Just What You Desire: Constrained Timeline Summarization with Self-Reflection for Enhanced Relevance

Muhammad Reza Qorib[1], **Qisheng Hu**[2*], **Hwee Tou Ng**[1]

[1]Department of Computer Science, National University of Singapore
[2]College of Computing and Data Science, Nanyang Technological University
mrqorib@u.nus.edu, qisheng001@e.ntu.edu.sg, nght@comp.nus.edu.sg

Abstract

Given news articles about an entity, such as a public figure or organization, timeline summarization (TLS) involves generating a timeline that summarizes the key events about the entity. However, the TLS task is too underspecified, since what is of interest to each reader may vary, and hence there is not a single ideal or optimal timeline. In this paper, we introduce a novel task, called Constrained Timeline Summarization (CTLS), where a timeline is generated in which all events in the timeline meet some constraint. An example of a constrained timeline concerns the legal battles of Tiger Woods, where only events related to his legal problems are selected to appear in the timeline. We collected a new human-verified dataset of constrained timelines involving 47 entities and 5 constraints per entity. We propose an approach that employs a large language model (LLM) to summarize news articles according to a specified constraint and cluster them to identify key events to include in a constrained timeline. In addition, we propose a novel self-reflection method during summary generation, demonstrating that this approach successfully leads to improved performance.

Code and Data — https://github.com/nusnlp/reacts

Introduction

In today's internet era, the rapid and massive flow of information makes it hard to stay updated, particularly for topics with extensive coverage over time. In the United States alone, Hamborg, Meuschke, and Gipp (2018) report that more than 5,000 news articles are being published every day. To help readers quickly grasp important information, many news platforms offer news in a timeline format, especially for important topics that progress over time, such as pandemics[1] or conflicts[2].

Original Timeline	
2003-11-19:	Receives the National Book Foundation Medal for Distinguished Contribution to American Letters.
2015-09-10:	Is awarded the National Medal of Arts by US President Barack Obama.
2015-11-03:	Releases a collection of short stories entitled "The Bazaar of Bad Dreams."
2018-07-25:	King is an executive producer of a show written for the streaming service Hulu. The series, "Castle Rock," is named after the fictional small Maine town that provides the setting for various King books and stories.
2020-04-21:	King's latest book, "If It Bleeds" is published. The book is a compilation of four novellas.
2021-06-04:	The miniseries "Lisey's Story," adapted by King and based on his 2006 novel of the same name, premieres on Apple TV+.
2022-09-06:	King's novel "Fairy Tale" is published.

Constrained Timeline	
Constraint: Focus on Stephen King's book releases.	
2015-11-03:	Releases a collection of short stories entitled "The Bazaar of Bad Dreams."
2020-04-21:	King's latest book, "If It Bleeds" is published. The book is a compilation of four novellas.
2022-09-06:	King's novel "Fairy Tale" is published.

Table 1: An unconstrained timeline of Stephen King and a constrained version focusing on Stephen King's book releases.

The task of summarizing news articles, or any collections of text documents, into timelines is called timeline summarization (TLS). Timeline summarization aims to automatically condense long-running news topics into temporally ordered time-stamped textual summaries of events on a particular topic. Timeline summarization aims to include any important events into the timeline without considering particular aspects that the readers are interested in.

To take a reader's interest into account when generating a timeline, we propose a new task called constrained timeline summarization (CTLS). Constrained timeline summa-

[*] Work done while Qisheng Hu was affiliated with the National University of Singapore.

[1] https://www.cdc.gov/museum/timeline/covid19.html

[2] https://www.usnews.com/news/best-countries/slideshows/a-timeline-of-the-russia-ukraine-conflict

rization offers personalization that TLS lacks. For example, a reader may want to automatically retrieve the timeline of Stephen King's book publication (Table 1). In this example, Stephen King's national awards are irrelevant to the reader even though they are generally considered important events in Stephen King's life.

The contributions of this paper are as follows:

- We propose a new task, constrained timeline summarization. The task has real-life applications.
- We present a new test set to benchmark models on the constrained timeline summarization task.
- We present an effective method that utilizes large language models without any need for training or fine-tuning.
- We propose a novel self-reflection method to produce a more relevant constrained event summary and demonstrate that self-reflection helps in generating more relevant constrained timelines.

Related Work

In this section, we briefly discuss related work on timeline summarization, query-based summarization, and update summarization. Constrained timeline summarization can be viewed as an amalgamation of the first two tasks.

Timeline Summarization

Previous work on timeline summarization can be categorized into three main approaches: direct summarization, date-wise approaches, and event detection approaches.

Direct Summarization In this approach, a collection of documents is treated as a set of sentences to be directly extracted. Sentence extraction can be performed by optimizing sentence combinations (Martschat and Markert 2018) or by ranking sentences (Chieu and Lee 2004). This category also includes methods that treat the task as an extension of multi-document summarization, where the goal is to generate a summary from multiple documents (Allan, Gupta, and Khandelwal 2001; Yu et al. 2021).

Date-wise Approach In this approach, the task is divided into two steps: identifying important dates and summarizing events that occurred on those dates. Most methods employ supervised techniques to select the dates. For instance, Ghalandari and Ifrim (2020) propose a classification or regression model to predict date importance, while Tran, Herder, and Markert (2015) utilize graph-based ranking for date selection.

Event Detection In this approach, the system first detects important events from the articles by clustering them based on similarity. It then ranks and selects the most important clusters and summarizes them into event descriptions. Various techniques have been proposed for clustering, including Markov clustering on bag-of-words features (Ghalandari and Ifrim 2020), dynamic affinity-preserving random walks (Duan, Jatowt, and Yoshikawa 2020), event graph compression (Li et al. 2021), date graph model (La Quatra et al. 2021), heterogeneous graph attention networks (You et al. 2022), and even large language models (Hu, Moon, and Ng 2024).

Query-Based Summarization

Query-based summarization, also called query-focused, topic-based, or user-focused summarization, aims to extract and summarize information that users are specifically interested in from a large number of texts. Essentially, it is a type of summarization that leverages user-provided query information.

Early approaches to query-based summarization mainly score or rank the relevance of each sentence in the document to the query based on predefined features (Rahman and Borah 2015). Sentences with the highest scores are then extracted to create the summary. Relevance scoring can be performed in an unsupervised manner by utilizing lexical and semantic features (Conroy, Schlesinger, and O'Leary 2006; Krishna, Kumar, and Reddy 2013) or in a supervised manner by training regressor models (Mani and Bloedorn 1998; Ouyang et al. 2011). Document graphs are also often employed when dealing with multiple documents (Mohamed and Rajasekaran 2006; Wang et al. 2013).

Due to the effectiveness of transformers, recent query-based summarization methods are predominantly based on transformer models, including large language models. For example, Laskar, Hoque, and Huang (2020) incorporate query relevance into BERTSUM (Liu and Lapata 2019), while Park and Ko (2022) integrate a query-attentive semantic graph with sequence-to-sequence transformers. Fine-tuning large language models has also been explored, such as in the work by Xu et al. (2023), who fine-tune BART (Lewis et al. 2020), and Cao et al. (2024), who fine-tune Llama 2 (Touvron et al. 2023) using custom adapters.

Update Summarization

Update summarization is the task of generating a short summary from a set of documents A under the assumption that users have read a set of documents B (Dang and Owczarzak 2009). Update summarization has a different objective from timeline summarization, but the methods proposed for it often bear some resemblance to the event detection approach for timeline summarization, notably in determining the novelty of the information from set A in relation to set B (Steinberger and Ježek 2009). In the context of timeline summarization, novelty detection involves determining whether the extracted events from set A are new events that are not present in set B.

Dataset

To benchmark the constrained timeline summarization task, we propose a novel test set called CREST (Constraint Restrictions on Entities to Subset Timelines). CREST consists of 235 timelines from 47 public figures or institutions (entities). We derive these timelines from the ENTITIES dataset (Ghalandari and Ifrim 2020), which were crawled from CNN Fast Facts. For each entity, we generate 5 pairs of constraints and corresponding subset timelines. The article

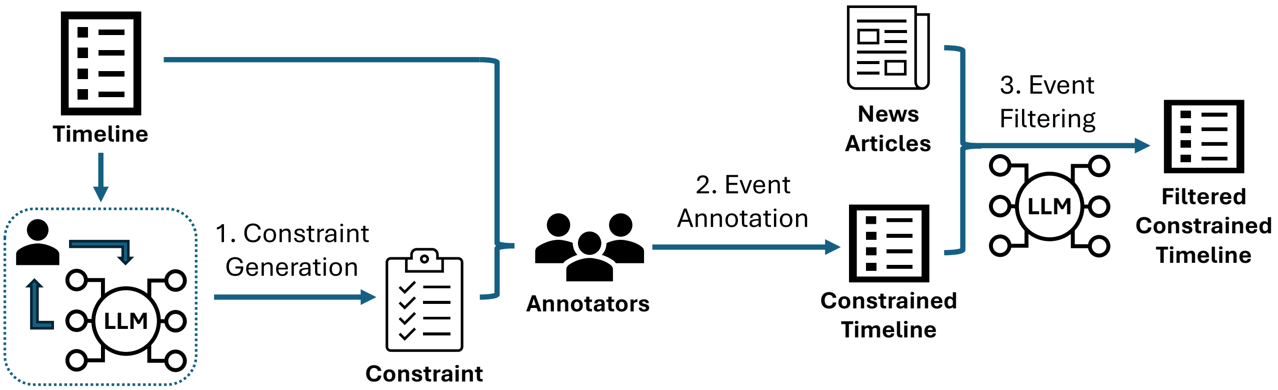


Figure 1: The dataset creation process consists of three steps: constraint generation, event annotation, and event filtering. Human annotators are tasked with determining whether an event in a timeline adheres to a constraint or not. The list of events that adheres to a constraint becomes a constrained timeline.

pool is sourced from the ENTITIES dataset, which was collected from The Guardian using the official API[3]. As such, our dataset is limited to British and American news sources. The dataset creation process involves constraint generation, event annotation, and event filtering.

Constraint Generation

We generate constraints by prompting GPT-4o[4] and manually selecting the best 5 constraints for each timeline. The prompt instructs GPT-4o to propose single-sentence constraints in the format "Focus on ...". To ensure that we have a variety of constraints, we query GPT-4o with four different types of prompts: general, numerical, relational, and geographical. The general prompt asks GPT-4o for constraint suggestions without any additional specification. The numerical prompt asks GPT-4o to generate constraints that contain ordinal phrases (e.g., first, second, etc.) or time indicators (e.g., timestamp, month, year, etc.). The relational prompt focuses on constraints involving some relationship between the entity and other public figures or institutions (e.g., "Focus on Stephen King's interactions with President Barack Obama."), while the geographical prompt asks for constraints with geographical information.

We find that GPT-4o generally suggests good constraints, but occasionally, the suggestions include hallucinations (e.g., constraints referencing non-existent events in the timeline) or are overly specific (e.g., only applicable to one event). Therefore, a human-in-the-loop process is essential to curate a good set of constraints for each timeline. Human intervention involves selecting the proposed constraints and modifying them to be more general.

Event Annotation

Given a list of events E_t from timeline t and a set of constraints C_t for timeline t, we build the constrained timelines by asking human annotators to label whether each event in the timeline adheres to each constraint. All constraints are applied to all events in the timeline, resulting in $|E_t| \times |C_t|$ assertions. Each annotator is provided with the complete timeline (containing all events) and the full set of constraints for that timeline.

We recruited 4 university students with strong English proficiency as annotators. To ensure high-quality annotations, the annotators completed a qualifying test by performing annotations on a different timeline. The annotators performed the task for approximately four hours and were compensated above standard rates ($22.65/hour). We found that our test set had high inter-annotator agreement, with an exact match percentage of 94.7% and a Cohen's kappa of 0.78 between the first and second annotators and an exact match percentage of 96.2% and a Cohen's kappa of 0.88 between the third and fourth annotators.

Event Filtering

One challenge with the ENTITIES dataset is that the article pool and timelines were collected independently from different sources. This causes a mismatch between the events covered by the articles and those included in the timelines. As a result, some important events in the ground-truth timelines are not covered by the article pool, making it impossible for the model to generate them without external knowledge. In such cases, even a human would not be able to achieve a perfect score.

To avoid unfairly penalizing an automatically constructed model, we provide an additional evaluation setting in which events in the timelines that are not covered by the article pool are filtered out. Given that the article pool contains more than forty-five thousand news articles, manually checking event coverage would be too costly and labor intensive. Following previous work (Gilardi, Alizadeh, and Kubli 2023), we utilize GPT-4o to check each article for information related to the events in question.

It is important to note that we only filter out events from the timelines, while the article pool remains unchanged. We assume that the ground truth timelines are comprehensive lists of all significant events related to the entities. We report the statistics of our dataset for both the full and filtered

[3] https://open-platform.theguardian.com/

[4] `gpt-4o-2024-08-06`

Statistics	All Events	Filtered
# topics	47	47
# timelines	233	201
# events	1031	667
Avg. # articles per topic	959	959
Avg. # timelines per topic	4.96	4.28
Avg. # events per timeline	4.42	3.32

Table 2: Statistics of our proposed dataset (CREST).

settings in Table 2.

Algorithm 1: Method

```
Require: A queue of articles $A$, a topic keyword $q$, a constraint $c$, a new article $a_i$ that arrived at time $i$, the event database $D$, the event clusters $G$, the retrieval limit $N$, the number of dates $l$ in the timeline, the number of sentences per date $k$ in the timeline.
Ensure: A timeline $T_{q,c}$ about topic $q$ following the constraint $c$, comprising $l$ timestamped event descriptions, each with $k$ sentences.
  $a_i \leftarrow \textsc{Dequeue}(A)$
  $e_i \leftarrow \textsc{ConstrainedTopicSum}(a_i, q, c)$
  if $e_i \neq \texttt{NULL}$ then
    if $\textsc{AdhereToConstraint}(e_i, q, c)$ then
      $Edges \leftarrow \{\}$
      for all $e_j \in \textsc{Retrieve}(D, e_i, N)$ do
        if $\textsc{SameEvent}(e_i, e_j)$ then
          $Edges \leftarrow Edges \cup \{[e_i, e_j]\}$
        end if
      end for
      $G \leftarrow \textsc{UpdateClusters}(G, Edges)$
      $D \leftarrow \textsc{Insert}(D, e_i)$
    end if
  end if
  $Clusters \leftarrow \textsc{RankClusters}(G, l)$
  $T_{q,c} \leftarrow []$
  $j \leftarrow 1$
  for all $v \in \textsc{SortByTime}(Clusters)$ do
    $T_{q,c}[j] \leftarrow \textsc{Summarize}(v, k)$
    $j \leftarrow j + 1$
  end for
  return $T_{q,c}$
```

Problem Definition

Constrained timeline summarization is a task to generate a timeline T that includes important events related to a topic and adhering to a constraint, given a list of input documents. The input comprises temporally ordered documents $A = \{a_1, a_2, ...\}$ related to a specific topic q, a constraint c, the expected number of dates l in the timeline, and the expected number of sentences per date k. The system-generated timeline T will be evaluated against a ground-truth timeline R. Similar to most timeline summarization datasets, the list of documents in this dataset is a list of chronologically ordered news articles. The constraint is a natural language sentence that specifies the kind of events related to the topic q that should be included in the timeline.

Method

Following the LLM-TLS method (Hu, Moon, and Ng 2024), we propose a new approach for the constrained timeline summarization task by leveraging a large language model (LLM) for summarization and clustering, which we call REACTS (REflective Algorithm for Constrained Timeline Summarization). Our method consists of four main steps: event summarization, self-reflection, event clustering, and finally, cluster and sentence selection. The core idea is to summarize each document according to the constraint, cluster the summaries that relate to the same event, and transform the clusters into event descriptions with corresponding dates. We illustrate our method in Figure 2.

Event Summarization

Inspired by the effectiveness of LLMs in query-based summarization (Jiang et al. 2024), we employ large language models for event summarization. The summary is expected to be in the format of a date followed by a one-sentence summary of a key event in the article related to the keyword and that adheres to the constraint, such as, ”2021-06-04: The miniseries *Lisey’s Story*, adapted by King and based on his 2006 novel of the same name, premieres on Apple TV+.” If there is nothing to summarize that meets the constraint, the model is expected to output NULL. We refer to this process as ConstrainedTopicSum in Algorithm 1.

Each article includes a publication date, but the important event may occur sometime before the publication date without an explicit mention of the exact date in the article. To assist the model in generating the correct date, we preprocess the news articles by prepending the sentence with the exact date whenever a time reference is mentioned. For example, if the publication date is 14 August 2024, and a sentence in the article contains a time reference like “yesterday,” the sentence is prepended with “(2024-08-13)”. Similarly, if the article mentions “last Friday,” the sentence is prepended with “(2024-08-09)”. The time references are parsed using HeidelTime[5] (Strötgen and Gertz 2015).

Self-Reflection

Self-evaluation techniques have been reported to improve the reasoning capabilities of LLMs (Weng et al. 2023; Xie et al. 2024). We observe that LLMs often produce an event summary even when it does not adhere to the specified constraint. To mitigate this, we employ self-reflection as an additional verification step by prompting the same LLM to assess whether the summary it just generated, e_i, for topic t adheres to the constraint c. If the model determines that it does not, e_i is discarded and excluded from the timeline generation. We refer to this process as AdhereToConstraint in Algorithm 1. The input prompt to perform self-reflection is given in Table 3.

[5] https://github.com/HeidelTime/heideltime

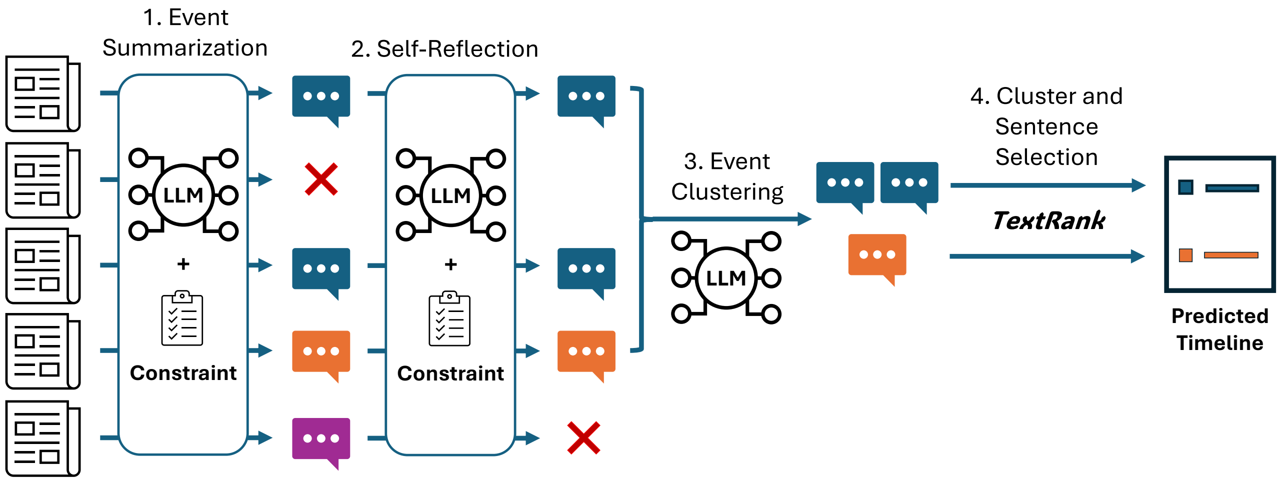


Figure 2: Illustration of our method for constrained timeline summarization (REACTS). The method consists of four steps: event summarization, self-reflection, event clustering, and cluster and sentence selection.

```
Review the timestamped event description related to keyword, accompanied by a constraint. Please determine whether the event description complies with or corresponds to the constraint. Respond with ‘Yes’ if the event description aligns with the constraint, or with ‘No’ if it does not.
#################
{positive example}

#################
{negative example}

#################
### Event
{event}

### Constraint
{constraint}
### Answer
```

Table 3: Prompt template for self-reflection.

Event Clustering

For every event summary e_i that passes the self-reflection step, the event description is encoded using the General Text Embeddings (GTE) model (Li et al. 2023). The summaries are transformed into embedding vectors so that semantically similar event summaries describing the same event can be accurately grouped together into a cluster. GTE performs exceptionally well on the Massive Text Embedding Benchmark (MTEB) while maintaining a modest number of parameters, making it an ideal choice for encoding summaries from tens of thousands of articles.

To generate clusters, from a vector database D, we retrieve N event descriptions that have the closest embedding vectors to the current event description being processed (the RETRIEVE function in Algorithm 1). For each pair consisting of the current event description and its retrieved neighbor, we use an LLM with few-shot prompting to check whether they describe the same event. In addition to this description matching by the LLM, we also check whether the event dates are the same, as events with similar descriptions but different dates likely represent distinct occurrences. This process is denoted as the SAMEEVENT function in Algorithm 1. If the pair passes both checks, the current event description is added to the cluster of its first matching neighbor. If the event description does not match any of its top N neighbors, it forms its own cluster. Finally, the embedding vector of each summary is stored in D to facilitate grouping with similar subsequent event descriptions.

Cluster and Sentence Selection

Each cluster represents an event related to the topic. To generate a timeline with l events, we select the best l clusters and summarize the event descriptions within each cluster into k sentences as specified by the user. We employ a heuristic to choose the top l clusters based on size, with the intuition that more significant events are covered by more articles, especially in the news domain. Subsequently, we apply TextRank (Mihalcea and Tarau 2004) to select the best k sentences within each cluster.

Baseline Method

A straightforward approach to perform constrained timeline summarization using an LLM is to concatenate the articles into the prompt and directly ask the model to produce a constrained timeline. However, LLMs have a limited context window size, so it is not always possible to fit the entire article pool within the input. To address this limitation, the baseline method involves randomly sampling articles and incrementally adding them to the input prompt one by one until the input context size limit is reached (taking into account the space needed for the instruction prompt and the output). Next, an instruction is added to the prompt, asking

		AR-1			AR-2			Date F1		
Model	**LLM**	**P**	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**
All events										
REACTS w/o SR	L3.1-8B	0.0580	0.0541	0.0561	0.0262	0.0264	0.0263	0.1399	0.1391	0.1395
REACTS	L3.1-8B	**0.0859**	**0.0695**	**0.0768**	**0.0381**	**0.0326**	**0.0351**	**0.1809**	**0.1710**	**0.1758**
REACTS w/o SR	L3.1-70B	0.0701	0.0957	0.0809	0.0326	0.0496	0.0393	0.1773	0.1773	0.1773
REACTS	L3.1-70B	**0.0970**	**0.1246**	**0.1091**	**0.0434**	**0.0592**	**0.0501**	**0.2425**	**0.2396**	**0.2411**
Filtered events										
Baseline	L3.1-8B	0.0387	0.0180	0.0246	0.0042	0.0026	0.0032	0.1005	0.0574	0.0731
REACTS w/o SR	L3.1-8B	0.0769	0.0693	0.0729	0.0337	0.0339	0.0338	0.1749	0.1743	0.1746
REACTS	L3.1-8B	**0.1095**	**0.0882**	**0.0977**	**0.0497**	**0.0427**	**0.0459**	**0.2266**	**0.2211**	**0.2238**
Baseline	L3.1-70B	0.0687	0.0939	0.0793	0.0324	0.0480	0.0387	0.1341	0.2524	0.1751
REACTS w/o SR	L3.1-70B	0.0906	0.1220	0.1040	0.0405	0.0614	0.0488	0.2315	0.2315	0.2315
REACTS	L3.1-70B	**0.1152**	**0.1533**	**0.1316**	**0.0483**	**0.0703**	**0.0572**	**0.2925**	**0.2925**	**0.2925**
Filtered events (10% data)										
Baseline	GPT-4o	0.0487	**0.1451**	0.0729	0.0216	0.0730	0.0334	0.2065	**0.3176**	0.2506
REACTS	GPT-4o	**0.0652**	0.1386	**0.0887**	**0.0281**	**0.0752**	**0.0409**	**0.3000**	0.3000	**0.3000**

Table 4: Score comparison of the baseline method, our method (REACTS), our method without self-reflection (REACTS w/o SR) using Llama 3.1 8B (L3.1-8B), Llama 3.1 70B (L3.1-70B), and GPT-4o on our dataset (CREST). We evaluate the models on precision (P), recall (R), and F1 scores using alignment-based ROUGE-1 (AR-1), alignment-based ROUGE-2 (AR-2), and date F1-score metrics. The best scores for each experiment setting are boldfaced.

the model to generate a timeline comprising l events, each described with a date and a k-sentence summary that adheres to the constraint c. The model then generates the timeline token-by-token until it either determines that it should stop or reaches the token limit.

Experiments

We run experiments to investigate whether self-reflection helps in generating more relevant timelines. We evaluate our method on our proposed dataset, both against the ground-truth timelines with all events and ground-truth timelines with filtered events. We employ Llama-3.1 8B[6] (Llama Team 2024), Llama-3.1 70B, and GPT-4o (OpenAI 2024) as the LLMs for our proposed method and the baseline method. However, we only evaluate models with GPT-4o on 10% of the test set due to cost consideration. In all experiments, we set the generation temperature of the LLMs to zero to make the results reproducible.

As previously explained, the baseline method is inherently limited by the maximum context length of the LLM. Therefore, it can only consider a limited number of articles when generating a timeline. To evaluate the best possible performance of the baseline method, we also conduct additional experiments with an oracle article retriever. From the article pool, the oracle retriever retrieves only articles relevant to the events in the unconstrained ground-truth timeline. Even though the oracle retriever helps to filter out noisy articles, the models still need to determine whether the events in the articles adhere to the constraints. The set of articles kept by the oracle retriever is then randomly sampled to fit the context length of the baseline method and used as the final article pool for all methods. That is, in this experiment, all methods receive the exact same set of input articles to generate the timeline summary. We use GPT-4o as the oracle retriever; therefore, we do not use it as the backbone LLM for the methods.

Evaluation

We evaluate the experiments with the standard metrics for the timeline summarization task, which are alignment-based ROUGE F1-score (Martschat and Markert 2017) and date F1-score (Martschat and Markert 2018). We employ an approximate randomization test (Riezler and Maxwell 2005; Chinchor, Hirschman, and Lewis 1993) with 100 trials and a p-value of 0.05 to measure statistical significance.

Alignment-Based ROUGE F1-score The alignment-based ROUGE F1-score measures the text overlap of the event descriptions between the predicted timeline and the ground-truth timeline. It first aligns the events in the predicted timeline with events in the ground-truth timeline based on the closeness of the dates and the similarity of the event descriptions. Following (Ghalandari and Ifrim 2020), we use the alignment setting that allows many-to-one alignment.

For each pair of aligned predicted event and ground-truth event, the metric[7] measures the n-gram similarity between the event descriptions. Precision is proportional to the ratio of the overlap compared to the predicted event description, while recall is proportional to the ratio of the overlap compared to the ground-truth event description.

Date F1-Score The date F1-score simply measures the F1 score of the dates covered in the ground-truth timeline

[6] https://llama.meta.com/

[7] We use ROUGE v1.5.5 by Chin-Yew Lin with `-n 2 -m -s` arguments, which measures up to 2-gram similarity of stemmed words, ignoring stopwords.

		AR-1			AR-2			Date F1		
Model	**LLM**	**P**	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**
With oracle retriever on filtered events										
BASELINE	L3.1-8B	0.0761	0.0829	0.0794	0.0299	0.0310	0.0305	0.2160	**0.2935**	0.2489
REACTS w/o SR	L3.1-8B	0.0966	0.0800	0.0875	0.0399	0.0341	0.0367	0.2172	0.2151	0.2161
REACTS	L3.1-8B	**0.1505**	**0.1102**	**0.1272**	**0.0687**	**0.0490**	**0.0572**	**0.2916**	0.2766	**0.2839**
BASELINE	L3.1-70B	0.1149	0.1764	0.1392	0.0429	0.0767	0.0550	0.2539	**0.4811**	0.3324
REACTS w/o SR	L3.1-70B	0.1142	0.1488	0.1292	0.0482	0.0655	0.0555	0.3165	0.3132	0.3148
REACTS	L3.1-70B	**0.1810**	**0.2213**	**0.1991**	**0.0735**	**0.0959**	**0.0832**	**0.4769**	0.4485	**0.4622**

Table 5: Results of the experiments with oracle retriever of the baseline method, our method (REACTS), our method without self-reflection (REACTS w/o SR) using Llama 3.1 8B (L3.1-8B) and Llama 3.1 70B (L3.1-70B). We evaluate the models on precision (P), recall (R), and F1 scores using alignment-based ROUGE-1 (AR-1), alignment-based ROUGE-2 (AR-2), and date F1-score metrics. The best scores for each experiment setting are boldfaced.

against the dates in the predicted timeline. Unlike alignment-based ROUGE, date F1-score performs hard matching of the dates and does not consider the event descriptions.

Results

We present our main experimental results in Table 4. Our findings indicate that our method significantly outperforms the baseline. The baseline method using Llama struggles to produce a coherent timeline. It often fails to determine when to stop and occasionally generates nonsensical outputs, especially with Llama-3.1 8B. Even when we use GPT-4o, our method still achieves better F1 scores than the baseline. However, note that GPT-4o may have a slight advantage over the other LLMs, as it was used in the dataset creation process.

We also observe that self-reflection significantly improves all scores (i.e., precision, recall, and F1) across all metrics (i.e., AR-1, AR-2, and date F1) in all experimental settings with Llama-3.1. With Llama-3.1 70B, when evaluated against ground-truth timelines without event filtering (all events), self-reflection improves the AR-1 F1 score by 2.82%, the AR-2 F1 score by 1.08%, and the date F1 score by 6.38%. When evaluated against filtered ground-truth timelines, self-reflection improves the AR-1 F1 score by 2.76%, the AR-2 F1 score by 0.84%, and the date F1 score by 6.10%.

With the oracle retriever (Table 5), using Llama-3.1 8B, our method still significantly outperforms the baseline by 4.78%, 2.67%, and 3.50% on AR-1 F1, AR-2 F1, and date F1 respectively. The score improvements are even greater with the larger Llama-3.1 70B model, reaching 5.99%, 2.82%, and 12.98% on AR-1 F1, AR-2 F1, and date F1 respectively.

The baseline method is impractical for real-world applications where hundreds of thousands of news articles are published each month. Regardless of the context window size, it cannot keep up with the speed and volume of information flowing through the internet. We have shown that even with the same set of articles (in the oracle retriever setup), our method is superior. Furthermore, the baseline method is unsuitable for online (streaming) processing. Every time a new article is added, previous articles need to be reprocessed to update the timeline, leading to significant computational inefficiency.

Conclusion

In this paper, we propose a new task with high relevance to current needs, called constrained timeline summarization. We present a new test set for the task (CREST), which was built by generating the constraints using human-in-the-loop collaboration with an LLM, hiring annotators to annotate the adherence of the events in the ground-truth timeline to the constraints, and filtering the events without supporting articles by utilizing an LLM.

We also propose an effective method that utilizes LLMs for the task. Our method does not require any training and can work with different LLMs. Our method works by summarizing the articles according to the constraint, employing a self-reflection procedure to filter out irrelevant summaries, clustering the summaries that describe the same event, and selecting the top l clusters and the top k sentences for each cluster to generate the constrained timeline.

Lastly, we demonstrate the effectiveness of our method by comparing it against a baseline method that generates the timeline directly by concatenating all the articles into its input prompt. We show that our method successfully outperforms the baseline on all metrics. Similarly, we demonstrate the effectiveness of self-reflection by comparing our method to a variant of our method that does not employ self-reflection, and show that self-reflection effectively improves the F1 scores on all metrics. With this work, we hope that constrained timeline summarization can gain more attention and more progress can be achieved on this task in future.

Acknowledgments

This work is fully supported by the Advanced Research and Technology Innovation Centre (ARTIC), the National University of Singapore under Grant (project number: ELDT-RP1).

References

Allan, J.; Gupta, R.; and Khandelwal, V. 2001. Temporal summaries of new topics. In *Proceedings of SIGIR*, 10–18.

Cao, J.; Jiao, D.; Yan, Q.; Zhang, W.; Tang, S.; and Zhuang, Y. 2024. IDEAL: Leveraging infinite and dynamic characterizations of large language models for query-focused summarization. arXiv:2407.10486.

Chieu, H. L.; and Lee, Y. K. 2004. Query based event extraction along a timeline. In *Proceedings of SIGIR*, 425–432.

Chinchor, N.; Hirschman, L.; and Lewis, D. D. 1993. Evaluating message understanding systems: An analysis of the third Message Understanding Conference (MUC-3). *Computational Linguistics*, 19(3): 409–450.

Conroy, J. M.; Schlesinger, J. D.; and O'Leary, D. P. 2006. Topic-focused multi-document summarization using an approximate oracle score. In *Proceedings of COLING/ACL*, 152–159.

Dang, H.; and Owczarzak, K. 2009. Overview of the TAC 2008 update summarization task. In *Proceedings of TAC*.

Duan, Y.; Jatowt, A.; and Yoshikawa, M. 2020. Comparative timeline summarization via dynamic affinity-preserving random walk. In *Proceedings of ECAI*, 1778–1785.

Ghalandari, D. G.; and Ifrim, G. 2020. Examining the state-of-the-art in news timeline summarization. In *Proceedings of ACL*, 1322–1334.

Gilardi, F.; Alizadeh, M.; and Kubli, M. 2023. ChatGPT outperforms crowd workers for text-annotation tasks. *Proceedings of the National Academy of Sciences of the United States of America*, 120(30).

Hamborg, F.; Meuschke, N.; and Gipp, B. 2018. Bias-aware news analysis using matrix-based news aggregation. *International Journal on Digital Libraries*, 21: 129–147.

Hu, Q.; Moon, G.; and Ng, H. T. 2024. From moments to milestones: Incremental timeline summarization leveraging large language models. In *Proceedings of ACL*, 7232–7246.

Jiang, P.; Xiao, C.; Wang, Z.; Bhatia, P.; Sun, J.; and Han, J. 2024. TriSum: Learning summarization ability from large language models with structured rationale. In *Proceedings of NAACL*, 2805–2819.

Krishna, R. V. V. M.; Kumar, S. Y. P.; and Reddy, C. S. 2013. A hybrid method for query based automatic summarization system. *International Journal of Computer Applications*, 68(6): 39–43.

La Quatra, M.; Cagliero, L.; Baralis, E.; Messina, A.; and Montagnuolo, M. 2021. Summarize dates first: A paradigm shift in timeline summarization. In *Proceedings of SIGIR*, 418–427.

Laskar, M. T. R.; Hoque, E.; and Huang, X. 2020. Query focused abstractive summarization via incorporating query relevance and transfer learning with transformer models. In *Proceedings of Canadian AI*, 342–348.

Lewis, M.; Liu, Y.; Goyal, N.; Ghazvininejad, M.; Mohamed, A.; Levy, O.; Stoyanov, V.; and Zettlemoyer, L. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of ACL*, 7871–7880.

Li, M.; Ma, T.; Yu, M.; Wu, L.; Gao, T.; Ji, H.; and McKeown, K. 2021. Timeline summarization based on event graph compression via time-aware optimal transport. In *Proceedings of EMNLP*, 6443–6456.

Li, Z.; Zhang, X.; Zhang, Y.; Long, D.; Xie, P.; and Zhang, M. 2023. Towards general text embeddings with multi-stage contrastive learning. arXiv:2308.03281.

Liu, Y.; and Lapata, M. 2019. Text summarization with pretrained encoders. In *Proceedings of EMNLP*, 3730–3740.

Llama Team, A. 2024. The Llama 3 herd of models. arXiv:2407.21783.

Mani, I.; and Bloedorn, E. 1998. Machine learning of generic and user-focused summarization. In *Proceedings of AAAI*, 821–826.

Martschat, S.; and Markert, K. 2017. Improving ROUGE for timeline summarization. In *Proceedings of EACL*, 285–290.

Martschat, S.; and Markert, K. 2018. A temporally sensitive submodularity framework for timeline summarization. In *Proceedings of CoNLL*, 230–240.

Mihalcea, R.; and Tarau, P. 2004. TextRank: Bringing order into text. In *Proceedings of EMNLP*, 404–411.

Mohamed, A. A.; and Rajasekaran, S. 2006. Improving query-based summarization using document graphs. In *Proceedings of ISSPIT*, 408–410.

OpenAI. 2024. GPT-4 technical report. arXiv:2303.08774.

Ouyang, Y.; Li, W.; Li, S.; and Lu, Q. 2011. Applying regression models to query-focused multi-document summarization. *Information Processing and Management*, 47(2): 227–237.

Park, C.; and Ko, Y. 2022. QSG Transformer: Transformer with query-attentive semantic graph for query-focused summarization. In *Proceedings of SIGIR*, 2589–2594.

Rahman, N.; and Borah, B. 2015. A survey on existing extractive techniques for query-based text summarization. In *Proceedings of ISACC*, 98–102.

Riezler, S.; and Maxwell, J. T. 2005. On some pitfalls in automatic evaluation and significance testing for MT. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, 57–64.

Steinberger, J.; and Ježek, K. 2009. Update summarization based on novel topic distribution. In *Proceedings of DocEng*, 205–213.

Strötgen, J.; and Gertz, M. 2015. A baseline temporal tagger for all languages. In *Proceedings of EMNLP*, 541–547.

Touvron, H.; Martin, L.; Stone, K.; Albert, P.; Almahairi, A.; Babaei, Y.; Bashlykov, N.; Batra, S.; Bhargava, P.; Bhosale, S.; Bikel, D.; Blecher, L.; Ferrer, C. C.; Chen, M.; Cucurull, G.; Esiobu, D.; Fernandes, J.; Fu, J.; Fu, W.; Fuller, B.; Gao, C.; Goswami, V.; Goyal, N.; Hartshorn, A.; Hosseini, S.; Hou, R.; Inan, H.; Kardas, M.; Kerkez, V.; Khabsa, M.; Kloumann, I.; Korenev, A.; Koura, P. S.; Lachaux, M.-A.; Lavril, T.; Lee, J.; Liskovich, D.; Lu, Y.; Mao, Y.; Martinet, X.; Mihaylov, T.; Mishra, P.; Molybog, I.; Nie, Y.; Poulton, A.; Reizenstein, J.; Rungta, R.; Saladi, K.; Schelten, A.; Silva, R.; Smith, E. M.; Subramanian, R.; Tan, X. E.; Tang, B.; Taylor, R.; Williams, A.; Kuan, J. X.; Xu, P.; Yan, Z.; Zarov, I.; Zhang, Y.; Fan, A.; Kambadur, M.; Narang, S.; Rodriguez, A.; Stojnic, R.; Edunov, S.; and Scialom, T. 2023. Llama 2: Open foundation and fine-tuned chat models. arXiv:2307.09288.

Tran, G.; Herder, E.; and Markert, K. 2015. Joint graphical models for date selection in timeline summarization. In *Proceedings of ACL*, 1598–1607.

Wang, W.; Li, S.; Li, J.; Li, W.; and Wei, F. 2013. Exploring hypergraph-based semi-supervised ranking for query-oriented summarization. *Information Sciences*, 237: 271–286.

Weng, Y.; Zhu, M.; Xia, F.; Li, B.; He, S.; Liu, S.; Sun, B.; Liu, K.; and Zhao, J. 2023. Large language models are better reasoners with self-verification. In *Findings of EMNLP*, 2550–2575.

Xie, Y.; Kawaguchi, K.; Zhao, Y.; Zhao, J. X.; Kan, M.-Y.; He, J.; and Xie, M. Q. 2024. Self-evaluation guided beam search for reasoning. In *Proceedings of NeurIPS*, 41618–41650.

Xu, R.; Wang, S.; Liu, Y.; Wang, S.; Xu, Y.; Iter, D.; He, P.; Zhu, C.; and Zeng, M. 2023. LMGQS: A large-scale dataset for query-focused summarization. In *Findings of EMNLP*, 14764–14776.

You, J.; Li, D.; Kamigaito, H.; Funakoshi, K.; and Okumura, M. 2022. Joint learning-based heterogeneous graph attention network for timeline summarization. In *Proceedings of NAACL*, 4091–4104.

Yu, Y.; Jatowt, A.; Doucet, A.; Sugiyama, K.; and Yoshikawa, M. 2021. Multi-TimeLine Summarization (MTLS): Improving timeline summarization by generating multiple summaries. In *Proceedings of ACL*, 377–387.

Appendix

Annotation Instruction

Before performing the event-constraint annotation, all annotators are required to read the annotation guidelines provided in Table 6. They will then perform the annotation by answering "yes" or "no" for each event-constraint pair, indicating whether the event adheres to the constraint. The annotation interface is shown in Figure 3.

Instructions

The goal of this annotation is to identify events in a timeline that satisfy certain constraints.

1. For an event in the timeline and each constraint (shown on Google Form), please click the 'Yes' button if the event meets the constraint, and 'No' if it does not.
2. The events in the timeline are real events reported by journalists about public figures or institutions.
3. We try to make the constraints as clear as possible, but sometimes they may require common knowledge (e.g., Obama was the president of the United States or California is in the United States). Please annotate the events based on your best knowledge, without making any unnecessary assumptions.
4. Annotating the events should not require internet searches, but you are free to do so if needed.
5. An event may satisfy more than one constraint (Example 2) or none at all (Example 3).
6. Please refer to the examples below to see how some sample events are annotated in relation to each of five constraints.

Table 6: Annotation instruction.

Experimental Details

Implementation Details

We use the vllm library[8] (version 0.5.4) to serve the local language models (Llama 3.1 8B and Llama 3.1 70B) and OpenAI Python API[9] library (version 0.28.1) to communicate with the language models. The GPT-4o model used in our experiments is `gpt-4o-2024-08-06`.

Compute Resource

We deploy Llama 3.1 8B on 2x NVIDIA A100 80GB GPUs (server #1 in Table 8) and Llama 3.1 70B on 4x NVIDIA H100 80GB GPUs (server #2 in Table 8). All experimental code was run on a server with 2x AMD EPYC 9554P CPUs and 792 GB of memory (server #1).

Prompts

We provide the prompt templates for our experiments in the tables below.

Summary Generation Table 10

Self-Reflection Table 11

Event Similarity Table 12

Baseline method Table 13

Hyper-Parameters

Our method does not require any training or fine-tuning, so we only need to set the decoding hyper-parameters for the language model. For clustering, the only hyper-parameter is determining the number of nearby events to retrieve from the vector database. We used the hyper-parameters of LLM-TLS (Hu, Moon, and Ng 2024) and did not perform a hyper-parameter search ourselves.

Hyper-parameter	Value
Temperature	0.
Top-p	1.
Max. tokens	
- Summary generation	256
- Self-reflect	256
- Event similarity	2
N	20

Table 7: The final hyper-parameters of our method. "Max. tokens" refers to the maximum number of tokens generated by the large language model, and N refers to the number of neighboring events retrieved during clustering.

Additional Results

To ensure reproducibility, we ran the large language model with the decoding temperature set to 0. We conducted our experiments over three trials using three different few-shot examples for clustering, which are provided as JSON files in our code and can be loaded with the `--few_shot` argument when running the code. For the baseline method, we set the random seed for sampling the articles by specifying the `--seed` argument in the code. We report the average and standard deviation in Table 9.

We evaluated the statistical significance of our results using approximate randomization (Riezler and Maxwell 2005; Chinchor, Hirschman, and Lewis 1993) with 100 trials. Notably, we found that REACTS (with self-reflection) demonstrated statistically significant improvements ($p < 0.05$) compared to REACTS without self-reflection and the BASELINE method across all experimental settings.

[8] https://github.com/vllm-project/vllm

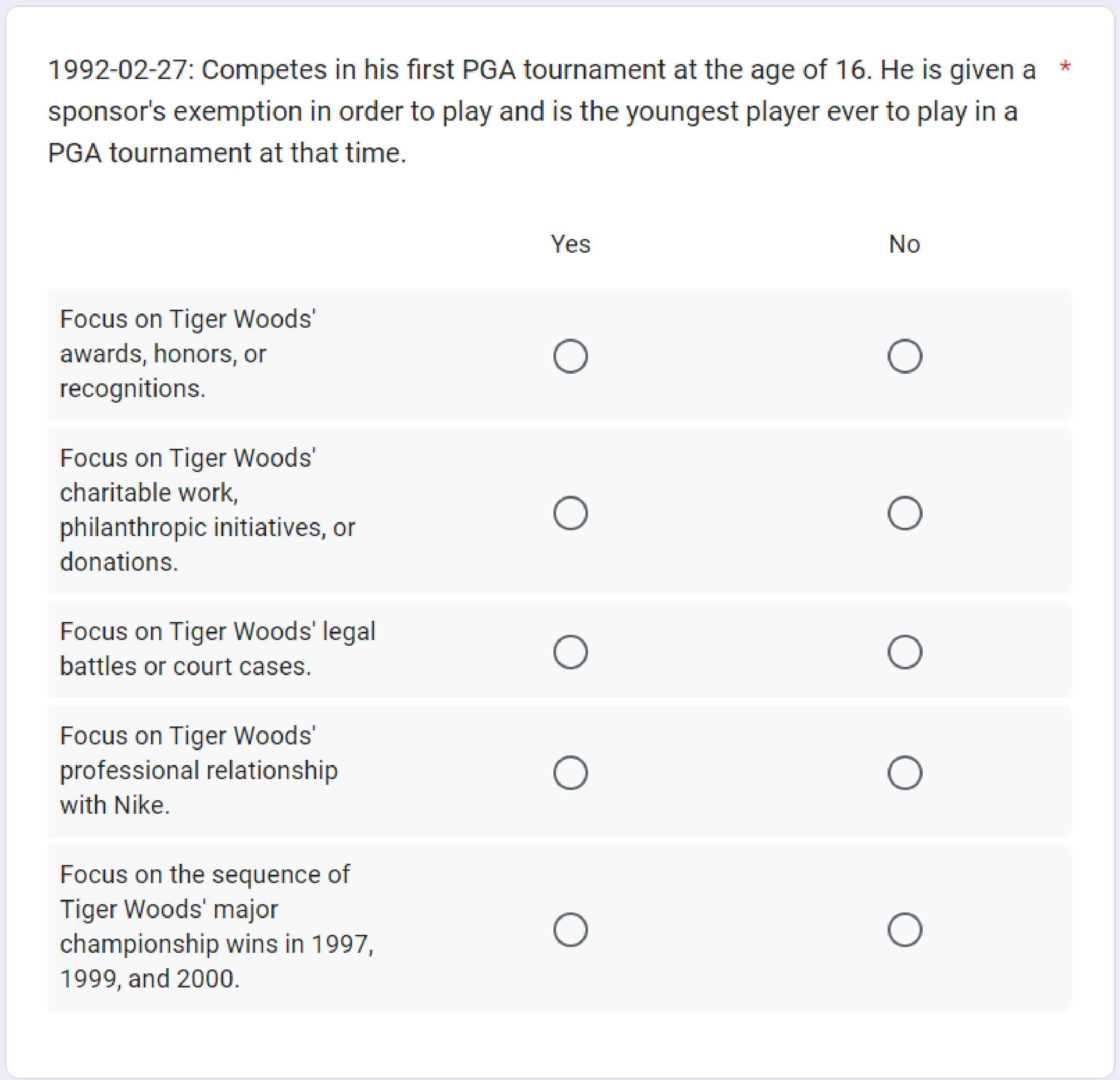
[9] https://github.com/openai/openai-python

Server	GPU	CPU	RAM
#1	4x NVIDIA A100 80GB	2x AMD EPYC 9554P	792 GB
#2	4x NVIDIA H100 80GB	2x INTEL XEON 8562Y+	528 GB

Table 8: Specifications of the servers used in our experiments. Server #1 has 4x NVIDIA A100 80GB GPUs, but we used only two GPUs throughout our experiments.

Model	**LLM**	**AR-1**	**AR-2**	**Date F1**
All events				
REACTS w/o SR	L3.1-8B	0.0575 ± 0.0012	0.0271 ± 0.0007	0.1415 ± 0.0018
REACTS	L3.1-8B	0.0770 ± 0.0004	0.0351 ± 0.0004	0.1761 ± 0.0004
REACTS w/o SR	L3.1-70B	0.0804 ± 0.0005	0.0388 ± 0.0005	0.1771 ± 0.0004
REACTS	L3.1-70B	**0.1074** ± 0.0015	**0.0488** ± 0.0011	**0.2407** ± 0.0006
Filtered events				
BASELINE	L3.1-8B	0.0248 ± 0.0008	0.0047 ± 0.0013	0.0857 ± 0.0110
REACTS w/o SR	L3.1-8B	0.0738 ± 0.0007	0.0339 ± 0.0002	0.1767 ± 0.0019
REACTS	L3.1-8B	0.0976 ± 0.0007	0.0456 ± 0.0006	0.2233 ± 0.0010
BASELINE	L3.1-70B	0.0871 ± 0.0068	0.0407 ± 0.0018	0.1866 ± 0.0102
REACTS w/o SR	L3.1-70B	0.1035 ± 0.0006	0.0484 ± 0.0004	0.2307 ± 0.0014
REACTS	L3.1-70B	**0.1311** ± 0.0006	**0.0566** ± 0.0005	**0.2923** ± 0.0013

Table 9: Average and standard deviation ($\bar{x} \pm \sigma$) of the F1 scores from our main experimental results.



1992-02-27: Competes in his first PGA tournament at the age of 16. He is given a sponsor's exemption in order to play and is the youngest player ever to play in a PGA tournament at that time. *

	Yes	No
Focus on Tiger Woods' awards, honors, or recognitions.	○	○
Focus on Tiger Woods' charitable work, philanthropic initiatives, or donations.	○	○
Focus on Tiger Woods' legal battles or court cases.	○	○
Focus on Tiger Woods' professional relationship with Nike.	○	○
Focus on the sequence of Tiger Woods' major championship wins in 1997, 1999, and 2000.	○	○

Figure 3: Example of the annotation interface for the event-constraint pairs in Tiger Woods's timeline.

```
### Instruction
Review the news article associated with the provided keyword and constraint. If the article's content does not relate to the keyword and specified constraint, output 'None'. Otherwise, summarize the most significant event related to the keyword while adhering to the constraint.

### Format
YYYY-MM-DD: One-sentence Summary

#################
### Keyword
Stephen King

### Constraint
Focus on Stephen King's book releases.

### Content
(example article about the release of the TV series adaptation of "Lisey's story", based on Stephen King's book)

### Related Event Summary
None.

#################
### Keyword
Stephen King

### Constraint
Focus on Stephen King's involvement in television and streaming projects.

### Content
(example article about the release of the TV series adaptation of "Lisey's story", based on Stephen King's book)

### Related Event Summary
2021-06-04: The miniseries "Lisey's Story," adapted by King and based on his 2006 novel of the same name, premieres on Apple TV+.

#################
### Keyword
{topic keyword}

### Constraint
{constraint}

### Content
{content}

### Related Event Summary
```

Table 10: Prompt template for summary generation.

```
Review the timestamped event description related to keyword, accompanied by a constraint. Please determine whether the event description complies with or corresponds to the constraint. Respond with 'Yes' if the event description aligns with the constraint, or with 'No' if it does not.
#################
{positive example}

#################
{negative example}

#################
### Event
{event}

### Constraint
{constraint}
### Answer
```

Table 11: Prompt template for self-reflection.

```
Taking the timestamps into account, evaluate whether two prior news events are referring to the same event related to the keyword. If the two events occur on the same date, and they are about the same topic related to the keyword, then they should be considered as referring to the same event. If so, please respond directly with 'yes'. If not, respond with 'no'. Then explain your answer.
----
{example #1}
----
{example #2}
----
{example #3}
----
# Keyword
{topic keyword}
# Event 1
{event 1}
# Event 2
{event2}
# Answer
```

Table 12: Prompt template for event similarity checking.

```
{article #1}
#################
...
{article #n}
#################
### Instruction
Using the articles about {topic keyword} above, please create a concise timeline with {l} events following the constraint below. Using only the information from the articles, provide the date and a {k}-sentence summary for each important event.

### Constraint
{constraint}

### Format
YYYY-MM-DD: One-sentence Summary
YYYY-MM-DD: One-sentence Summary

### Answer
```

Table 13: Prompt template for the baseline method.