Unsupervised Event Chain Mining from Multiple Documents

Yizhu Jiao
UIUC
Illinois, USA
yizhuj2@illinois.edu

Ming Zhong
UIUC
Illinois, USA
mingz5@illinois.edu

Jiaming Shen
Google Research
New York, USA
jmshen@google.com

Yunyi Zhang
UIUC
Illinois, USA
yzhan238@illinois.edu

Chao Zhang
Georgia Institute of Technology
Atlanta, USA
chaozhang@gatech.edu

Jiawei Han
UIUC
Illinois, USA
hanj@illinois.edu

ABSTRACT

Massive and fast-evolving news articles keep emerging on the web. To effectively summarize and provide concise insights into real-world events, we propose a new event knowledge extraction task *Event Chain Mining* in this paper. Given multiple documents about a super event, it aims to mine a series of salient events in temporal order. For example, the event chain of super event *Mexico Earthquake in 2017* is {*earthquake hit Mexico*, *destroy houses*, *kill people*, *block roads*}. This task can help readers capture the gist of texts quickly, thereby improving reading efficiency and deepening text comprehension. To address this task, we regard an event as a cluster of different mentions of similar meanings. In this way, we can identify the different expressions of events, enrich their semantic knowledge and replenish relation information among them. Taking events as the basic unit, we present a novel unsupervised framework, EMiner. Specifically, we extract event mentions from texts and merge them with similar meanings into a cluster as a single event. By jointly incorporating both content and commonsense, essential events are then selected and arranged chronologically to form an event chain. Meanwhile, we annotate a multi-document benchmark to build a comprehensive testbed for the proposed task. Extensive experiments are conducted to verify the effectiveness of EMiner in terms of both automatic and human evaluations.

CCS CONCEPTS

• **Information systems** → **Data mining**; • **Computing methodologies** → **Natural language processing**.

KEYWORDS

event chain, event extraction, text mining, unsupervised learning

ACM Reference Format:
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 (WWW '23), April 30–May 04, 2023, Austin, TX, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3543507.3583295

WWW '23, April 30–May 04, 2023, Austin, TX, USA

ACM ISBN 978-1-4503-9416-1/23/04.
https://doi.org/10.1145/3543507.3583295

1 INTRODUCTION

In the information age, a plethora of information resources is at the fingertips of every user. Faced with a variety of complex and lengthy news on the web, how to quickly understand their core idea has become a critical problem with increasing concerns. Generally, massive unstructured news articles can be regarded as the chains of salient events arranged in order [1, 11]. Therefore, extracting event chain knowledge from them is a crucial step in text understanding. Recently, various event-centric tasks have gained significant interest, such as event relation extraction [2, 13, 37], salient event identification [25, 39], and event process understanding [8, 43]. However, most of these studies highly rely on expert annotations, which are expensive and time-consuming. Subsequent research [20, 38] attempts to alleviate this issue under an unsupervised setting. They extract event schema from large corpus as prior knowledge to assist downstream tasks, such as story generation [40], question answering [34], text summarization [49], and reading comprehension [46]. However, explorations on how to mine event chains from large amounts of unstructured text remain lacking. Such a task can provide a brief summary and help readers to capture the skeleton of texts quickly.

To this end, we propose a new task of knowledge extraction, **Event Chain Mining**. It aims to mine a series of salient events in a temporal order, which can serve as a concise highlight of texts. Specifically, given a super event[1], multiple documents usually report it from different perspectives. Moreover, these reports usually share the most salient events among their texts. For example, Figure 1 shows three documents on a super event, Mexico Earthquake in 2017. Most of them mention four essential events in the earthquake, including *earthquake hit Mexico*, *damage houses*, *kill people*, and *block roads*. These events can provide a sequential highlight of how this disaster occurred. With such a chain of salient events, readers can effectively grasp the whole picture of the news, thus improving reading efficiency and deepening reading comprehension. This observation leads to the **Event Chain Mining** problem, which poses the following challenges:

(1) *variability of events*. An event can be expressed in different descriptions. For example, in Figure 1, *earthquake rocked southern Mexico* and *earthquake hit southern Mexico* are two different mentions, but they describe similar meanings in Mexico Earthquake.

[1]A super event (or a key event in recent studies [47]) is a more coarse-grained event by itself. We use this term to distinguish it from events.

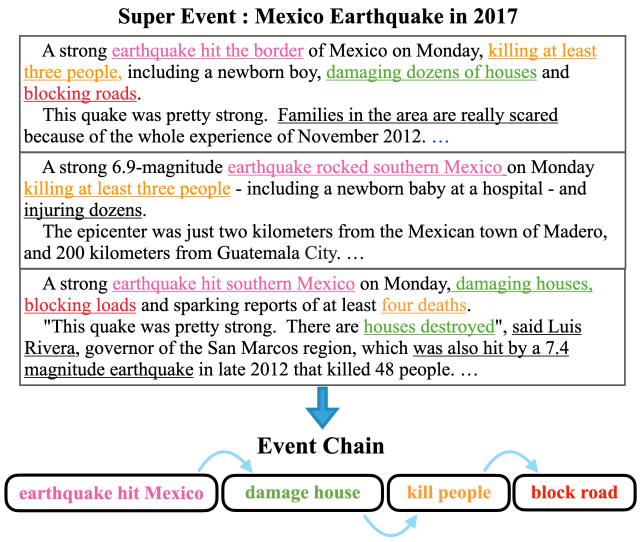

Figure 1: Example of Event Chain Mining task. The underlined text indicates an event mention, while the same colored texts represent salient mentions with similar meanings. We extract salient events and arrange them in chronological order to form an event chain.

The desired event chain should not include both events, as this brings in information redundancy.

(2) *salience inequality of events.* Not all events in the news are equally important. Some of the events could be too general and contain little information, such as *say it*. Others could be too specific, not closely tied to the main points, such as *The state has 3.44 million people*. These events should be filtered to ensure that the final event chain includes only the central point of the news.

(3) *ambiguity of event relationship.* Existing methods heavily rely on local contexts to determine the temporal relationships between different events. However, due to different narrative styles, clear clues are not always provided in the news. Especially when events scatter separately in a long text or multiple documents, it is more challenging to capture their long-distance relations.

To address these challenges, we regard an event as a cluster of mentions with similar meanings. In this way, there are three significant benefits. First, it naturally helps to address the first challenge – different expressions of the same event. Second, it enhances event semantics by including multiple related mentions, which makes it easier to deepen event understanding and recognize salient events. Third, it enriches the order of information between events. By introducing multiple mentions, more clues about event relations can be obtained from the contexts. As a result, event ordering can be more convincing with the support from the majority of mentions.

On top of that, we propose a novel unsupervised event chain mining framework, EMiner, which contains four major steps. Concretely, given a set of texts on the same super event, we first decompose them into multiple event mentions. We elaborate frequently-occurring syntactic patterns and extract all possible event mentions. Then, event mentions of similar meanings are merged into clusters as distinct events. We formulate the event mention merging problem as an online text stream clustering task without requiring a fixed number of clusters. Next, we measure the salience of events to select important ones according to event frequency counting. Finally, we propose a language model empowered event ordering method, which jointly incorporates both of content and commonsense to arrange the salient events in a sequential chain. It not only leverages explicit clues from the content description in multiple documents, but also exploits commonsense knowledge in the pre-trained generation models by re-framing the ordering problem as a generation task. By combining two perspectives together, the final chronological chain of events can be produced.

To build a testbed for the proposed task, we re-annotate an existing multi-document dataset [28] to develop a new benchmark for evaluating event chain mining systems. For multiple documents on a super event, we manually annotate salient events as a brief summary. In addition, we design a comprehensive evaluation system, which can evaluate the model from multiple aspects, such as event semantics and sequential orders. We conduct extensive experiments and the results verify that EMiner can produce a chain of salient events to guide people to understand texts.

The major contributions of this paper are summarized as follows:

(1) We propose a new task, Event Chain Mining, to summarize and provide concise insights to real-world events. It assists people in quickly capturing the central points of a large amount of unstructured textual data on the web, thus improving reading efficiency and deepening comprehension.

(2) We present a novel framework EMiner to automatically extract an event chain in an unsupervised setting. According to the cluster completeness, term occurrence, and semantic similarity, our model extracts and merges essential events from multiple documents. EMiner can also reorder salient events by incorporating contents and commonsense knowledge to form the final chain.

(3) To facilitate research in this direction, we establish a comprehensive testbed with a human-annotated benchmark and evaluation metrics. Experimentally, our proposed EMiner outperforms all baselines in terms of both automatic and human evaluation.

2 PROBLEM FORMULATION

In this section, we first introduce some important concepts and then present our task definition. An **event mention**, is a phrase that contains multiple words $(w_1, w_2, \ldots, w_z)$, where z is the number of words, and $w_1, w_2, \ldots, w_m$ are all in the vocabulary. A pair of words (w_i, w_j) in an event mention m may follow a specific syntactic relation. An **event** is a cluster of event mentions in similar meanings $\{m_1, m_2, \ldots, m_x\}$, where x is the number of mentions. A **super event** refers to a more coarse-grained event described by multiple documents. A **salient event** is one that provides sufficient and important information about the super event. An **event chain** stands for a series of salient events arranged in the order of occurrence.

Task Definition. Given a set of documents $\mathcal{D}$ on a super event, our task of *event chain mining* is to produce a sequence of salient events $(e_1, e_2, \ldots, e_n)$, which can give a brief summary of the texts. n is the number of events.

Intuitively, humans are accustomed to understanding texts in a sequential order instead of modeling a relation graph. Thus, our task aims to mine an event chain. However, notably, our proposed method can also extract the partial order relationship among events (introduced in the next section) to form a relation graph. As to complex event relation graph modeling, we leave it as future work.

3 FRAMEWORK

The proposed framework, EMiner, outlines the event chain mining task in four major steps: (1) event mention extraction, (2) event mention merging, (3) salient event selection, and (4) salient event ordering. The architecture is illustrated in Figure 2.

3.1 Event Mention Extraction

We adopt a lightweight method to extract event mentions in the texts without relying on manually-labeled training data. It aims to decompose texts into multiple event mentions. For example, there is a sentence about Mexico Earthquake in 2017: *A strong earthquake shook Mexico on Monday, killing at least three people and damaging dozens of buildings*. It mainly contains three event mentions, *earthquake shook Mexico*, *killing people*, and *damaging buildings*. To handle the complex structure of event mentions, we elaborate frequently-occurring syntactic patterns inspired by Zhang et al. [44]. By pattern matching, all possible event mentions are extracted from texts based on sentence dependency tree structures.

Specifically, given a sentence, we first use a dependency parser[2] to obtain its dependency parse tree. As the centers of event mentions are verbs, we extract all verbs from each sentence. To ensure that all the extracted event mentions are semantically complete and frequently occurring without being too complicated, we elaborate 76 syntactic patterns based on those in Zhang et al. [44] (some examples are in Table 5). For each verb, we check its dependent words and their dependency labels. If they match one of the syntactic patterns, we extract the corresponding words as an event mention. We give priority to more complex patterns to make event mention contain more details. That is, once a pattern is exactly matched, we will no longer consider the remaining simpler ones. By such a strategy, all possible event mentions can be extracted from texts. Notably, we treat the sentences with clauses equally, so this method can decompose long sentences completely into event mentions.

3.2 Event Mention Merging

In this step, we merge similar event mentions into the same cluster, which is indispensable for the framework. To improve generalization capability for different topics or texts, the number of clusters should not be fixed. Thus, we formulate the event mention merging as a short text stream clustering task [7, 18, 41]. Specifically, event mentions are regarded as a stream and each of them is processed incrementally. In each process, for a mention, we decide whether to add it to an existing cluster or create a new cluster. Then, we update the corresponding cluster to prepare for subsequent events.

For example, there are three mentions, *kill people*, *damage houses*, and *destroy homes*. Initially, we create a new cluster for the first mention *kill people*. Then, when *damage houses* comes, there is an existing cluster {*kill people*}. Since this mention is not related to the cluster, we still create a new cluster for it. Thus, there exist two clusters now, {*kill people*} and {*damage houses*}. Later, for the third mention, we compare the probability of it joining these two clusters and creating a new one. This mention was decided to be grouped into the second cluster. Finally, we obtain two clusters, {*kill people*} and {*damage houses*, *destroy homes*}.

Such an evolutionary clustering automatically increases the number of clusters with event mentions. Nonetheless, it comes to three questions: (1) how to represent and update a cluster; (2) how to estimate the probabilities of a mention belonging to existing clusters and a new cluster; and (3) how to avoid the mention order affecting the clustering process. We will solve these problems in turn.

3.2.1 Cluster Feature. We first represent an event with the cluster feature (CF) vector, which essentially is a cluster with its event mentions. The CF vector of an event is defined as a tuple $\left\{\vec{f_e}, n_e, x_e\right\}$, where $\vec{f_e}$ contains a list of mention frequencies in event e; n_e is the number of mentions in event e; and x_e is the number of words in event e. The cluster feature vector presents desirable addition and deletion properties.

- Addition Property. A mention m can be efficiently added to cluster e by updating its CF vector as follows.

$$\begin{aligned} f_e^w &= f_e^w + N_m^w, \quad \forall w \in m, \\ n_e &= n_e + 1, \\ x_e &= x_e + N_m. \end{aligned}$$

- Deletion Property. A mention m can be efficiently deleted from cluster e by updating its CF vector as follows.

$$\begin{aligned} f_e^w &= f_e^w - N_m^w, \quad \forall w \in m \\ n_e &= n_e - 1, \\ x_e &= x_e - N_m, \end{aligned}$$

where N_m^w and N_m are the number of occurrences of word w in mention m and the total number of words in mention m, respectively, and $N_m = \sum_{w \in m} N_m^w$. Besides, f_e^w is the number of occurrences of the word w in cluster e. With the addition and deletion properties, we can update the CF vectors when including or excluding a mention.

3.2.2 Model Formulation. We assume the mentions are generated by the Dirichlet Process Multinational Mixture (DPMM) model [42]. Its generative process is as follows.

$$\begin{aligned} \theta \mid \gamma &\sim GEM(1, \gamma), \\ e \mid \theta &\sim \text{Multi}(\theta), \\ \mathcal{N}_k \mid \beta &\sim \text{Dir}(\beta) \quad k = 1, \ldots, \infty, \\ m \mid e, \{\mathcal{N}_k\}_{k=1}^{\infty} &\sim p\left(m \mid \mathcal{N}_e\right). \end{aligned}$$

Here, when generating mention m, the model first selects a mixture component (cluster e) according to the mixture weights. Then mention m is generated by the selected mixture component (cluster e) from distribution $p\left(m \mid \mathcal{N}_e\right)$. θ is generated by a stick-breaking construction [36]. $\mathcal{N}_k$ are also generated by a Dirichlet distribution [36]. γ and β are two hyper-parameters.

Following Kumar et al. [18], the probability of mention m generated by cluster e is:

[2] https://nlp.stanford.edu/software/stanford-dependencies.html

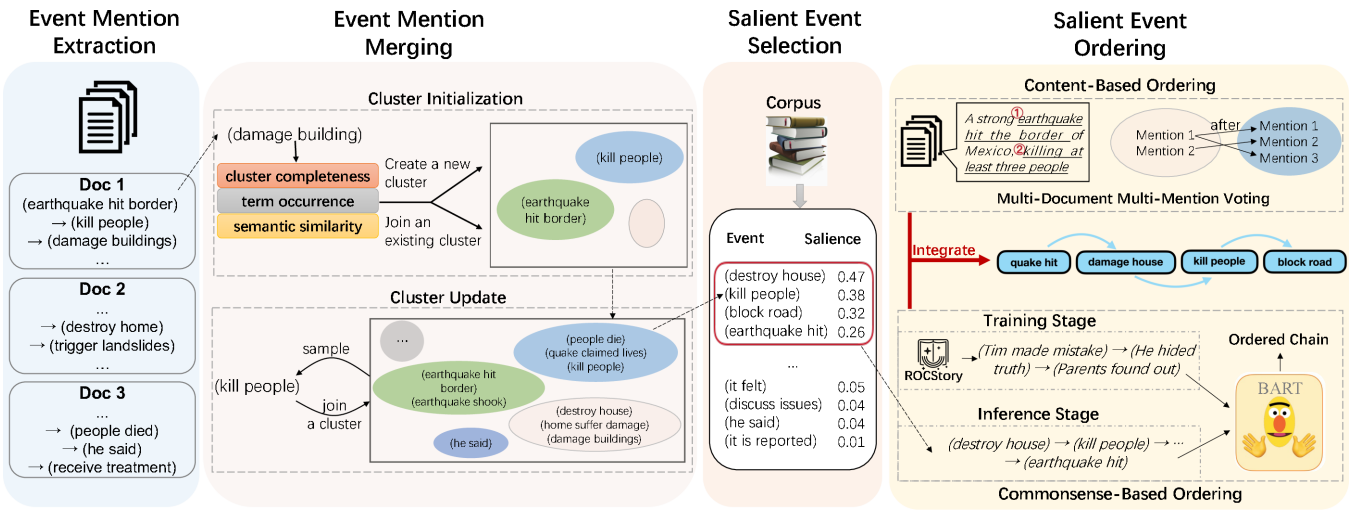

Figure 2: Architecture of EMiner. Given multiple documents about a super event, we decompose texts into event mentions and merge those with similar meanings into the same clusters. Then essential events are then selected and arranged chronologically to form an event chain. For convenience, each event in the last two steps is represented by a representative mention.

$$
\begin{aligned}
& p\left(e_m = e \mid e, m, \alpha, \beta\right) \\
& \propto \left(\frac{n_e}{D - 1 + \alpha D}\right) \cdot \\
& \left(\frac{\prod_{w \in m} \prod_{j=1}^{f_m^w} f_e^w + \beta + j - 1}{\prod_{i=1}^{x_m} x_e + V\beta + i - 1}\right) \cdot \\
& \left(1 + \frac{1}{n_e} \sum_{i=1}^{n_e} Sim\left(m, m_i^e\right)\right).
\end{aligned} \tag{1}
$$

In the above equation, the first term represents the completeness of the cluster. A new mention gives priority to clusters with more mentions. Thus, although the number of clusters can be unlimited, only a limited number of clusters will be created. Here, n_e is the number of mentions contained by the cluster e, D is the number of current mentions in the existing clusters, and α is the concentration parameter of the model.

The second term defines the term occurrence between a cluster and a mention. It is based on multinomial distribution with pseudo weight of words β. x_m and f_m^w represent total number of words and term frequency of word w in mention m, respectively. The symbol f_e^w is the term frequency of the word w in the cluster e. The current vocabulary size of the model is represented by V, and x_e is the number of words in the cluster e.

The third term reflects the semantic similarity between a cluster and a mention. For a mention m, we calculate the average semantic similarity between each mention m_i^e in the cluster e. Here, given a pair of mentions, $Sim(\cdot)$ is the function for their semantic similarity scores. Following Zhang et al. [45], we first use a pre-trained language model [9] to obtain contextual representations of the two mentions. Then, their similarity is computed as a sum of cosine similarities between their word embeddings.

So far, we have defined the probability of a mention choosing an existing cluster. Then we have to consider the probability of a mention to create a new cluster. By following the DPMM, the probability of creating a new cluster is as follows.

$$
\begin{aligned}
& p\left(e_m = K + 1 \mid \vec{e}, \vec{m}, \alpha, \beta\right) \\
& \propto \frac{\alpha D}{D - 1 + \alpha D} \cdot \frac{\prod_{w \in m} \prod_{j=1}^{f_m^w} (\beta + j - 1)}{\prod_{i=1}^{x_m} (V\beta + i - 1)},
\end{aligned} \tag{2}
$$

where K is the number of the existing clusters; αD denotes the pseudo number of clusters related mentions, and β is the pseudo term frequency of each word (exist in mention) of the new cluster.

3.2.3 Merging Process. The merging method allows the processing of each event mention incrementally and updating the model accordingly. Initially, it creates a new cluster for the first mention. We initialize the cluster feature of this new cluster with the first mention. Later, for each event mention, it either belongs to an existing cluster or generates a new cluster. It depends on the corresponding probability computed with Eqs. (1) and (2). We choose the cluster with the highest probability. The CF vector is updated according to the addition property. In this way, we can detect new clusters more naturally without a fixed number of clusters. Based on this process, we can obtain the initial clustering result.

Since all the mentions are processed one by one, their order may affect the clustering results. Therefore, to improve the robustness of the model, we then update the clustering results. For each mention, we delete it from its current cluster with the deletion property. Then, we reassign it to an existing cluster or create a new cluster for it. According to Eqs. (1) and (2), the choice with the highest probability will be made.

3.3 Salient Event Selection

In this step, we filter too general or too specific events, thereby selecting salient ones. Since each event is involved with multiple event mentions, it helps to enhance event semantic understanding.

For example, *Jessica said on conference* is a general event mention with specific arguments. Existing frequency-based methods [35, 46] might fail to handle it. However, with the help of typical general mentions in the same cluster, such as *it says*, the selection algorithm can filter it more easily.

Based on this observation, we define the salience score for an event cluster. All mentions of this event are taken into consideration. An event has a high value if its mentions occur frequently in the texts and rarely exist in a general-domain background corpus. Computationally, given an event e, we define its salience as follows:

$$\begin{aligned}\text{Salience}(e) &= \frac{1}{N}\sum_{i=1}^{N}\text{Salience}(m_i)\\ &= \frac{1}{N}\sum_{i=1}^{N}(1+\log(\text{freq}(m_i))^2)\log(\frac{N_{bs}}{bsf(m_i)}),\end{aligned}$$

where m_i is a mention of event e. N is the number of mentions. Besides, $freq(m_i)$ denotes the frequency of mention m_i, N_{bs} is the number of background texts, and $bsf(w)$ refers to the background text frequency of mention m_i. For each event, its mention with the highest salience score is selected as the representative mention. Since the representative mention is salient and informative, we will use it to represent an event for ordering in the next step.

3.4 Salient Event Ordering

The typical strategy for event temporal ordering is to formulate it as a classification problem. Specifically, given a short paragraph and two event mentions involved in it, these methods classify their relationship as "before" or "after" [14, 30, 50]. However, such paradigms hit bottlenecks in our task, as they rely heavily on the local context to make decisions. They cannot handle the events from different documents, which is likely to happen in the case of the multi-document scenario. Even if two events come from the same document, existing work fails to capture long-distance relationships in long texts. On top of that, recent studies only deal with event pairs and lack exploration of the overall sequence of multiple events.

To address these aforementioned challenges, we propose a novel event ordering method that incorporates both contents and commonsense to jointly determine the order of an event sequence. On the one hand, the contents of multiple documents can provide explicit clues about the event relationships. We adopt a straightforward strategy to arrange the event pairs based on the content description. On the other hand, commonsense knowledge can also assist this task when context clues are insufficient, such as generally sentencing often occurs after a crime. In this case, we re-frame the event ordering problem as a generation task, which exploits commonsense knowledge in pre-trained language models to help arrange events into a chain. These two perspectives can lead to two different sorting results. By combining them, the final chronological chain of events can be produced.

3.4.1 Content-Based Ordering. To leverage the content of source documents, we can compare the relative order in which these events are described in the text. However, this direct approach brings two obstacles. First, considering several documents may use flashbacks, their narratives cannot directly reflect the temporal relationships of events. Second, notably, the basic unit of order is an event, which is a cluster of multiple mentions. Therefore, the relations of different mentions in the different documents might come into conflict. To handle these problems, we introduce a multi-document multi-mention voting mechanism, which makes the decision that is supported by the majority of mentions in most documents. In this way, multiple documents can reduce the distraction caused by the differences in narrative styles, and multiple mentions can provide richer information for decision-making. Concretely, the ordering score of an event e is defined as:

$$\text{ContentScore}(e) = \frac{1}{N_d}\sum_{i=1}^{N_d}\min_{m\in e}\text{index}(d_i, m),$$

where N_d is the number of documents, d_i represents the i-th document, m is a mention for event e, and $\text{index}(d_i, m)$ refers to the index of mention m in the event sequences extracted from the text d_i. By comparing the order of each event, we can rank them in the content-based order and produce an event sequence.

3.4.2 Commonsense-Based Ordering. Pre-trained language models have demonstrated the ability to generate coherent passages and reason with commonsense [5, 33], thus we further integrate it to discover implicit event ordering. Specifically, we re-frame the event ordering problem as a generation task, and let the pre-trained language model outputs an event chain according to the general order of event occurrence.

Formally, given an event sequence $(e_1, e_2, e_3, ..., e_n)$ containing n events, the model is required to generate a new permutation of this sequence $(e'_1, e'_2, e'_3, ..., e'_n)$, which is more consistent with the sequence order that might occur in the real world. To simplify the format of the inputs and outputs, we use representative mentions of events, rather than entire clusters of events. For example, for a sequence of events: {*criminal was caught*} ⇒ {*criminal robbed a bank*} ⇒ {*criminal was put into jail*}, the pre-trained language model should realize that the arrest should not occur before the robbery, and consequently adjust the sequence of events with the following output: {*criminal robbed a bank*} ⇒ {*criminal was caught*} ⇒ {*criminal was put into jail*}.

We employ BART [19] as our generation model because it includes the task of restoring randomly ordered sentences to the normal order during pre-training, thereby gaining a better commonsense of the order of the events. To adapt BART to our defined generation task, we utilize a story dataset, ROCStory [29] for the training by extracting the event mentions as the distant supervision. After training on these synthetic data, BART can output an adjusted event sequence based on commonsense.

3.4.3 Overall Ordering. For a salient event sequence, both content-based and commonsense-based ordering methods can produce a well-ordered chain, which should be integrated together for the final decisions. Therefore, we propose to calculate the overall ordering score of an event based on its ranking in both chains. For an event e_i, $\text{Rank}_{\text{Content}}(e_i)$ and $\text{Rank}_{\text{Common}}(e_i)$ refer to its ranking in the above two chains, respectively. The final ordering score of e_i is:

$$\text{Order}(e_i) = \lambda\cdot\text{Rank}_{\text{Content}}(e_i) + (1-\lambda)\cdot\text{Rank}_{\text{Common}}(e_i),$$

Table 1: Dataset Statistics.

Domain	#SuperEvent	#Event	#Doc	#Word/Doc
Sport	19	101	483	458.3
Politics	22	129	539	569.1
Economics	32	171	711	365.5
Social Issues	37	190	784	511.6
Overall	100	591	2517	472.4

where λ is a hyperparameter. The smaller the ordering score $\text{Order}(e_i)$, the earlier the event e_i occurred. By rearranging the set of events by $\text{Order}(e_i)$, we can acquire the final salient event chain.

4 EXPERIMENTS

In this section, we conduct both automatic and human evaluations to show EMiner can mine meaningful event chains from unstructured texts, which can assist people to acquire information quickly.

4.1 Dataset

We re-annotate an existing multi-document dataset [28] to develop a new benchmark. This benchmark involves 100 super events and 2517 articles. These super events cover four popular domains in news reports, including sport, politics, economics, and social issues, to broaden the domain coverage For each super event, there are 25 articles describing it on average. The average number of words in an article is 472. We manually annotate 5-10 salient events as a brief summary. For the sake of convenience, each event is described with one mention. Each mention includes less than 10 words to ensure brevity. More dataset statistics are shown in Table 1.

Our annotation team consists of 3 graduate students majoring in data mining. For each super event, the annotators are required to read all the related articles and write a chain of events. Annotated events should be mentioned in most of the related articles. In addition, the whole event chain should make readers understand the main plot of this super event without original texts. Each annotated chain is required to be reviewed by another annotator, after which they discuss and revise until reaching an agreement.

4.2 Baselines

Considering the novelty of the event chain mining task, there is no existing approach to directly solve this task. Therefore, we apply several related studies to our scenario, including ASER [44], ODEE [23], CEE-IEA [16], SalienceAwareModel [46]. In addition, we present a randomly produced event chain as the lower bound for this task. We also introduce LEAD as a strong baseline, which extracts all events from the first three sentences of the texts as a chain. For the ablation study, we remove each component or replace it with related methods as the baselines, including ASER [44] for mention extraction, HDBSCAN [27] and OSDM [18] for mention merging, ETDisc for event selection, and SYMTIME [50] for event ordering (More details in Appendix A.1).

4.3 Evaluation Metrics

We build a comprehensive evaluation system, which evaluates the quality of the produced event chains from multiple perspectives. Motivated by Lin [22], we propose three kinds of event-based ROUGE F1 scores, including ERouge-1, ERouge-2, and ERouge-L. Specifically, similar to ROUGE, we evaluate how much percentage of events, event pairs, and the longest common event subsequence in the induced chains are covered by human-provided references. Since each event contains multiple mentions, we measure the overlap of all mention pairs in two mentions, and then take the average as the similarity of two events. More details can be found in Appendix A.2.

Following Zhang et al. [43], we provide two overlap standards of two event mentions to better understand the mining quality, "String Match" and "Hypernym Allowed". The first standard requires all words in the produced mention to be the same as the referent mention. This setting is rather strict. The second standard allows the hypernyms of words in mentions to relax the restrictions on the comparison. For example, two event mentions, *damage houses* and *damage buildings*, are counted as a match. This setting helps check if our framework selects relevant events.

4.4 Implementation details

Details about our implementation are introduced in Appendix A.3.

4.5 Automatic Evaluation

Following Glavas et al. [12] and Zhang et al. [43], we provide two settings to make the evaluation comprehensive: (1) Basic: evaluate events based on only verbs; (2) Advanced: evaluate events based on all words. From Table 2, in these two settings, we can see the improvement of ERouge scores when adopting our proposed framework to mine event chains compared with the baselines. Compared with ASER and ODEE, our method can work on multi-documents and the extracted results are not limited by pre-defined event ontologies. Thus, our method can greatly outperform them. CEE-IEA is trained for limited event types, thereby failing to handle open-domain documents. On top of that, its extracted events are disordered, which shows obvious weakness on the ERouge2 and ERougeL scores. Although the graph from SalienceAwareModel can reflect the orders among event mentions to some extent, its output has redundant mentions and can't ensure including all important ones. Overall, EMiner surpasses the baselines by a large margin.

4.6 Ablation Study

To verify the effectiveness of each component in our framework, we conduct an ablation study including component removal and replacement. The experiment results are shown in Table 3. We first remove each component from the full framework. Without event mention merging, we regard each mention as an event, and then perform event selection and ordering. If event selection is detached, the merged events are ordered according to their occurrences. After removing event ordering, we directly compare the selected salient events with human references. Our framework can already obtain a relatively high performance compared to the variant without merging. It reveals the significance of identifying similar event mentions, which can reduce information redundancy. In addition,

Table 2: Experimental results. Basic Setting refers to only evaluating the verb for each event while Advanced Setting refers to evaluating all the words. String Match and Hypernym Allowed are two overlap standards of two event mentions. The first requires all words to be the same and the second allows the hypernyms to relax the restrictions.

Basic Setting						
Models	**String Match**			**Hypernym Allowed**		
	ER-1	**ER-2**	**ER-L**	**ER-1**	**ER-2**	**ER-L**
RANDOM	4.2500	0.0759	2.7500	12.1428	3.5298	7.9702
LEAD	10.8095	1.9076	9.4345	16.3273	4.3404	15.3630
ASER	4.3915	0.2356	3.1245	14.4910	3.4934	5.0103
ODEE	6.1304	1.5395	5.7498	15.5597	4.6941	7.1258
CEE-IEA	12.1395	2.2945	6.4198	19.3922	4.9875	7.1871
SalienceAwareModel	13.6948	3.1566	10.1307	20.2479	6.4299	12.3195
EMiner	**17.7195**	**4.1674**	**15.5333**	**25.0217**	**7.4400**	**19.7000**
Advanced Setting						
Models	**String Match**			**Hypernym Allowed**		
	ER-1	**ER-2**	**ER-L**	**ER-1**	**ER-2**	**ER-L**
RANDOM	1.6250	0.3489	1.1250	4.5833	1.2721	3.5833
LEAD	9.6369	1.6723	8.8869	13.5654	2.9875	11.8154
ASER	1.7594	0.3957	1.2592	4.6140	1.3015	3.1357
ODEE	4.5105	0.7816	1.6893	5.1076	2.5361	4.1629
CEE-IEA	8.3976	1.5791	3.5420	12.1467	3.0968	6.0687
SalienceAwareModel	9.6240	2.1881	9.0317	14.0746	3.0817	10.0985
EMiner	**13.9196**	**3.0671**	**12.2307**	**16.9015**	**4.0622**	**14.6999**

Table 3: Ablation Study (Hypernym Allowed in Advanced Setting). 'w/o' means removing the component from the full framework. 'A → B' means replacing component A in the full framework with method B.

Model	ER-1	ER-2	ER-L
Component Removal			
EMiner	**16.9015**	**4.0622**	**14.6999**
w/o Merging	7.6785	0.8554	7.4743
w/o Selection	12.1011	2.5872	10.7619
w/o Ordering	16.8654	3.0450	14.2261
w/o Ordering$_{Content}$	16.8621	3.9781	14.2678
w/o Ordering$_{Common}$	16.7916	4.0134	14.4740
Component Replacement			
EMiner	**16.9015**	**4.0622**	**14.6999**
Extraction → ASER	13.4234	2.5079	10.3810
Merging → HDBSCAN	15.4940	3.7529	13.5714
Merging → OSDM	15.5654	3.0450	14.2261
Selection → ETDisc	15.2083	2.7651	12.5654
Ordering → SYMTIME	16.7440	3.9113	12.9446

removing the selection component affects the results sightly. It is supposed that, due to the lead bias problem, most salient events are arranged at the front of the chain after ordering. In addition, the obvious drop of the ERouge-L score in the fourth row reflects the important role that event ordering plays in this task.

In addition, we replace each component with other related methods. Compared with ASER, our extractor is better at processing long sentences. Based on this, EMiner show a clear advantage in the subsequent steps. In addition, our method is superior although the two comparative clustering methods also work. Moreover, filtering events instead of mentions can introduce more semantic information, which plays an important role in selecting salient events. Finally, although SYMTIME is a powerful pre-trained temporal model, it fails to utilize rich relation information between multiple mentions. Thus, EMiner can achieve better results.

4.7 Case Study

Figure 3 shows an interesting example on the super event of the Mexico Earthquake in 2017. It presents the outputs of each component in our framework to provide a straightforward view of our work. This case includes the extracted event mentions from the first step, one event cluster from the second step, the representative mentions of salient events from the third step, and the final event chain. For better comparison, we also show the human-annotated reference. Notably, here we show a representative event mention rather than all mentions for the salient events and the event chain. Through the case study, we want to verify the effectiveness and analyze the limitations of our framework. We can see that our method can successfully discover most of the salient events for a super event. For example, the occurrence and consequences of this earthquake have been saved in the event chain. Although some relationships between events are different from the reference, it

Super Event: Mexico Earthquake in 2017				
Extracted Event Mentions	**Event Cluster**	**Representative Mention in Salient Events**	**Event Chain**	**Reference**
earthquake strike state near coast earthquake felt in city trigger landslide area be fill with vacationer section closed by rockslide they separate to temblor kill people home suffer damage in town …	{earthquake strike state near coast, mexico locate at point, earthquake strike border on monday, evacuate in region, area strike by quake, hit by earthquake, quake strike off coast, quake occur on coast }	trigger landslide kill people family feel scared because experience house destroy block road quake felt in salvador report quake at magnitude suffer disruption to communication	1. hit by earthquake 2. house destroy 3. trigger landslides 4. people died 5. block roads 6. crack open in building	1.earthquake rock Mexico 2.damage houses 3.kill people 4.trigger landslides 5.block roads

Figure 3: Case study about Mexico Earthquake in 2017. It shows the outputs of each component in our framework including the extracted event mentions from the first step, one event cluster from the second step, the representative mentions of salient events from the third step, and the final event chain. The human-annotated reference is also shown here.

Table 4: Results of human evaluation by ranking. Relev., Infor., and Cohen. represent relevance, informativeness, and coherence to original texts, respectively. Reference refers to the human-annotated event chains.

Model	Relev.	Infor.	Cohen.
Merging → HDBSCAN	4.14	4.54	3.21
Selection → ETDISC	4.66	3.28	3.52
Ordering → SYMTIME	2.84	3.53	4.37
EMiner	**2.13**	**2.56**	**2.85**
Reference	1.23	1.09	1.05

doesn't affect our understanding of the entire super event. Please find more case studies in Appendix A.4

4.8 Human Evaluation

To better understand the model performance, we also conduct the human evaluation. Specifically, we ask 10 graduate students to rank five different event chains (produced by our framework, its variants, and groundtruth) according to three metrics: relevance, informativeness, and coherence to the texts. Ranking first means the best performance on this metric. We randomly select 20 samples from our dataset for evaluation. The results are provided in Table 4. From the perspective of relevance, our framework can output more relevant event chains to the super events. Compared with Selection → ETDISC, other methods can mine more relevant and salient events thanks to event-based selection introducing more semantic information. In terms of the informativeness metrics, our framework substantially extracts distinct events and reduces information redundancy when compared to other baselines. The capacity to grasp different events in similar meanings is largely responsible for this improvement. However, Merging → HDBSCAN performs poorly on the informativeness because it lacks semantic knowledge to identify synonyms in the mentions. Coherence depends on whether event chains can reflect the plot of texts smoothly. Benefiting from rich event relationships, event-based ordering can obtain high scores. However, the performance of all automatic models is still far from the human-annotated answers.

5 RELATED WORK

Considering the importance of events in understanding unstructured texts, many efforts have been devoted to representing and understanding events. FrameNet [3] proposes to represent events with schema, which has one predicate and several arguments. Event detection and extraction attract lots of research interests in this field [10, 15, 21, 24, 26, 31]. Ahmad et al. [2], Han et al. [13], Wang et al. [37] pay attention on event relation extraction and predicting. Salient event identification is also a popular research topic [16, 25, 39]. Apart from these, there have been recent interests in event process understanding [8, 43]. However, these studies cost expensive expert annotations. Some studies [20, 38] alleviate this problem under an unsupervised setting. The pioneering work [6] induces event chains as a new representation of structured knowledge. Chambers and Jurafsky [6] and Radinsky and Horvitz [32] extended such event chain modeling for news prediction and timeline construction. Berant et al. [4] extracted events and their relationships in biological processes for biological reading comprehension. More recently, Zhang et al. [46] walks on the event graph built based on a large corpus to obtain event chains for narrative understanding. These studies extract event schemas from a large number of texts as prior knowledge to assist downstream tasks [34, 40].

However, most of the existing works rely on human-curated event ontologies. They are limited to pre-defined event types and fail to extract events from open-domain texts. Also, they mainly focus on sentence-level or document-level extraction. Multiple documents would be a more complex and informative scenario. On top of that, although a few studies work on the event ordering task, they focus on the relationship between a pair of events. Yet, automatic mining of event chains is still under-explored.

6 CONCLUSION

In this paper, we propose a new task, Event Chain Mining, to summarize the skeleton of texts by extracting event chains. To address it automatically, a novel unsupervised framework EMiner is suggested. Besides, we develop a benchmark dataset and a comprehensive evaluation system for this task. Extensive experiments verify the effectiveness of the proposed framework and the quality of the produced event chains.

ACKNOWLEDGMENTS

Research was supported in part by US DARPA KAIROS Program No. FA8750-19-2-1004 and INCAS Program No. HR001121C0165, National Science Foundation IIS-19-56151, IIS-17-41317, and IIS 17-04532, and the Molecule Maker Lab Institute: An AI Research Institutes program supported by NSF under Award No. 2019897, and the Institute for Geospatial Understanding through an Integrative Discovery Environment by NSF under Award No. 2118329.

REFERENCES

[1] H Porter Abbott. 2020. *The Cambridge introduction to narrative*. Cambridge University Press.

[2] 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

[3] Collin F. Baker, Charles J. Fillmore, and John B. Lowe. 1998. The Berkeley FrameNet Project. In *36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, COLING-ACL '98, August 10-14, 1998, Université de Montréal, Montréal, Quebec, Canada. Proceedings of the Conference*, Christian Boitet and Pete Whitelock (Eds.). Morgan Kaufmann Publishers / ACL, 86–90. https://doi.org/10.3115/980845.980860

[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] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems* 33 (2020), 1877–1901.

[6] 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/

[7] Junyang Chen, Zhiguo Gong, and Weiwen Liu. 2019. A nonparametric model for online topic discovery with word embeddings. *Information Sciences* 504 (2019), 32–47.

[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] 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, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, Jill Burstein, Christy Doran, and Thamar Solorio (Eds.). Association for Computational Linguistics, 4171–4186. https://doi.org/10.18653/v1/n19-1423

[10] 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

[11] Edward Morgan Forster. 1985. *Aspects of the Novel*. Vol. 19. Houghton Mifflin Harcourt.

[12] Goran Glavas, Jan Snajder, Marie-Francine Moens, and Parisa Kordjamshidi. 2014. HiEve: A Corpus for Extracting Event Hierarchies from News Stories. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation, LREC 2014, Reykjavik, Iceland, May 26-31, 2014*, Nicoletta Calzolari, Khalid Choukri, Thierry Declerck, Hrafn Loftsson, Bente Maegaard, Joseph Mariani, Asunción Moreno, Jan Odijk, and Stelios Piperidis (Eds.). European Language Resources Association (ELRA), 3678–3683. http://www.lrec-conf.org/proceedings/lrec2014/summaries/1023.html

[13] 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

[14] Rujun Han, Qiang Ning, and Nanyun Peng. 2019. Joint Event and Temporal Relation Extraction with Shared Representations and Structured Prediction. 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)*. 434–444.

[15] 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

[16] 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

[17] Nikhil Ketkar. 2017. Introduction to pytorch. In *Deep learning with python*. Springer, 195–208.

[18] Jay Kumar, Junming Shao, Salah Uddin, and Wazir Ali. 2020. An Online Semantic-enhanced Dirichlet Model for Short Text Stream Clustering. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel R. Tetreault (Eds.). Association for Computational Linguistics, 766–776. https://doi.org/10.18653/v1/2020.acl-main.70

[19] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. 7871–7880.

[20] 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

[21] 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/v1/2021.naacl-main.69

[22] Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*. 74–81.

[23] Xiao Liu, Heyan Huang, and Yue Zhang. 2019. Open Domain Event Extraction Using Neural Latent Variable Models. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, Anna Korhonen, David R. Traum, and Lluís Màrquez (Eds.). Association for Computational Linguistics, 2860–2871. https://doi.org/10.18653/v1/p19-1276

[24] 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

[25] 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

[26] 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

[27] Leland McInnes, John Healy, and Steve Astels. 2017. hdbscan: Hierarchical density based clustering. *J. Open Source Softw.* 2, 11 (2017), 205. https://doi.org/10.21105/joss.00205

[28] Sebastião Miranda, Arturs Znotins, Shay B. Cohen, and Guntis Barzdins. 2018. Multilingual Clustering of Streaming News. 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, 4535–4544. https://aclanthology.org/D18-1483/

[29] Nasrin Mostafazadeh, Nathanael Chambers, Xiaodong He, Devi Parikh, Dhruv Batra, Lucy Vanderwende, Pushmeet Kohli, and James Allen. 2016. A Corpus and Evaluation Framework for Deeper Understanding of Commonsense Stories. *arXiv e-prints* (2016), arXiv–1604.

[30] Qiang Ning, Zhili Feng, and Dan Roth. 2017. A Structured Learning Approach to Temporal Relation Extraction. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. 1027–1037.

[31] 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

[32] Kira Radinsky and Eric Horvitz. 2013. Mining the web to predict future events. In *Sixth ACM International Conference on Web Search and Data Mining, WSDM 2013, Rome, Italy, February 4-8, 2013*, Stefano Leonardi, Alessandro Panconesi, Paolo Ferragina, and Aristides Gionis (Eds.). ACM, 255–264. https://doi.org/10.1145/2433396.2433431

[33] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, Peter J Liu, et al. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.* 21, 140 (2020), 1–67.

[34] Siva Reddy, Danqi Chen, and Christopher D. Manning. 2019. CoQA: A Conversational Question Answering Challenge. *Trans. Assoc. Comput. Linguistics* 7 (2019), 249–266. https://transacl.org/ojs/index.php/tacl/article/view/1572

[35] Jiaming Shen, Yunyi Zhang, Heng Ji, and Jiawei Han. 2021. Corpus-based Open-Domain Event Type Induction. 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, 5427–5440. https://doi.org/10.18653/v1/2021.emnlp-main.441

[36] Yee Whye Teh. 2010. Dirichlet Process.

[37] 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

[38] 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

[39] 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

[40] Lili Yao, Nanyun Peng, Ralph M. Weischedel, Kevin Knight, Dongyan Zhao, and Rui Yan. 2019. Plan-and-Write: Towards Better Automatic Storytelling. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*. AAAI Press, 7378–7385. https://doi.org/10.1609/aaai.v33i01.33017378

[41] Jianhua Yin, Daren Chao, Zhongkun Liu, Wei Zhang, Xiaohui Yu, and Jianyong Wang. 2018. Model-based Clustering of Short Text Streams. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2018, London, UK, August 19-23, 2018*, Yike Guo and Faisal Farooq (Eds.). ACM, 2634–2642. https://doi.org/10.1145/3219819.3220094

[42] Jianhua Yin and Jianyong Wang. 2016. A model-based approach for text clustering with outlier detection. In *32nd IEEE International Conference on Data Engineering, ICDE 2016, Helsinki, Finland, May 16-20, 2016*. IEEE Computer Society, 625–636. https://doi.org/10.1109/ICDE.2016.7498276

[43] 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

[44] Hongming Zhang, Xin Liu, Haojie Pan, Yangqiu Song, and Cane Wing-Ki Leung. 2020. ASER: A Large-scale Eventuality Knowledge Graph. In *WWW '20: The Web Conference 2020, Taipei, Taiwan, April 20-24, 2020*, Yennun Huang, Irwin King, Tie-Yan Liu, and Maarten van Steen (Eds.). ACM / IW3C2, 201–211. https://doi.org/10.1145/3366423.3380107

[45] Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. BERTScore: Evaluating Text Generation with BERT. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net. https://openreview.net/forum?id=SkeHuCVFDr

[46] 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.emnlp-main.107

[47] 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

[48] Ming Zhong, Pengfei Liu, Danqing Wang, Xipeng Qiu, and Xuanjing Huang. 2019. Searching for Effective Neural Extractive Summarization: What Works and What's Next. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, Anna Korhonen, David R. Traum, and Lluís Màrquez (Eds.). Association for Computational Linguistics, 1049–1058. https://doi.org/10.18653/v1/p19-1100

[49] Ming Zhong, Yang Liu, Suyu Ge, Yuning Mao, Yizhu Jiao, Xingxing Zhang, Yichong Xu, Chenguang Zhu, Michael Zeng, and Jiawei Han. 2022. Unsupervised Multi-Granularity Summarization. In *Findings of the Association for Computational Linguistics: EMNLP 2022*. 4980–4995.

[50] Ben Zhou, Kyle Richardson, Qiang Ning, Tushar Khot, Ashish Sabharwal, and Dan Roth. 2021. Temporal Reasoning on Implicit Events from Distant Supervision. 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, 1361–1371. https://doi.org/10.18653/v1/2021.naacl-main.107

Table 5: Several event mention patterns and the corresponding examples. "v" stands for verbs, "n" stands for nouns, and "a" stands for adjectives. "nsubj", "dobj", "xcomp", and "nsubjpass" are syntactic relations.

Pattern	Example
n_1-*nsubj*-v_1	people die
n_1-*nsubj*-v_1-*dobj*-n_2	earthquake hit Mexico
n_1-*nsubj*-v_1-*xcomp*-*a*	residents felt scared
n_1-*nsubj*-v_1-*xcomp*-v_2-*dobj*-n_2	he wants to drink water
n_1-*nsubjpass*-v_1	people was injured

A APPENDIX

A.1 Baselines

Considering the novelty of the event chain mining task, there is no existing approach to directly solve this task. Therefore, we apply several related works to our scenario:

- ASER [44] proposes a lightweight open-domain event extractor based on syntactic pattern matching.
- ODEE [23] adopts a neural latent variable model for event mention extraction.
- CEE-IEA [16] uses a contextual model to detect salient event mentions from single documents;
- SalienceAwareModel [46] builds an event mention graph from the corpus, and starts a walk along the directed edge to obtain maximum event mention chains. In our experiment, the graph is built based on multi-documents.

In addition, we present a randomly produced event chain as the lower bound for this task. Moreover, due to the lead bias problem in the news domain [48], we introduce LEAD as a strong baseline. It refers to extracting all events from the first three sentences of the texts as a chain. Since these baselines target event mentions, each mention is used as an event cluster for evaluation. Except for SalienceAwareModel, other methods target single documents. Therefore, in our multi-documents scenario, we use them to process each document and report the highest evaluation scores. Also, considering these methods cannot handle event orders, the extracted event mentions are sorted by the order in which the text describes them.

To verify the effectiveness of each component in EMiner, we also conduct the ablation study. We remove each component or replace it with related methods as the baselines.

- For mention extraction: ASER [44] has been introduced above.
- For mention merging: HDBSCAN [27] is a hierarchical density-based spatial clustering method; OSDM [18] is an online semantic-enhanced dirichlet model for short text stream clustering. These two methods don't require fixing the number of clusters.
- For event selection: ETDisc [35] provides a frequency-based salient word selector and we expand it to filter event mentions.
- For event ordering: SYMTIME [50] is a neuro-symbolic temporal reasoning pretrained model, which can determine the order between events.

A.2 Evaluation Metrics

Given the groundtruth $E = [e_1, e_2, \dots e_n]$ and the model output $Z = [z_1, z_2, \dots z_m]$ (n and m are the lengths of the event chains), the ERouge-1 scores (including precision and recall) can be calculated as followed:

$$\text{ERouge-1}_{pre} = \frac{1}{m}\sum_{i=1}^{m}\max_{j\in(1,\dots,n)} overlap(z_i, e_j)$$

$$\text{ERouge-1}_{rec} = \frac{1}{n}\sum_{i=1}^{n}\max_{j\in(1,\dots,m)} overlap(z_j, e_i)$$

Here, *overlap* is a function to measure the overlap of two events. we provide two overlap standards of two event mentions to better understand the mining quality, "String Match" and "Hypernym Allowed", following Zhang et al. [43]. The first standard requires the words in the produced mention to be the same as the referent mention. This setting is rather strict. The second standard allows the hypernyms of words in mentions to relax the restrictions on the comparison. Different from Rouge-1 evaluating single events, Rouge-2 focuses on adjacent event pairs:

$$\text{ERouge-2}_{pre} = \frac{1}{m-1}\sum_{i=1}^{m-1}\max_{j\in(1,\dots,n-1)} overlap([z_i, z_{i+1}], [e_j, e_{j+1}])$$

$$\text{ERouge-2}_{rec} = \frac{1}{n-1}\sum_{i=1}^{n-1}\max_{j\in(1,\dots,m-1)} overlap([z_j, z_{j+1}], [e_i, e_{i+1}])$$

Here, given two equal-length event chains, their *overlap* score is the average of the overlap scores of the corresponding events to the two sequences. As to ERouge-L, it evaluates the longest common event subsequence in the model output and the groundtruth. Specifically, E' and Z' are the subsequences of E and Z respectively. They are of equal length, v. Then ERouge-L can be denoted as:

$$\text{ERouge-L}_{pre} = \frac{1}{m}\max_{E'\subseteq E, Z'\subseteq Z}\frac{1}{v} overlap(E', Z')$$

$$\text{ERouge-L}_{rec} = \frac{1}{n}\max_{E'\subseteq E, Z'\subseteq Z}\frac{1}{v} overlap(E', Z')$$

Based on the precision and recall of three ERouge scores, we can obtain the corresponding F_1 score:

$$\text{ERouge-k}_{F_1} = \frac{2 * \text{ERouge-k}_{pre} * \text{ERouge-k}_{rec}}{\text{ERouge-k}_{pre} + \text{ERouge-k}_{rec}}, \quad k \in \{1, 2, L\}$$

A.3 Implementation Details

We implement EMiner using PyTorch [17]. All the experiments are conducted on 1 NVIDIA TITAN Xp GPU.

For event mention extraction, we give priority to more complex patterns to make event mentions contain more details. That is, once a complex pattern is exactly matched, we will no longer consider the remaining simpler ones. By such a strategy, all possible event mentions can be extracted from texts. Also, the extracted events will not overlap. (the selected syntactic patterns are shown in Table 5)

For event mention merging, to decide the parameters, we first decide the parameter scopes following related studies on news

Super Event: Mexico Earthquake in 2017				
Extracted Event Mentions	**Event Cluster**	**Representative Mention in Salient Events**	**Event Chain**	**Reference**
peterson accused of fires, fire led investigation, he arrested on allegations, peterson entered plea at court, peterson hails from community, scenario rounded with megawatts, organization preserved parcels, transaction culminated years, peterson suspected of fires,	{firefighter arrested, peterson arrested on charges, firefighter arrested on suspicion, peterson arrested, peterson arrested on Tuesday, peterson arrested after capt., he arrested on allegations, } {destroy structures, destroyed by fire, destroying structures, damaging by flames, destroyed by fire, structures claimed by fire, burn homes}	arrest came after months turn into fire body found inside home destroy dozen homes left person injured peterson face years for charge suffer losses fire burn throughout area	1. arrest came after months 2. turn into fire 3. body found inside home 4. destroy dozen homes 5. leave person injured 6. peterson faces years for charge 7. suffer losses 8. fire burn throughout area	1. man started fires 2. fire destroyed dozen homes 3. fire left person dead 4. officers arrested man on suspicion 5. man entered plea at court 6. fire burned miles over week 7. fire fanned by winds

Figure 4: Case study about A Former Firefighter Arrested for Starting Fires. It shows the outputs of each component in our framework including the extracted event mentions from the first step, two event clusters from the second step, the representative mentions of salient events from the third step, and the final event chain. The human-annotated reference is also shown here.

clustering [41], and then slightly change the values and choose the best parameters according to the experiment results. Finally, we set $\alpha = 0.3$, $\beta = 0.03$, and the number of iterations to 10. For different super events, we use the same set of parameters. All the extracted events are grouped and we do not manually de-duplicate events in the same group. For salient event selection, we select the events with the top 20 salience scores as they can cover the main content of the texts.

Regarding salient event ordering, we use BART-large as the backbone generation model for commonsense-based ordering. To adapt BART to our defined generation task, we utilize a story dataset, ROCStory [29] as the distant supervision for the training. Each story in this dataset consists of 5 narrative sentences. Typically, each sentence describes an event, and the narrative description is in line with the order of event occurrences. For each story, we extract all the event mentions (as introduced in Section 3.1) and arrange them in the order they occur in the story to form the target sequence in the training data. Also, we randomly shuffle the event mentions of each story to produce the input sequence. BART is trained on these synthetic data for 5 epochs with a batch size of 32 and a learning rate of 2e-5. Then it can become familiar with the given task format and output an adjusted the sequence of events based on commonsense. During the inference, we take the representative mentions of salient events as input and obtain a new ranking of each event using BART. λ used in the overall ordering score is 0.1 in all the experiments.

A.4 Case Study

Figure 4 presents the case study. The super event is *A Former Firefighter Arrested for Starting Fires*. It shows the outputs of each component in our framework including the extracted event mentions from the first step, two event clusters from the second step, the representative mentions of salient events from the third step, and the final event chain. The human-annotated reference is also shown here.