What Would Happen Next? Predicting Consequences from An Event Causality Graph

Chuanhong Zhan [1] † **Wei Xiang** [2] † **Chao Liang**[1] **Bang Wang**[1] *
[1] School of Electronic Information and Communications,
Huazhong University of Science and Technology, Wuhan, China
{zhanch, liangchao111, wangbang}@hust.edu.cn
[2] Faculty of Artificial Intelligence in Education,
Central China Normal University, Wuhan China.
xiangwei@ccnu.edu.cn

Abstract

Existing script event prediction task forcasts the subsequent event based on an event script chain. However, the evolution of historical events are more complicated in real world scenarios and the limited information provided by the event script chain also make it difficult to accurately predict subsequent events. This paper introduces a *Causality Graph Event Prediction*(CGEP) task that forecasting consequential event based on an Event Causality Graph (ECG). We propose a *Semantic Enhanced Distance-sensitive Graph Prompt Learning* (SeDGPL) Model for the CGEP task. In SeDGPL, (1) we design a *Distance-sensitive Graph Linearization (DsGL)* module to reformulate the ECG into a graph prompt template as the input of a PLM; (2) propose an *Event-Enriched Causality Encoding (EeCE)* module to integrate both event contextual semantic and graph schema information; (3) propose a *Semantic Contrast Event Prediction (ScEP)* module to enhance the event representation among numerous candidate events and predict consequential event following prompt learning paradigm. Experiment results validate our argument our proposed SeDGPL model outperforms the advanced competitors for the CGEP task.[1]

1 Introduction

Event prediction aims to forecast the consequential event that are most likely to happen next, based on historical events and their relationships. It has significant applications in many scenarios and industries, such as dialogue systems (Chen et al., 2017), discourse understanding (Lee and Goldwasser, 2019), and story generation (Chaturvedi et al., 2017). Existing script event prediction task (Wang et al., 2023) predicts the subsequent event given a sequence of events, named event script chain. However, we argue that the evolution of historical events are more complicated than a script event chain in real world scenarios. Besides, the limited information provided by event chains also make it difficult to accurately predict subsequent events.

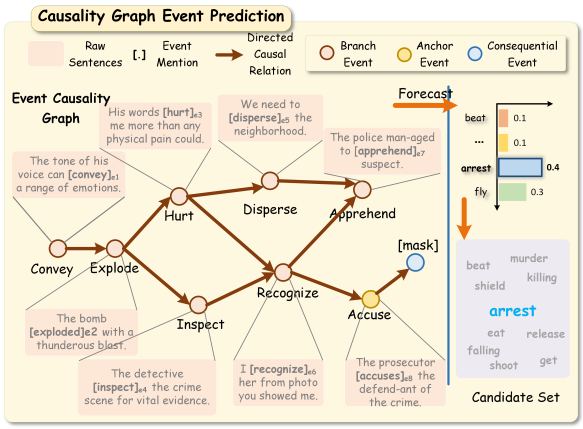

Figure 1: Illustration of the Causality Graph Event Prediction task, forecasting consequential event based on an Event Causality Graph.

Motivated from such considerations, this paper introduces a *Causality Graph Event Prediction*(CGEP) task that forecasting consequential event based on an Event Causality Graph (ECG). As illustrated in Fig. 1, the CGEP task is to select the most likely consequential event from candidate set based on an input ECG and an selected anchor event. Instead of using event script chain for subsequent event prediction, we model the connection between events by an ECG, which can better reveal the evolution of historical events. Besides, an ECG may have more than one consequential event that are likely to happen next. As such, we predict a consequential event for each tail node event (i.e. the anchor event) in an ECG, to achieve a more comprehensive understanding of events' evolution.

†. These authors contributed equally to this work.
*. Corresponding author: Bang Wang
1. We released the code at: https://github.com/zhanchuanhong/SeDGPL.

Findings of the Association for Computational Linguistics: EACL 2024, pages 820–832
November 12-16, 2024

Traditional event prediction methods either encode the contextual semantic of events (Du et al., 2022a) or model the information of graph structure (Shirai et al., 2023) for event forcasting. The recently emerged prompt learning paradigm, based on pre-trained language model (PLM), exhibits outstanding ability in logical reasoning and has been applied in many natural language processing tasks (Xiang et al., 2022). However, most PLMs take text sequences as input and struggle to process graph-structured inputs. In this paper, we use graph prompt learning paradigm to linearize the input ECG, so as to utilize the encyclopedia-like knowledge embedded in a PLM for prediction.

Besides, some studies obtain common sense knowledge from external knowledge bases to augment event prediction (Li et al., 2018). This again validates our argument that the event chain input contains insufficiency information of historical event. By contrast our ECG input itself has included abundant historical events and causalities information. To this end, we enrich the event representation by integrating event contextual semantic and graph schema information from the input ECGs. Furthermore, we select the consequential event from a significantly larger candidate set than that of event script prediction task. And a semantic contrastive learning is used to enhance the event representation among numerous candidate events.

In this paper, we introduce a CGEP task to forecast consequential event based on an ECG, and propose a *Semantic Enhanced Distance-sensitive Graph Prompt Learning* (SeDGPL) Model for the CGEP task. The SeDGPL model contains three modules: (1) The *Distance-sensitive Graph Linearization (DsGL)* module reformualtes the ECG into a graph prompt template as the input of a PLM; (2) The *Event-Enriched Causality Encoding (EeCE)* module enriches the event representation by integrating both event contextual semantic and graph schema information; (3) The *Semantic Contrast Event Prediction (ScEP)* module enhances the event representation among numerous candidate events and predicts consequential event following prompt learning paradigm.

We construct two CGEP datasets based on existing event causality corpus MAVEN-ERE and Event StoryLine Corpus (ESC). Experiment results validate our argument that predicting events based on ECG is more reasonable than that based on event script chain, and our proposed SeDGPL model outperforms the advanced competitors for the task.

2 Related Work

2.1 Script Event Prediction

Script Event Prediction focuses on predicting future events based on a narrative event chain with shared entities. Previous studies (Zhou et al., 2022; Wang et al., 2021; Huang et al., 2021) employ word2vec to encode the events, and predict subsequent events based on the similarity between candidate events and script events. With respect to temporal ordering, Pichotta and Mooney (2016); Wang et al. (2017) employ Long Short-Term Memory (LSTM) to model the temporal dependencies between events. Contemporary event modeling methods utilize the Pre-trained Language Models, e.g. BERT (Devlin et al., 2018) and RoBERTa (Liu et al., 2019). However, these models lack discourse-awareness as they are trained using Masked Language Modeling, which does not effectively capture the causal and temporal relations between multi-hop events. To address this problem, some research (Li et al., 2018; Zheng et al., 2020) also explore specific event graphs as external knowledge base to assist event prediction. For example, Wang et al. (2022b) proposes a novel Retrieval-Enhanced Temporal Event forecasting framework, which dynamically retrieves high-quality sub-graphs based on the corresponding entities.

2.2 Event Graph Reasoning

Event Graph Reasoning aims to leverage the structure and connections within the graph to identify new patterns (Roy et al., 2024) that do not explicitly exist in the event graph. Depending on the goal of reasoning, the task can be further categorized into relational reasoning (Huang et al., 2024) and event prediction (Li et al., 2021). For relation reasoning, Tang et al. (2023) adopts different event attributes to learn the semantic representations of events, and reasons event relation based on their similarities. Tang et al. (2021) combines LSTM and attention mechanisms to dynamically generate event sequence representations, thereby predicting event relationships. For event prediction, prior studies (Du et al., 2021, 2022b) perform subgraph matching between instance graph and schema graph to identify subsequent events. However, such methods predict event types rather than the events themselves. Moreover, Li et al. (2023b); Islam et al. (2024) predict potential events for the next timestamp by dividing the event graph into a series of subgraphs based on event timestamps.

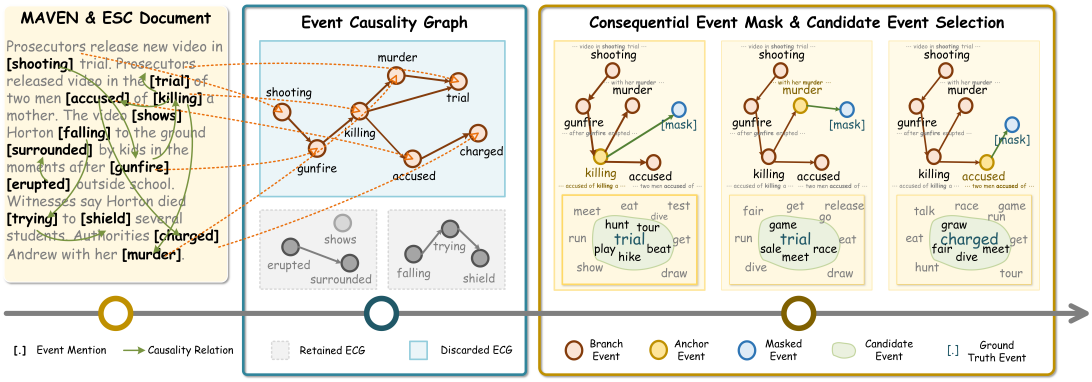

Figure 2: Data Processing Flowchart: The data processing involves transforming an original ECG into multiple data instances, with each instance specifically predicting a single consequential event.

3 Causality Graph Event Prediction

3.1 Task Definition

We define the Causality Graph Event Prediction (CGEP) task as predicting the most likely consequential events that will occur next in an event causality graph (ECG). As illustrated in Figure 1, the ECG is a directed graph consisting of some past events as nodes and the causal relations between them as directed edges, denoted by $\mathcal{G}(\mathcal{E}, \mathcal{R})$. Where an event node $e_i \in \mathcal{E}$ contains the event mention word(s) Em_i and the raw sentence S_i it belongs to; A causality edge $r_{ij} \in \mathcal{R}$ is a directed causal relation from the event node e_i to the event node e_j, indicating that e_i causes e_j (i.e. $e_i \rightarrow e_j$). Each tail node in an ECG, which has no edge starting from to any other event node, is used as the anchor event $e_t \in \mathcal{E}$ for next event prediction. The objective of CGEP task is to select the most likely consequential events e_c from the candidate event set $\mathcal{E}_c$ for an anchor event node e_t in an ECG.

3.2 Datasets Construction

We construct two CGEP datasets based on the public event causality dataset MAVEN-ERE (Wang et al., 2022c) and EventStoryLine Corpus (ESC) (Caselli and Vossen, 2017), annotating event mentions and directed causal relations between events within documents. Figure 2 illustrates the process of CGEP dataset construction.

We first construct ECGs based on the annotations in each document from ESC and MAVEN-ERE datasets, using the annotated events as nodes and the annotated directed causal relation between events as edges. Note that multiple disconnected ECGs may be constructed from a single document, and only *weakly connected graphs*[2] with more than 4 event nodes are retained to ensure a complete event causality graph structure for event prediction. We then mask one of the tail event node in an ECG as a CGEP instance, where the masked event is the consequential event e_c to be predicted and its cause event is the anchor event e_t. In case that the masked event is caused by multiple events or an anchor event causes multiple effect events, it is further divided into multiple CGEP instances to ensure that each instance has an unique anchor event and ground truth consequential event.

For each CGEP instance, we randomly select a large number of tail node events from all other ECGs in the dataset as negative samples to construct a candidate set of consequential events $\mathcal{E}_c$. The ground truth event e_c is the one that has been masked aforementioned. Considering that the ground truth event mention may also appears in the sentence of other event nodes, that is the sentence it belongs to contains multiple event mentions, we replace them by a PLM-specific token [PAD] to prevent answer leakage. Finally, we construct two CGEP dataset CGEP-MAVEN and CGEP-ESC[3] for the CGEP task, in which each instance contains an event causality graph $\mathcal{G}(\mathcal{E}, \mathcal{R})$, an anchor event e_t, a candidate event set $\mathcal{E}_c$, and a ground truth consequential event e_c.

Considering the varying instance sizes of the CGEP-MAVEN and CGEP-ESC datasets, the number of candidate sets for consequential events is randomly selected to be 512 and 256 , respectively. Table 1 summarizes the statistics of our constructed

2. A graph is considered weakly connected if every pair of vertices in the graph is connected by a path, regardless of the direction of the edges.
3. Datasets will be released publicly after the anonymous review.

Datasets	Docs	ECGs	Avg.Nodes	Avg.Edges	Instances	CandiSet
CGEP-MAVEN	3,015	5,308	8.4	12.9	12,200	512
CGEP-ESC	243	363	11	24.9	1,191	256

Table 1: Statistics of our CGEP-MAVEN and CGEP-ESC datasets.

CGEP-MAVEN and CGEP-ESC datasets.

4 Methodology

We propose a *Semantic Enhanced Distance-sensitive Graph Prompt Learning Model* (SeDGPL) for causality graph event prediction. As illustrated in Figure 3, the SeDGPL contains three modules: (1) *Distance-sensitive Graph Linearization* (DsGL); (2) *Event-Enriched Causality Encoding* (EeCE); (3) *Semantic Contrast Event Prediction* (ScEP).

4.1 Distance-sensitive Graph Linearization

The DsGL module is to reformulate the Event Causality Graph (ECG) of an input CGEP instance into a graph prompt template $\mathcal{T}(\mathcal{G})$, as the input of a Pre-trained Language Model (PLM). As illustrated in Figure 3 (a), the graph prompt template $\mathcal{T}(\mathcal{G})$ is a concatenation of some event causality triple templates $\mathcal{T}_n$ and a simple prompt template $\mathcal{T}_m$, represented as follows:

$$\mathcal{T}(\mathcal{G}) = \texttt{[C]}, \mathcal{T}_1, \texttt{[S]}, ... \mathcal{T}_n, \texttt{[S]}, \mathcal{T}_m, \texttt{[S]}, \quad (1)$$

where `[C]` and `[S]` are the PLM-specific tokens `[CLS]` and `[SEP]`, respectively, indicating the beginning and ending of an input sequence. Additionally, `[S]` is also used to mark the boundary between each triple templates and the prompt template.

Given an ECG $\mathcal{G}$ with n directed causality edges, we can first obtain n event causality triples $T_r^{(n)} = (e_i, r_{ij}, e_j)$, each containing a cause event e_i, an effect event e_j and a directed causal relation r_{ij} from e_i to e_j. The template $\mathcal{T}_n$ for each event causality triple is formulated by concatenating of both the cause and the effect event mentions with an inserted conjunction word *causes*:

$$\mathcal{T}_n = Em_i \ \ causes \ \ Em_j, \quad (2)$$

where Em_i and Em_j are the event mentions of cause event e_i and effect event e_j, respectively.

We argue that the closer an event causality triple $T_r^{(n)}$ is to the anchor event e_t, the stronger its connection to the anchor event, and it can provide more critical information for consequential event prediction. To this end, we order the event causality triples based on their distance to the anchor event. The distance of an event causality triple $T_r^{(n)} = (e_i, r_{ij}, e_j)$ to the anchor event e_t is computed by the number of edges on the shortest undirected path from its cause event e_i to the anchor event e_t, as follows:

$$d_n(e_i, e_t) = \min_{p \in P(e_i, e_t)} |p|, \quad (3)$$

where $P(e_i, e_t)$ is the set of all undirected paths from the cause event e_i to the anchor event e_t, and $|p|$ is the number of edges on the path p.

We arrange the event causality triple templates $\mathcal{T}_n$ in decreasing order of their distance to the anchor event e_t. As in Equation 1, the distances are ordered such that $d_1 \geq d_2 \geq \ ... \ \geq d_n$, indicating that $\mathcal{T}_n$ is closest to the anchor event and $\mathcal{T}_1$ is the farthest one. At the end of graph prompt template $\mathcal{T}(\mathcal{G})$, we design and concatenate a simple prompt template $\mathcal{T}_m$ for event prediction:

$$\mathcal{T}_m = Em_t \ \ causes \ \ \texttt{[MASK]}, \quad (4)$$

where Em_t is the event mention of anchor event e_t and the PLM-specific token `[MASK]` is used to predict consequential event.

4.2 Event-Enriched Causality Encoding

To enrich the event representation for causality encoding, we propose an EeCE module that integrates both event contextual semantic and graph schema information into the ECG representation. After graph linearization, we input each graph prompt template $\mathcal{T}(\mathcal{G})$ into a Pre-trained Language Model (PLM) for ECG encoding, denoted as $\mathcal{P}_{\text{ECG}}$. As illustrated in Figure 3 (b), the input representation of PLM is constructed by summing the corresponding token embedding $\mathbf{h}_t^{(g)}$, the segment embedding $\mathbf{h}_s^{(g)}$, and the position embedding $\mathbf{h}_p^{(g)}$:

$$\mathbf{h}^{(g)} = \mathbf{h}_t^{(g)} + \mathbf{h}_s^{(g)} + \mathbf{h}_p^{(g)}. \quad (5)$$

For contextual semantic encoding, we input the raw sentence S_i of each event into another PLM $\mathcal{P}_c$

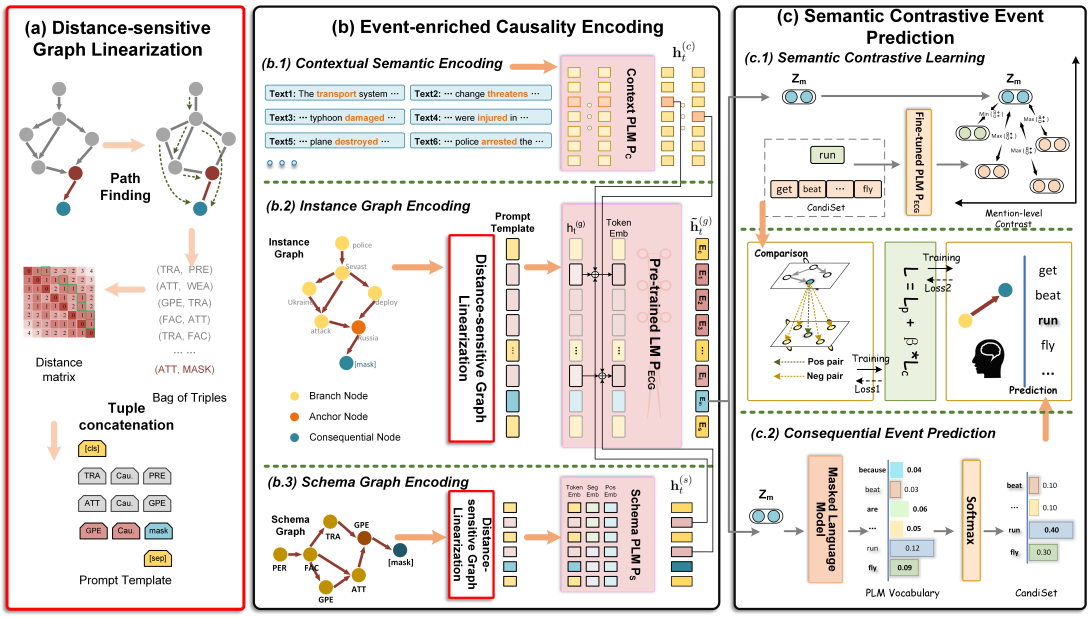

Figure 3: The SeDGPL model consists of three modules: (1) *Distance-sensitive Graph Linearization* (DsGL); (2) *Event-Enriched Causality Encoding* (EeCE); (3) *Semantic Contrast Event Prediction* (ScEP).

to obtain its contextual representation $\mathbf{h}^{(c)}$, as illustrated in Figure 3 (b.1). For schema information encoding, we first construct an event schema graph by replacing each event node in an ECG with its corresponding annotated event type, like (Zhuang et al., 2023; Groz et al., 2021), and etc. After the same graph linearization operation, we input each schema graph template into another PLM $\mathcal{P}_s$ to obtain the event's schema representation $\mathbf{h}^{(s)}$, as illustrated in Figure 3 (b.3). We note that only the token embeddings of event's contextual representation $\mathbf{h}_t^{(c)}$ and schema representation $\mathbf{h}_t^{(s)}$ are used for next enrichment fusion. The segment embedding $\mathbf{h}_s^{(g)}$ and position embedding $\mathbf{h}_p^{(g)}$ of ECG encoding, which contain graph structure information, are directly used without fusion.

To fuse the features of event's contextual semantic and schema information into the ECG representation, we use the fusion gate to integrate their event's representations $\mathbf{h}_t^{(c)}$ and $\mathbf{h}_t^{(s)}$ into the event's representation of ECG $\mathbf{h}_t^{(g)}$. Specifically, we first use a fusion gate to integrate the contextual representation $\mathbf{h}_t^{(c)}$ schema representation $\mathbf{h}_t^{(s)}$, and output $\mathbf{h}_t^{(r)} \in \mathbb{R}^{d_h}$ as the event enrichment vector. The transition functions are:

$$\mathbf{g}_r = \text{sigmoid}(\mathbf{W}_r \mathbf{h}_t^{(c)} + \mathbf{U}_r \mathbf{h}_t^{(s)}), \quad (6)$$

$$\mathbf{h}_t^{(r)} = \mathbf{g}_r \odot \mathbf{h}_t^{(c)} + (1 - \mathbf{g}_r) \odot \mathbf{h}_t^{(s)}, \quad (7)$$

where $\mathbf{W}_r \in \mathbb{R}^{d_h \times d_h}$, $\mathbf{U}_r \in \mathbb{R}^{d_h \times d_h}$ are learnable parameters and $\odot$ donates the element-wise product of vectors.

We next use another fusion gate to integrate the event enrichment vector $\mathbf{h}_t^{(r)} \in \mathbb{R}^{d_h}$ into the token embeddings of event's representation in ECG $\mathbf{h}_t^{(g)}$. The transition functions are:

$$\mathbf{g}_e = \text{sigmoid}(\mathbf{W}_e \mathbf{h}_t^{(g)} + \mathbf{U}_e \mathbf{h}_t^{(r)}), \quad (8)$$

$$\tilde{\mathbf{h}}_t^{(g)} = \mathbf{g}_e \odot \mathbf{h}_t^{(g)} + (1 - \mathbf{g}_e) \odot \mathbf{h}_t^{(r)}, \quad (9)$$

where $\mathbf{W}_e \in \mathbb{R}^{d_h \times d_h}$, $\mathbf{U}_e \in \mathbb{R}^{d_h \times d_h}$ are learnable parameters. With the fusion gate, we enrich the event's representation in ECGs by integrating both event's contextual semantic and schema information features. Note that only the representations of event mention in ECGs are fused, the other tokens in graph prompt template $\mathcal{T}(\mathcal{G})$, such as *causes*, `[CLS]`, `[SEP]`, `[MASK]`, and etc., are originally encoded by the ECG encoding PLM $\mathcal{P}_{\text{ECG}}$.

Finally, the PLM $\mathcal{P}_{\text{ECG}}$ outputs a hidden state vector $\mathbf{z}$ for each input token in the graph prompt template $\mathcal{T}(\mathcal{G})$, using the fused event's token embeddings as input representations.

4.3 Semantic Contrast Event Prediction

Following the prompt learning paradigm (Xiang et al., 2023; Li et al., 2023a), we use the hidden state vector of `[MASK]` token $\mathbf{z}_m$ for consequential event prediction. To enhance the PLM's ability of understanding event semantic among numerous candidate events, we apply a kind of semantic contrastive learning to improve the `[MASK]` token presentation $\mathbf{z}_m$.

Model	CGEP-MAVEN						CGEP-ESC					
	MRR	Hit@1	Hit@3	Hit@10	Hit@20	Hit@50	MRR	Hit@1	Hit@3	Hit@10	Hit@20	Hit@50
CSProm-KG	22.3	18.1	23.2	31.0	38.4	50.7	14.2	11.9	11.3	21.0	25.6	34.6
SimKG	9.3	4.5	9.2	18.0	25.3	35.0	14.9	10.3	13.5	18.4	22.3	34.0
BARTbase	24.7	19.5	24.5	34.8	42.6	53.6	16.0	12.5	16.8	21.1	28.6	38.9
MCPredictor	18.1	13.0	18.4	27.3	32.0	43.2	9.7	8.4	10.9	17.4	22.2	37.5
Llama3-7B	9.6	5.0	11.1	20.2	24.5	26.6	6.7	1.1	8.9	20.2	26.3	29.2
GPT-3.5-turbo	14.6	8.1	17.1	28.1	33.3	39.5	10.1	4.9	11.4	20.5	25.2	31.5
SeDGPL	**27.9**	**21.9**	**28.9**	**40.8**	**48.1**	**57.9**	**19.6**	**15.2**	**18.1**	**22.3**	**29.9**	**41.9**

Table 2: Overall results of Causality Graph Event Prediction on the CGEP-MAVEN and CGEP-ESC datasets.

Model	CGEP-MAVEN				
	MRR	Hit@1	Hit@3	Hit@10	Hit@50
CSProm-KG	7.1 (↓15.2)	4.8 (↓13.3)	6.4(↓16.8)	10.6(↓20.4)	22.4(↓28.3)
SimKG	5.0(↓4.3)	2.2(↓2.3)	4.3(↓4.9)	8.5(↓9.5)	25.7(↓9.3)
BARTbase	11.8(↓12.9)	8.2(↓11.3)	11.2(↓13.3)	16.6(↓18.2)	34.2(↓19.4)
MCPredictor	7.3(↓10.8)	3.6(↓9.4)	7.3(↓14.5)	14.8(↓19.7)	29.4(↓13.8)
SeDGPL	**16.0**(↓11.9)	**12.4**(↓9.5)	**15.4**(↓13.5)	**23.0**(↓17.8)	**39.4**(↓18.5)

Table 3: Overall results of Script Event Prediction on CGEP-MAVEN dataset.

Semantic Contrastive Learning: As illustrated in Figure 3 (c.1), we first obtain a representation vector $\mathbf{z}_c$ for each candidate event e_c using the fine-tuned PLM $\mathcal{P}_{ECG}$. Then, the hidden state of `[MASK]` token $\mathbf{z}_m$ is used as the anchor sample, and the candidate event representations $\mathbf{z}_c$ are used as contrastive samples, where the ground truth event is the positive sample $\mathbf{z}_c^+$ and the other candidate events are negative samples $\mathbf{z}_c^-$. We employ the Supervised contrastive loss (Khosla et al., 2020) to compute the semantic contrast loss, as follows:

$$L_c = -\log \frac{\exp(\mathbf{z}_m \cdot \mathbf{z}_c^+/\tau)}{\sum_{c \in \mathcal{C}} \exp(\mathbf{z}_m \cdot \mathbf{z}_c/\tau)}, \quad (10)$$

where τ is a scalar temperature parameter and $\mathcal{C}$ is the candidate set containing the positive sample and negative samples.

Consequential Event Prediction: As illustrated in Figure 3 (c.2), the PLM $\mathcal{P}_{\text{ECG}}$ estimates the probability of each word within its vocabulary V for the hidden state of `[MASK]` token $\mathbf{z}_m$. We use the predicted probability of the event mention word e_c in the event candidate set $\mathcal{E}_c$ as the ranking score, to form an event prediction list:

$$\mathcal{P}(\texttt{[MASK]} = e_c \in \mathcal{E}_c \mid \mathcal{T}(\mathcal{G})). \quad (11)$$

We employ the cross entropy loss to compute the event prediction loss, as follows:

$$L_p = -\frac{1}{K}\sum_{k=1}^{K} \mathbf{y}^{(k)} \log(\hat{\mathbf{y}}^{(k)}) + \lambda\|\theta\|^2, \quad (12)$$

where $\mathbf{y}^{(k)}$ and $\hat{\mathbf{y}}^{(k)}$ are the gold label and predicted label of the k-th training instance respectively. λ and θ are the regularization hyper-parameters. We use the AdamW optimizer (Loshchilov and Hutter, 2017) with $L2$ regularization for model training.

Training Strategy: The cost function of our SeDGPL is optimized as follows:

$$L = L_p + \beta * L_c, \quad (13)$$

where β is a weight coefficient to balance the importance of the event prediction loss and semantic contrast loss.

5 Experiment

5.1 Experiment Settings

Our experiments are conducted using the constructed CGEP-MAVEN and CGEP-ESC datasets. Following the standard data splitting of the underlying ESC (Caselli and Vossen, 2017) corpus, we use the last two topics as development set and conduct 5-fold cross-validation on the remaining 20 topics. The average results of each fold are adopted as performance metrics. Since the underlying MAVEN-ERE corpus did not release the test set, following (Tao et al., 2023), we use the original development set as our test set and sample 20% of the data from the original training set to form the development set.

We adopt the MRR (Mean Reciprocal Rank) and Hit@n (Hit Rate at n) as the evaluation metrics. Details about experimental settings and evaluation metrics can be found in Appendix B.

5.2 Competitors

We replicate some advanced event prediction models to conduct causality graph event prediction as benchmarks, including methods in knowledge graph completion tasks (CSProm-KG (Chen et al., 2023), SimKG (Wang et al., 2022a)) and script event prediction (BARTbase (Zhu et al., 2023), MCPredictor (Bai et al., 2021)). Furthermore, we validate the effectiveness of large language models on the CGEP task, including Llama3-7B (Touvron et al., 2023) and GPT-3.5-turbo (Gao et al., 2023). For more details about its specific implementation, please refer to the Appendix A and Appendix C.

5.3 Overall Results

Table 2 compares the overall performance between our SeDGPL and the competitors on both CGEP-MAVEN and CGEP-ESC datasets.

We can observe that our SeDGPL has achieved significant performance improvements overall competitors in terms of much higher MRR and Hit@n. We attribute its outstanding performance to two main factors: 1) The transformation of the event causality graph into an ordered triple sequence for graph prompt learning, which enables our SeDGPL to effectively leverage both the structure information of event causality graph and the encyclopedic knowledge in a PLM for event prediction; 2) The enrichment of event representation through contextual semantic and schema information fusion encoding. Besides, We can also observe that the BARTbase outperforms the other competitors in Table 2. This might be attributed to the fine-tuning of a pre-trained language model in advance using an event-centric pre-training objective, which injects event-level knowledge into the PLM before making predictions. 3) The performance of the Llama3-7B and GPT-3.5-turbo surpasses some models trained on the entire dataset, e.g. the SimKG model , indicating that large language models have great potential in understanding event relationships and reasoning event patterns.

To validate our argument that predicting consequential events based on event causality graph is more effective than predicting based on the event script chain, we also employ our SeDGPL and the competitors to conduct script event prediction for comparison, using the longest event chain in each event causality graph from CGEP-MAVEN dataset.[4] Table 3 presents the performance of script event prediction between our SeDGPL and the competitors, as well as the performance variation compared with causality graph event prediction. We can observe that the performance of event prediction suffers significantly due to the transformation of the causality graph input into the even chain input. This is not unexpected. The event causality graph has a more complex structure than the script event chain, as it includes additional event nodes and causal connections, that can provide comprehensive prior knowledge for event prediction. Besides, it can be observed that our SeDGPL also outperforms all competitors in script event prediction, again approving our design object.

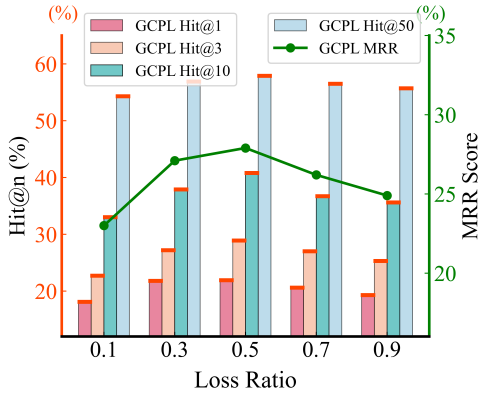

Figure 4: Results on CGEP-MAVEN with different loss ratio β.

5.4 Ablation Study

Module Ablation To examine the effectiveness of different modules, we design the following ablation study: (1) SeDGPL w/o Dist. randomly orders the event causality triples without considering distance sensitivity; (2) SeDGPL w/o Ctxt. enriches event representation with only schema information, but without its contextual semantic; (3) SeDGPL w/o Schm. enriches event representation with only contextual semantic, but without its schema information; (4) SeDGPL w/o Ctrst. predicts consequential events without semantic contrastive learning. Table 4 presents the results of our module ablation study.

The first observation is that neither the SeDGPL w/o Ctxt. and the SeDGPL w/o Schm. can outperform the Full SeDGPL model. This indicates that our fusion of both event contextual semantic and graph schema information is an effective ap-

4. Considering the instance number of event chains in CGEP-ESC dataset, we only conduct script event prediction based on the CGEP-MAVEN dataset.

Model	CGEP-MAVEN						CGEP-ESC					
	MRR	Hit@1	Hit@3	Hit@10	Hit@20	Hit@50	MRR	Hit@1	Hit@3	Hit@10	Hit@20	Hit@50
SeDGPL w/o Dist.	26.4	20.4	26.2	39.2	47.0	57.2	13.9	7.8	15.6	18.8	23.9	37.8
SeDGPL w/o Ctxt.	5.3	4.0	4.2	9.6	13.9	23.6	12.2	8.8	11.0	17.9	21.7	33.8
SeDGPL w/o Schm.	22.0	17.0	21.9	31.5	40.9	54.3	15.6	11.5	12.4	20.4	24.3	37.4
SeDGPL w/o Ctrst.	21.2	15.8	21.0	32.0	41.4	53.8	13.2	8.5	14.5	20.0	25.2	38.0
Full SeDGPL	**27.9**	**21.9**	**28.9**	**40.8**	**48.1**	**57.9**	**19.6**	**15.2**	**18.1**	**22.3**	**29.9**	**41.9**

Table 4: Experiment results of ablation study on both CGEP-MAVEN corpus and CGEP-ESC corpus.

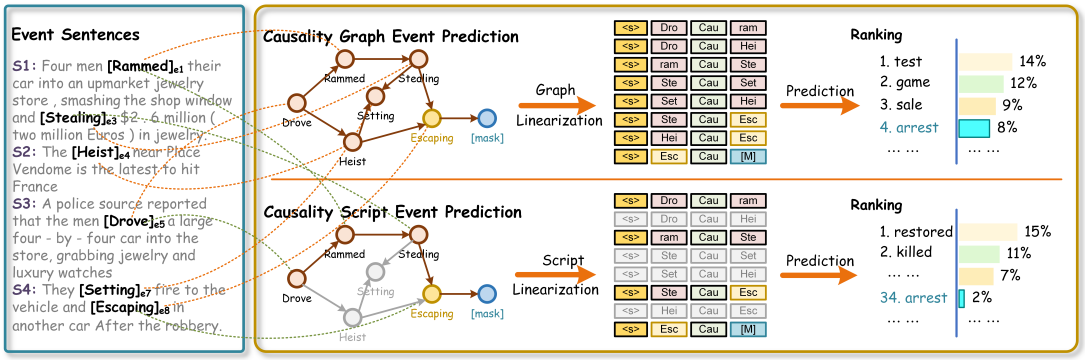

Figure 5: A case of SeDGPL on causality graph event prediction and causality script event prediction tasks.

proach to enrich event's representation learning for consequential event prediction. On the other hand, the SeDGPL w/o Ctxt. performs the worst among all ablation models. As it merely uses event mention words for representation learning, ignoring the event contextual semantic and existing linguistic ambiguities. The second observation is that the SeDGPL w/o Dist. cannot outperform the Full SeDGPL model, even the performance gap is not obvious. This suggests that it is essential to order event causality triples based on distance sensitivity, as different triples in an event causality graph may have different importance for prediction consequential events. We can also observe that the SeDGPL w/o Ctrst. cannot outperform the Full SeDGPL model, validating the effectiveness of contrastively learning the `[MASK]` token presentation $\mathbf{z}_m$ among numerous candidate events.

Hyper-parameter Ablation To further examine the impact of semantic contrastive learning module, we compare the performance of our SeDGPL against using different contrastive loss weight coefficient β on the CGEP-MAVEN dataset, as plotted in Figure 4. It can be observe that our SeDGPL achieves the best overall performance when the contrastive loss weight coefficient is set to 0.5. Yet it suffers from either a large or small value of the loss weight coefficient. Indeed, a small weight coefficient weakens the impact of semantic contrastive learning; By contrast, a large weight coefficient ignores the event prediction loss in back-propagation.

6 Case study

Figure 5 illustrates an example of SeDGPL applied to the causality graph event prediction (CGEP) task and the causality script event prediction (CSEP) task. For the CGEP task, SeDGPL linearizes the entire event graph into an event chain, comprehensively considering all the causality triples in the event graph. In contrast, for the CSEP task, SeDGPL extracts only a subset of the causality triples from the event graph to form the main event chain, disregarding the other nodes in the event graph, which undermines the structural information of the event graph. From Figure 5, we observe that incorporating information beyond the main event chain can effectively aid the model in predicting subsequent events more accurately. For instance, in the CGEP task, given the causality triples *(Drove, causes, Heist)* and *(Heist, causes, Escaping)* as prior knowledge, our model can readily infer that the subsequent event following "Escaping" is "arrest". In contrast, for the CSEP task, the model only relies on the main event chain to judge the relationship between events. Therefore, the model

can not effectively capture the causal relationships between events at different levels and the complex structure information in the event causality graph, leading to a decline in performance.

7 Concluding Remarks

In this paper, we argue that predicting consequential events based on the event causality graph is more effective than predicting based on the event script chain. To validate our argument, we propose the SeDGPL Model, a distance-sensitive graph prompt model that integrate both event contextual semantic and graph schema information, and conduct abundant experiments on both CGEP and SEP task. Experiment results validate our argument, and our proposed SeDGPL model outperforms the advanced competitors for the CGEP task.

In our future work, we shall attempt to integrate other types of event relationships, e.g. temporal relations, to assist in event prediction.

8 Limitation

Due to the input length limitations of PLMs, we may have to discard some triplets during the linearization process, which could result in the loss of information beneficial for prediction.

9 Acknowledge

This work is supported in part by National Natural Science Foundation of China (Grant No:62172167). The computation is completed in the HPC Platform of Huazhong University of Science and Technology.

10 Ethics Statement

This paper has no particular ethic consideration.

References

Long Bai, Saiping Guan, Jiafeng Guo, Zixuan Li, Xiaolong Jin, and Xueqi Cheng. 2021. Integrating deep event-level and script-level information for script event prediction. *arXiv preprint arXiv:2110.15706*.

Tommaso Caselli and Piek Vossen. 2017. The event storyline corpus: A new benchmark for causal and temporal relation extraction. In *Proceedings of the Events and Stories in the News Workshop*, pages 77–86.

Snigdha Chaturvedi, Haoruo Peng, and Dan Roth. 2017. Story comprehension for predicting what happens next. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1603–1614.

Chen Chen, Yufei Wang, Aixin Sun, Bing Li, and Kwok-Yan Lam. 2023. Dipping plms sauce: Bridging structure and text for effective knowledge graph completion via conditional soft prompting. *arXiv preprint arXiv:2307.01709*.

Hongshen Chen, Xiaorui Liu, Dawei Yin, and Jiliang Tang. 2017. A survey on dialogue systems: Recent advances and new frontiers. *Acm Sigkdd Explorations Newsletter*, 19(2):25–35.

Alla Chepurova, Aydar Bulatov, Yuri Kuratov, and Mikhail Burtsev. 2023. Better together: Enhancing generative knowledge graph completion with language models and neighborhood information. *arXiv preprint arXiv:2311.01326*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Li Du, Xiao Ding, Kai Xiong, Ting Liu, and Bing Qin. 2021. Excar: Event graph knowledge enhanced explainable causal reasoning. 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)*, pages 2354–2363.

Li Du, Xiao Ding, Yue Zhang, Kai Xiong, Ting Liu, and Bing Qin. 2022a. A graph enhanced bert model for event prediction. *arXiv preprint arXiv:2205.10822*.

Xinya Du, Zixuan Zhang, Sha Li, Pengfei Yu, Hongwei Wang, Tuan Lai, Xudong Lin, Ziqi Wang, Iris Liu, Ben Zhou, et al. 2022b. Resin-11: Schema-guided event prediction for 11 newsworthy scenarios. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: System Demonstrations*, pages 54–63.

Jinglong Gao, Xiao Ding, Bing Qin, and Ting Liu. 2023. Is chatgpt a good causal reasoner? a comprehensive evaluation. *arXiv preprint arXiv:2305.07375*.

Benoît Groz, Aurélien Lemay, Sławek Staworko, and Piotr Wieczorek. 2021. Inference of shape expression schemas typed rdf graphs. *arXiv preprint arXiv:2107.04891*.

Xingyue Huang, Miguel Romero, Ismail Ceylan, and Pablo Barceló. 2024. A theory of link prediction via relational weisfeiler-leman on knowledge graphs. *Advances in Neural Information Processing Systems*, 36.

Zhenyu Huang, Yongjun Wang, Hongzuo Xu, Songlei Jian, and Zhongyang Wang. 2021. Script event prediction based on pre-trained model with tail event

enhancement. In *Proceedings of the 2021 5th International Conference on Computer Science and Artificial Intelligence*, pages 242–248.

Muhammed Ifte Khairul Islam, Khaled Mohammed Saifuddin, Tanvir Hossain, and Esra Akbas. 2024. Dygcl: Dynamic graph contrastive learning for event prediction. *arXiv preprint arXiv:2404.15612*.

Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. 2020. Supervised contrastive learning. *Advances in neural information processing systems*, 33:18661–18673.

I-Ta Lee and Dan Goldwasser. 2019. Multi-relational script learning for discourse relations. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4214–4226.

Manling Li, Sha Li, Zhenhailong Wang, Lifu Huang, Kyunghyun Cho, Heng Ji, Jiawei Han, and Clare Voss. 2021. The future is not one-dimensional: Complex event schema induction by graph modeling for event prediction. *arXiv preprint arXiv:2104.06344*.

Sha Li, Ruining Zhao, Manling Li, Heng Ji, Chris Callison-Burch, and Jiawei Han. 2023a. Opendomain hierarchical event schema induction by incremental prompting and verification. In *Proc. The 61st Annual Meeting of the Association for Computational Linguistics (ACL2023)*.

Zhipeng Li, Shanshan Feng, Jun Shi, Yang Zhou, Yong Liao, Yangzhao Yang, Yangyang Li, Nenghai Yu, and Xun Shao. 2023b. Future event prediction based on temporal knowledge graph embedding. *Computer Systems Science & Engineering*, 44(3).

Zhongyang Li, Xiao Ding, and Ting Liu. 2018. Constructing narrative event evolutionary graph for script event prediction. *arXiv preprint arXiv:1805.05081*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Karl Pichotta and Raymond Mooney. 2016. Learning statistical scripts with lstm recurrent neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30.

Kaushik Roy, Alessandro Oltramari, Yuxin Zi, Chathurangi Shyalika, Vignesh Narayanan, and Amit Sheth. 2024. Causal event graph-guided language-based spatiotemporal question answering.

Sola Shirai, Debarun Bhattacharjya, and Oktie Hassanzadeh. 2023. Event prediction using case-based reasoning over knowledge graphs. In *Proceedings of the ACM Web Conference 2023*, pages 2383–2391.

Tingting Tang, Wei Liu, Weimin Li, Jinliang Wu, and Haiyang Ren. 2021. Event relation reasoning based on event knowledge graph. In *International Conference on Knowledge Science, Engineering and Management*, pages 491–503. Springer.

Wei Tang, Qingchao Kong, Yin Luo, and Wenji Mao. 2023. Neuro-logic learning for relation reasoning over event knowledge graph. In *2023 IEEE International Conference on Intelligence and Security Informatics (ISI)*, pages 1–6. IEEE.

Zhengwei Tao, Zhi Jin, Xiaoying Bai, Haiyan Zhao, Chengfeng Dou, Yongqiang Zhao, Fang Wang, and Chongyang Tao. 2023. Seag: Structure-aware event causality generation. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 4631–4644.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Liang Wang, Wei Zhao, Zhuoyu Wei, and Jingming Liu. 2022a. Simkgc: Simple contrastive knowledge graph completion with pre-trained language models. *arXiv preprint arXiv:2203.02167*.

Lihong Wang, Juwei Yue, Shu Guo, Jiawei Sheng, Qianren Mao, Zhenyu Chen, Shenghai Zhong, and Chen Li. 2021. Multi-level connection enhanced representation learning for script event prediction. In *Proceedings of the Web Conference 2021*, pages 3524–3533.

Ruijie Wang, Zheng Li, Danqing Zhang, Qingyu Yin, Tong Zhao, Bing Yin, and Tarek Abdelzaher. 2022b. Rete: retrieval-enhanced temporal event forecasting on unified query product evolutionary graph. In *Proceedings of the ACM Web Conference 2022*, pages 462–472.

Xiaozhi Wang, Yulin Chen, Ning Ding, Hao Peng, Zimu Wang, Yankai Lin, Xu Han, Lei Hou, Juanzi Li, Zhiyuan Liu, et al. 2022c. Maven-ere: A unified large-scale dataset for event coreference, temporal, causal, and subevent relation extraction. *arXiv preprint arXiv:2211.07342*.

Zhongqing Wang, Yue Zhang, and Ching Yun Chang. 2017. Integrating order information and event relation for script event prediction. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 57–67.

Zikang Wang, Linjing Li, and Daniel Zeng. 2023. Integrating relational knowledge with text sequences for script event prediction. *IEEE Transactions on Neural Networks and Learning Systems*.

Wei Xiang, Chao Liang, and Bang Wang. 2023. Teprompt: Task enlightenment prompt learning for implicit discourse relation recognition. *arXiv preprint arXiv:2305.10866*.

Wei Xiang, Zhenglin Wang, Lu Dai, and Bang Wang. 2022. Connprompt: Connective-cloze prompt learning for implicit discourse relation recognition. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 902–911.

Jianming Zheng, Fei Cai, Yanxiang Ling, and Honghui Chen. 2020. Heterogeneous graph neural networks to predict what happen next. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 328–338.

Pengpeng Zhou, Bin Wu, Caiyong Wang, Hao Peng, Juwei Yue, and Song Xiao. 2022. What happens next? combining enhanced multilevel script learning and dual fusion strategies for script event prediction. *International Journal of Intelligent Systems*, 37(11):10001–10040.

Fangqi Zhu, Jun Gao, Changlong Yu, Wei Wang, Chen Xu, Xin Mu, Min Yang, and Ruifeng Xu. 2023. A generative approach for script event prediction via contrastive fine-tuning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 14056–14064.

Ling Zhuang, Po Hu, and Weizhong Zhao. 2023. Event relation extraction using type-guided attentive graph convolutional networks. In *International Conference on Database Systems for Advanced Applications*, pages 3–20. Springer.

A Competitors

CSProm-KG We first linearize instance graphs and input them into the PLM, rather than modeling based on triplets as in the original model. And then we generate the conditional soft prompts based on the node embedding E_h and relation embedding E_r, obtained by the embedding layer of RoBERTa. Then the conditional soft prompts will be concatenated with the text embedding and input into the PLM. Notice due to the specificity of the task, we cannot employ the KGC model ConvE for prediction. Consequently, we input the anchor node embedding into an MLP classification layer to obtain the final prediction.

SimKG Similar to our proposed model, we initially transform the event instance graph to a linearized input sequence. To fully consider the impact of the candidate event set on the model, we also incorporate the candidate event set into contrastive learning, referring to them as *candidate event negatives*. Since the consequence node to be predicted must be the 1-hop neighbors of the anchor node, we set the *Re-Ranking* factors α to 0. Finally, we combine four types of contrastive losses, referred to as *Candi-event Negatives*, *In-batch Negatives*, *Pre-batch Negatives*, and *Self-Negativest*, and utilize back-propagation to update the model parameters.

BARTbase We first populate the event instance graph according to the template described in the article, and then randomly mask out some events for Event-Centric Pretraining. Notice only the training set is used for Pretraining. In the Task-Specific Contrastive Fine-tuning phase, we first replace all events with virtual tokens, whose initial representation is obtained by averaging the embedding of all tokens of the event. Finally, we input the mask embedding into an MLP classification layer to obtain the prediction probabilities for each candidate event.

MCPredictor We first encode the event mentions and event texts, using the *[CLS]* vector as the text representation. Then, we concatenate the two embeddings to obtain the initial representation $\mathbf{e}_i$ of the event. Node that since the large number of candidate events, the computational complexity significantly increases when concatenating candidate events to the template to obtain event scores. So instead of employing *Event-Level Scoring* and *Script-Level Scoring*, We employ an MLP classifier on the template embeddings to obtain scores for each candidate event.

All baseline experiments are conducted on PyTorch framework with CUDA on NVIDIA GTX 3090 Ti GPUs. We employ the RoBERTa-base(Liu et al., 2019) for the base model, and set the length of sequence to 200, the mini-batch to 1, the training epoch to 15 on CGEP-ESC, while CGEP-MAVEN is 10.

B Details about Experimental Settings

Our method is implemented based on the pretrained RoBERTa-base model (Liu et al., 2019) with 768-dimension provided by HuggingFace transformers [5], and run PyTorch framework with CUDA on NVIDIA GTX 3090 GPU. We set the learning rate l_{tr} for the PLM to $5e-6$, the weight coefficient β to 0.5, and the temperature τ to 1.

We use the average MRR (Mean Reciprocal Rank) and Hit@n (Hit Rate at n) overall impressions as the evaluation metrics, which are widely used in prediction and retrieval tasks (Chepurova et al., 2023). For the *HIT@n* metric, given an input sample (G, n_c), if the model's top n predictions include n_c, then the model's prediction is deemed correct. Then the *HIT@n* calculation formula is as follows:

$$\textit{HIT@n} = \frac{1}{N}\sum_{i=0}^{N}\mathbb{I}(Rank_i \leq n)$$

where N means the total number of the data. For the *MRR* metric, denote the predicted ranking of n_c of the i-th sample among the candidate events as $Rank_i$, then the calculation formula for the metric is as follows:

$$\textit{MRR} = \frac{1}{N}\sum_{i=0}^{N}\frac{1}{Rank_i}$$

In this task, we employ *MRR*, *HIT@1*, *HIT@3*, *HIT@10*, and *HIT@50* to measure the excellence of a model.

C GPT-3.5-Turbo Prompt Detail

We evaluate GPT-3.5-turbo performance on the CGEP task under zero-shot settings. Figure 6 illustrate a demonstration of GPT-3.5-turbo reasoning process. We first provide a formal definition of the causal event graph, then sequentially concatenate

5. https://github.com/huggingface/transformers

MAVEN-ERE	Docs	The Length of Event Chain											Sum
		3	4	5	6	7	8	9	10	11	12	13	
Train	1552	2660	739	233	91	41	16	8	5	2	1	1	**3797**
Valid	258	454	119	43	19	7	2	2	1	1	1	2	**651**
Test	258	454	140	50	20	5	1	0	0	0	0	0	**670**
Sum	**2068**	**3568**	**998**	**326**	**130**	**53**	**19**	**10**	**6**	**3**	**2**	**3**	**5118**

Table 5: Specific statistics of different division sets of the CGEP-MAVEN dataset for the *script event prediction*.

Demonstration of GPT-3.5-turbo' s reasoning

Description: We annotated the events and their causal relationships in the given document and constructed an event causality graph with the events as nodes and the causal relationships as edges.

Input: The Expedition of the Thousand (Italian `` Spedizione dei Mille '') was an event of the Italian Risorgimento that took place in 1860 . a corps of volunteers led by giuseppe garibaldi sailed from quarto , near genoa (now quarto dei mille) and landed in marsala , … … an independent Sicily as part of the kingdom of Italy , and for common people , land distribution and the end of oppression.

Candidates: [Moving , turned , released , growing , becomes , expanding , sworn , winning , recovered , said , death , refused , standing , acknowledged … … returning , seized , besieged , sentenced , cutting].

Question: We have an event causality graph : ['bolster', 'cause', 'conquer'], ['Expedition', 'cause', 'end'], ['bolster', 'cause', 'venture'], ['bolster', 'cause', 'sailed'], ['decided', 'cause', 'venture'], ['bolster', 'cause', 'distribution'], ['bolster', 'cause', 'Expedition'], ['bolster', 'cause', 'landed'], ['Expedition', 'cause', 'distribution'], ['bolster', 'cause', 'end'], ['landed', 'cause', 'conquer'] , and what are the consequences of ' bolster ' ? Please select the 50 most likely event outcomes from the candidate events based on their probability . Note that only the event list needs to be generated.

Response: ['destroyed', 'killed', 'death', 'damaged', 'damage', 'obliterated', 'annihilated', 'crushed', 'deaths', 'losses', 'destroying', 'sinking', 'degenerating', 'tore', 'wounded', 'flooding', **'removing'**, 'dying', 'struck', 'burned', 'collapsing', 'erupted', 'crashing', 'crushed by', 'eliminated', 'collapsed', 'collapsed in', 'exploded', 'crushing', 'bash', 'collapsed into', 'fall', 'shattered', 'combusted', 'decimated', 'kill', 'terminating', 'falling', 'obliterate', 'eradicated', 'blow up', 'torn', 'sink', 'burst', 'demolished', 'crash', 'shatter', 'burn', 'fall into', 'break', 'run over']

Label: removing

Figure 6: A demonstration of GPT-3.5-turbo reasoning CGEP task.

the text of each event in the event graph according to their indices to obtain the textual input. Simultaneously, we include the candidate events as input and linearize the event graph based on the weights of triplet. Finally, the model is queried with: "What are the subsequent events of *Anchor Event*?" and then asked to select the top 50 most likely events from the candidate set.

Note that when calculating the metrics, we remove events from the generated list that are not in the candidate set before computing the Hit@n metric. Additionally, when calculating MRR, if the golden event is not in the generated list, we uniformly assign it a rank of 256/512. Therefore, when generating the list, we typically allow it to generate more than 50 events, e.g. 60.

D The Datasets Process of SEP task

In this section, we will provide a detailed description of the data preprocessing for the *script event prediction* task. For each CGEP-MAVEN instance, which contains an event causality graph, an anchor event, a candidate event set, and aground truth consequential event, we extract the longest event causality chains from the graph that terminate at the respective consequential nodes, and the other property, such as the candidate event set, the anchor event, will be maintained from the instance.

Note that for each data instance, the maximum number of event chains extracted is limited to 1. Additionally, we remove any event chains containing fewer than 2 nodes. Ultimately, we divide the documents into training, validation, and test sets, with a split ratio of 75%, 12.5%, and 12.5%, respectively. Table 5 summarizes the statics of final processed dataset for the task.