Cascading Large Language Models for Salient Event Graph Generation

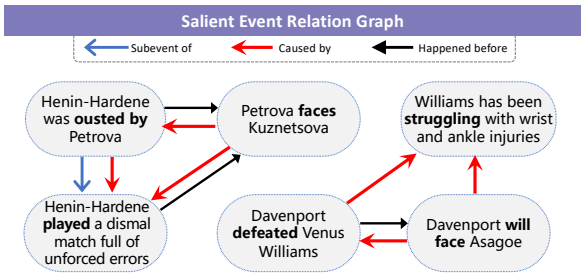
Xingwei Tan[1,2], **Yuxiang Zhou**[2], **Gabriele Pergola**[1], **Yulan He**[1,2,3]
[1]Department of Computer Science, University of Warwick, UK
[2]Department of Informatics, King's College London, UK
[3]The Alan Turing Institute, UK
{Xingwei.Tan, Gabriele.Pergola.1}@warwick.ac.uk
{Yuxiang.Zhou, Yulan.He}@kcl.ac.uk

Abstract

Generating event graphs from long documents is challenging due to the inherent complexity of multiple tasks involved such as detecting events, identifying their relationships, and reconciling unstructured input with structured graphs. Recent studies typically consider all events with equal importance, failing to distinguish salient events crucial for understanding narratives. This paper presents CALLMSAE, a CAscading Large Language Model framework for SAlient Event graph generation, which leverages the capabilities of LLMs and eliminates the need for costly human annotations. We first identify salient events by prompting LLMs to generate summaries, from which salient events are identified. Next, we develop an iterative code refinement prompting strategy to generate event relation graphs, removing hallucinated relations and recovering missing edges. Powered by CALLMSAE, we present *NYT-SEG*, a large-scale automatically annotated event graph dataset which can serve as distant supervision signals. Fine-tuning contextualised graph generation models on *NYT-SEG* outperforms the models trained on CAEVO data. Results on a human-annotated test set show that the proposed method generates salient and more accurate graphs, outperforming competitive baselines[1].

1 Introduction

Events are fundamental discourse units which form the backbone of human communication. They are interconnected through various event relations such as hierarchical, temporal, or causal relations. Event relation graphs are vital for representing and understanding complex event narratives, with nodes representing events and edges denoting relationships between them. High-quality event relation graphs can enhance numerous downstream tasks, such as question answering (Lu et al., 2022) and reasoning (Melnyk et al., 2022).



NYT Article: «*Top-Seeded Henin-Hardenne Surprised by No. 14 Petrova*»

After **Lindsay Davenport defeated Venus Williams**, 7-5, 6-4, in a match that was ho-hum until the last game, **Nadia Petrova surprised top-seeded Justine Henin-Hardenne**, [...].

.....

Now she can take the whole week off. **Henin-Hardenne, the defending Open champion, made 30 unforced errors**, double-faulted four times and hit just 14 winners.

.....

Petrova will meet Svetlana Kuznetsova in the quarterfinals.

.....

In what could be a letdown for Davenport, **she will play her quarterfinal match against unseeded Shinobu Asagoe** of Japan. Asagoe, who has never won a singles title, defeated No. 29 Eleni Daniilidou of Greece, 7-6 (4), 4-6, 6-3. **Williams, trying to recover from wrist and ankle injuries**, said: "I can't be hard on myself.[..]".

Figure 1: An example of salient event relation graph (top) generated from the NYT article (bottom).

Existing studies on contextualized event graph generation primarily focus on fine-tuning language models for end-to-end generation of linearized graphs from documents (Madaan et al., 2021; Tan et al., 2024). These methods rely on distant supervision, such as events and event temporal relations detected using CAEVO (McDowell et al., 2017), due to the data-intensive nature of language models and the significant manual effort required for annotating event graphs. However, CAEVO's bottom-up, predicate-centric extraction often results in sparse, low-quality event graphs populated with non-salient events. For example, CAEVO frequently identifies all verbs as events, including trivial ones like "*say*" and "*think*," which have minimal connections to other events and contribute little to narrative understanding, thus introducing noise. In recent research involving human annotation (Dunietz et al., 2014; Liu et al., 2018), a *summarisation test* is leveraged to identify salient events or entities, where an event or entity is considered salient

[1]Code and data: https://github.com/Xingwei-Tan/CALLMSAE.

Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers), pages 2223–2245

if a human-written summary is likely to include it.

Motivated by these insights, we aim to improve the quality of distant supervision and mitigate noise in event graph generation by incorporating event saliency. Our hypothesis is that effective graph generation benefits from a top-down strategy, where the main events are identified first, rather than extracted solely in a bottom-up manner based on the predicate. Capitalizing on the powerful summarization capabilities of large language models (LLMs), we propose a method that first instructs LLMs to summarize documents before identifying salient events. Specifically, we introduce a code-prompting strategy for constructing event relation graphs encompassing hierarchical, temporal, and causal relations[2] (see Figure 1). Unlike vanilla prompting, which queries each potential event pair individually, our code prompt format generates each relation type in a single pass. Furthermore, we incorporate an iterative refinement process using a *hallucination grader* to filter spurious edges and iterative generation to recover missing ones. Finally, the abstractive nature of salient events presents an evaluation challenge, as they rarely match gold standard events exactly, even when semantically equivalent. To address this, we propose evaluation metrics based on semantic text embeddings for assessing the generated event relation graphs.

Our experimental results on the New York Times corpus (Sandhaus and Sandhaus, 2008) show that our proposed CALLMSAE, a novel CAscading Large Language Model framework for SAlient Event graph generation, outperforms the baselines in terms of event saliency and edge quality. The fine-tuned model surpasses previous models trained with CAEVO-generated graphs. Our contributions are summarised as follows:

- We propose CALLMSAE, a CAscading Large Language Model framework for SAlient Event graph generation, serving as a distant signal generator for contextualised graph generation models.
- We propose a novel contextualised evaluation metric for comparing salient event graphs. Our extensive experimental evaluation on LLM-generated event relation graphs in terms of event saliency and event relation on the NYT corpus, demonstrating how higher quality salient event graphs can improve contextualised graph generation.
- We provide a large-scale LLM-generated salient event graph dataset *NYT-SEG* with three major event relation types for distant supervision (10,231 documents), along with a human-annotated test set (100 documents).

[2]We extend beyond the CAEVO's temporal-only relations to encompass multiple relation types.

2 Related Work

Event Relation Graph Construction The early idea of event relation graph construction comes from UzZaman et al. (2013), which introduces a dataset for evaluating an end-to-end system which takes raw text as input and output TimeML annotations (i.e., temporal relations). CAEVO (McDowell et al., 2017) and Cogcomptime (Ning et al., 2018) both utilise a wide range of manually designed features to train MaxEnt and averaged perception for extracting events and relations. Han et al. (2019b) proposed a joint event and relation extraction model based on BERT (Devlin et al., 2019) and BiLSTM (Panchendrarajan et al., 2018). Other researchers focus on developing specialised sub-systems to classify extracted event pairs for relations (Ning et al., 2019; Han et al., 2019a; Wang et al., 2020; Pergola et al., 2021a,b; Tan et al., 2021, 2023). ATOMIC (Sap et al., 2019) is a large-scale commonsense knowledge graph containing the causes and effects of events. MAVEN-ERE (Wang et al., 2022) is built with event coreference, temporal, causal and subevent relations. However, ATOMIC and MAVEN-ERE completely rely on crowdsourcing and thus are difficult to extend. MAVEN-ERE is less than half the size of our dataset and does not consider the saliency of events.

Madaan et al. (2021) fine-tune GPT-2 to generate linearised graphs from documents in an end-to-end manner. Their temporal relation graphs used for training are produced by CAEVO. Following this direction, Tan et al. (2024) instead view the task as set generation and propose a framework based on set property regularisation and data augmentation. In this paper, we focus on generating multi-relation graphs via in-context learning, prompt interaction, and iterative refinement.

Salient Event Identification Several existing papers investigate the problem of identifying salient events. Choubey et al. (2018) build a rule-based classifier to identify central events by exploiting human-annotated event coreference relations. They find the central events either have large numbers of coreferential event mentions or have large stretch

sizes. Jindal et al. (2020) propose a contextual model to identify salient events based on BERT and BiLSTM. They also mention several features, such as event trigger frequency, which are essential features to identify the salient events. Liu et al. (2018) propose a feature-based method using LeToR (Liu et al., 2009) and a neural-based method called Kernel-based Centrality Estimation. To train and evaluate their methods, they build a dataset based on the *summarisation test*: an event is considered salient if a summary written by a human is likely to include it. Zhang et al. (2021) combine the Kernel-based Centrality Estimation with the event and temporal relation extraction model of Han et al. (2019b) to build a salience-aware event chain modelling system. However, they only focus on single-dimensional chains and only model temporal relations.

3 Cascading LLMs to Generate Salient Event Graphs

CALLMSAE combines various prompts in a pipelined manner to generate salient event graphs. In this section, we introduce the prompts for generating salient events, and then the method for generating relation graphs based on the salient events. Lastly, we define an evaluation metric for comparing event graphs: *Hungarian Graph Similarity*.

3.1 Generate Salient Events

The *summarisation test* (Section 1) is often used to guide the annotation of salient events or entities (Dunietz et al., 2014; Liu et al., 2018; Lu et al., 2023; Lyu et al., 2024b). These studies identify events or entities included in human-written summaries as salient. Similarly, we instruct LLMs to generate a summary first and then extract events from it. Examples of the summarisation and salient event generation prompts are shown in the Appendix (Table 11 and 12).

3.2 Generate Graphs as Code Completion

While LLMs can extract salient events, they often struggle with identifying event relations (Chan et al., 2023; Sun et al., 2022, 2024; Tan et al., 2024). Prompt engineering for extracting event relations is complex due to the need to incorporate various terminologies and graph constraints. Moreover, prompt efficiency is crucial as generating a large-scale dataset with LLMs can still incur significant computational costs, albeit less than crowdsourcing.

In our method, the main prompt for generating the event relation graph is formulated as a Python code completion task. The graph is defined using the NetworkX[3] package in Python, with nodes representing the salient events generated in Section 3.1. LLMs are instructed to complete the code by adding relation edges using NetworkX's APIs.

Recent research suggests that formulating prompts as code can enhance LLMs' reasoning abilities (Wang et al., 2023; Zhang et al., 2023). In our task, the Python code format effectively incorporates all necessary terminologies, enabling LLMs to understand them without confusion. The Python code format also allows for the inclusion of constraints (e.g., ensuring the graph is a directed acyclic graph) and additional instructions (e.g., ask for explanations) as comments. LLMs can generate explanations as comments without disrupting the main content of the graph, which is difficult to achieve in other formats (e.g., JSON). Moreover, the code template simplifies parsing the response, as LLMs are directed to use the ".add_edge()" function to add the relations.

Since hierarchical, temporal, and causal relations are asymmetric, each can be represented by a Directed Acyclic Graph (DAG). We formulate three distinct prompts to guide LLMs in generating three DAGs, each representing one of these relation types. This approach avoids the complexity of a multi-label graph, and LLMs can focus on a single relation type and carefully consider the topological structure of the graph. We can also use the ".find_cycle()" function from NetworkX to detect constraint violations reliably. In addition, if relation types are interconnected, the initially generated graphs can help the generation of subsequent graphs (as will be explained in Section 3.4). We provide an example of the code prompt in Appendix (Table 13).

3.3 Iterative Refinement

Hallucination Grader The code prompt efficiently guides LLMs to generate graphs, but it often generates hallucinated relations. Based on our preliminary experiments, these hallucinations stem from the models' overconfidence in their relation predictions. Specifically, LLMs tend to infer event relations without explicit linguistic cues or strong evidence for logical inference. Consequently, LLMs predict far more relations than the gold standards, leading to low precision.

[3] https://networkx.org/documentation/stable/

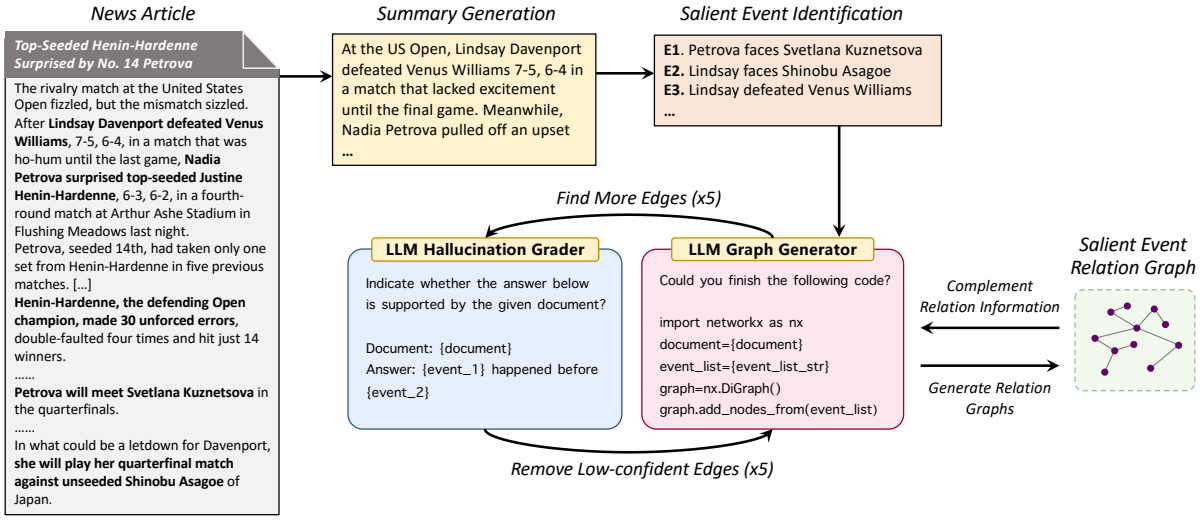


Figure 2: The proposed CALLMSAE framework.

Recent studies show that LLMs can evaluate and correct their own outputs (Madaan et al., 2023; Asai et al., 2024). Thus, we introduce a hallucination grader to address hallucination. For each relation edge generated, we pose a question to the LLMs to determine whether the relation is grounded in the given document. If the LLMs respond with a "yes", the edge is retained; otherwise, it is discarded. An example of the hallucination grader prompt is shown in Appendix (Table 14).

Recover Missing Edges The main benefit of the hallucination grader is that it increases precision by removing low-confident edges. However, this process inevitably reduces recall. To mitigate this side effect, we introduce an iterative refinement process. After discarding hallucinated edges, we reinsert the code block containing the relation edges into the graph generation prompt and ask the LLMs to complete the code again. In this way, the LLMs can reconsider whether there are any missing relations in the document, thereby improving recall.

Once the LLMs generate a new graph, the hallucination grader checks the relation edges again. This process is repeated for a fixed number of times. We set the maximum number of iterations to 5 in our experiments, as the LLMs stop discovering new edges after 2 or 3 iterations in most documents.

3.4 Complement Relation Types

Hierarchical, temporal, and causal relations are not independent of each other. In this case, we found that providing the graph for the first relation can benefit the generation of the dependent relation's graph. Specifically, we predict the hierarchical relation graph first. Then, we provide this graph to the LLMs and ask them to generate the temporal relation graph. Lastly, with both the hierarchical and temporal relation graphs available, the LLMs predict the causal relation graph.

The hierarchical relation describes two closely related events at different granularity levels. It focuses on the inherent semantics of the events and does not depend on other relation types. For example, "*writing a dissertation*" is a subevent of "*doing a PhD*". Therefore, we choose to predict the hierarchical relations first. Temporal relations can depend on hierarchical relations. For example, knowing "*doing a PhD*" happened before "*being prompted to Professor*" allows us to deduce that "*writing a thesis*" also happened before "*being prompted to Professor*". Thus, we predict temporal relations after hierarchical relations. Lastly, causal relations depend on both hierarchical and temporal relations, as the antecedent event in a causal relation must occur before the consequence. Therefore, the causal relation is predicted in the last step. For more details about the entire prompting process, please refer to the pseudocode in Appendix D.

3.5 Hungarian Graph Similarity

It is challenging to compare event relation graphs generated by LLMs due to the abstractive nature of generation, making it difficult to align the generated events with the gold standard events (Li et al., 2023). Moreover, salient events are often high-level and abstract rather than fine-grained and concrete, which means some variations in wording are not only acceptable but also expected. Instead of using exact matching (Zhao et al., 2024) or rule-based

token matching (Tan et al., 2024) on events and relations to calculate F_1, adopting semantic-based evaluation metrics is more reasonable and fair. As more tasks adopt text generation frameworks, many researchers are also turning to metrics based on language models rather than traditional token matching metrics like ROUGE and BLUE (Goyal et al., 2022; Pratapa et al., 2023; Lyu et al., 2024a).

In this study, we propose a novel metric for evaluating LLM-generated event graphs, called **Hungarian Graph Similarity (HGS)**. The metric is based on the Hungarian assignment algorithm (Kuhn and Kuhn, 1955), which is widely used in object detection to match generated objects and target objects (Carion et al., 2020). It can find the optimal assignment given a cost matrix containing the distance between elements in two lists of objects. We adapt this algorithm to match predicted edges with edges in the gold standard graphs as follows:

1. Encode the events using SFR-Embedding-Mistral (Meng et al., 2024).

2. Given two edges of the same relation type, let $\bar{e}_1^h, \bar{e}_1^t$ be the embeddings of the head event and the tail event in the first edge. Let $\bar{e}_2^h, \bar{e}_2^t$ be the embeddings of the head and tail events in the second edges. We define the distance between the edges as $\max\big(D_{cos}(\bar{e}_1^h, \bar{e}_2^h), D_{cos}(\bar{e}_1^t, \bar{e}_2^t)\big)$, where $D_{cos}(\cdot, \cdot)$ is the cosine distance.

3. Build a cost matrix by computing the distance between every edge pair in the gold and predicted edge sets. Pad the matrix to a square matrix with the maximum cost value of 1.

4. Apply the Hungarian algorithm to the cost matrix to get the minimal cost value. The final score is $1 -$ cost value, making the value more intuitive (higher is better). To compute the HGS over all the documents, we weight the scores by the number of gold edges to obtain an average value.

In step 2, we take the maximum value of the distances between head and tail events because relation edges are considered matched only if both the head and tail events match.

For more detailed analysis, we define precision-oriented HGS and recall-oriented HGS. We match edges without padding the cost matrix in step 3 to obtain the total cost values of all matched edge pairs. Then, the total matched similarity is the number of matched edges minus the total cost. **Precision-oriented HGS** is computed by dividing the total matched similarity by the total number of predicted edges. **Recall-oriented HGS** is computed by dividing the total matched similarity by the total number of edges in the target graph.

4 Dataset

In this section, we describe how we construct the *NYT-SEG* which includes LLM-generated distant training pairs and a human-annotated dataset based on the New York Times (NYT) corpus.

4.1 Document Selection

We follow the same procedures as in (Tan et al., 2024) to select documents from the NYT corpus, one of the largest news datasets, with additional filtering based on document length. We select $10,347$ documents based on their descriptors indicating they are related to event narratives instead of opinions and discussions, such as sports and international politics. Among them, 100 documents are randomly sampled as the test set to be annotated by humans. More details about data selection are shown in Appendix A.1.

4.2 Annotation Settings

We recruited annotators from Prolific[4]. There are two subtasks: *salient event identification* and *event relation identification*. In the first subtask, the participants are asked to identify the salient event triplets: *actor*, *trigger*, and *object* (optional). We provide the definition of an event and several examples in the guidelines. They are instructed to do the summarisation test: the salient events should be the events they would include in the summary of the given article. Moreover, we provide some prominent features for helping annotators to identify salient events (Choubey et al., 2018; Jindal et al., 2020):

- Frequency: salient events are frequently mentioned in the articles.
- First appearance: salient events are often mentioned at the beginning of the article.
- Stretch size: salient events are often mentioned throughout the articles. Stretch size is the distance between the location where the event is first mentioned and last mentioned. A salient event usually has a large stretch size.

[4] http://www.prolific.com

In the second stage, we ask participants to identify relation triplets: *a source event*, *a relation type*, and *a target event*. Both the source and target events should be among the salient events identified in the first stage. In the guideline, we define three relation types: *happened_before*, *caused_by*, and *is_subevent_of*. *happened_before* indicates that the source event happened earlier than the target event. *caused_by* means the source event would not have happened if the target event did not happen. *is_subevent_of* signifies that the source event is a subevent of the target event. Annotators were informed that relations would be either explicitly mentioned in the article or inferred based on evidence within the article. Further details about the guidelines and user interface can be found in Appendix A.4.

4.3 Inter Annotator Agreement

Identifying salient events and event relations is complicated and time-consuming. We found it challenging to educate participants about these concepts because, in daily life, the meanings of events and relations differ from their definitions in the field of information extraction. Moreover, the technical definitions are much less intuitive to those outside the academic field. As a result, thorough training of participants is important to obtain high-quality annotations.

In total, we recruited 3 annotators to annotate 100 documents. Due to their varying availability, annotator 1 and 2 each annotated 45 documents, while annotator 3 annotated 20 documents. Among these, 5 documents were annotated by all three annotators. Following prior research in information extraction (Gurulingappa et al., 2012; Zhao et al., 2024), we used F_1 to measure the inter-annotator agreement on these 5 documents. To compute inter-annotator agreement, events or relations identified by one annotator are represented as set S_1. Another annotator's annotation S_2 serves as a pseudo-reference to compute precision $= \frac{|S_1 \cap S_2|}{|S_1|}$, recall $= \frac{|S_1 \cap S_2|}{|S_2|}$, and the F_1 score $= \frac{2|S_1 \cap S_2|}{|S_1|+|S_2|}$.

Table 1 shows the agreement scores for stages 1 and 2. Identifying salient events is subjective, which makes it difficult to reach a complete agreement. Moreover, event relation identification is even more subjective and dependent on the previous stage, leading to less unanimous agreement.

Annotator	Stage 1	Stage 2
1 & 2	0.838	0.676
1 & 3	0.771	0.645
2 & 3	0.847	0.710
Average	0.819	0.677

Table 1: Inter-annotator agreement measured by F_1.

4.4 Dataset Statistics

Table 2 shows the distributions of the relation types after applying the transitive closure to the annotated graphs. *happened_before* emerges as the most frequent relation type, reflecting the predominant focus on temporal sequences in news articles, and they are relatively straightforward to identify. Conversely, *caused_by* is the least frequent as it is the most challenging to identify.

Relation Type	Number
happened_before	310
caused_by	202
is_subevent_of	245
Total	757

Table 2: The distributions of the relation types.

5 Experiments

5.1 Model Settings

We compare against the following baselines:

- CAEVO (McDowell et al., 2017) is a pipeline system based on a MaxEnt classifier and manual features for extracting events and relations.
- Madaan et al. (2021) trained language models on CAEVO-generated linearised graphs with LM objective. We implemented their method to train a Flan-T5 model.
- Tan et al. (2024) also trained language models on CAEVO-generated graphs, but applied data augmentations and regularisations to mitigate the set element misalignment issue. We applied their method to train a Flan-T5.
- Han et al. (2019b) proposed a joint event and temporal relation extraction model. We adapted the model to hierarchical and causal relations by training on MAVEN-ERE. We also replaced BERT with Longformer (Beltagy et al., 2020) to make it suitable for long documents.
- GPT-4 (OpenAI, 2024) and GPT-3.5 are based on the generative pre-train framework.

	Mean event number	Event Frequency ↑	First Appearance ↓	Stretch Size ↑
CAEVO	34.71	0.05	0.46	0.07
Human	8.26	**0.11**	0.31	**0.20**
GPT-4	6.49	0.09	0.37	0.18
Llama3	5.17	0.09	**0.30**	0.19
Mixtral	10.60	0.10	0.33	**0.20**

Table 3: The average number of extracted events and the saliency features (in percentage values).

We used "gpt-4-1106-preview" and "gpt-3.5-turbo" respectively.

- MIXTRAL is an LLM based on the Mistral model and the mixture of expert framework. We used the 8x7B instruct (Jiang et al., 2024).
- LLAMA3 is an LLM based on the Llama framework. We used the 70B-instruct 8bit version for a balance between speed and performance.

We fine-tuned a Flan-T5-base (250M) with the relation graphs generated by CALLMSAE, following the same method as in Tan et al. (2024). The baseline prompt evaluates whether each event pair is supported by the document, akin to the hallucination grader described in Section 3.3. Thus, it serves as an ablation of our method without incorporating the code prompt. Another baseline, which asks for the relation type given an event pair, is also tested and included in our experiment.

CALLMSAE is designed to be model-agnostic. Due to budget constraints and the preliminary test results, we chose Llama3 as the backbone of all the prompt-based methods detailed in Table 5.

5.2 Event Saliency Evaluation

Table 3 shows the salient features (defined in Section 4.2, computation formulas in Appendix B) extracted from various backbone LLMs using summarisation prompts, alongside comparison with CAEVO and human annotations. The LLM-generated events are much more salient than CAEVO-generated events and exhibit similarity to human annotations.

We also use human annotations to evaluate the saliency. In the salient event identification annotation, we provide the events generated by CAEVO and Mixtral as candidate salient events. Note that only the top CAEVO events ranked in saliency features are shown. Half of the candidates are from CAEVO and the other half are from Mixtral. They are randomly shuffled and then shown to the annotators. We compute the precision, recall, and F_1 based on how the annotators select them. We also compute HGS using human-annotated salient events as references (Table 4). It is clear that although CAEVO extracted more events than Mixtral, many of them are not salient. Mixtral outperforms CAEVO significantly across all evaluation metrics.

	P	R	F_1	HGS
CAEVO	3.29	3.72	3.49	18.18
Mixtral	48.97	56.77	52.59	67.15

Table 4: Precision, recall, and F_1 based on the choices of the annotators. Hungarian graph similarity (HGS) is defined in Section 3.5. The values are in percentage.

5.3 Salient Event Relation Graph Evaluation

The salient event relation graph evaluation results are shown in Table 5. The compared methods (row 1 - 4) are outperformed by *Baseline Prompt* (row 5) on all relation types. *Baseline Prompt (rel type)* (row 6), which asks the model to generate all possible relation types given the event pair, performs slightly better than *Baseline Prompt*. However, *Baseline Prompt* and *Baseline Prompt (relation type)* are slow and costly because the number of prompts they need for building one graph is $O(n^2)$, where n is the number of events in the document. On the other hand, the time complexity of *Code Prompt* is $O(1)$. Moreover, *Code Prompt*'s overall HGS is significantly higher than *Baseline Prompt* and *Baseline Prompt (relation type)* on all relation types. *Baseline Prompt* check the event pairs more thoroughly and thus have higher recall but its precision is much lower. The complete CALLMSAE combines the code prompt and hallucination grader for iterative refinement, checking missing relations and verifying them to prevent hallucination. *Code Prompt (dependent rels)* (the 8th row) is an ablation of *CALLMSAE* (the 9 row), differing only in the

	Hierarchical			Temporal			Causal		
	PHGS	*RHGS*	*HGS*	*PHGS*	*RHGS*	*HGS*	*PHGS*	*RHGS*	*HGS*
Han et al. (2019b)	0.158	0.247	0.098	0.092	0.352	0.148	0.084	0.316	0.116
CAEVO	-	-	-	0.030	0.558	0.092	-	-	-
Madaan et al. (2021)	-	-	-	0.061	0.439	0.116	-	-	-
Tan et al. (2024)	-	-	-	0.126	0.335	0.187	-	-	-
Baseline Prompt	0.076	**0.651**	0.248	0.085	0.627	0.195	0.062	**0.657**	0.207
Baseline Prompt (rel type)	0.288	0.375	0.268	0.135	0.604	0.261	0.185	0.513	0.256
Code Prompt	0.174	0.559	0.315	0.153	**0.678**	0.283	0.121	0.632	0.272
Code Prompt (dependent rels)	N.A.	N.A.	N.A.	0.211	0.601	0.341	0.135	0.599	0.272
CALLMSAE (ours)	0.196	0.544	0.334	**0.294**	0.509	0.327	0.198	0.529	0.295
Fine-tuned T5 (CALLMSAE)	**0.314**	0.434	**0.339**	0.244	0.544	**0.362**	**0.366**	0.397	**0.343**

Table 5: The Hungarian Graph Similarity (HGS) of the LLM-generated graphs on the human-annotated NYT dataset. PHGS is precision-oriented HGS. RHGS is recall-oriented HGS. *Code Prompt (dependent rels)* means adding hierarchical graphs in the prompts for temporal graphs; and adding hierarchical and temporal for causal graphs. *Fine-tuned T5 (CALLMSAE)* means fine-tuning a flan-T5 using the graphs generated by CALLMSAE. All prompt-based methods (row 5 - 9) are based on *Llama3-70B-instruct*.

absence of iterative refinement. These results highlight the effectiveness of the hallucination grading approach, which effectively increases the precision and strikes a balance with recall.

In the temporal category, the results of *Code Prompt (dependent rels)* are obtained when provided with hierarchical graphs generated by CALLMSAE to LLMs. It has much higher overall HGS and precision than *Code Prompt* without hierarchical information, showing that hierarchical information can mitigate hallucinations during the temporal graph generation. In the casual category, the results of *Code Prompt (dependent rels)* are obtained when given both hierarchical and temporal graphs generated by CALLMSAE. The additional information also increases precision.

Fine-tuned T5 outperform all the methods based on CAEVO (McDowell et al., 2017; Madaan et al., 2021; Tan et al., 2024), showing that the high-quality graphs generated by CALLMSAE can boost the contextualised graph generation. Interestingly, the performance of the *Fine-tuned T5*, fine-tuned on CALLMSAE-generated data, exceeds that of CALLMSAE itself, implying that the fine-tuned model can effectively adapt the reasoning patterns provided by Llama3 and generalise them.

	Hier	Temp	Causal	Overall
Baseline (rel type)	0.58	0.66	0.48	0.60
CALLMSAE	0.71	0.71	**0.65**	0.69
Fine-tuned T5	**0.72**	**0.74**	0.62	**0.70**

Table 6: The human evaluation scores.

5.4 Human Evaluation of Event Graph

We recruited additional annotators to evaluate the generated graph on 50 of the test documents. We asked them whether the relation edges are correct. These scores ($\frac{\text{correct edges}}{\text{generated edges}}$) can be viewed as precision (Table 6). The human evaluation correlates with the HGS scores, verifying our conclusion.

	Format Error ↓	Cycle ↓
GPT-3.5	0%	10.67%
GPT-4	3.67%	1.67%
Mixtral	3.33%	2.33%
Llama3	**0%**	**0%**

Table 7: The average number of CALLMSAE-generated graphs out of 100 with format errors or cycles.

5.5 Format Error and Cycles in the Graphs

We specified the relation graphs as directed acyclic graphs in the prompt. If there is a cycle in the generated graph, it means that the LLM failed to follow the instructions. A cycle also indicates constraint violations because all the relations in the graphs are asymmetric. We use the APIs in *NetworkX* package to detect cycles in the transitive closure of the graphs. If the Python interpreter returns an error, it is classified as a format error. We prompt each LLM three times on the test set. Table 7 shows the average number of documents encountering format errors or cycles. All LLMs have low rates of format errors which shows that state-of-the-art LLMs can understand the instruction well and generate executable Python code. Among them, GPT-3.5 and Llama3 have no errors. About 10% of graphs generated by GPT-3.5 have cycles, suggesting that

GPT-3.5 may have inferior reasoning ability compared to other LLMs. GPT-4 and Mixtral both have low rates of cycle occurrence, but they are beaten by Llama3 which has no cycle in all generations, showing its remarkable understanding of the transitive and asymmetric constraints in the complex event relation graphs.

6 Conclusion

This study explored utilising LLMs to generate salient event relation graphs from news documents without relying on human annotations. We studied how the events generated by LLMs are compared to the traditional methods in terms of event saliency. We further demonstrated that CALLMSAE-generated graphs can serve as distant signals to fine-tune smaller models and outperform those based on CAEVO. The CALLMSAE-generated event graphs, together with the human-annotated test set are collected as *NYT-SEG*.

Limitations

CALLMSAE is more demanding than CAEVO in terms of computational power and time. To generate the *NYT-SEG*, we spent $2,200$ wall clock hours in inference. More details about resource cost are disclosed in Appendix D.

Although we have tested many prompting methods and included several of the most effective ones in this paper, we have not explored all possible combinations due to the extensive volume of recent literature on prompt engineering. There might still exist combinations of prompts that could further improve performance (Alzaid et al., 2024; Zhou et al., 2024). However, we are almost certain that any potential combinations, if they exist, are likely to be more complex and thus less efficient for building large-scale datasets. For example, we did not add demonstrations in graph generation because the code template is already quite lengthy. Adding more documents could potentially exceed the context windows of some LLMs, making it challenging for them to interpret the instructions effectively. The main goal of this work is to demonstrate the potential of LLM-based generation can help the data-demanding event graph generation task.

Ethics Statement

Event relation graph generation is a powerful tool for understanding text. A potential misuse of the proposed method is mining user behaviours on their private data. For example, salient event relation graphs can be extracted from users' tweets to analyse their potential reactions to advertisements and scams. That could be a huge risk to social media users.

Another potential risk is that the saliency may introduce bias. LLMs may have their preferences in selecting a specific group of events as important events due to the data they were trained on. This is a question which requires further large-scale investigation. However, we think this risk is negligible in this study because we work on document-level information. There is little room for selection given that the news articles are already the products of choice and distillation. If the system is used to extract information from a border information source, such as social media, the risk must be carefully assessed.

Acknowledgements

This work was supported in part by the UK Engineering and Physical Sciences Research Council (EPSRC) through a Turing AI Fellowship (grant no. EP/V020579/1, EP/V020579/2). Xingwei Tan was supported by the Warwick Chancellor's International Scholarship. This work was conducted on the Sulis Tier-2 HPC platform hosted by the Scientific Computing Research Technology Platform at the University of Warwick. Sulis is funded by EPSRC Grant EP/T022108/1 and the HPC Midlands+ consortium.

References

Ethar Alzaid, Gabriele Pergola, Harriet Evans, David Snead, and Fayyaz Minhas. 2024. Large multimodal model-based standardisation of pathology reports with confidence and its prognostic significance. *The Journal of Pathology: Clinical Research*, 10(6):e70010.

Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. 2024. Self-RAG: Learning to retrieve, generate, and critique through self-reflection. In *The Twelfth International Conference on Learning Representations*.

Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *CoRR*, abs/2004.05150.

Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. 2020. End-to-end object detection with transformers. In *European conference on computer vision*, pages 213–229. Springer.