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

Abstract

Predicting the consequences based on some past events has a huge potential in various Natural Language Processing applications. However, existing work faces two shortcomings: (1) Simple modeling scenarios, such as Script Event Prediction task, which predict subsequent events only based on an event chain; (2) Interpolation scenarios, such as Event Knowledge Graph Completion task, where the predicted event has already occurred in known events. In this paper, we propose a new task named Event Causality Graph Prediction, which forecast the consequence event based on an event causality graph constructed from a document describing complex event scenarios. To that end, we propose two corresponding datasets and an **G**raph **C**ontrastive **P**rompt **L**earning model(GCPL), which utilize the benefits of graph prompt learning and introduce the Dual Encoder to integrate node text and graph structure information. We conduct extensive experiments on two datasets and our GCPL achieves state-of-the-art performance among all competitors.

1 Introduction

What Happens Next? Predicting the potential consequences that may arise by knowing some past events is of great significance to various Natural Language Processing applications, e.g. Sentiment Analysis (Zhang et al., 2022), Dialogue Systems (Tang et al., 2021b; Chen et al., 2017), Planning Decisions (Arnold and Sally, 1997).

Script Event Prediction (SEP) (Zhou et al., 2022a; Jans et al., 2012), as a classic event prediction task, aims to predict subsequent event from a set of candidate events, based on a simple event script[1] (Granroth-Wilding and Clark, 2016; Huang et al., 2021). For example, given the script "read menu – order food – take food – eat foot", the model is required to predict "pay the food" as the subsequent event from the candidate set. However, the task focuses on simple scenarios, just a single event chain, which fails to fully capture the interactions between events, making this task unsuitable for complex event modeling.

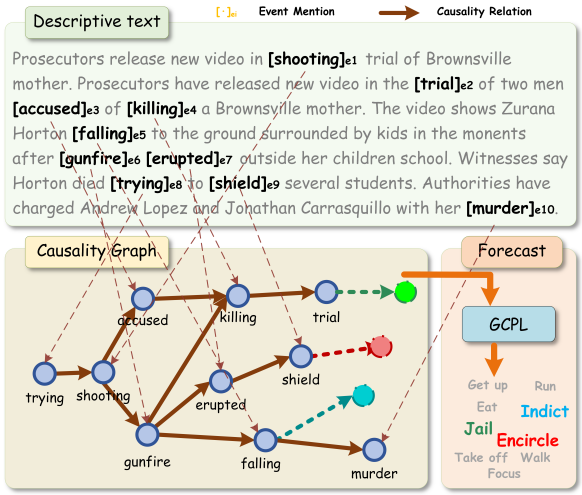

Figure 1: An example of Consequence Event Prediction based on Causality Event Knowledge Graph.

Once event interactions become complex, they are often represented in the form of graph, known as event knowledge graph (EKG) (Gottschalk and Demidova, 2018; Guan et al., 2022). *Event Knowledge Graph Completion* (EKGC) (Wang et al., 2022a), similar to event prediction, aims to predict missing events in instance graphs from the schema graph[2]. However, the candidate events for this task may have already appeared in the schema graph, meaning that EKGC is an interpolation task. Furthermore, schema graph needs to be derived from multiple instance graphs, which is known as Event Schema Induction task (Chambers, 2013; Huang et al., 2021; Li et al., 2020), but EKGC utilizes it solely as prior knowledge to guide the model to make predictions. It is worth noting that EKGC is inspired by *Knowledge Graph Completion* (KGC) (Lv et al., 2022; Lovelace et al., 2021), but instead of completing a single entity knowledge graph, EKGC focuses on completing multiple event instance graphs.

To address the aforementioned shortcomings, this paper introduces a new task called **C**ausality **G**raph **E**vent **P**rediction (CGEP). As illustrated in Fig. 1, CGEP aims to forecast the consequence

[1]Script refers to a standardized event sequence pattern that commonly occurs in a particular situation or field. It describes the relationships and sequential order between events.

[2]Event schema graph serves as a general and abstract representation of a particular type of complex event.

event based on an event causality graph, which is constructed from a document describing complex event scenarios. And the predicted event does not appear in the known information, such as the causality graph, indicating that our task involved extrapolation.

Furthermore, in order to tackle this task, we construct two corresponding datasets, named MAVEN-GEP and ESG, containing 14667 and 1378 pieces of data respectively. Simultaneously, we propose the **G**raph **C**ontrastive **P**rompt **L**earning model(GCPL), based on graph prompt learning. Specifically, we introduce the Dual Encoder module to integrate node text and graph structure information effectively. Secondly, we leverage the benefits of contrastive learning to guide PLM to understand the potential differences between golden events and other events. We conducted an extensive experiments on the proposed datasets, and the results demonstrate that our model achieve state-of-the-art performance.

Our main contributions can be summarized as follows:

1. We propose a more challenging but practical **C**ausality **G**raph **E**vent **P**rediction (CGEP) task, representing the first migration of event prediction to event causality graphs.

2. We create two suitable datasets and propose an effective baseline model GCPL, based on graph prompt learning, for this new task.

3. Extensive experiments on two datasets demonstrate the superior performance of our model, indicating that GCPL can serve as a robust baseline.

2 Event Causality Graph Prediction

2.1 Task Descriptions

The ECGP task is defined as predicting some *consequential events* that are most likely to happen next, given some past events and their causal relations.

We construct an *event causality graph* (ECG) consisting of past events as nodes and their causal relations as directed edges. Let $G = (V, E)$ denote the ECG, where an event node $e_i \in V$ contains the event mention and $(e_i, e_j) \in E$ is a directed edge from the event e_i to the event e_j, indicating that e_i causes e_j, i.e. $e_i \rightarrow e_j$. Notice that an ECG G is a *directed acyclic graph*. We call a node $e_t \in G$ as a *tail node*, if there does not exist an edge starting from e_t to any other node in the ECG. We assume to have a candidate set of all possible consequential events. Let $\mathcal{C}$ denote the candidate set, where $c \in \mathcal{C}$ is a consequential event containing its mention word(s). The objective of the ECGP task is to select the most likely candidate event $c^* \in \mathcal{C}$ for a tail node $e_t \in G$.

Figure 1 illustrates an ECG example, which is constructed based on the annotated *event mentions* (nodes) and annotated *events' causal relations* (directed edges) in a document.

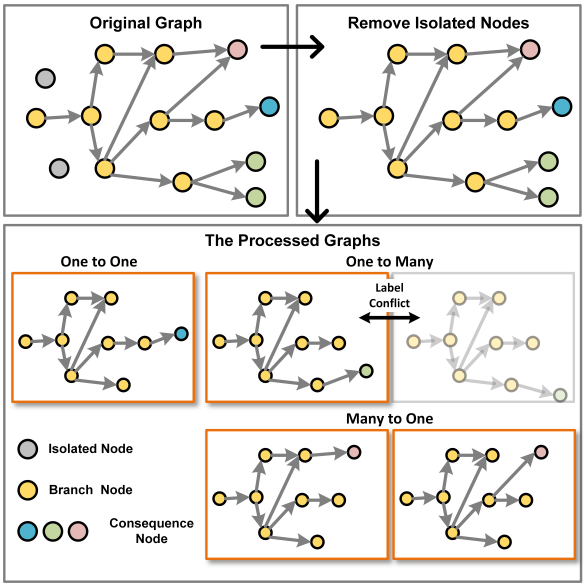

Figure 2: Data Processing Flowchart: The data processing involves transforming an original EKG into multiple data instances, with each instance specifically predicting a single leaf event.

2.2 Task Datasets

We construct two ECGP datasets based on the public dataset the MAVEN (Wang et al., 2022d) and Event StoryLine Corpus (ESC v0.9) (Caselli and Vossen, 2017), in which annotations for events' mentions and relations are available on a per document basis.

Construct an instance ECG Based on the annotations, we first construct an *original* ECG G_o for each document. Notice that some events in a document have no causal relations to other events, i.e., they are isolated events. Based on G_o, we construct an *instance* ECG G_i by removing all isolated event nodes in G_o and also removing all tail nodes and their corresponding edges in G_o.

Construct a data instance We construct a data instance containing a *prophetic event* and its be-

Dataset	Topic	Doc.	Relation				Event Nodes		
			Causal	Temporal	Coreference	Sub-Event	Leaf Node	Isolated Node	Branch node
MAVEN-ERE	90	4480	57992	1216217	103193	15841	23006*	43032*	18247*
ESC	22	258	1770	8111	1032	-	1193	2266	1875

Table 1: Statistics of MAVEN-ERE and ESC dataset.

	MAVEN-GEP				ESG			
	Train	Valid	Test	Total	Train	Valid	Test	Total
Nodes	9.8	9.6	9.9	9.8	10.5	8.7	10.3	10.3
Edges	6.8	7.1	8.4	7.3	15.7	3.3	14.7	7.0
Sample	8735	2167	3765	14667	976	143	259	1378

Table 2: Statistics of our processed dataset. ESG takes fold 1 as an example.

longing instance ECG. Note that a prophetic event is also a tail node in G_i. Three cases need to be considered in the original graph G_o: (1) One-to-One: Only one event e_i causes e_t and e_i does not cause any other tail event, then a data instance is created with e_i as a prophetic event in G_i. (2) One-to-Many: Only one event e_i causes e_t and e_i also causes other tail event $e_{t'}$. We only choose one of such tail nodes for e_i to construct a data instance. (3) Many-to-One: Two or more events $\{e_i\}$ cause e_t, each of which is as a prophetic event to construct one data instance.

Construct a candidate set We construct a candidate set for each prophetic event as follows: We include the event mention of the corresponding tail node $e_t \in G_o$ with $e_i \rightarrow e_t$ as a candidate event (i.e., the ground truth), and randomly select the rest candidate events from all other tail nodes in all original ECGs of the dataset. In particular, we select in total 512 candidate events from the MAVEN-ERE and 256 from the EventStoryLine v0.9 (ESC) to form a candidate set of for each prescient event.

Prevent answer leakage There is a possibility of answer leakage where the mention of a candidate event also appears in the contextual sentence. To deal with answer leakage, we replace the token of a candidate event mention by a special token [PAD] in the contextual sentence. For multi-tokens and discontinuous event mentions, we only replace the tokens in the corresponding positions.

Dataset statistics We divide all the data instances into the training, validation, and test set. Table 2 presents the statistics of the constructed ECGP datasets, namely, the MAVEN-ECGP and ESC-ECGP dataset.

3 GCPL Model

Fig. 3 illustrates our GCPL model, including the dual encoder module, the event classification module and the event contrast module.

3.1 Dual Encoder

To enjoy both advantages of event causality graph structure and document contextual semantic, we propose a dual encoder to learn a representation for past events and their causal relations.

Event Text Encoder Given the description text $\mathcal{S} = [s_i]_{t=1}^{L_\mathcal{S}}$ of event e_i, where $L_\mathcal{S}$ is the length of the text and $\mathcal{S}$ contains the tokens of e_i, to indicate the location of event mention in the text, we incorporate virtual locators <c> and </c> on both sides of the mention, i.e.

$$\mathcal{S} = [s_1, s_2, ..., < c >, e_i, < /c >, ..., s_{L_\mathcal{S}}]$$

Then, we take the sequence $\mathcal{S}$ as input to get the embedding matrix H_T of the text $\mathcal{S}$:

$$H_T = [h^T_{s_1}, h^T_{s_2}, ...] = TEncoder(\mathcal{S})$$

where *TEncoder* represents the text encoder and we take $h^T_{e_i}$ as the textual representation of event e_i.

Graph Structure Encoder For a given graph $G = (V, E)$, it contains only one type of edge, i.e. $(e_i, \texttt{causes}, e_j) \in E$. To fully model the directionality of causality graph, we randomly transform half of edges in E of type "causes" into edges of type "causedBy", i.e. $(e_i, \texttt{causes}, e_j) \rightarrow (e_j, \texttt{causedBy}, e_i)$. At the same time, we also regard non-causal event pairs as a type of edge, denoted as None, i.e. $(e_i, \texttt{none}, e_k)$.

To construct the graph prompt templates T, we employ the three types of edges mentioned above with the same ratio and random order:

$$\begin{aligned} T = [\texttt{CLS}] + e_1 \ \texttt{causedBy} \ e_2 [\texttt{SEP}] + ... \\ + e_i \ \texttt{none} \ e_j \ [\texttt{SEP}] \\ + e_c \ \texttt{causes} \ [\texttt{MASK}] \ [\texttt{SEP}]. \end{aligned}$$

Once template *T* exceeds the length limit of PLM, we will randomly discard a portion of the triples.

In order to integrate text information into graph structure information, we first input the template T into the encoder's embedding layer:

$$\begin{cases} H_{TE}, H_{PE}, H_{SE} = GEncoder.embedding(T) \\ GEncoder = \texttt{DeepCopy}(TEncoder) \end{cases}$$

where H_{TE}, H_{PE}, H_{SE} are the PLM-specific vector matrix, named Token Embedding, Position Embedding, Segment Embedding(Devlin et al., 2018). And DeepCopy means that the parameters of *GEncoder* come from the deep copy of *TEncoder*, that is, they are trained separately.

Then, we replace the embedding of event e_i in H_{TE} with $h^T_{e_i}$ to get $H^\dagger_{TE}$, where $h^T_{e_i}$ is the textual representation of event e_i mentioned above:

$$H^\dagger_{TE} = [h^{TE}_{[\texttt{CLS}]}, ..., \boldsymbol{h^T_{e_i}}, ..., h^{TE}_{[\texttt{MASK}]}, h^{TE}_{[\texttt{SEP}]}]$$

Finally, we use $H^\dagger_{TE}, H_{PE}, H_{SE}$ as subsequent inputs to obtain the embedding matrix H_{GT}:

$$H_{GT} = GEncoder.encode((H^\dagger_{TE}, H_{PE}, H_{SE}))$$

We then use the [MASK] vector $h^{GT}_{[\texttt{MASK}]}$ for subsequent prediction.

3.2 Event Classification Module

After the PLM encoding, we utilize the hidden state $h^{GT}_{[\texttt{MASK}]} \in R^{\mathbb{D}}$ of the input [MASK] to classify the entire vocabulary $\mathcal{V}$ and predict the final result. That is:

$$P([\texttt{MASK}] = e_i \in \mathcal{V} | h^{GT}_{[\texttt{MASK}]})$$

To address multi-token events in the candidate set, we replace them with a single virtual token <A$_i$> before training, and their initial embeddings are obtained by averaging the vectors of each token in the event mention. Subsequently, we apply a softmax layer to normalize the predicted probabilities of the candidate set events:

$$P(e_i \in \mathcal{C} | h^{GT}_{[\texttt{MASK}]}) = \frac{\exp((p_{e_i}))}{\sum_{j=1}^{|\mathcal{C}|} \exp((p_{e_j}))}$$

Then, we select the event with the highest probability from the candidates as the final prediction result.

During the training stage, we utilize cross-entropy loss to calculate the loss of the module:

$$\mathcal{L}_P = -\frac{1}{K} \sum_{i}^{\mathcal{N}} \sum_{j=1}^{|\mathcal{C}|} \mathbf{y}_i^j \log(\hat{\mathbf{y}}_i^j) + \lambda \|\theta\|^2$$

where $\mathbf{y}_i^j$ represents the true value of the j-th candidate event of the i-th data, and $\hat{\mathbf{y}}_i^j$ represents the corresponding prediction probability of the model. λ and θ are the regularization hyper-parameters.

3.3 Event Contrast Module

Due to the large number of events in the candidate set, relying solely on [MASK] for classification becomes challenging, particularly when the event graph is sparse and the training data is limited. To address this issue, we perform semantic comparison between the [MASK] vector and the candidate event vector in the semantic space.

During the training stage, for the event $e_{c_i} \in \mathcal{C}$, we first utilize the embedding layer of the PLMs to obtain its embedding vector $h_{e_{c_i}}$:

$$h_{e_{c_i}} = \text{Embedding}(e_{c_i})$$

To fully consider the semantic information of the candidate set events, we employ the [MASK] embedding as the anchor sample, the label event as a positive sample, and the remaining candidate events as negative samples for semantic comparison:

$$\begin{cases} D_A = Linear(h^{GT}_{[\texttt{MASK}]}) \\ D_+ = Linear(h_{e_g}) \\ D_- = Linear(h_{e_{c_i}}), h_{e_{c_i}} \in \mathcal{C}/e_g \end{cases}$$

Then, we utilize Supcon(Khosla et al., 2020) to compute the contrastive loss as follows:

$$\mathcal{L}_C = -\log \sum_{i=1}^{\mathcal{N}} \frac{\exp(sim(D^i_A, D^i_+)/\tau)}{\sum_{d_j \in \mathcal{D}_i} \exp(sim(D^i_A, d_j)/\tau)}$$

where sim uses cosine similarity and $\mathcal{D}_i = \{D^i_+\} \cup \{D^i_{-j}\}_{j=1}^{|\mathcal{C}|-1}$ means the embedding of candidate events. τ is a hyper-parameter used to flatten the similarity between anchor sample and positive and negative samples.

3.4 Training Strategy

To jointly train the model, we linearly combine the losses of the two modules using the hyperparameter β. Make the model adapt to this task in both vocabulary space and semantic space. Then, we get the final loss as follows:

$$\mathcal{L}_{total} = \mathcal{L}_P + \beta * \mathcal{L}_C$$

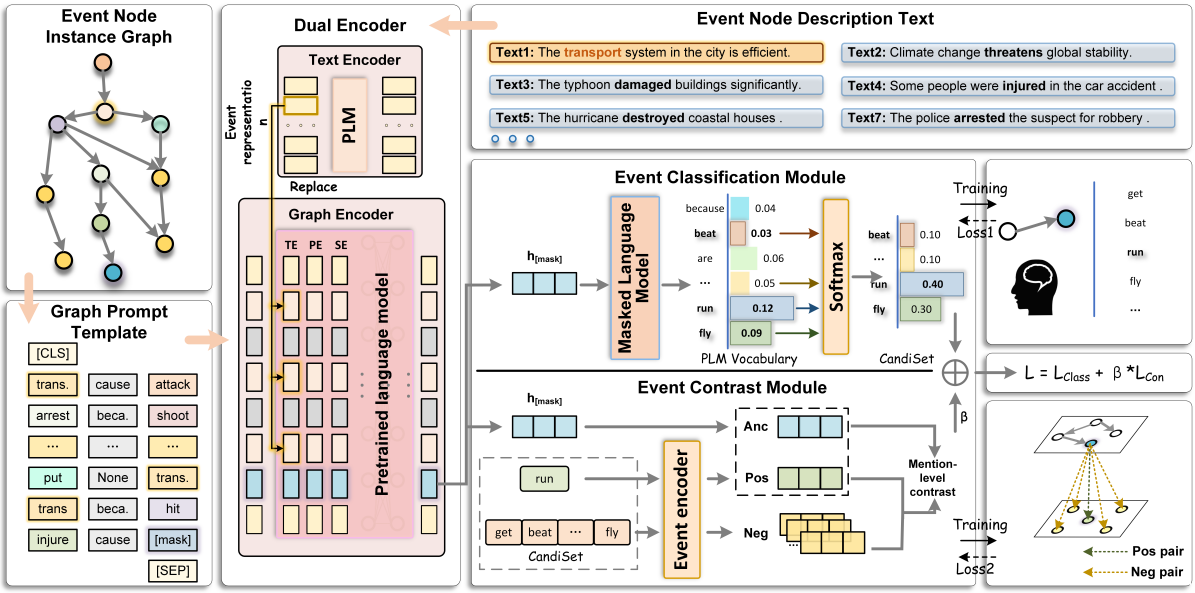

Figure 3: Illustration of GCPL. TE, PE, SE means **T**oken **E**mbedding, **P**osition **E**mbedding, **S**egment **E**mbedding respectively.

where β is a weight coefficient utilized to balance the prompt loss and contrast loss. During the validation and test phases, as the event contrast operates solely on the semantic space, we rely on the prompt module alone to predict the final consequence event.

4 Experiments

4.1 Experiments Setup

Dataset Setting As the amount of processed ESG data is limited, we take measures to ensure the validity of the experimental results. Specifically, following (Zhao et al., 2021; Liu et al., 2021), we designate the last two topics as the development set, while the remaining 20 topics are used for conducting 5-fold cross-validation experiments. In the case of MAVEN-GEP, as the original dataset does not provide test set labels, we designate the validation set as the test set. Additionally, we allocate 20% of the data from the train set to create a separate validation set.

Parameter Setting We implement the overall model under the pytorch framework of Huggingface Transformer (Wolf et al., 2020). We use RoBERTa-base (Liu et al., 2019) as Dual encoders while training the entire model on NVIDIA GTX 3090 GPUs. We optimize the entire model using the AdamW (Loshchilov and Hutter, 2017) optimizer, with a learning rate of 1e-6, 5e-6, 1e-5. We perform early stop with the loss ratio β setting to 0.5 and τ setting to 1.

4.2 Competitors

We selected four models with the most advanced performance in KGC and SEP tasks for comparison. Since the original task models are not designed to handle graph-structured data, we transform the instance graph into a one-dimensional sequence as input, similar to GCPL.

- CSProm-KG (Chen et al., 2023) constructs soft parameters to fuse the graph structure and text information for graph completion.
- SimKG (Wang et al., 2022b) employs three types of negative examples in contrastive learning to effectively complete the graph.
- BARTbase (Zhu et al., 2023) designs an event-centered pre-training target to fine-tune PLM, and then predicts subsequent events.
- MCPredictor (Bai et al., 2021) leverages the transformer architecture to integrate event-level and script-level information for script event prediction.

4.3 Overall Results

Table 3 compares the overall performance between our GCPL and the competitors on the MAVEN-GEP and ESG datasets.

Text-Enabled : We can first observe that: **(1)** GCPL achieved outstanding results in both datasets,

Model	MAVEN-GEP					ESG				
	MRR	Hit@1	Hit@3	Hit@10	Hit@50	MRR	Hit@1	Hit@3	Hit@10	Hit@50
CSProm-KG	22.3(5.4)	17.8(3.8)	22.2(4.2)	31.2(7)	50.9(20.4)	14.8(14.2)	9.1(8.6)	14.7(13.9)	28.7(26.2)	44.5(42.1)
SimKG	8(7.3)	3.6(3.5)	8.2(6.9)	15.2(13.2)	32.4(28.7)	14.7(13.1)	7.7(5.4)	16.6(14.7)	25(23.8)	40(38.4)
BARTbase	22.5(5.5)	17.7(3.7)	22.1(5)	33.3(7.9)	52.5(21.4)	14.5(11.3)	7.9(5.4)	14.9(11.8)	29.2(24.8)	44.7(41.1)
MCPredictor	16.1(5.7)	11.3(2.8)	16.4(4.7)	25.9(11.3)	43.1(25.3)	14.1(9.2)	6.1(3)	15.5(9.3)	28.1(20.8)	41.7(37.6)
GCPL (RoBERTa)	**30.3**(10.7)	**24.5**(8.7)	**31.7**(10.1)	**40.8**(13.4)	**58.2**(26.2)	**20.5**(19.3)	**15.4**(9.6)	**24**(22.4)	**32.8**(31.8)	**49.1**(47.2)

Table 3: Overall results of comparison models on the MAVEN-GEP and ESG dataset. •(○) represents the results of GCPL with and without text settings.

outperforming other baseline models significantly. We attribute its excellent performance to the Dual Encoder architecture, avoiding coding conflicts in PLM adaptation to graph structure and sentence coding. **(2)** The performance of CSProm-KG surpasses that of SimKG by a significant margin. This can be attributed to the incorporation of soft prompt parameters in each layer during the training process in CSProm-KG. **(3)** Despite BARTbase and MCPredictor outperforming CSProm-KG and SimKG, their performance still falls short of GCPL. This can be attributed to BARTbase and MCPredictor models' limited consideration of a single event script, resulting the valuable information in the causality graph remaining underutilized.

Text-Disabled : We can observe that: **(1)** Even without textual information,GCPL still achieved optimal performance. We attribute this to the comparison of event semantics, which shortens the distance between [MASK] and the golden event and distances it from other events in the semantic space. **(2)** The SimKG model achieved optimal Hit@50 performance on the MAVEN-GEP. This can be attributed to the contrastive learning paradigm's compatibility with scenarios involving numerous negative examples and the use of a candidate set size of 512 in MAVEN-GEP. **(3)** BARTbase and MCPredictor perform slightly worse than CSProm-KG and SimKG due to the limited candidate set of 5 events in the SEP task, posing challenges for accurate predictions when faced with hundreds of candidates.

Finally, as shown in B, our GCPL with all three PLMs have achieved better performance than the competitors. We attribute its outstanding performance to graph prompt learning that enables PLM to effectively encode graph structure information, and it also proves that GCPL can serve as an effective baseline model for the task.

4.4 Ablation Study

To evaluate the impact of GCPL's main modules on experimental performance, we conduct some ablation experiments on two datasets. As shown in table 4:

• GCPL W/o Dual Encoder: Based on GCPL full, remove the Dual Encoder module.

• GCPL W/o Event Contrast: Based on GCPL full, remove the Event Contrast module.

• GCPL W/ Pearson Correlation Coefficient: The Pearson Correlation coefficient is a statistic that measures the linear relationship between two variables.

• GCPL W/ Euclidean Distance: Euclidean distance uses the square root of the sum of squared differences between corresponding positions of vectors.

• GCPL W/ Manhattan Distance: Manhattan distance measures vector distance by summing the absolute differences between them.

Module Ablation: In the first group, we can observe that Dual-encoder architecture outperforms the single-encoder architecture, emphasizing its ability to separate graph structure encoding and sentence encoding to avoid confusion. Additionally, removing the cosine similarity loss results in a notable performance decrease, indicating that the absence of similarity reduces the distance to the correct label while increasing the distance from the negative label.

Similarity Ablation: In the second group, cosine similarity achieves the best performance among the four similarity losses, while Manhattan similarity performs the worst. This is because cosine similarity is particularly effective in high-dimensional vector spaces, as it focuses solely on the direction of the vectors and disregards their length. Furthermore, the non-differentiability of the Manhattan distance function at points other than the origin prevents the direct use of conventional gradient descent algorithms for parameter updates.

Model	MAVEN-GEP					ESG				
	MRR	Hit@1	Hit@3	Hit@10	Hit@50	MRR	Hit@1	Hit@3	Hit@10	Hit@50
GCPL Full	**30.3**	**24.5**	**31.7**	**40.8**	**58.2**	**20.5**	**15.4**	**24**	**32.8**	**49.1**
GCPL W/o Dual Encoder	23.1	17.6	23.9	33.3	53.5	15.7	8.2	15.8	30.1	44.7
GCPL W/o Event Contrast	27.2	22.3	28.1	38.1	55.6	16.6	10.5	17.8	32.6	45.2
GCPL W/ Pearson Correlation Coefficient	22.3	17.7	22.3	31.2	49.9	20.7	15.1	23.1	36.8	48.4
GCPL W/ Euclidean Distance	27.3	22.1	28.2	36.1	55.9	18.7	11.8	23	30.1	49.4
GCPL W/ Manhattan Distance	2.7	0.7	1.7	4.6	19.5	2.3	0.5	0.7	3.6	18.9

Table 4: Ablation experiment in text setting. The three block are module ablation and similarity ablation respectively.

4.5 Effect of Loss Ratio

To assess the influence of the two losses on model performance, we evaluate the performance using different loss ratio to observe their effects. As shown in Fig 4, GCPL achieves the best performance when the loss ratio is set to 0.5. The model performance gradually decreases when the ratio $\beta \in [0, 0.5)$ or $\beta \in (0.5, 1.5]$. This is because the model's prediction of correct events mainly relies on prompt loss, and contrast loss only allows the model to distinguish golden events from other events in the semantic space. When the ratio is set to 0, it indicates that the model lacks the ability to pre-differentiate events in the semantic space, leading to a degradation in performance.

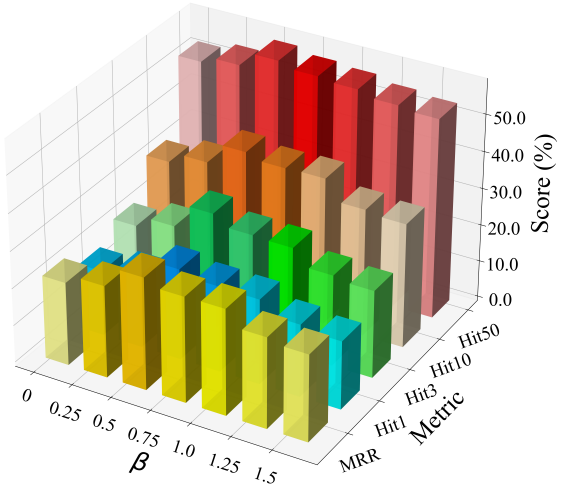

Figure 4: Results on MAVEN-GEP with different loss ratio β.

4.6 Few Shot

In real-world scenarios, dataset annotation can be prohibitively expensive. Following (Sharma et al., 2023; AlKhamissi et al., 2022), we decided to assess the effectiveness of GCPL in low-resource scenarios. Figure 5 shows the comparison results between GCPL under low resources and BARTbase under full resources. We can observe that GCPL with 60% of the data achieves comparable performance to BARTbase with 100% of the data. Furthermore, when the data volume is reduced from 100% to 40%, GCPL's Hit@50 only experiences an 8.5% decrease, while the MRR drops by 7.9%. This suggests that GCPL significantly reduces the reliance on the data volume.

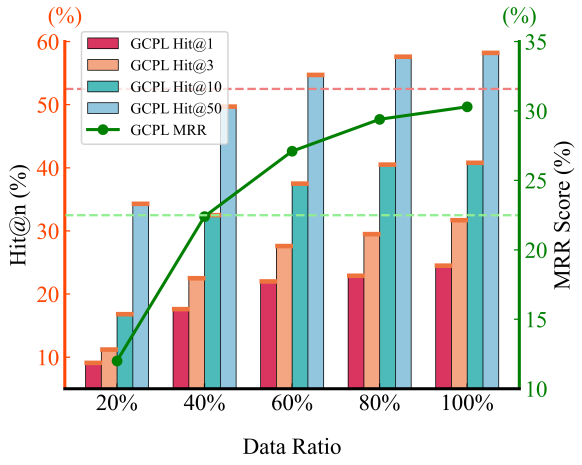

Figure 5: Results on MAVEN-GEP in low-resource scenarios. The orange and green dashed lines represent BARTbase's Hit@50 and MRR respectively in the full resource scenario.

4.7 Visualization

To fully assess the contrast loss's efficacy in distinguishing golden events from other negatives in the semantic space, we visually display the semantic distances between [MASK] and golden event representation. As shown in Fig 6, the cosine similarity of the two vectors is significantly higher in the model with the Event Contrast module compared to the model without it (The first three rows are brighter than the last three rows). Additionally, we incorporate the module into BARTbase and CSProm-KG models, and the semantic trend remains the same. This confirms the effectiveness of

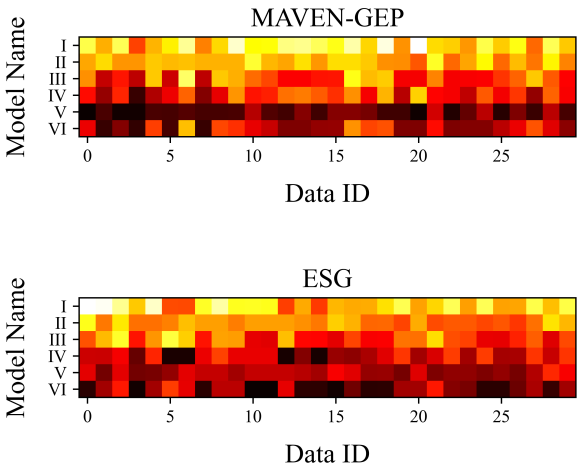

Figure 6: Visualization of semantic cosine similarity of [MASK] and golden events on MAVEN-GEP and ESG. We compare GCPL with BARTbase and CSProm-KG. I-VI respectively represent GCPL W/EC, GCPL Wo/EC, BART W/EC, BART W/o EC, CSProm W/EC, CSProm W/o EC. EC means our Event Contrast Module. Brighter colors indicate higher similarity.

the contrastive loss in distinguishing golden events from other negatives within the semantic space.

5 Related Work

5.1 Script Event Prediction

Script Event Prediction (Zhou et al., 2022b; Wang et al., 2021; Huang et al., 2021) focus on predicting future events based on a narrative event chain with shared entities. Currently, a common approach is to utilize the similarity between candidate events and script events.

For example, Granroth-Wilding and Clark (2016) use word2vec to obtain event representations, and then predict subsequent events based on the similarity between candidate events and script events. However, this approach does not consider the temporal relationship between events. To address this limitation, some studies (Pichotta and Mooney, 2016; Wang et al., 2017; Lv et al., 2019) employ Long Short-Term Memory (LSTM) to model the temporal dependencies between events. Du et al. (2022) use BERT to encode event text information, and use Graph Neural Network (GNN) to fuse the event graph information into the event representation. Zhu et al. (2023) employ Prompt Learning paradigm to train their neural network model and design a likelihood-based contrastive loss for fine-tuning.

5.2 Event Knowledge Graph Completion

Event Knowledge Graph Completion (Wang et al., 2022a) aims to predict whether a candidate event node from the schema graph (Dror et al., 2022; Li et al., 2023; Jin et al., 2022; Li et al., 2020) is missing for the instance graph.

Based on the different completion targets, EKGC can be categorized into two types of tasks: node completion and edge completion. Wang et al. (2022a) match event nodes in the instance graph to the event schema graph, model neighbor nodes of candidate nodes to predict missing event nodes. While Tang et al. (2021a) utilize LSTM and attention mechanism for the prediction of missing edges in the graph. Mirtaheri et al. (2023) presents an incremental training framework for event-centric KGC that addresses the issue of catastrophic forgetting. Certainly, EKG can be deployed for various downstream tasks as well (Mao et al., 2021; Zhang and Tang, 2022). Li and Liu (2022) utilize prior knowledge in the EKG, combine event scene representation and calculation of multiple prediction results to predict events.

Although many works utilize event evolution graph (Wang et al., 2022c; Gao et al., 2021; Hu et al., 2021) for event prediction, they typically treat it as an external knowledge base rather than directly predicting events within the event graph. We motivate our work against this aspect and predict consequence events based on event causality graphs and construct two corresponding datasets.

6 Conclusion and Future Work

In this paper, we propose the Causality Graph Event Prediction task that aims to forecast the consequence event based on an event causality graph. We design two datasets, an evaluation framework, and several baseline models for the task. And our model GCPL, based on graph prompt learning, achieved the best results among all competitors.

We identify a few directions for future work. First, we hope to be able to build larger and more complex causality graph datasets. Even MAVEN-ERE with 4480 documents ended up with only 14667 pieces of data. Secondly, we only considered event causality. We hope to add other event relationships in subsequent work while avoiding redundancy in the event graph. Finally, the baseline model GCPL we proposed did not fully cope with the sparsity of the event graph, so there is still a lot of room for improvement in this task.

7 Limitation

Our GCPL converts the graph structure into a one-dimensional sequence as the input of PLM. However, due to the input length limit of PLM, this solution easily loses the graph structure information. In addition, when designing the ECGP task, we believe that each piece of data has a consequence event. However, this is unreasonable. We need to add a None event, that is, this piece of data will not have any consequences.

Ethics Statement

This paper has no particular ethic consideration.

References

Badr AlKhamissi, Faisal Ladhak, Srini Iyer, Ves Stoyanov, Zornitsa Kozareva, Xian Li, Pascale Fung, Lambert Mathias, Asli Celikyilmaz, and Mona Diab. 2022. Token: Task decomposition and knowledge infusion for few-shot hate speech detection. *arXiv preprint arXiv:2205.12495*.

Wright Arnold and Wright Sally. 1997. The effect of industry experience on hypothesis generation and audit planning decisions. *Ssrn Electronic Journal*, 9.

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.

Nathanael Chambers. 2013. Event schema induction with a probabilistic entity-driven model. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1797–1807.

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.

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*.

Rotem Dror, Haoyu Wang, and Dan Roth. 2022. Zero-shot on-the-fly event schema induction. *arXiv preprint arXiv:2210.06254*.

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

Jianqi Gao, Xiangfeng Luo, and Hao Wang. 2021. An uncertain future: Predicting events using conditional event evolutionary graph. *Concurrency and Computation: Practice and Experience*, 33(9):e6164.

Simon Gottschalk and Elena Demidova. 2018. Eventkg: A multilingual event-centric temporal knowledge graph. In *The Semantic Web: 15th International Conference, ESWC 2018, Heraklion, Crete, Greece, June 3–7, 2018, Proceedings 15*, pages 272–287. Springer.

Mark Granroth-Wilding and Stephen Clark. 2016. What happens next? event prediction using a compositional neural network model. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30.

Saiping Guan, Xueqi Cheng, Long Bai, Fujun Zhang, Zixuan Li, Yutao Zeng, Xiaolong Jin, and Jiafeng Guo. 2022. What is event knowledge graph: A survey. *IEEE Transactions on Knowledge and Data Engineering*.

Wenjie Hu, Yang Yang, Ziqiang Cheng, Carl Yang, and Xiang Ren. 2021. Time-series event prediction with evolutionary state graph. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, pages 580–588.

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.

Bram Jans, Steven Bethard, Ivan Vulic, and Marie-Francine Moens. 2012. Skip n-grams and ranking functions for predicting script events. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2012)*, pages 336–344. ACL; East Stroudsburg, PA.

Xiaomeng Jin, Manling Li, and Heng Ji. 2022. Event schema induction with double graph autoencoders. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2013–2025.

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.

Manling Li, Qi Zeng, Ying Lin, Kyunghyun Cho, Heng Ji, Jonathan May, Nathanael Chambers, and Clare Voss. 2020. Connecting the dots: Event graph schema induction with path language modeling. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 684–695.

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

Yanhao Li and Wei Liu. 2022. Sudden event prediction based on event knowledge graph. *Applied Sciences*, 12(21):11195.

Jian Liu, Yubo Chen, and Jun Zhao. 2021. Knowledge enhanced event causality identification with mention masking generalizations. In *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence*, pages 3608–3614.

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*.

Justin Lovelace, Denis Newman-Griffis, Shikhar Vashishth, Jill Fain Lehman, and Carolyn Penstein Rosé. 2021. Robust knowledge graph completion with stacked convolutions and a student re-ranking network. In *Proceedings of the conference. Association for Computational Linguistics. Meeting*, volume 2021, page 1016. NIH Public Access.

Shangwen Lv, Wanhui Qian, Longtao Huang, Jizhong Han, and Songlin Hu. 2019. Sam-net: Integrating event-level and chain-level attentions to predict what happens next. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 6802–6809.

Xin Lv, Yankai Lin, Yixin Cao, Lei Hou, Juanzi Li, Zhiyuan Liu, Peng Li, and Jie Zhou. 2022. Do pretrained models benefit knowledge graph completion? a reliable evaluation and a reasonable approach. Association for Computational Linguistics.

Qianren Mao, Xi Li, Hao Peng, Jianxin Li, Dongxiao He, Shu Guo, Min He, and Lihong Wang. 2021. Event prediction based on evolutionary event ontology knowledge. *Future Generation Computer Systems*, 115:76–89.

Mehrnoosh Mirtaheri, Mohammad Rostami, and Aram Galstyan. 2023. History repeats: Overcoming catastrophic forgetting for event-centric temporal knowledge graph completion. *arXiv preprint arXiv:2305.18675*.

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.

Arushi Sharma, Abhibha Gupta, and Maneesh Bilalpur. 2023. Argumentative stance prediction: An exploratory study on multimodality and few-shot learning. *arXiv preprint arXiv:2310.07093*.

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

Zhiwen Tang, Hrishikesh Kulkarni, and Grace Hui Yang. 2021b. High-quality diversification for task-oriented dialogue systems. *arXiv preprint arXiv:2106.00891*.

Hongwei Wang, Zixuan Zhang, Sha Li, Jiawei Han, Yizhou Sun, Hanghang Tong, Joseph P Olive, and Heng Ji. 2022a. Schema-guided event graph completion. *arXiv preprint arXiv:2206.02921*.

Liang Wang, Wei Zhao, Zhuoyu Wei, and Jingming Liu. 2022b. 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. 2022c. 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. 2022d. 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.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations*, pages 38–45.

Wenxuan Zhang, Xin Li, Yang Deng, Lidong Bing, and Wai Lam. 2022. A survey on aspect-based sentiment analysis: Tasks, methods, and challenges. *IEEE Transactions on Knowledge and Data Engineering*.

Yaru Zhang and Xijin Tang. 2022. Introducing trigger evolutionary graph and event segment for event prediction. In *International Symposium on Knowledge and Systems Sciences*, pages 186–201. Springer.

Kun Zhao, Donghong Ji, Fazhi He, Yijiang Liu, and Yafeng Ren. 2021. Document-level event causality identification via graph inference mechanism. *Information Sciences*, 561:115–129.

Bo Zhou, Yubo Chen, Kang Liu, and Jun Zhao. 2022a. Script event prediction via multilingual event graph networks. *ACM Transactions on Asian and Low-Resource Language Information Processing*, 22(2):1–17.

Pengpeng Zhou, Bin Wu, Caiyong Wang, Hao Peng, Juwei Yue, and Song Xiao. 2022b. 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.

A Performance Metrics

We use *HIT@n* and *Mean Reciprocal Ranking (MRR)* as evaluation metrics in this task. For the *HIT@n* metric, given an input sample $(\mathcal{G}_i, e_e^i)$, if the model's top n predictions include e_e^i, then the model's prediction is deemed correct. If the total number of data is N, then the *HIT@n* calculation formula is as follows:

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

For the *MRR* metric, if the predicted ranking of the i-th sample is denoted 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 use 5 indicators: *MRR*, *HIT@1*, *HIT@3*, *HIT@10*, and *HIT@50* to measure the excellence of a model.

B PLM Ablation

GCPL utilizes experimental results from four different PLMs with the node text available.

Model	Text	MAVEN-GEP				
		MRR	Hit@1	Hit@3	Hit@10	Hit@50
GCPL (RoBERTa)	✔	**30.3**	**24.5**	**31.7**	**40.8**	58.2
GCPL (BERT)	✔	26.9	20.7	27.4	39.7	**58.4**
GCPL (ERNIE)	✔	26.6	21.1	27.3	37.7	56.3
GCPL (DeBERTa)	✔	24.2	19.9	24.1	32.4	51.3
Model	**Text**	**ESG**				
		MRR	Hit@1	Hit@3	Hit@10	Hit@50
GCPL (RoBERTa)	✔	**20.5**	**15.4**	**24**	**32.8**	**49.1**
GCPL (BERT)	✔	19	13.9	19	31.2	46.2
GCPL (ERNIE)	✔	19.6	13	20.7	32	47.1
GCPL (DeBERTa)	✔	20.2	13.6	21.6	29.9	42.2

Table 5: PLM ablation of our GCPL model on MAVEN-GEP and ESG datasets.