Unsupervised Episode Detection for Large-Scale News Events

Priyanka Kargupta pk36@illinois.edu **UIUC** Illinois, USA

Yunyi Zhang yzhan238@illinois.edu **UIUC** Illinois, USA

Yizhu Jiao yizhuj2@illinois.edu **UIUC** Illinois, USA

Siru Ouyang siruo2@illinois.edu **UIUC** Illinois, USA

Jiawei Han hanj@illinois.edu **UIUC** Illinois, USA



Figure 1: Example of an event structure hierarchy. We take a key event corpus as input and aim to detect each episode node with a corresponding cluster of relevant text segments.

fore, struggle with these crucial characteristics. This paper introduces a novel task, episode detection, aimed at identifying episodes from a news corpus containing key event articles. An episode describes a cohesive cluster of core entities (e.g., "protesters", "police") performing actions at a specific time and location. Furthermore, an episode is a significant part of a larger group of episodes under a particular key event. Automatically detecting episodes is challenging because, unlike key events and atomic actions, we cannot rely on explicit mentions of times and locations to distinguish between episodes or use semantic similarity to merge inconsistent episode co-references. To address these challenges, we introduce EpiMine, an unsupervised episode detection framework that (1) automatically identifies the most salient, key-event-relevant terms and segments, (2) determines candidate episodes in an article based on natural episodic partitions estimated through shifts in discriminative term combinations, and (3) refines and forms final episode clusters using large language model-based reasoning on the candidate episodes. We construct three diverse, real-world event datasets annotated at the episode level. EpiMine outperforms all baselines on these datasets by an average 59.2% increase across all metrics.

Episodic structures are inherently interpretable and adaptable to

evolving large-scale key events. However, state-of-the-art automatic event detection methods overlook event episodes and, there-

CCS Concepts

Abstract

• Computing methodologies \rightarrow Information extraction; • Information systems \rightarrow Data mining.

Keywords

Unsupervised Episode Detection, Event Detection, Segment Classification, Large Language Models, Key Term Extraction

Introduction

Given the saturation of real-time news accessible at our fingertips, reading and processing the critical information of a key event has become an increasingly daunting challenge. Consequently, research on automatic textual event detection has recently attempted to integrate the manner in which humans neurologically perceive/store events into textual event detection methods. Specifically, neuroscientists studying event representations in human memory find that events are stored in a top-to-bottom hierarchy, as demonstrated

in Figure 1. The deeper the hierarchical event level, the more finegrained its corresponding text granularity [48]: we consider a theme as corpus-level (all articles discussing the 2019 Hong Kong Protests), key event as document-level (an article typically discusses a full one to two day key event), episode as segment-level, and atomic action as sentence or phrase-level.

Furthermore, neurological research [3, 21] indicates that events are encoded into memory as episodic structures. Representing events as discrete episodes helps us piece together a coherent narrative by considering the sequence of actions, reactions, and developments over time. This empowers several downstream tasks (e.g., event schema generation, event prediction), which may benefit from insights into the causes and consequences of events that might be missed with a more generalized analysis. Despite its strengths, existing automatic event extraction works fail to consider the episode-level.

For example, key event detection specifically seeks to output "a set of thematically coherent documents" for each key event [29, 48]. However, it is challenging to manually parse through a large cluster of relevant articles in order to gain an efficient and compact understanding of a given event. To address the lack of interpretability of such article clusters, the adjacent task of timeline summarization [9, 14, 23, 39] aims to identify the dates and a compact summary for each key event. However, high-level timelines are typically applicable for historical themes and thus unrealistic for currently evolving key events where fine-grained timeline summarization is more suitable. Hence, event chain mining [18] attempts to address this by mining a series of temporally-ordered atomic actions at the phraselevel; however, the granularity level is often too fine-grained to properly represent a large-scale key event. Thus, we aim to tackle the novel task of **episode detection** to pave the way for a more effective event representation.

Episode detection aims to detect episodes from a news corpus containing key event articles. An episode can be described as a cohesive cluster of subjects performing actions at a certain time and location, occurring as part of a larger sequence of episodes under a specific key event. Episode detection introduces a *unique set of challenges*, which we address using our novel framework, **EpiMine**. EpiMine is an <u>unsupervised</u> episode detection framework that automatically detects meaningful episodic events and their corresponding text segments in a large key event corpus, all without any level of human supervision or labeled training data. EpiMine is comprised of the following components: (1) discriminative co-occurrence detection, (2) episode partitioning, (3) candidate episode estimation, and (4) episode-segment classification. Collectively, they tackle the unique challenges of episode detection, detailed below:

Challenge 1: Journalists do not timestamp episodes. Key event detection partitions a thematic corpus into document-level clusters by heavily relying on explicit temporal features, like publication dates [48]. For example, an article primarily discusses one key event, which can then be roughly mapped to the article's publication date. However, this assumption fails at the episode-level, where there is no guarantee to have a distinct timestamp associated with each text segment that discusses a new episode. Fortunately, we can take advantage of the idea that journalists naturally partition news articles by sequentially discussing distinct episodes:

Example: An article likely completes its discussion of the episode A, protesters storming the Legislaive Council, before episode B, "protesters vandalized the Legislative Chamber" (Figure 3).

Hence, in order to partition articles into distinct episode segments, EpiMine must identify whether or not two consecutive segments are discussing the same or different episodes, which brings us to our next challenge.

Challenge 2: Episodes contain semantically diverse actions. Each episode features a *set of unique atomic actions*, which we can utilize for determining whether or not two segments discuss the same episode. However, for clustering actions, existing methods [18] rely heavily on semantic similarity. This not realistic for episode-segment clustering:

Example: "protesters spray-painted slogans" and "they unfurled the colonial-era flag" will fall under the same episode, but are semantically different and unlikely to be clustered.

Alternatively, we can identify salient terms that reflect the same episode ("barriers" and "shoved" are unique to Episode A; "defaced" and "walls" for Episode B), by exploiting corpus-level signals. Specifically, if we frequently see "defaced" mentioned together with "walls" (or their respective synonyms) and not with other terms (and vice-versa), then we consider them a discriminative co-occurrence. Consequently, when such co-occurrences remain consistent across text segments, this indicates the same episode being discussed. Conversely, if a sufficient shift in the combination of terms occurs, then this indicates that a different episode is being discussed.

Challenge 3: Articles often do not feature all episodes. Given the real-time nature of reporting, an article may feature only a subset or none of ground-truth episodes from a multi-day key event (e.g., focusing on a high-level analysis of the key event or mentioning other related key events). To minimize this noise,

EpiMine seeks to identify the set of articles which maximizes the quantity and quality of potential episodes. It then merges any article partitions across these articles which likely discuss the same episode and employs a large language model (LLM) to provide a more fluent interpretation of the candidate episodes, accounting for the episode's core entity, actions, object, location, and time period. This allows EpiMine to finally map the remaining non-salient article segments to these episodes, pruning any candidates which are not sufficiently supported by the remaining articles. We summarize our core contributions:

- (1) We introduce the novel task of episode detection, which takes in a key-event corpus and outputs multiple detected episodes and their related segments extracted from the corpus.
- (2) We propose a novel unsupervised episode detection method, EpiMine, which exploits discriminative term co-occurrences to estimate candidate episode partitions from top articles. It refines these candidate episodes using LLM-based reasoning to output a comprehensive and diverse set of episodes.
- (3) We construct **three novel datasets**, reflecting a diverse set of real-world themes and thirty global key events, as no large-scale key event-specific news corpus exists for this task where the key events are guaranteed to contain distinguishable episodes.
- (4) We compare EpiMine with five other baselines through an extensive set of experiments and case studies performed on our real-world datasets, demonstrating that it outperforms all baselines by, on average, a 59.2% increase across all metrics.

Reproducibility: We provide our dataset and source code¹ to facilitate further studies.

2 Related Works

2.1 Event Extraction

Understanding events in unstructured texts is crucial, leading to substantial research in event detection and extraction [11, 17, 25, 27, 30, 35]. Existing work, such as Ahmad et al. [1], Han et al. [15], Wang et al. [40], have focused on event relation extraction and prediction, while salient event identification remains a significant area of study [19, 28, 43]. Furthermore, event process understanding has garnered recent attention [8, 46]. However, these studies typically require costly expert annotations. Some efforts have sought to mitigate this challenge through unsupervised approaches [24, 42]. The pioneering work [7] introduced event chains as a novel representation of structured news knowledge. In the realm of biological reading comprehension, Berant et al. [4] extracted events and their relationships in biological processes. More recently, Zhang et al. [47] utilized event graphs constructed from large corpora to derive event chains for narrative understanding. Despite these advancements, most existing studies concentrate on phrase-level or sentence-level events (analogous to actions in this paper), neglecting the hierarchical structures of events at varying granularities within textual data. Additionally, these approaches often rely on human-curated event ontologies, are restricted to predefined event types, and fail to capture events from open-domain texts. Although key event extraction has been explored in some studies [18, 48], the automatic detection of episodes remains under-explored and critically important.

 $^{^{1}}https://github.com/pkargupta/epimine\\$

2.2 Timeline Summarization

Timeline summarization (TLS) automatically identifies key dates of major events and provide concise descriptions of occurrences on those dates. Previous approaches to TLS have predominantly focused on extractive methods. For instance, Nguyen et al. [34] built thematic timelines for general-domain topics by selecting and ranking events relevant to an input query. Martschat and Markert [31] introduced a temporally sensitive submodularity framework, designing objective functions and constraints that model the temporal dimension inherent in timeline summarization. Steen and Markert [39] were the first to propose an abstractive timeline summarization system. Their approach clusters sentences likely describing the same event and uses multi-sentence compression to generate summary sentences for each cluster. More recently, Li et al. [23] represented news articles as event-graphs, transforming the summarization task into compressing the graph to its salient sub-graph. Additionally, You et al. [45] presented a joint learning-based heterogeneous graph attention network, integrating event detection into a unified framework to eliminate redundant sentences. While these methods involve key event grouping and understanding, they typically apply to historical themes, producing higher-level timelines. However, these approaches may not be practical for ongoing news events, where a detailed timeline summary (distinguishing between episodes) is more appropriate.

2.3 Topic Discovery

The goal of topic discovery is to group news articles by their themes, an area of study dating back to the early days of automated event theme identification from textual data [2]. One approach [5, 50] treats this task as a topic modeling issue, using LDA [6] or its enhanced versions. Another line of research [37, 38] constructs a graph based on keyword frequency and identifies themes from keyword clusters, mapping these themes to documents via keyword-driven feature similarities. However, these techniques do not consider temporal information, limiting them to discerning different thematic events. To address this, some studies [12, 16] introduce tracking keyword spikes to pinpoint events. Specifically, [49] uses burst detection [22] to identify and utilize bursty terms for document representation and clustering. Building on this, [13] creates a burst information network with recognized bursts and their attributes for clustering. Finally, [44] proposes unsupervised stream-based story discovery, which computes article embeddings their shared temporal themes. Overall, these methods solely focus on the key-event or atomic action level.

3 Preliminaries

3.1 Problem Definition

Definition 3.1 (Episode). An episode E_i is one of a partially ordered sequence of subevents, $\{E_1, \ldots, E_i, \ldots E_k\}$, of a key (major) event E, where typically $2 \ge k \ge 20$, and E_i does not overlap with E_j if $i \ne j$. Actions in the episode E_i can be either semantically similar or diverse, but typically have relatively tight time, location, and/or thematic (entities, actions, objects) proximity between them.

EpiMine aims to extract episodes from a news corpus, where an episode occurs as a significant component of a larger group of episodes that fall under a specific key event. For instance, in Figure 1, without knowing Episode #1, "Protesters stormed the Legislative Council Complex", readers would not have fully understood Episode #3, "Police dispersed protesters". Hence, episodes help us understand the overall key event and are especially useful for events that are currently evolving, where finer context is required for sufficiently understanding them.

Definition 3.2 (Episode Detection). Given a corpus \mathcal{D} about one key event, where each document $d \in \mathcal{D}$ is a news article, the goal of the task is to obtain the set of text segment clusters $\mathcal{E} = \{E_1, E_2, \dots, E_K\}$. Each episode cluster $E_i \subset \mathcal{S} = \{s_1^1, s_2^1, \dots, s_{|\mathcal{D}|}^{|\mathcal{D}|}\}$, where \mathcal{S} contains all of the text segments identified in each document $d \in \mathcal{D}$, and every two clusters do not have overlapping text segments (i.e., $E_i \cap E_i = \emptyset$ for $i \neq j$).

It is important to note that the number of episodes K is not known in advance and oftentimes, a news article segment may discuss either episodes of a different key event (e.g., an episode with similar aspects that occurred in a different historical key event) or multiple episodes of the current key event. Nonetheless, our goal is to detect the most relevant episodes to the current key event at hand and consequently mine the most distinctive text segments for each of these (hence our constraint of $E_i \cap E_j = \emptyset$ for $i \neq j$).

3.2 Key Event News Article Pre-Processing

Given that the expected output for the episode detection task is a cluster of text segments, we first must segment each key event news article. We would like to ideally preserve both the primary aspects (e.g., core entities and their actions) and peripheral aspects (e.g., reactions to a core entity's action) relevant to that episode, which may be helpful for cross-document episode co-reference resolution. In order to do this, we utilize the text segmentation method, C99 [10]. Furthermore, in order to assist with the cohesiveness of the segment, we employ entity co-reference resolution before performing segmentation, which assists with retaining the context across text segments ("They surrounded the legislative building [...]") \rightarrow "The protesters surrounded the legislative building [...]"). In the following section, we present our core methodology given these text segments (in their raw form, without co-references resolved) and their source articles as the primary inputs.

4 Methodology

In order to tackle the episode detection task, we propose a novel unsupervised framework, EpiMine. As shown in Figure 2, EpiMine consists of the following four core components: (1) **episode indicative term mining**, which identifies combinations of salient terms that are likely to discriminatively co-occur *within* an episode and not *across* episodes; (2) **episode partitioning**, which partitions each article into approximate isolated episodes based on consecutive shifts in the term co-occurrence distribution, (3) **candidate episode estimation**, which clusters the top partitions into candidate episodes and utilizes LLM-based reasoning to produce fluent and meaningful episodes, and (4) **episode-segment classification**, which maps confident segments to their respective episode clusters.

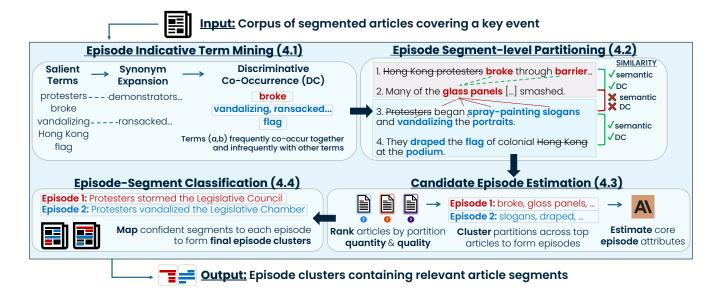


Figure 2: We detail the overall framework of EpiMine.

Hong Kong protesters broke through the final barrier separating them and the interior of the city's Legislative Council Complex. Many of the glass panels lining the building's exterior had already been smashed in the afternoon, but it took another hour or so before protesters were able to breach the metal shutters.

Within minutes, <u>protesters</u> began spray-painting slogans onto the corridor walls and vandalizing the portraits of <u>Legislative Council</u> presidents. They also draped the flag of colonial-era <u>Hong Kong</u> at the podium of the legislative chamber.

Figure 3: Natural partition between two episodes within a key event article. An episode's discriminative terms are bolded, while salient but non-discriminative terms are underlined.

4.1 Episode Indicative Term Mining

In the absence of supervision, our goal is to identify *potential* candidates for episodes. Episodes are often described in relation to each other and usually lack specific timestamps or locations consistently mentioned within their segments. For example, the phrase "police dispersed protesters" may not have a precise timestamp because it is a *response* to "protesters stormed the Legislative Council Complex," and some journalists may consider the implicit ordering adequate.

Additionally, the same episode can be described using different entities and actions, as journalists may highlight different perspectives when reporting an episode. For example, both "protesters shoved against the barricades" and "the police used pepper-spray on the protesters" describe the episode "protesters stormed the Legislative Council Complex". However, they are semantically different, focusing on different core entities and actions. Therefore, we cannot depend on a consistent subject-action-object triple or an explicit time and location associated with each episode in the articles.

Salient Terms. To circumvent this challenge, we exploit the idea that journalists naturally partition news articles according to episodes, forming episode fragments. For example, as shown in Figure 3, an article will likely complete its discussion of episode #1, "Protesters stormed the Legislative Council Complex" (red), before fully shifting to discussing episode #2, "protesters vandalized the Legislative Chamber" (blue). Across these episode fragments, certain salient terms are featured (e.g., protesters, legislative, vandalizing, podium). We modify the idea of event salience from [18] for the task of episode detection:

Definition 4.1 (Salience). A term is <u>salient</u> if it is (1) distinct and significant to understanding a given key event, as well as (2) frequently found in a key event's segments and infrequently in other background/general articles.

We define the salience score of a term w_i within segment s as the following function, where $freq(w_i)$ is the number of key event segments that w_i is contained in, N_{bg} is the number of news articles in the background corpus we construct (using general New York Times articles), and $bgf(w_i)$ is the number of background articles that w_i is present in.

Salience
$$(w_i) = \left(1 + \log^2\left(freq(w_i)\right)\right) \log\left(\frac{N_{bg}}{bqf(w_i)}\right)$$
 (1)

Stop words and infrequent terms ($freq(w_i) < 5$) are assigned a salience score of -1. A key event's set of salient terms T is comprised of the terms with a salience score *above the mean salience* across the entire vocabulary. In the case of infrequent synonyms used by a journalist as a stylistic choice (e.g., "demonstrations", "marches"), we expand T with terms that are similar (cosine-similarity) to their static word representations (average of its contextualized word embeddings across entire key event corpus).

Discriminative Co-occurrence. Now, each text segment within the key event corpus has a corresponding set of salient terms identified. In Figure 3, we can see that the first episode fragment, and Episode #1 in general, features a combination of similar terms, such as "protesters", "barrier", and "breach". Likewise, the second episode may include a combination of terms similar to "protesters", "spray-painting", and/or "flag". Hence, we introduce the significance of co-occurrence for episode detection. Despite some journalists choosing to only describe the protesters spray-painting while others focus on the protesters draping the colonial-era flag, we must be able to recognize that their respective salient terms are likely to co-occur within the same episode.

However, we make a **novel distinction** between a co-occurrence and a *discriminative* co-occurrence. A salient term a (e.g., "protesters") may often co-occur with another salient term b ("spray-painting") within an episode. On the other hand, if a frequently co-occurs with many other terms in various episodes ("protesters broke"), a and its co-occurrences are less useful for distinguishing episodes. Thus, (a, b) is not a discriminative co-occurrence.

Definition 4.2 (Discriminative Co-occurrence). A pair of terms (a,b) discriminatively co-occur if (1) they frequently appear together in episode E_i , and (2) neither a nor b appear as frequently with other terms w in other episodes $E_{\notin i}$.

We compute the discriminative occurrence d between salient term pair (a, b) using the following equation:

$$\mathbf{d}(a,b) = \log\left(\frac{freq(a,b)}{\max(\bar{f}_a,\bar{f}_b)}\right) \times \log\left(\frac{|T|}{\max(|F_a|,|F_b|)}\right),$$
where $\bar{f}_a = \frac{1}{|T|} \sum_{\forall w_i \in T} freq(a,w_i)$, and
$$F_a = \{freq(a,w_i) > 1 \ \forall \ w_i \in T\}$$
(2)

The first log term ensures that the pair's co-occurrence (freq(a, b)) is **statistically significant** (\geq the max of *a* and *b*'s mean vocabularywide co-occurrence respectively). The second log term ensures that the salient term pair is a discriminative matching, penalizing pairings where either a or b co-occurs with a large proportion of the salient term set T). For example, co-occurrences with "protesters" are not discriminative because "protesters" is a core entity in all episodes and thus frequently co-occurs with many terms in T. In contrast, ("slogans", "flags") is a discriminative co-occurrence since both terms frequently appear together in segments discussing episode #2 and rarely co-occur with other terms $w_i \in T$. If a and bare the same term or close synonyms (determined by statistically significant semantic similarity), they have maximum co-occurrence. By leveraging multiple articles in a large key event corpus, we have sufficient statistical support to ensure our output reflects the average realistic reporting of the key event and its episodes.

4.2 Episode Partitioning

With the ability to identify discriminative co-occurrences, we can use a key transitive property to resolve episode co-references within and across articles, where *not all combinations of an episode's discriminative terms explicitly co-occur*:

If (a, b) and (b, c) are both discriminative co-occurrences, then (a, c) is also likely to be a discriminative co-occurrence.

To illustrate this, we have the following text segment excerpts of a news article (the salient and discriminative terms are *italicized*):

- (1) Protesters defaced the emblem of Hong Kong, spray-painted slogans, and unfurled the colonial-era flag.
- (2) The portrait of LegCo president Andrew Leung was defaced.
- (3) A *slogan* on the *wall* reads: "The government forced us to revolt".
- (4) *Police* said at least 13 people had been *arrested* on *suspicion* of involvement in the pro-democracy protest.

We can naturally see that segments 1-3 all discuss the "protesters vandalized the Legislative Chamber" episode, while segment 4 discusses the "police dispersed protesters" episode. We can systematically replicate this partitioning process by considering the discriminative co-occurrence score between all pairwise combinations of terms from segments (i-1) and (i). If the average discriminative co-occurrence and static semantic similarity between each term a from (i-1) and b from (i) is statistically significant $(\geq \mu_d - \sigma_d)$ for that specific article d (e.g., notably (slogans, defaced) for segments 1-3), we hypothesize that the **same episode** is being discussed and merge them into one episode fragment. If not (e.g., (slogans, arrested) for segments 3-4), this indicates that a **different episode** is being discussed, and we partition them into two episode fragments. Further implementation details are provided in Appendix B.

4.3 Candidate Episode Estimation

After partitioning each article into episode fragments, we can identify the set of articles that maximizes the *quantity and quality* of potential episodes. This step is crucial as it enables EpiMine to **minimize noise from articles** that may discuss *only a subset or none of the ground-truth episodes* (e.g., focusing on high-level analyses or other relevant key events). Simultaneously, it improves our method's efficiency by not relying on the entire key event corpus. We compute the per-article rank by multiplying two metrics:

- (1) Quality of episode fragments: A top article should primarily consist of episode fragments containing salient terms that discriminatively co-occur. This reduces the rank of general fragments which summarize/analyze the event. We use the average of each episode fragment's mean inner-discriminative co-occurrence (across all pairwise combinations of a fragment's salient terms).
- (2) *Quantity of episode fragments*: A top article should ideally contain the highest number of ground-truth episodes. Therefore, we take the log of the number of episode fragments in the article.

After ranking all articles, we select the top $\delta\%$ to serve as the basis for candidate episode estimation. To estimate the candidate episodes, we must first resolve potential co-references to the same episode across these top articles. We apply agglomerative clustering [33] to the top episode fragments using a pre-computed distance matrix. The distance between two fragments (inversed) is calculated using the same discriminative and static semantic similarity score used in Section 4.2). We then prune clusters that do not contain a statistically significant number of episode fragments.

Finally, we use an LLM to resolve (1) the lack of time and location stamps in an episode fragment and (2) semantic inconsistency between the fragments within a cluster. The LLM leverages its reasoning capabilities to **estimate the candidate episodes** by identifying the **core attributes** of each unique cluster (entities, actions, objects, location, and time). It outputs the *episode attributes*,

relevant keywords (assists with extraction), and top extracted text segments based on the clusters (prompt & example in Appendix C).

4.4 Episode-Segment Classification

With these core summaries of the episode clusters, we obtain a generalized description of each candidate episode. This allows us to compute a simple episode representation for assigning an episode and confidence score to each input segment.

Episode & Segment Representations. We compute a candidate episode representation by encoding both its LLM core attributes and extracted segments using SentenceTransformers (ST) [36]. Following extremely weakly supervised text classification works, such as [20, 41], we take the harmonic mean of these representations, as the latter extracted segments are likely not as significant as the earlier extractions and core attributes. We similarly encode all input segments with the same ST model.

Episode-Segment Confidence Estimation. Following recent work on extremely weakly supervised text classification [20], we compute a confidence score specific to the episode-segment classification task. Simply mapping a text segment to its top episode (based on highest cosine similarity) overlooks the presence of episode-irrelevant text segments and fails to address segments discussing multiple episodes (e.g., a journalist's key event summary at the beginning of an article). We aim to avoid assigning such segments to any cluster, ensuring no overlap between episode clusters, as discussed in Section 3.1.

Thus, we compute segment s_i 's cosine similarity to its top two episodes (e_i^0 and e_i^1). A larger gap ($e_i^0 - e_i^1$) between the two indicates higher confidence in mapping s_i to its top episode. We normalize each gap by the sum of all segment episode gaps within the entire corpus, ensuring the confidence is relative to the key event:

$$s_{i,\text{confidence}} = \frac{e_i^0 - e_i^1}{\sum_{l=1}^{|S|} (e_l^0 - e_l^1)}$$
(3)

After computing each segment's confidence in its top episode, we assign only those segments with a statistically significant confidence level to their respective episode clusters E. We then prune candidate episode clusters with ≤ 0 assigned segments, resulting in our **final detected episodes and their clusters**, \mathcal{E} .

5 Experimental Setup

For implementing **EpiMine**, we use the following hyperparameters across all datasets: $\delta = 25\%$, $sim_thresh = 0.75$. All other hyperparameters are set to their respective default values. To determine statistical significance, we check for whether or not a value is $\geq \mu - \sigma$. For our word representations, we use bert-base-uncased. For our sentence representations, we use all-mpnet-base-v2. We choose to use Claude-2.1 2 for fluent candidate episode estimation due to its strong structured JSON/XML input and output formatting abilities. However, this proprietary model can be replaced with any open-source model as EpiMine is model-agnostic. We use only one NVIDIA GeForce GTX 1080 for all experiments.

²claude.ai/

Table 1: Statistics of our collected datasets. The numbers are averaged per key event.

Theme	# docs	# episodes	# segments
Terrorism/Attacks	32.2	5.9	290.3
Natural Disasters	36.2	7.4	324.6
Political Events	70.2	7.5	667.7

5.1 Datasets

We conduct our experiments on three thematic, real-world news corpora selected from Wikipedia³ over the last decade. For each theme, we manually collect approximately 10 key events composed of multiple articles and ensure that distinct *episodes* exist in these articles. These articles are originally obtained from the hyperlinks of references from the Wikipage of each key event, and then filtered with constraints in time, language, and relevance. We avoid excessive filtering (allowing for articles that may contain either a subset or even none of the ground truth episodes identified) in order to **preserve the challenging, real-world nature** of the task.

During the collection process, we targeted selecting a diverse set of key events topics within a theme. For instance, we attempted to cover every type of "natural disaster", including tornados, wild-fires, and etc. When selecting key events, we leave out those with less than 20 hyperlinks in the Wikipage. Table 1 summarizes the statistics for these datasets. We also construct a background news corpus of approximately 4,000 long news articles using the New York Times corpus for topic categorization [32]. We provide further details on the dataset collection criteria/process, each theme, and corresponding key events in Appendices E and F.

5.2 Baseline Models

We compare against the following methods using the evaluation metrics specified in Appendix D. For the two most competitive baselines, K-means and EvMine, we also integrate EpiMine's same exact LLM model/prompt (Appendix C) for candidate episode estimation (Section 4.3) and confidence-based episode-segment classification process (Section 4.4) to demonstrate any LLM-based improvements.

- K-means [26]: No. of ground-truth episodes is given; clusters segments based on semantic similarity of ST [36] embeddings.
- EvMine [48]: Unsupervised framework for key event detection that leverages peak phrases and detects communities using event-indicative features. We extend the original document-level method to the segment level for episode detection.
- EMiner [18]: Unsupervised event chain mining that performs atomic action clustering. For episode detection, we map its final output, a list of events, back to the original sentences from which each event was extracted, treating these sentences as segments. To retrieve more episode-associated segments, we use ST [36] to select the *k* most similar segments to each cluster sentence.

We also include the following full and partial ablations of EpiMine (clusters segments from all articles vs. top $\delta\%$ articles, respectively):

³https://en.wikipedia.org/wiki/

Table 2: Results averaged across each theme, including the mean # of episodes EpiMine identifies per theme. Results are computed on each key event corpus using the top-5 documents for each detected episode. Due to variance in LLM generation, we run it 10 times and report the average of each measure. Bold values denote the top method; † denotes the second-best method.

Terrorism (avg. # of eps = 5			eps = 5.36)	Natural Disasters (avg. # of eps = 7.4)			Politics (avg. # of eps = 7.5)		
Methods	5-prec	5-recall	5-F1	5-prec	5-recall	5-F1	5-prec	5-recall	5-F1
EMiner	0.0864	0.0025	0.0048	0.1037	0.0019	0.0037	0.0866	0.0016	0.0032
K-means K-means + LLM	0.2123 0.2858	0.2123 0.1404	0.2123 0.1826	0.2785 0.3740	0.2847 0.1658	$0.2814^{\dagger} \\ 0.22$	0.1604 0.2718	0.1604 0.1736	0.1604 0.1825 [†]
EvMine EvMine + LLM	0.2303 0.3788	0.1502 0.1570	0.1745 0.2133	0.2815 0.4356	0.0802 0.1322	0.1225 0.1940	0.0536 0.3273	0.04 0.1298	0.0458 0.1728
EpiMine - No Confidence	0.7121 0.6197 [†]	0.2207 [†] 0.3019	0.3243 [†] 0.3845	0.7098 0.4366 [†]	$0.2846^{\dagger} \\ 0.2078$	0.3453 0.2776	0.6267 0.6029 [†]	0.2154 [†] 0.2773	0.2923 [†] 0.3385

Table 3: Ablation studies conducted on top 25% of article episode clusters (Section 4.3).

	Terrorism			Natural Disasters			Politics		
Ablations	5-prec	5-recall	5-F1	5-prec	5-recall	5-F1	5-prec	5-recall	5-F1
EpiMine-Top	0.2292	0.2435	0.2144	0.3817	0.2232	0.2450	0.1051^{\dagger}	0.2233	0.1201^{\dagger}
TF-IDF No DC	0.0985 0.1968 [†]	$0.1403 \\ 0.1752^{\dagger}$	$0.1059 \\ 0.1707^{\dagger}$	$0.3284^{\dagger} \\ 0.2520$	0.1919 [†] 0.1546	$0.2221^{\dagger} \\ 0.1785$	0.0907 0.1126	$0.1908 \\ 0.2108^{\dagger}$	0.0916 0.1299

Table 4: Compares the top-5 salient terms which (1) have the highest cosine-similarity (CS) and (2) highest discriminative co-occurrence (DC), with the given keyword.

Keyword	CS	DC		
broke	stormed, ransacked,	glass, doors, metal, build-		
	dashed, occupied, rushed	ing, teargas		
slogans	spray, placards, painted,	reads, wall, damage,		
	defaced, pictures	started, portraits, spray		

- **No Confidence**: A full ablation, where all input segments are classified based on the episode with max cosine similarity instead of using the confidence score from Equation 3.
- EpiMine-Top: A partial ablation which directly outputs the intermediate episode clusters formed based on the top articles identified in Section 4.3 without inputting them into the LLMbased episode estimation step.
- TF-IDF: A partial ablation which replaces the salience and synonym expansion step (Section 4.1) with TF-IDF.
- No DC: A partial ablation which replaces the discriminative co-occurrence score (Equation 2) with raw pair frequency.

5.3 Overall Results & Analysis

In Table 2, EpiMine shows an average **80.8**% increase in 5-precision, a **34.0**% increase in 5-recall, and a **62.8**% increase in 5-F1 over all baselines. Notably, despite both 'K-means' and 'K-means + LLM' being *given the ground-truth number of episodes*, they are **significantly outperformed by EpiMine** (both the base model and no confidence ablation). Additionally, EvMine and EMiner, originally designed for key event and atomic action levels of event granularity,

fail to address the unique challenges of episode detection.

We further analyze our results through extensive quantitative and qualitative studies, including a detailed case study on the "2019 Hong Kong Legislative Protest" (as shown in Figure 1), leading to the following takeaways:

(1) The strengths of discriminative co-occurrence complement those of cosine similarity:

Table 4 illustrates the qualitative strengths of our novel discriminative co-occurrence metric. Both cosine similarity (CS) and discriminative co-occurrence (DC) offer different, complementary strengths. CS identifies similar words that play a *similar role or are synonyms* within an episode (e.g., "broke", "ransacked"), while DC identifies the *key surrounding actions and objects* that co-occur within the same episode (e.g., "slogans", "wall"). This is quantitatively supported by the ablation studies in Table 3, which show a significant decrease in the quality of our top articles (partitioned into episodes) without our salience and discriminative co-occurrence measures. The politics dataset does show slight improvements in precision and F1 when discriminative co-occurrence is replaced, due to more term overlap across episodes, resulting in less distinct, sequential episodes.

(2) LLMs require effective episode fragment clusters for quality episode estimation:

In Table 5, we compare EpiMine with the most competitive baselines integrated with our LLM-based candidate episode estimation and classification, as well as Claude-2.1 and ChatGPT-4⁴. Without any initial clusters as guidance, LLMs alone (Claude,

⁴https://chat.openai.com/

Table 5: Gold and detected episodes (a maximum of five are included for brevity) for the "2019 Hong Kong Legislative Protests" key event. We specify the gold/detected episode attributes for each episode cluster in the following semicolon-separated format: core entity; action; object; time; location. "Not detected" denotes that no more episodes were generated by the model.

Model	Episode #1	Episode #2	Episode #3	Episode #4	Episode #5
Gold (total 5 eps)	Activists; headed; towards the Leg- islative Council Complex; 1 July 2019; Hong Kong	Protesters; stormed; the Legislative Council Complex; around 9:00 pm; Hong Kong;	Protesters; damaged/defaced; por- traits, furniture, emblem, etc.; 1 July 2019; Legislative Council Complex	Police; started using; tear gas to disperse protesters; 12:05 am 2 July; around the Legislative Coun- cil complex	Police; arrested; individuals in con- nection with the incident; between 3 July and 5 July; Hong Kong
K-means + LLM (total 4 eps)	Protesters; storm and vandalize; Legislative Council building; July 1, 2019; Legislative Council com- plex in Admiralty, Hong Kong	Hong Kong government; con- demns; protesters storming legisla- tive building; July 1, 2019; Hong Kong	Hong Kong protesters; express; de- mands for freedom and democ- racy; July 1, 2019; Hong Kong Leg- islative Council	Hong Kong police; adopt; more re- strained tactics; July 1, 2019; Hong Kong Legislative Council	Not detected
EvMine + LLM (total 4 eps)	Protesters; vandalize; Hong Kong legislative building; July 1, 2019; Hong Kong legislative building	Protesters; occupy and vandalize; Hong Kong legislative chamber; July 1, 2019; Hong Kong legislative building	Protesters; spray paint; slogans and demands; July 1, 2019; Hong Kong legislative building	Protesters; deface; Hong Kong emblem; July 1, 2019; Hong Kong legislative building	Not detected
Claude (total 3 eps)	Protesters; storm; Hong Kong's Legislative Council; July 1, 2019; Hong Kong's Legislative Council building	Police; retreat and avoid confronta- tion; protesters storming Hong Kong's Legislative Council; July 1, 2019; Hong Kong's Legislative Council building	Brian Leung Kai-ping; pulls off mask and reads protesters' de- mands; inside Hong Kong's Leg- islative Council; July 1, 2019; Leg- islative Council chamber	Not detected	Not detected
GPT-4 (total 2 eps)	Hong Kong protesters; storm Leg- islative Council; government and police; July 1, 2019; Legislative Council Complex, Hong Kong	Hong Kong citizens; march against extradition bill; "Carrie Lam and Chinese government; June 2019; Various locations in Hong Kong	Not detected	Not detected	
EpiMine (total 7 eps)	protesters; broke into and oc- cupied; Hong Kong's legislative building; July 1, 2019; Hong Kong	protesters; vandalized; the legisla- tive building; after breaking in; Hong Kong	police; fired tear gas at; protesters; after midnight on July 1; outside the legislative building	Carrie Lam; condemned; the protesters' actions; in a news conference at 4am on July 2; Hong Kong	police; began making arrests of; protesters involved; in the days af- ter; Hong Kong

GPT-4) fail to detect high-quality episodes, missing most ground-truth episodes and including irrelevant atomic actions (e.g., "Brian Leung pulls off mask"). Similarly, using low-quality clusters from the baselines results in poor performance. EvMine detects episodes that all reflect the same event, "Protesters vandalized the Legislative Chamber". While K-means produces more distinct episodes, it does not capture the most critical, gold episodes. In contrast, EpiMine's episodes are both distinct and meaningful.

An interesting observation is that EpiMine's clusters elicit the LLM to identify more meaningful temporal information. Unlike most baseline episodes, which simply have "July 1, 2019" as the time attribute, EpiMine's episodes feature more descriptive temporal cues: "after breaking in", "after midnight", "in a news conference at 4 am on July 2". Moreover, EpiMine's "incorrect" episode #4 is a significant sub-event of the key event discussed by many articles. This strongly demonstrates the impact of EpiMine's candidate episode clusters as input into the LLM; LLMs alone cannot perform quality episode detection.

(3) Fragment ranking effectively identifies top articles:

In Figure 4, we conduct a sensitivity analysis of the top articles chosen to estimate candidate episodes (Section 4.3). We compare the gold episodes contained in the set of the top $\delta\%$ articles as we vary δ . By ranking the articles based on their likelihood of containing both high-quality and numerous episodes, we find that *EpiMine's top article selection covers the vast majority of episodes* by $\delta=25\%$ and more comprehensively around $\delta=45\%$. This is significant as it helps *minimize both the noise and the amount of data* needed to accurately detect all episodes.

(4) Confidence leads to more conservative classification:

The "No-Confidence" ablation in Table 2 shows that using confidence (Equation 3) improves EpiMine's precision but reduces

recall. This conservative approach results in EpiMine being cautious when assigning segments to episode clusters, excluding segments with insufficient confidence. This indicates that our method *relies on both our clustering method and confidence scoring* for *precise* episode detection. EpiMine users can determine if they prefer a confidence or no-confidence method depending on their use-case (higher precision versus higher recall). Nonetheless, both version still outperform all baselines.

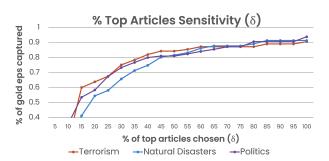


Figure 4: Percentage of key event's gold episodes captured in the $\delta\%$ top articles chosen during the candidate episode estimation. Results averaged across themes.

6 Conclusion

In this work, we propose our novel method **EpiMine** for unsupervised episode detection for large-scale news events. EpiMine performs (1) episode indicative term mining, which identifies combinations of salient terms that are likely to discriminatively co-occur within an episode and not across episodes, (2) episode partitioning, which partitions each article into approximate isolated episodes

based on consecutive shifts in the term co-occurrence distribution, (3) candidate episode estimation, which clusters the top partitions into candidate episodes and utilizes LLM-based reasoning to produce fluent and meaningful episodes, and (4) episode-segment classification, which maps confident segments to their respective episode clusters. EpiMine significantly outperforms all baselines on the vast majority of key events, as shown through extensive quantitative and qualitative analysis. Further work towards the temporal analysis of episodes within articles can be explored, as well as extending our work to primarily multilingual news settings with low resources.

References

- [1] Wasi Uddin Ahmad, Nanyun Peng, and Kai-Wei Chang. 2021. GATE: Graph Attention Transformer Encoder for Cross-lingual Relation and Event Extraction. In Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021. AAAI Press, 12462–12470. https://ojs.aaai.org/ index.php/AAAI/article/view/17478
- [2] James Allan, J. Carbonell, G. Doddington, J. Yamron, and Yiming Yang. 1998.
 Topic Detection and Tracking Pilot Study Final Report.
- [3] Christopher Baldassano, Janice Chen, Asieh Zadbood, Jonathan W Pillow, Uri Hasson, and Kenneth A Norman. 2017. Discovering event structure in continuous narrative perception and memory. *Neuron* 95, 3 (2017), 709–721.
- [4] Jonathan Berant, Vivek Srikumar, Pei-Chun Chen, Abby Vander Linden, Brittany Harding, Brad Huang, Peter Clark, and Christopher D. Manning. 2014. Modeling Biological Processes for Reading Comprehension. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL, Alessandro Moschitti, Bo Pang, and Walter Daelemans (Eds.). ACL. https://doi.org/10.3115/v1/d14-1159
- [5] Adham Beykikhoshk, Ognjen Arandjelovic, Dinh Phung, and Svetha Venkatesh. 2018. Discovering topic structures of a temporally evolving document corpus. Knowledge and Information Systems 55 (06 2018).
- [6] David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. Journal of machine Learning research 3, Jan (2003), 993-1022.
 [7] Nathanael Chambers and Daniel Jurafsky. 2008. Unsupervised Learning of
- [7] Nathanael Chambers and Daniel Jurafsky. 2008. Unsupervised Learning of Narrative Event Chains. In ACL 2008, Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics, June 15-20, 2008, Columbus, Ohio, USA, Kathleen R. McKeown, Johanna D. Moore, Simone Teufel, James Allan, and Sadaoki Furui (Eds.). The Association for Computer Linguistics, 789–797. https://aclanthology.org/P08-1090/
- [8] Muhao Chen, Hongming Zhang, Haoyu Wang, and Dan Roth. 2020. What Are You Trying to Do? Semantic Typing of Event Processes. In Proceedings of the 24th Conference on Computational Natural Language Learning, CoNLL 2020, Online, November 19-20, 2020, Raquel Fernández and Tal Linzen (Eds.). Association for Computational Linguistics, 531–542. https://doi.org/10.18653/v1/2020.conll-1.43
- [9] Xiuying Chen, Mingzhe Li, Shen Gao, Zhangming Chan, Dongyan Zhao, Xin Gao, Xiangliang Zhang, and Rui Yan. 2023. Follow the Timeline! Generating an Abstractive and Extractive Timeline Summary in Chronological Order. ACM Transactions on Information Systems 41, 1 (2023), 1–30.
- [10] Freddy Y. Y. Choi. 2000. Advances in domain independent linear text segmentation. In 1st Meeting of the North American Chapter of the Association for Computational Linguistics. https://aclanthology.org/A00-2004
- [11] Xinya Du and Claire Cardie. 2020. Event Extraction by Answering (Almost) Natural Questions. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020, Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu (Eds.). Association for Computational Linguistics, 671-683. https://doi.org/10.18653/v1/2020.emnlp-main.49
- [12] G. Fung, J. X. Yu, Philip S. Yu, and Hongjun Lu. 2005. Parameter Free Bursty Events Detection in Text Streams. In VLDB.
- [13] Tao Ge, Lei Cui, Baobao Chang, Zhifang Sui, and Ming Zhou. 2016. Event Detection with Burst Information Networks. In *COLING*.
- [14] Demian Gholipour Ghalandari and Georgiana Ifrim. 2020. Examining the State-of-the-Art in News Timeline Summarization. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault (Eds.). Association for Computational Linguistics, Online, 1322–1334. https://doi.org/10.18653/v1/2020.acl-main.122
- [15] Rujun Han, I-Hung Hsu, Mu Yang, Aram Galstyan, Ralph M. Weischedel, and Nanyun Peng. 2019. Deep Structured Neural Network for Event Temporal Relation Extraction. In Proceedings of the 23rd Conference on Computational Natural Language Learning, CoNLL 2019, Hong Kong, China, November 3-4, 2019,

- Mohit Bansal and Aline Villavicencio (Eds.). Association for Computational Linguistics, 666–106. https://doi.org/10.18653/v1/K19-1062
- [16] Qi He, Kuiyu Chang, and Ee-Peng Lim. 2007. Using Burstiness to Improve Clustering of Topics in News Streams. Seventh IEEE International Conference on Data Mining (ICDM 2007) (2007), 493–498.
- [17] Yizhu Jiao, Sha Li, Yiqing Xie, Ming Zhong, Heng Ji, and Jiawei Han. 2022. Open-Vocabulary Argument Role Prediction For Event Extraction. In Findings of the Association for Computational Linguistics: EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022, Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang (Eds.). Association for Computational Linguistics, 5404–5418. https://aclanthology.org/2022.findings-emnlp.395
- [18] Yizhu Jiao, Ming Zhong, Jiaming Shen, Yunyi Zhang, Chao Zhang, and Jiawei Han. 2023. Unsupervised Event Chain Mining from Multiple Documents. In Proceedings of the ACM Web Conference 2023. 1948–1959.
- [19] Disha Jindal, Daniel Deutsch, and Dan Roth. 2020. Is Killed More Significant than Fled? A Contextual Model for Salient Event Detection. In Proceedings of the 28th International Conference on Computational Linguistics, COLING 2020, Barcelona, Spain (Online), December 8-13, 2020, Donia Scott, Núria Bel, and Chengqing Zong (Eds.). International Committee on Computational Linguistics, 114–124. https://doi.org/10.18653/v1/2020.coling-main.10
- [20] Priyanka Kargupta, Tanay Komarlu, Susik Yoon, Xuan Wang, and Jiawei Han. 2023. MEGClass: Extremely Weakly Supervised Text Classification via Mutually-Enhancing Text Granularities. In Findings of the Association for Computational Linguistics: EMNLP 2023, Houda Bouamor, Juan Pino, and Kalika Bali (Eds.). Association for Computational Linguistics, Singapore, 10543–10558. https: //aclanthology.org/2023.findings-emnlp.708
- [21] Sangeet S Khemlani, Anthony M Harrison, and J Gregory Trafton. 2015. Episodes, events, and models. Frontiers in human neuroscience 9 (2015), 590.
- [22] J. Kleinberg. 2004. Bursty and Hierarchical Structure in Streams. Data Mining and Knowledge Discovery 7 (2004), 373–397.
- [23] Manling Li, Tengfei Ma, Mo Yu, Lingfei Wu, Tian Gao, Heng Ji, and Kathleen McKeown. 2021. Timeline Summarization based on Event Graph Compression via Time-Aware Optimal Transport. In Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics, Online and Punta Cana, Dominican Republic, 6443–6456. https: //doi.org/10.18653/v1/2021.emnlp-main.519
- [24] Manling Li, Qi Zeng, Ying Lin, Kyunghyun Cho, Heng Ji, Jonathan May, Nathanael Chambers, and Clare R. Voss. 2020. Connecting the Dots: Event Graph Schema Induction with Path Language Modeling. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020, Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu (Eds.). Association for Computational Linguistics, 684-695. https: //doi.org/10.18653/v1/2020.emnlp-main.50
- [25] Sha Li, Heng Ji, and Jiawei Han. 2021. Document-Level Event Argument Extraction by Conditional Generation. In Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021, Kristina Toutanova, Anna Rumshisky, Luke Zettlemoyer, Dilek Hakkani-Tür, Iz Beltagy, Steven Bethard, Ryan Cotterell, Tanmoy Chakraborty, and Yichao Zhou (Eds.). Association for Computational Linguistics, 894-908. https://doi.org/10.18653/y1/2021.naacl-main.69
- [26] Aristidis Likas, Nikos Vlassis, and Jakob J Verbeek. 2003. The global k-means clustering algorithm. Pattern recognition 36, 2 (2003), 451–461.
- [27] Xiao Liu, Zhunchen Luo, and Heyan Huang. 2018. Jointly Multiple Events Extraction via Attention-based Graph Information Aggregation. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics, Brussels, Belgium, 1247–1256. https://doi.org/10.18653/v1/D18-1156
- [28] Zhengzhong Liu, Chenyan Xiong, Teruko Mitamura, and Eduard H. Hovy. 2018. Automatic Event Salience Identification. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 -November 4, 2018, Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii (Eds.). Association for Computational Linguistics, 1226–1236. https://doi.org/10.18653/v1/d18-1154
- [29] Zheng Liu, Yu Zhang, Yimeng Li, and Chaomurilige. 2023. Key News Event Detection and Event Context Using Graphic Convolution, Clustering, and Summarizing Methods. Applied Sciences 13, 9 (2023), 5510.
- [30] Yaojie Lu, Hongyu Lin, Jin Xu, Xianpei Han, Jialong Tang, Annan Li, Le Sun, Meng Liao, and Shaoyi Chen. 2021. Text2Event: Controllable Sequence-to-Structure Generation for End-to-end Event Extraction. In Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers). Association for Computational Linguistics, Online, 2795–2806. https://doi.org/10.18653/v1/2021.acl-long.217
- [31] Sebastian Martschat and Katja Markert. 2018. A Temporally Sensitive Sub-modularity Framework for Timeline Summarization. In Proceedings of the 22nd Conference on Computational Natural Language Learning, Anna Korhonen and Ivan Titov (Eds.). Association for Computational Linguistics, Brussels, Belgium,

- 230-240. https://doi.org/10.18653/v1/K18-1023
- [32] Yu Meng, Jiaxin Huang, Guangyuan Wang, Zihan Wang, Chao Zhang, Yu Zhang, and Jiawei Han. 2020. Discriminative topic mining via category-name guided text embedding. In *Proceedings of The Web Conference 2020*. 2121–2132.
- [33] Fionn Murtagh and Pedro Contreras. 2012. Algorithms for hierarchical clustering: an overview. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery 2, 1 (2012), 86–97.
- [34] Kiem-Hieu Nguyen, Xavier Tannier, and Véronique Moriceau. 2014. Ranking multidocument event descriptions for building thematic timelines. In COLING 2014, the 25th International Conference on Computational Linguistic. 1208–1217.
- [35] Zheng Qi, Elior Sulem, Haoyu Wang, Xiaodong Yu, and Dan Roth. 2022. Capturing the Content of a Document through Complex Event Identification. In Proceedings of the 11th Joint Conference on Lexical and Computational Semantics. Association for Computational Linguistics, Seattle, Washington, 331–340. https://doi.org/10.18653/v1/2022.starsem-1.29
- [36] 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). Association for Computational Linguistics, Hong Kong, China, 3982–3992. https://doi.org/10.18653/v1/D19-1410
- [37] H. Sayyadi, Matthew F. Hurst, and A. Maykov. 2009. Event Detection and Tracking in Social Streams. In ICWSM.
- [38] H. Sayyadi and L. Raschid. 2013. A Graph Analytical Approach for Topic Detection. ACM Trans. Internet Techn. 13 (2013), 4:1–4:23.
- [39] Julius Steen and Katja Markert. 2019. Abstractive Timeline Summarization. In Proceedings of the 2nd Workshop on New Frontiers in Summarization. Association for Computational Linguistics, Hong Kong, China, 21–31. https://doi.org/10. 18653/v1/D19-5403
- [40] Haoyu Wang, Muhao Chen, Hongming Zhang, and Dan Roth. 2020. Joint Constrained Learning for Event-Event Relation Extraction. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020, Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu (Eds.). Association for Computational Linguistics, 696-706. https://doi.org/10.18653/v1/2020.emnlp-main.51
- [41] Zihan Wang, Dheeraj Mekala, and Jingbo Shang. 2021. X-Class: Text Classification with Extremely Weak Supervision. In Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Kristina Toutanova, Anna Rumshisky, Luke Zettlemoyer, Dilek Hakkani-Tur, Iz Beltagy, Steven Bethard, Ryan Cotterell, Tanmoy Chakraborty, and Yichao Zhou (Eds.). Association for Computational Linguistics, Online, 3043–3053. https://doi.org/10.18653/v1/2021.naacl-main.242
- [42] Noah Weber, Leena Shekhar, Niranjan Balasubramanian, and Nate Chambers. 2018. Hierarchical Quantized Representations for Script Generation. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018, Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii (Eds.). Association for Computational Linguistics, 3783–3792. https://doi.org/10.18653/v1/d18-1413
- [43] David Wilmot and Frank Keller. 2021. Memory and Knowledge Augmented Language Models for Inferring Salience in Long-Form Stories. In Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021, Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih (Eds.). Association for Computational Linguistics, 851–865. https://aclanthology.org/ 2021.emnlp-main.65
- [44] Susik Yoon, Dongha Lee, Yunyi Zhang, and Jiawei Han. 2023. Unsupervised story discovery from continuous news streams via scalable thematic embedding. In Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval. 802–811.
- [45] Jingyi You, Dongyuan Li, Hidetaka Kamigaito, Kotaro Funakoshi, and Manabu Okumura. 2022. Joint Learning-based Heterogeneous Graph Attention Network for Timeline Summarization. In Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. 4091–4104.
- [46] Hongming Zhang, Muhao Chen, Haoyu Wang, Yangqiu Song, and Dan Roth. 2020. Analogous Process Structure Induction for Sub-event Sequence Prediction. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020, Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu (Eds.). Association for Computational Linguistics, 1541–1550. https://doi.org/10.18653/v1/2020.emnlp-main.119
- [47] Xiyang Zhang, Muhao Chen, and Jonathan May. 2021. Salience-Aware Event Chain Modeling for Narrative Understanding. In Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021, Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih (Eds.). Association for Computational Linguistics, 1418–1428. https://aclanthology.org/2021.emnlpmain.107
- [48] Yunyi Zhang, Fang Guo, Jiaming Shen, and Jiawei Han. 2022. Unsupervised Key Event Detection from Massive Text Corpora. In Proceedings of the 28th ACM

- SIGKDD Conference on Knowledge Discovery and Data Mining (Washington DC, USA) (KDD '22). Association for Computing Machinery, New York, NY, USA, 2535–2544. https://doi.org/10.1145/3534678.3539395
- [49] Wayne Xin Zhao, Rishan Chen, Kai Fan, Hongfei Yan, and Xiaoming Li. 2012. A novel burst-based text representation model for scalable event detection. In Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers). 43–47.
- [50] Deyu Zhou, Haiyang Xu, and Yulan He. 2015. An Unsupervised Bayesian Modelling Approach for Storyline Detection on News Articles. In EMNLP.

A Ethics Statement

Based on our current methodology and results, we do not expect any significant ethical concerns, given that subtasks like episode detection within the news event extraction and analysis is a standard problem domain across data mining applications. Furthermore, having the method rely on zeor supervision helps as a barrier to any user-inputted biases. However, one minor factor to take into account is any hidden biases that exist within the large language models used as a result of any potentially biased data that they were trained on. We used these pre-trained language models for refining the fluency of the detected episode clusters and did not observe any concerning results, as it is a low-risk consideration for the domains that we studied.

B Additional Details for Episode Partitioning

We note that for determining semantic similarity between the terms of two segments, we use both (1) the average cosine similarity between all unordered pairs of terms between segment (i-1) and (i), and (2) the cosine similarity between the average of static term representations in (i-1) and the average of static term representations in (i). Furthermore, we filter out any non-salient segments before episode partitioning to avoid any influence of noisy segments (e.g., journalist's analysis, summary statements, historical comparisons, and other generic noise) on the quality of our episode fragments.

Finally, following [20, 41], we take the harmonic mean of all pairwise discriminative co-occurrence scores instead of a simple average. This allows us to prioritize the more salient *and* discriminative terms when determining the episode partitions. For instance, if "protesters" consistently occurs throughout the majority of episodes and thus has a low average discriminative co-occurrence, then it is not as informative for episode partitioning.

C Claude-2 Prompt & Example for Candidate Episode Estimation

Prompt. We use the following prompt for estimating fluent candidate episodes from our input episode fragment clusters. We denote k as the number of episode fragment clusters outputted after clustering the top article episode fragments in Section 4.3.

Task: You are a key news event analyzer that is aiming to detect episodes (a representative subevent that reflects a critical sequence of actions performed by a subject at a certain and/or location) based on text segments from different news articles. Given the above groups of article segments, predict at least 2 and at most {k} potential episodes of the key event. Some groups may fall under the same episode. Output your answer inside the tags <answer></answer> as a JSON object where each item is also a JSON with the key

"title" with the value containing the [subject, action, object, time, location] of the episode, a key "keywords" with the string value being a list of 5-10 associated keywords unique to that specific episode, and a final key "example_sentences" with a value being a list of 2-5 extracted sentences from the input segment groups. Feel free to output less than {k} episodes if you feel that any are redundant (could fit under an existing candidate episode). The title, keywords, and example sentences of a predicted episode should not be able to be placed under another different predicted episode.

Example. Below, we provide an example of EpiMine's candidate episode estimation step (Section 4.3). Specifically, the LLM identifies the core attributes (subject, action, object, time, location) of each unique cluster, relevant keywords, and top extracted text segments given the input clusters:

'title': ['Protesters', 'storm and vandalize', 'Hong Kong's Legislative Council building', 'July 1, 2019', 'Legislative Council building in Admiralty, Hong Kong'] 'keywords': ['vandalism, graffiti, violence, escalation, ransacking']

'example_sentences': ['Hundreds of anti-extradition bill protesters finally broke into the legislature after many hours of attacking the public entrance and ransacked the building, including displaying the colonial Hong Kong flag in the chamber.', 'Slogans on the wall read: "Murderous regime", and "There are no rioters only a tyrannical regime."]

D Evaluation Metrics

Following a recent work on key event detection [48], we adapt the k-prec, k-recall, and k-F1 to quantitatively evaluate the episode detection results. We use these metrics to evaluate how the model output matches the ground truth episodes using the top-k segments within each detected episode. Formally, suppose there are N ground truth episodes $\mathcal{G} = \{G_1, G_2, \ldots, G_N\}$, each of which is a set of text segments related to its corresponding episode. $\mathcal{E} = \{E_1, E_2, \ldots, E_K\}$ are the model predicted episodes, each of which is a ranked list of segments, and $E_{j,k}$ means the top-k segments within E_j . Then, the k-metrics are defined as follows:

$$\begin{aligned} \text{k-prec} &= \frac{\sum_{G_i \in \mathcal{G}} \mathbbm{1}(\exists E_j \in \mathcal{E}, E_{j,k} \cap G_i \geq \frac{k}{2})}{\sum_{E_j \in \mathcal{E}} \mathbbm{1}(|E_j| \geq k)} \\ \text{k-recall} &= \frac{\sum_{G_i \in \mathcal{G}} \mathbbm{1}(\exists E_j \in \mathcal{E}, E_{j,k} \cap G_i \geq \frac{k}{2})}{N} \\ \text{k-F1} &= \frac{2 \cdot \text{k-prec} \cdot \text{k-recall}}{\text{k-prec} + \text{k-recall}} \end{aligned}$$

E Key Event Corpus Retrieval

Given that our task is novel and no large-scale key event-specific news corpus is available for this task where the key events are guaranteed to contain distinguishable episodes, we briefly discuss how we collect the input corpus from online news data, even if it is out of scope for our fundamental methodology. Given our set of key events (as listed in Section 5.1), we first scrape the external reference list from their corresponding Wikipedia page and select the news articles that have been published within two months of given key event's start date (e.g., all articles selected for "January 6 2021 Capitol Attack" would have been published between November 6-March 6). This is important as we want to prioritize the news articles which focus on describing the episodes of the key event and their corresponding aspects as opposed to primarily opinions or analyses. This allows us to motivate our task as one critical for currently evolving key events which required a more fine-grained episodic timeline. Furthermore, it is consequently unlikely for a single article to cover all of the episodes and exclusively episodes under a key event. Despite this being more challenging, it is acceptable as the goal of our task is to extract only the key event-related episodes, which must be substantiated by multiple documents in either case. We list all of our selective themes and their corresponding key events included in the dataset that we construct:

- Terrorism and attacks: 2021 Atlanta spa shootings; 2014 Montgomery County Shootings; 2021 Indianapolis FedEx shooting; 2022 Cincinnati FBI field office attack; 2019 Jersey City shooting; 2019 Naval Air Station Pensacola shooting; 2022 Greenwood Park Mall shooting, 2018 Capital Gazette shooting; 2021 Collierville Kroger shooting; 2019 Kyoto Animation arson attack
- Natural disasters: 2023 Tornado outbreak sequence; 2023 Hawaii Wildfires; 2021 Western Kentucky tornado; 2017 Mocoa landslide; 2010 Haiti earthquake; 2021 Henan floods; 2019 Nyonoksa radiation accident; 2022 North American winter storm; 2011 Fukushima nuclear accident
- Political Events: 2020 Kyrgyz Revolution, 2019 Storming of the Hong Kong Legislative Council Complex; 2019 Siege of the Hong Kong Polytechnic University; 2017 Zimbabwean coup; 2018 Italian government formation; 2021 January 6 United States Capitol attack; 2018 Thai Cave Rescue Operation; 2018 Armenian Revolution; 2017 Lebanon–Saudi Arabia dispute; 2013 Tunisian political crisis

F Claude-2 Prompt for Dataset Annotation

We automatically annotate our dataset using Claude-2.1 using the prompt below (before an additional human-verification stage):

You are a news event analyzer that labels text segments of a news article with their matching event episode description. I will give you several text segments, and several episodes of a key event in tuples. We define an episode as the following: an episode is a set of thematically coherent text segments discussing a particular set of core entities performing actions for or towards an object(s) at a certain time and/or location during a real-world key event. The entities, actions, objects, time, and location can all be considered aspects of an episode.

[one-shot demonstration & format specification] Please help classify the text segments under different episodes (the output value for each segment should be an integer key of each episode). If you think a text

segment cannot be used to describe any episodes, please use "X" in the output to indicate the lack of an episode tuple number for that segment. If a text segment is very general, does not describe the key event at hand, or can be matched to multiple episodes, then please use a "M"

in the output to indicate the multiple episode mapping for that segment. There should be a value assigned to each of the len(segments) segments (segment_0, \dots , segment_len(segments)-1).