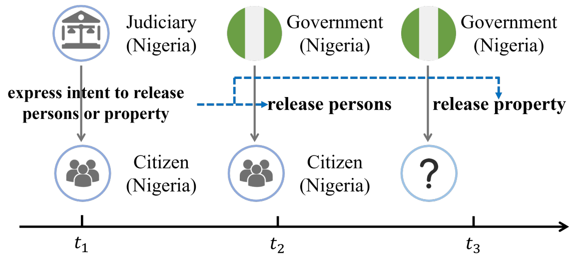
SRM-LLM: Semantic Relationship Mining with LLMs for Temporal Knowledge Graph Extrapolation

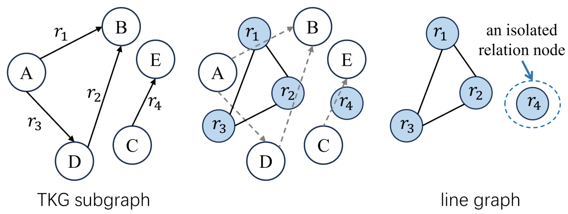
Fu Zhang†*, **Panfeng Zhang**†, **Jingwei Cheng**
School of Computer Science and Engineering, Northeastern University, China
Key Laboratory of Intelligent Computing in Medical Image of Ministry of Education
Northeastern University, China
{zhangfu,chengjingwei}@neu.edu.cn; 2301997@stu.neu.edu.cn

Abstract

Temporal knowledge graph (TKG) extrapolation aims to predict future facts by modeling the dynamic evolution of historical facts within TKGs. Existing methods often neglect the complex semantic relationships between relations when modeling their dynamic evolution, leading to incomplete relation representations and affecting the accuracy of reasoning. Inspired by the advancements in large language models (LLMs), we propose Semantic Relationship Mining based on LLMs (SRM-LLM), a novel approach for extracting semantic relationships to achieve TKG extrapolation. By leveraging LLMs to analyze the types of relations, we first identify several common relation types (e.g., causal, synonymous) in TKGs. We then design the LLM-based prompting strategy to capture latent semantic connections between relations, enabling the construction of relational association subgraphs for relation representation learning. In addition, SRM-LLM further enhances reasoning capabilities by incorporating structured logical constraints to guide inference. Experiments on five TKG datasets show significant performance gains and achieve new state of the art (SOTA) results, confirming the effectiveness of our method on TKG extrapolation tasks.



(a) An example of TKG reasoning, where the blue dashed lines indicate the causal relationships between relations. Through the latent causal relationships, the missing entity Citizen (Nigeria) can be correctly predicted.



(b) The process of transforming the TKG subgraph into the line graph.

Figure 1: Part (a) illustrates the crucial role of latent information between relations in TKG reasoning. Part (b) illustrates the method of learning associations between relations by transforming a TKG subgraph into a line graph. However, the resulting line graph may contain isolated relation nodes (e.g., r_4), which can impact the learning of relation representations.

1 Introduction

A knowledge graph (KG) is a semantic network that structurally represents facts about the real world using entities and relations in the form of triples *(s, r, o)* (Bollacker et al., 2008; Zhao et al., 2021; Song et al., 2021). However, most facts in the real world are not static and change over time (Li et al., 2023). As for the fact "*Barack Obama is the president of USA*", it only holds true from *2009* to *2017*. To capture and represent the evolution of facts over time, the temporal knowledge graph (TKG) is designed. With temporal information, facts in TKGs are typically represented as quadruples *(s, r, o, t)*, where the timestamp *t* denotes the specific time at which the fact occurs.

Reasoning tasks in TKGs can be classified into two types: **interpolation reasoning** and **extrapolation reasoning** (Jin et al., 2020a). Interpolation reasoning aims to infer missing facts within the known timestamps, while extrapolation reasoning attempts to predict potential unknown facts that may occur in the future timestamps. The extrapolation reasoning is more challenging and has practical value in scenarios such as event prediction

† Equal contribution. * Corresponding author.

Findings of the Association for Computational Linguistics: EMNLP 2025, pages 16007–16021
November 4-9, 2025 ©2025 Association for Computational Linguistics

(Deng et al., 2020) and stock forecasting (Carta et al., 2021). Therefore, in this work, we focus on the extrapolation reasoning.

To achieve accurate TKG reasoning, it is essential to construct models capable of fully capturing temporal dynamics and the complexity of relations. In recent years, numerous studies have focused on improving entity representations to better capture the evolutionary patterns of entities. For example, RE-GCN (Li et al., 2021) leverages recursive modeling of KG sequences to learn evolutionary representations of entities. However, most studies overlook the rich latent information within relations. When aggregating relations, they primarily focus on directly adjacent entities, resulting in incomplete relation representations. As bridges connecting entities, relations encapsulate abundant semantic information and dynamic evolutionary patterns. Taking Figure 1(a) as an example, when predicting the query *(Nigeria, release property, ?, t_3)*, the causal relationships between the relation "*express intent to release persons or property*" and the relations "*release persons*" and "*release property*" enable us to accurately predict that the correct entity is *Citizen(Nigeria)*.

For learning information between relations, two recent studies (Liu et al., 2023; Liang et al., 2023) extend the concept of line graph (Harary and Norman, 1960) to the domain of TKG reasoning. As shown in Figure 1(b), for the subgraph at a given timestamp (the white part), if two edges are adjacent, their corresponding relation nodes in the line graph (the blue part) will be connected by an edge. These two methods learn information between relations by generating the relational correlation graph. However, the relational correlation graph obtained through this approach may contain isolated relation nodes that cannot be effectively connected to other relation nodes. This isolation hinders the propagation and aggregation of information within the graph, thus affecting the learning of relation representations.

To address the aforementioned issues, we propose a novel method, SRM-LLM, for learning information between relations in TKG reasoning. Specifically, for each dataset, we first leverage LLMs to analyze the latent association types between relations it contains (e.g., causal, synonymous). We then design the LLM-based prompting strategy to mine the latent semantic connections among relations within each subgraph, enabling the construction of relational association subgraphs for relation representation learning. Unlike methods that convert the graph into a line graph, our approach avoids the emergence of isolated relation nodes, thereby facilitating more accurate relation representation learning. In addition, inspired by TLogic (Liu et al., 2022), we propose a rule-based historical relation retrieval module, which retrieves historical quadruples related to the query relation by applying logical rules to historical subgraphs. This facilitates the construction of a global graph for learning entity representations at a global level. Finally, we combine these global entity representations with local entity representations to perform prediction. Experimental results clearly demonstrate the effectiveness of our method. Our main contributions are summarized as follows:

- We propose a novel approach for applying LLMs to the domain of TKG reasoning, leveraging the semantic understanding capabilities of LLMs to mine the latent semantic correlations between relations. This approach avoids the emergence of isolated relation nodes, thereby enhancing the effectiveness of relation representation.

- We propose a rule-based historical relation retrieval module, which facilitates the construction of a global graph for learning entity representations at a global level.

- We conduct extensive experiments on five commonly used TKG datasets and the experimental results demonstrate the effectiveness of our method. On the ICEWS14 dataset, the Hits@1 metric improves by 5.4% and on the ICEWS05-15 dataset, it improves by 4.1%.

2 Related Work

2.1 Static KG Reasoning

Static KG reasoning methods can be broadly categorized into four types: translation-based methods, matrix factorization-based methods, convolutional neural network-based methods and graph neural network-based methods. Graph neural network-based methods, such as R-GCN (Schlichtkrull et al., 2018) and CompGCN (Vashishth et al., 2020), update node representations by aggregating information from neighboring nodes.

2.2 TKG Reasoning

TKG Interpolation Reasoning In the domain of TKG interpolation reasoning, most methods are extensions of static KG reasoning methods. For example, TTransE (Leblay and Chekol, 2018) extends the TransE (Bordes et al., 2013) method from static KGs by integrating temporal information into relation representations. However, these methods still face challenges when addressing extrapolation tasks.

TKG Extrapolation Reasoning TKG extrapolation aims to predict future facts using historical information. Some recent methods utilize graph neural networks to model the associative constraints between entities and relations. RE-GCN (Li et al., 2021) learns the evolutionary representations of entities and relations at each timestamp by recursively modeling KG sequences. To enhance explainability, TLogic (Liu et al., 2022) leverages logical rules to predict future events. With the emergence of LLMs, TKG reasoning methods based on LLMs have become a research hotspot. GenTKG (Liao et al., 2024a) combines the retrieval strategy based on temporal logical rules with instruction fine-tuning for generative predictions.

3 Problem Definition

TKG and TKG Extrapolation Let $\mathcal{E}$ and $\mathcal{R}$ represent a set of entities and relations. A quadruple (s, r, o, t) represents a relation $r \in \mathcal{R}$ that occurs between subject entity $s \in \mathcal{E}$ and object entity $o \in \mathcal{E}$ at timestamp t. All the quadruples that occur at timestamp t form a knowledge graph $\mathcal{G}_t$. A TKG $\mathcal{G}$ is defined as a sequence of knowledge graphs at different timestamps, i.e., $\mathcal{G} = \{\mathcal{G}_1, \mathcal{G}_2, \cdots, \mathcal{G}_t\}$. TKG extrapolation involves predicting a missing object $(s, r, ?, t)$, a missing subject entity $(?, r, o, t)$, or a missing relation $(s, ?, o, t)$ based on the historical KG sequences $\mathcal{G}_{<t} = \{\mathcal{G}_1, \mathcal{G}_2, \cdots, \mathcal{G}_{t-1}\}$. Without loss of generality, the inverse relation quadruples (o, r^{-1}, s, t) are incorporated into the TKG dataset, thereby reducing the entity prediction task to the prediction of object entities.

4 Methodology

In this section, we propose a novel LLM-based and logical constraint method, SRM-LLM, which is aimed at mining semantic associations between relations to achieve TKG extrapolation tasks. The overall framework of our model is illustrated in Figure 2. In the following, we provide a detailed description of the model components.

4.1 Relational Semantic Association Module

4.1.1 Relational Semantic Association Mining

To mine latent semantic associations among relations in TKGs while capturing prevalent dynamic interaction patterns between events—such as causal chains, conflicts, and hierarchical structures—we categorize relation associations into five types: (1) Causal, (2) Synonymous, (3) Oppositional, (4) Inclusion and (5) Progressive relationships. Specific explanations of the five types of association are shown in **Appendix A.1**.

By identifying the types of association, we can better understand and analyze the complex connections between relations. In Table 8 of **Appendix A.2**, some specific examples are presented, such as the relation "*Praise or endorse*" and the relation "*Criticize or denounce*", which are in a oppositional relationship with each other. These types of relationships help construct a logical framework for understanding the interactions between events.

Next, we design a prompt to extract the latent associations between all existing relations in the subgraph $\mathcal{G}_{t_i}$. By utilizing the LLM, the latent associations between relations are identified and represented in the format $[r_i, p_r, r_j]$, where $r_i, r_j \in \mathcal{R}$ and p_r is one of the association types. A specific example of the prompt is shown in **Appendix B**.

4.1.2 Relation Representation Learning

Through the relation associations mined by the LLM, we construct a corresponding relational association subgraph $\mathcal{RG}_{t_i}$ for each subgraph $\mathcal{G}_{t_i}$. We use a relational graph convolutional network (Schlichtkrull et al., 2018) as a semantic aggregator to obtain embeddings for each relational node in the relational association subgraph $\mathcal{RG}_{t_i}$. The definition of the relational aggregator is as follows:

$$\mathbf{r}^{l+1}_{r_o,t} = \sigma\left(\frac{1}{c_r}\sum_{r_s\in\mathcal{R}^{p_r}_{r_o}}\mathbf{W}^l_1\left(\mathbf{r}^l_{r_s,t}+\mathbf{p_r}^l\right)+\mathbf{W}^l_2\mathbf{r}^l_{r_o,t}\right), \tag{1}$$

where $\mathbf{r}^{l+1}_{r_o,t}, \mathbf{r}^l_{r_o,t} \in \mathbb{R}^{|\mathcal{R}|\times d}$ represent the embeddings of relations in the $l+1^{th}$ and l^{th} layers of the relation aggregation R-GCN in each relational association subgraph $\mathcal{RG}_t$. $\mathcal{R}^{p_r}_{r_o}$ represents the set of relations adjacent to the relation node r_o through the relationship type p_r. $\mathbf{r}^l_{r_s,t}$ and $\mathbf{p_r}^l$ represent the adjacent relation embeddings and the corresponding relationship type embeddings in the l^{th} layer of the R-GCN. $\mathbf{W}^l_1$ and $\mathbf{W}^l_2 \in \mathbb{R}^{d\times d}$ are trainable weight parameter matrices in l^{th} layer. c_r is a nor-

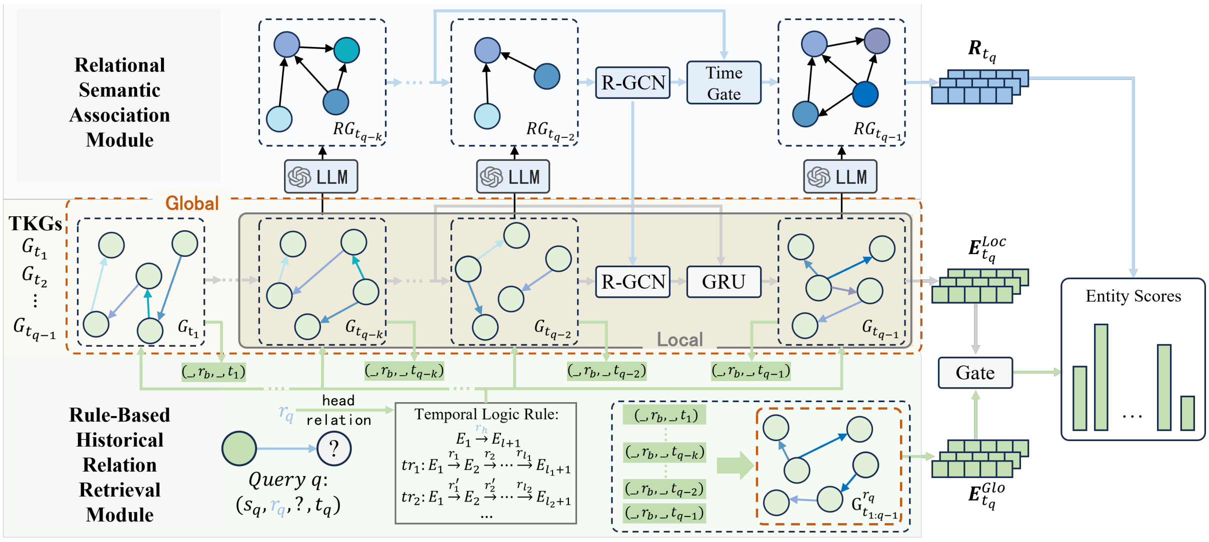


Figure 2: Framework of SRM-LLM. For a given query q, we first consider the most recent k timestamp subgraphs (e.g., $G_{t_{q-k}}$). Through the **Relational Semantic Association Module**, we leverage LLM to mine the latent semantic associations between relations within each subgraph and construct the corresponding relational association subgraphs (e.g., $RG_{t_{q-k}}$) to learn the relation representations $\mathbf{R}_{t_q}$. Meanwhile, from these k historical subgraphs ($G_{t_{q-k}}, \ldots, G_{t_{q-1}}$), we learn the local entity representations $\mathbf{E}_{t_q}^{Loc}$. In addition, for all historical subgraphs of the query q ($G_{t_1}, \ldots, G_{t_{q-1}}$), we utilize the **Rule-Based Historical Relation Retrieval Module** to extract quadruples related to the query relation r_q from each subgraph. These quadruples are used to construct a global graph $G_{t_{1:q-1}}^{r_q}$, from which we learn the global entity representations $\mathbf{E}_{t_q}^{Glo}$. Finally, we fuse the global and local entity representations to obtain the final entity representations and perform prediction.

malization constant that equals the in-degree of relation. $\sigma(\cdot)$ is the RReLU activation function.

For updating the relation evolution representations, we use the time gate recurrent component (Li et al., 2021) to update the relation embeddings at timestamp t. The specific form is as follows:

$$\mathbf{r}_t^p = pooling(\mathbf{E}_{t,r}) + \mathbf{r}, \tag{2}$$

$$\mathbf{U}_t = \sigma_2(\mathbf{W}_3 \mathbf{R}_t^p + \mathbf{b}), \tag{3}$$

$$\mathbf{R}_{t+1} = \mathbf{U}_t \mathbf{R}_t^p + (1 - \mathbf{U}_t)\mathbf{R}_t, \tag{4}$$

where $\sigma_2(\cdot)$ is the sigmoid function. $\mathbf{W}_3$ represents the weight matrix of the time gate. $\mathbf{E}_{t,r}$ represents the entity embeddings connected to relation r at timestamp t. $\mathbf{R}_t^p \in \mathbb{R}^{|\mathcal{R}| \times d}$ consists of r_t^p of all relations at timestamp t. $\mathbf{R}_{t+1}$ represents the updated relation embedding matrix obtained through the time gate.

4.2 Local Entity Representation

After obtaining the relational embeddings of the corresponding relational association subgraph $\mathcal{RG}_t$, we employ the entity aggregation R-GCN to obtain the embeddings of entity nodes in the corresponding historical subgraph $\mathcal{G}_t$. Similar to the learning of relational node embeddings, the entity aggregation R-GCN is defined as follows:

$$\mathbf{e}_{o,t}^{l+1} = \sigma\left(\frac{1}{c_o}\sum_{s \in \mathcal{E}_o^r} \mathbf{W}_4^l\left(\mathbf{e}_{s,t}^l + \mathbf{r}^l\right) + \mathbf{W}_5^l \mathbf{e}_{o,t}^l\right), \tag{5}$$

where $\mathbf{e}_{o,t}^{l+1}, \mathbf{e}_{o,t}^l \in \mathbb{R}^{|\mathcal{E}| \times d}$ represent the embeddings of entities in the $l+1^{th}$ and l^{th} layers of the entity aggregation R-GCN in each historical subgraph $\mathcal{G}_t$. $\mathcal{E}_o^r$ represents the set of entities adjacent to the entity node o through the relation r. $\mathbf{e}_{s,t}^l$ and $\mathbf{r}^l$ represent the adjacent entity embeddings and the corresponding relation embeddings in the l^{th} layer of the R-GCN. Among them, $\mathbf{r}^l$ is obtained by aggregating the corresponding relational association subgraph $\mathcal{RG}_t$ using Equation 1. $\mathbf{W}_4^l$ and $\mathbf{W}_5^l \in \mathbb{R}^{d \times d}$ are trainable weight parameter matrices in l^{th} layer. c_o is a normalization constant that equals the in-degree of entity. $\sigma(\cdot)$ is the RReLU activation function.

For a given query $q = (s_q, r_q, ?, t_q)$, after obtaining the structural semantic embeddings of entities in each historical subgraph $\mathcal{G}_t$, it is necessary to further model the sequential dependencies of entities across the historical subgraphs from the most recent k timestamps. We achieve this by progressively updating the entity representations using a

Gated Recurrent Unit (GRU) (Cho et al., 2014). The update process is formulated as follows:

$$\mathbf{E}_{G_{t+1}} = GRU_E(\mathbf{E}_{G_t}, \mathbf{E}_t), \tag{6}$$

where $\mathbf{E}_{G_t}$ represents the entity embedding matrix after aggregation on the subgraph at timestamp t. $\mathbf{E}_t$ represents the embeddings of entity.

4.3 Rule-Based Historical Relation Retrieval Module

In entity representation learning, in addition to capturing the evolving representations of the k most recent historical subgraphs associated with the query, we also employ logical rules to capture the global historical facts relevant to the query relations. First, we employ the random walk strategy in TLogic (Liu et al., 2022) to extract temporal logical rules. The definition of rules in TLogic is shown in **Appendix C**. Then, for the query $q = (s_q, r_q, ?, t_q)$, we obtain the logical rules with the query relation r_q as the head relation. Subsequently, in the global historical subgraphs $(\mathcal{G}_1, \mathcal{G}_2, \ldots, \mathcal{G}_{t_q-1})$, we extract the global historical facts associated with the corresponding body relation r_b, thereby obtaining the logical relation historical subgraph related to the query relation r_q.

After obtaining the logical relation historical subgraph, we model the structural semantic representations of the subgraph using R-GCN to update the global entity representations. The update process is similar to that in Equation 5.

By combining the local evolution representations and global logical representations, we obtain the final entity representations. Formally, the entity representations can be obtained as follows:

$$\mathbf{E}_{t_q} = \lambda \mathbf{E}^L_{t_q} + (1-\lambda)\mathbf{E}^G_{t_q}, \tag{7}$$

where $\mathbf{E}^L_{t_q}$ represents the local evolution representations of entities and $\mathbf{E}^G_{t_q}$ represents the global logical representations of entities. λ is a hyperparameter in the range of 0 to 1, used to control the weight of the local and global representations.

4.4 Prediction and Training

For a query $q = (s_q, r_q, ?, t_q)$, the entity representation $\mathbf{e}_{s_q,t_q}$ and relation representation $\mathbf{r}_{r_q,t_q}$ of the query at timestamp t_q are used as inputs to compute the predicted scores for each candidate entity representation $\mathbf{e}_o$. We choose to employ ConvTransE (Shang et al., 2019) as the decoder to perform the entity prediction task. The score calculation process for each candidate entity e_o is as follows:

$$\phi(s_q, r_q, e_o, t_q) = \sigma_3\left(\mathbf{e}_{o,t_q}\text{ConvTransE}\left(\mathbf{e}_{s_q,t_q}, \mathbf{r}_{r_q,t_q}\right)\right), \tag{8}$$

where $\sigma_3(\cdot)$ is the softmax activate function.

The training objective is to minimize the cross-entropy loss:

$$\mathcal{L} = \sum_{t=0}^{T} \sum_{(s_q,r_q,e,t_q)\in\mathcal{G}_{t_q}} \sum_{e_o\in\mathcal{E}} y^{e_o}_{t_q}\log\phi(s_q, r_q, e_o, t_q), \tag{9}$$

where $\phi(s_q, r_q, e_o, t_q)$ represents the matching score between the query q and the candidate entity e_o. $y^{e_o}_{t_q}$ is 1 if the fact exists, otherwise 0.

5 Experiments

In this section, we conduct a series of experiments on five TKG datasets to demonstrate the effectiveness of our model.

5.1 Experimental Setup

Datasets We select five representative datasets in TKG reasoning domain to evaluate our method. They are GDELT (Leetaru and Schrodt, 2013), WIKI (Leblay and Chekol, 2018), ICEWS14 (Garcia-Duran et al., 2018), ICEWS18 (Jin et al., 2020b) and ICEWS05-15 (Garcia-Duran et al., 2018).

Evaluation Metrics In the experiments, we adopt the widely used metrics Mean Reciprocal Rank (MRR) and Hits@{1, 3, 10} to evaluate the effectiveness of TKG reasoning methods. Higher values for these metrics are better. We report the experimental results under the time-aware filtered setup, which is commonly used in recent works.

5.2 Experimental Results

The experimental results of our method, along with the results of all baselines, are presented in Table 1 and Table 2. Specific details about the *selection of baselines* and the *experimental settings* are shown in **Appendix D**.

Entity Prediction By observing the results in Table 1, it can be seen that our method achieves the best performance on the ICEWS series datasets, indicating the effectiveness of our approach.

Specifically, our method significantly outperforms all the static methods compared, highlighting the importance of temporal information in TKG reasoning. As for the interpolation methods, they lack the ability to model the evolution of entities

Model	ICEWS14				ICEWS18				ICEWS05-15			
	MRR	Hits@1	Hits@3	Hits@10	MRR	Hits@1	Hits@3	Hits@10	MRR	Hits@1	Hits@3	Hits@10
ComplEx (2016)	32.54	23.43	36.13	50.73	22.94	15.19	27.05	42.11	32.63	24.01	37.50	52.81
ConvE (2018)	35.09	25.23	39.38	54.68	24.51	16.23	29.25	44.51	33.81	24.78	39.00	54.95
Conv-TransE (2019)	33.80	25.40	38.54	53.99	22.11	13.94	26.44	42.28	33.03	24.15	38.07	54.32
RotatE (2019)	21.31	10.26	24.35	44.75	12.78	4.01	14.89	31.91	24.71	13.22	29.04	48.16
TTransE (2018)	13.72	2.98	17.70	35.74	8.31	1.92	8.56	21.89	15.57	4.80	19.24	38.29
DE-SimplE (2020)	33.36	24.85	37.15	49.82	19.30	11.53	21.86	34.80	35.02	25.91	38.99	52.75
TNTComplEx (2020)	34.05	25.08	38.50	50.92	21.23	13.28	24.02	36.91	27.54	9.52	30.80	42.86
CyGNet (2021)	35.05	25.73	39.01	53.55	24.93	15.90	28.28	42.61	36.81	26.61	41.63	56.22
xERTE (2021)	40.02	32.06	44.63	56.17	29.98	22.05	33.46	44.83	46.62	37.84	52.31	63.92
RE-GCN (2021)	40.39	30.66	44.96	59.21	30.58	21.01	34.34	48.75	48.03	37.33	53.85	68.27
CEN (2022)	42.20	32.08	47.46	61.31	31.50	21.70	35.44	50.59	46.84	36.38	52.45	67.01
TiRGN (2022)	44.04	33.83	48.95	63.84	33.66	23.19	37.99	54.22	50.04	39.25	56.13	70.71
HisMatch (2022)	46.42	35.91	51.63	66.84	33.99	23.91	37.90	53.94	52.85	42.01	59.05	73.28
RETIA (2023)	42.76	32.28	47.77	62.75	32.43	22.23	36.48	52.94	47.26	36.64	52.90	67.76
RPC* (2023)	44.55	34.87	49.80	65.08	34.91	24.34	38.74	55.89	51.14	39.47	57.11	71.75
LogCL (2024)	48.87	37.76	54.71	70.26	35.67	24.53	40.32	57.74	57.04	46.07	63.72	77.87
CRAFT* (2024)	45.71	35.05	51.83	65.21	34.21	23.96	38.53	54.11	50.14	39.56	56.18	70.09
HTCCN* (2024)	45.39	36.58	50.84	-	35.63	24.90	39.26	-	51.94	40.32	57.79	-
GenTKG* (2024)	-	36.85	47.95	53.50	-	24.25	37.25	42.1	-	-	-	-
LLM-DA* (2024)	47.10	36.90	52.60	67.10	35.70	25.50	40.30	57.00	52.10	41.60	58.60	72.80
SRM-LLM	**50.84**	**39.80**	**56.82**	**71.92**	**37.23**	**25.92**	**42.01**	**59.67**	**57.97**	**46.98**	**64.91**	**78.54**

Table 1: Performance comparison of *entity prediction* on ICEWS datasets in terms of MRR(%) and Hits@{1,3,10}(%) (time-aware filtered). The best performance is highlighted in boldface, and the second-best is underlined. The baseline results marked with * are from the corresponding papers, while the remaining baseline results are from LogCL (Chen et al., 2024b).

Model	WIKI			GDELT		
	MRR	Hits@1	Hits@3	MRR	Hits@1	Hits@3
ComplEx	38.54	40.51	48.61	16.96	11.25	19.52
ConvE	36.41	39.45	49.25	16.55	11.02	18.88
Conv-TransE	35.54	38.43	47.13	16.20	10.85	18.38
RotatE	37.44	42.36	47.67	13.45	6.95	14.09
TTransE	29.27	21.67	34.43	5.50	0.47	4.94
DE-SimplE	45.43	42.60	47.71	19.70	12.22	21.39
TNTComplEx	45.00	40.00	49.30	19.53	12.41	20.75
CyGNet	58.78	47.89	66.44	18.48	11.52	19.57
xERTE	73.60	69.05	78.03	18.09	12.30	20.06
RE-GCN	78.53	74.50	81.59	19.64	12.42	20.90
CEN	78.93	75.05	81.90	20.39	12.96	21.77
TiRGN	80.05	75.15	84.35	21.67	13.63	23.27
HisMatch	78.07	73.89	81.32	22.01	14.45	23.80
RETIA	76.29	72.56	77.97	20.12	12.76	21.45
RPC	81.18	76.28	85.43	22.41	14.42	24.36
LogCL	76.94	72.91	79.76	23.75	14.64	25.60
CRAFT	81.32	**77.21**	85.36	23.78	15.38	26.23
HTCCN	-	-	-	23.46	15.18	25.21
GenTKG	-	-	-	-	13.90	22.55
SRM-LLM	**81.40**	76.86	**85.79**	**25.23**	**15.92**	**27.21**

Table 2: Performance comparison of *entity prediction* on WIKI and GDELT datasets in terms of MRR(%) and Hits@{1,3}(%) (time-aware filtered).

Model	ICEWS14	ICEWS18	ICEWS05-15	WIKI
ConvE	38.80	37.73	37.89	78.23
Conv-TransE	38.40	38.00	38.26	86.64
RE-GCN	41.06	40.53	40.63	97.92
TiRGN	42.57	41.78	42.12	93.58
RETIA	42.05	41.78	43.19	98.21
SRM-LLM	**44.05**	**42.61**	**44.17**	**98.96**

Table 3: Performance on the *relation prediction* task.

and relations over time. Therefore, our method also outperforms interpolation methods. Compared with our method, the approach of RETIA and RPC, which involves constructing relational association graphs by introducing line graph to learn relational representations, is affected by isolated relation nodes. This negatively impacts the full learning of relational representations and consequently degrades their performance. LogCL ignores the potential connections between relations and does not model the relationships between them, which leads to inaccurate relational representations and impacts its performance.

As observed in Table 2, our model has a less remarkable performance improvement on the WIKI dataset compared to other datasets. This is because the WIKI dataset has relatively few relation types (only 24 types). The relation representation learning for this dataset is already relatively sufficient, so the room for improvement is smaller.

Relation Prediction Since some models are not specifically designed for relation prediction tasks, we compare SRM-LLM only with representative models among the baselines (Liu et al., 2023). The static reasoning methods include ConvE and Conv-TransE, while the temporal reasoning methods comprise RE-GCN, TiRGN and RETIA. The MRR results for relation prediction are reported in Table 3.

Model	ICEWS14				ICEWS18				WIKI			
	MRR	Hits@1	Hits@3	Hits@10	MRR	Hits@1	Hits@3	Hits@10	MRR	Hits@1	Hits@3	Hits@10
w/o RSA	48.42	37.38	54.00	70.28	35.64	24.57	40.27	57.60	78.12	74.03	83.67	87.21
w/o RHR	49.90	38.44	56.32	71.86	36.17	25.08	41.32	58.55	80.08	75.78	84.36	87.22
SRM-LLM	**50.84**	**39.80**	**56.82**	**71.92**	**37.23**	**25.92**	**42.01**	**59.67**	**81.40**	**76.86**	**85.79**	**88.83**

Table 4: The ablation study results of MRR and Hits@{1,3,10} on ICEWS14, ICEWS18 and WIKI datasets.

Number of Types	ICEWS14			
	MRR	Hits@1	Hits@3	Hits@10
3 types of p_r	50.34	39.13	56.39	72.15
5 types of p_r	**50.84**	**38.80**	**56.82**	**71.92**
7 types of p_r	49.96	38.61	56.34	71.71

Table 5: Performance comparison on different types of association.

Model	Sparse ICEWS14			
	MRR	Hits@1	Hits@3	Hits@10
LogCL	22.34	14.62	24.39	38.08
RETIA	24.60	16.41	**27.60**	**40.71**
SRM-LLM	**24.83**	**17.12**	<u>27.47</u>	<u>39.95</u>
w/o RSA	22.12	14.31	23.97	37.69
w/o RHR	23.32	15.20	24.53	38.89

Table 6: Performance comparison on Sparse ICEWS14 dataset.

The experimental results demonstrate that SRM-LLM outperforms all baseline methods in relation prediction, indicating that modeling the latent associations between relations contributes to learning more accurate representations.

Moreover, similar to the entity prediction task, SRM-LLM achieves greater performance improvements on the ICEWS series datasets compared to the WIKI datasets. This is because the WIKI dataset contain fewer relation than the ICEWS series dateset, resulting in fewer candidate relations and making the relation prediction task relatively easier on WIKI, thereby leading to smaller performance gains.

5.3 Ablation Study

To analyze the contribution of each component in the model, we conduct ablation studies on ICEWS14, ICEWS18 and WIKI datasets. The results of different variants in terms of MRR and Hits@{1, 3, 10} are reported in Table 4.

Impact of Relational Semantic Association To analyze the impact of relational representation learning on the results, we remove the Relational Semantic Association module. The results are denoted as "w/o RSA" in Table 4. As can be seen, removing the RSA module leads to a significant decline in performance. This is because the RSA module can mine the rich semantic information between relations through LLM, which helps in learning more accurate relational representations.

Impact of Rule-Based Historical Relation Retrieval To thoroughly analyze the impact of the Rule-Based Historical Relation Retrieval module on model performance, we remove the module and observe the changes in model performance. The results are denoted as "w/o RHR" in Table 4. Removing this module leads to a decline in model performance. The role of the RHR module is to capture the facts from the historical KG sequences related to the query relation using temporal logical rules and use these facts to construct a global historical graph relevant to the query, thereby extracting global information. These results indicate that the RHR module is also of great importance.

Impact of Local and Global History To analyze the roles of local and global history, we separately remove the modeling of local history and global history, denoted as "w/o L" and "w/o G", respectively. The results are reported in **Appendix E**. The results show that modeling both local and global history for a query improves the model's reasoning performance. Furthermore, we observe that the performance drop caused by the "w/o L" variant is greater than that of the "w/o G" variant, indicating that modeling the evolution of local history is more effective than modeling global history.

In addition, we conduct ablation studies on certain components of RE-GCN. We remove the Time Gate and Static Graph modules. The corresponding results are denoted as "w/o TG" and "w/o SG". The results are also reported in **Appendix E**. The results of "w/o SG" demonstrate the importance of static graph information. The entity type information in the static graph helps provide better initial representations for entities. The results of "w/o

TG" show that the gating mechanism also plays a crucial role in learning the evolution of relations.

5.4 Study on the Number of Association Types

To investigate the impact of varying the number of association types on the construction of the relation association graph, we experiment with both the expansion and reduction of association types based on the initially five types. For relation type expansion, in addition to the initial five types, we leverage the semantic understanding capabilities of the LLM to discover additional associations within each temporal subgraph. For relation type reduction, we utilize the LLM to identify and retain only the top three most salient association types at each timestamp, based on their estimated importance. The experimental results are reported in Table 5.

Upon analyzing the results, we observe that increasing the number of association types leads to performance degradation, likely due to LLMs misclassifying fine-grained categories and introducing noisy edges, as well as increased model complexity. Conversely, reducing association types can also harm performance, as sparse relation nodes lack sufficient neighboring information for effective representation learning.

5.5 Study on Sparse Dataset

To quantitatively assess how the sparsity of TKGs affects model performance, we conduct additional experiments. Based on the ICEWS14 dataset, we construct a sparser version, referred to as Sparse ICEWS14. The comparison between the Sparse ICEWS14 and the original ICEWS14 dataset is shown in **Appendix F**. In the Sparse ICEWS14 dataset, only 10% of the original outgoing edges are retained for each entity, with a minimum of one edge per entity. The experimental results are reported in Table 6.

Experimental results show that although all models experience significant performance degradation under sparse data conditions, SRM-LLM still maintains superior performance in terms of MRR and Hits@1. We attribute this to two key factors: (1) SRM-LLM leverages semantic relations extracted by the LLM to construct a relation association graph. Even when some relational instances are missing, the semantic reasoning capabilities of the LLM can infer potential associations, thereby enriching the structural information of sparse subgraphs. (2) The RHR module extracts quadruples relevant to the query relation from all historical subgraphs, effectively expanding the semantic coverage of the current sparse context and reducing the model's dependence on local information.

Model	ICEWS14		ICEWS18		ICEWS05-15	
	MRR	Hits@1	MRR	Hits@1	MRR	Hits@1
RGCN	**50.84**	**39.80**	**37.23**	**25.92**	**57.97**	**46.98**
CompGCN	49.67	38.24	37.02	25.84	57.84	46.14
KBGAT	49.52	38.27	36.97	25.66	56.89	45.24

Table 7: The results of different GNN Aggregation.

5.6 Study on Different GNN Aggregation

To further investigate the impact of different types of Graph Neural Network (GNN) on relation aggregation and entity aggregation encoding, R-GCN is replaced with CompGCN (Vashishth et al., 2019) and KBGAT (Nathani et al., 2019) in the respective aggregation encoder. Table 7 reports the MRR and Hits@1 results on the ICEWS series datasets. Experimental results demonstrate that SRM-LLM (R-GCN) achieves the best performance on most metrics across all three datasets. This suggests that, although no explicit relation-aware attention mechanism is introduced, the semantic relation-type information extracted by the LLM provides the model with prior knowledge, enabling R-GCN to differentiate the distinctive features among various relation types.

5.7 Parameter Analysis

We also conduct experiments on ICEWS14 to analyze the impact of parameter variations in the model. The parameters analyzed include the rule length l, the number of layers in R-GCN, the weight λ that balances the local and global entity embedding representations and the length k of the local historical subgraphs. Due to space limitations, the detailed analysis are shown in **Appendix G**.

5.8 Study on Time Overhead

Considering both code openness and model performance, we select RETIA, LogCL and HTCCN for a comparative analysis of training and testing time overhead. The experimental results are reported in Table 13 of **Appendix H**. As shown in the table, compared to LogCL, our approach slightly increases the time overhead due to the relational aggregator. However, this also improves performance markedly. compared to RETIA, which also enhances relation representation learning, our approach has significantly lower time overhead and better performance. In summary, SRM-LLM ensures a limited increase in time cost while delivering superior extrapolation results.

6 Conclusion

In this paper, we propose a novel LLM-based TKG reasoning method, SRM-LLM. We leverage the powerful semantic understanding capability of LLMs to identify the types of semantic association between relations (e.g., causal and synonymous), and construct the relational association graph to enhance relation representations. The design of this module avoids the emergence of isolated relation nodes, optimizing the learning process of relation representations. In addition, we introduce a rule-based historical relation retrieval module, which extracts globally relevant historical facts related to the query relations through temporal logical rules, learning entity representations at a global level. Experimental results show that SRM-LLM significantly outperforms existing methods on multiple benchmark datasets, demonstrating its effectiveness in TKG extrapolation tasks.

Limitations

Although SRM-LLM achieves significant performance improvements, there are still limitations in this paper. While we have identified five types of association, the semantic associations between relations could be more complex. We could further expand and refine these relationship types to capture the dynamic semantic information between relations more comprehensively in the future. In this paper, we only use GPT-3.5-turbo to mine the semantic associations of relations, without exploring the differences between different LLMs. Different LLMs may vary in semantic understanding and reasoning capabilities, which could affect the accuracy and efficiency of relational semantic association mining. In the future, we plan to explore the differences between various LLMs to better understand their applicability and advantages in TKG reasoning. By comparing the performance of different LLMs, we can choose the model most suitable for TKG reasoning task, further enhancing reasoning capabilities.

Acknowledgments. The work is supported by the National Natural Science Foundation of China (62276057).

References

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 1247–1250.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. *Advances in neural information processing systems*, 26.

Elizabeth Boschee, Jennifer Lautenschlager, Sean O'Brien, Steve Shellman, James Starz, and Michael Ward. 2015. ICEWS Coded Event Data.

Salvatore Carta, Alessandro Sebastian Podda, Diego Reforgiato Recupero, and Maria Madalina Stanciu. 2021. Explainable ai for financial forecasting. In *International Conference on Machine Learning, Optimization, and Data Science*, pages 51–69. Springer.

Tingxuan Chen, Jun Long, Liu Yang, Zidong Wang, Yongheng Wang, and Xiongnan Jin. 2024a. Htccn: Temporal causal convolutional networks with hawkes process for extrapolation reasoning in temporal knowledge graphs. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 4056–4066.

Wei Chen, Huaiyu Wan, Yuting Wu, Shuyuan Zhao, Jiayaqi Cheng, Yuxin Li, and Youfang Lin. 2024b. Local-global history-aware contrastive learning for temporal knowledge graph reasoning. In *2024 IEEE 40th International Conference on Data Engineering (ICDE)*, pages 733–746. IEEE.

Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar. Association for Computational Linguistics.

Songgaojun Deng, Huzefa Rangwala, and Yue Ning. 2020. Dynamic knowledge graph based multi-event forecasting. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1585–1595.

Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. 2018. Convolutional 2d knowledge graph embeddings. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32.

Alberto Garcia-Duran, Sebastijan Dumančić, and Mathias Niepert. 2018. Learning sequence encoders for temporal knowledge graph completion. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4816–4821.

Rishab Goel, Seyed Mehran Kazemi, Marcus Brubaker, and Pascal Poupart. 2020. Diachronic embedding for temporal knowledge graph completion. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 3988–3995.

Zhen Han, Peng Chen, Yunpu Ma, and Volker Tresp. 2021. Explainable subgraph reasoning for forecasting on temporal knowledge graphs. In *International Conference on Learning Representations*.

Frank Harary and Robert Z Norman. 1960. Some properties of line digraphs. *Rendiconti del circolo matematico di palermo*, 9:161–168.

Woojeong Jin, Meng Qu, Xisen Jin, and Xiang Ren. 2020a. Recurrent event network: Autoregressive structure inferenceover temporal knowledge graphs. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6669–6683, Online. Association for Computational Linguistics.

Woojeong Jin, Meng Qu, Xisen Jin, and Xiang Ren. 2020b. Recurrent event network: Autoregressive structure inferenceover temporal knowledge graphs. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6669–6683.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Timothée Lacroix, Guillaume Obozinski, and Nicolas Usunier. 2020. Tensor decompositions for temporal knowledge base completion. In *International Conference on Learning Representations*.

Julien Leblay and Melisachew Wudage Chekol. 2018. Deriving validity time in knowledge graph. In *Companion proceedings of the the web conference 2018*, pages 1771–1776.

Kalev Leetaru and Philip A Schrodt. 2013. Gdelt: Global data on events, location, and tone, 1979–2012. In *ISA annual convention*, volume 2, pages 1–49. Citeseer.

Jiang Li, Xiangdong Su, and Guanglai Gao. 2023. TeAST: Temporal knowledge graph embedding via archimedean spiral timeline. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15460–15474, Toronto, Canada. Association for Computational Linguistics.

Yujia Li, Shiliang Sun, and Jing Zhao. 2022a. Tirgn: Time-guided recurrent graph network with local-global historical patterns for temporal knowledge graph reasoning. In *IJCAI*, pages 2152–2158.

Zixuan Li, Saiping Guan, Xiaolong Jin, Weihua Peng, Yajuan Lyu, Yong Zhu, Long Bai, Wei Li, Jiafeng Guo, and Xueqi Cheng. 2022b. Complex evolutional pattern learning for temporal knowledge graph reasoning. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 290–296.

Zixuan Li, Zhongni Hou, Saiping Guan, Xiaolong Jin, Weihua Peng, Long Bai, Yajuan Lyu, Wei Li, Jiafeng Guo, and Xueqi Cheng. 2022c. Hismatch: Historical structure matching based temporal knowledge graph reasoning. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 7328–7338.

Zixuan Li, Xiaolong Jin, Wei Li, Saiping Guan, Jiafeng Guo, Huawei Shen, Yuanzhuo Wang, and Xueqi Cheng. 2021. Temporal knowledge graph reasoning based on evolutional representation learning. In *Proceedings of the 44th international ACM SIGIR conference on research and development in information retrieval*, pages 408–417.

Ke Liang, Lingyuan Meng, Meng Liu, Yue Liu, Wenxuan Tu, Siwei Wang, Sihang Zhou, and Xinwang Liu. 2023. Learn from relational correlations and periodic events for temporal knowledge graph reasoning. In *Proceedings of the 46th international ACM SIGIR conference on research and development in information retrieval*, pages 1559–1568.

Ruotong Liao, Xu Jia, Yangzhe Li, Yunpu Ma, and Volker Tresp. 2024a. GenTKG: Generative forecasting on temporal knowledge graph with large language models. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 4303–4317, Mexico City, Mexico. Association for Computational Linguistics.

Ruotong Liao, Xu Jia, Yangzhe Li, Yunpu Ma, and Volker Tresp. 2024b. Gentkg: Generative forecasting on temporal knowledge graph with large language models. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 4303–4317.

Kangzheng Liu, Feng Zhao, Guandong Xu, Xianzhi Wang, and Hai Jin. 2023. Retia: relation-entity twin-interact aggregation for temporal knowledge graph extrapolation. In *2023 IEEE 39th International Conference on Data Engineering (ICDE)*, pages 1761–1774. IEEE.

Yushan Liu, Yunpu Ma, Marcel Hildebrandt, Mitchell Joblin, and Volker Tresp. 2022. Tlogic: Temporal logical rules for explainable link forecasting on temporal knowledge graphs. In *Proceedings of the AAAI conference on artificial intelligence*, volume 36, pages 4120–4127.

Deepak Nathani, Jatin Chauhan, Charu Sharma, and Manohar Kaul. 2019. Learning attention-based embeddings for relation prediction in knowledge graphs.

In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4710–4723, Florence, Italy. Association for Computational Linguistics.

Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *The semantic web: 15th international conference, ESWC 2018, Heraklion, Crete, Greece, June 3–7, 2018, proceedings 15*, pages 593–607. Springer.

Chao Shang, Yun Tang, Jing Huang, Jinbo Bi, Xiaodong He, and Bowen Zhou. 2019. End-to-end structure-aware convolutional networks for knowledge base completion. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 3060–3067.

Yichen Song, Aiping Li, Hongkui Tu, Kai Chen, and Chenchen Li. 2021. A novel encoder-decoder knowledge graph completion model for robot brain. *Frontiers in Neurorobotics*, 15:674428.

Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. 2019. Rotate: Knowledge graph embedding by relational rotation in complex space. In *International Conference on Learning Representations*.

Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex embeddings for simple link prediction. In *International conference on machine learning*, pages 2071–2080. PMLR.

Shikhar Vashishth, Soumya Sanyal, Vikram Nitin, and Partha P. Talukdar. 2019. Composition-based multi-relational graph convolutional networks. *CoRR*, abs/1911.03082.

Shikhar Vashishth, Soumya Sanyal, Vikram Nitin, and Partha P. Talukdar. 2020. Composition-based multi-relational graph convolutional networks. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Jiapu Wang, Sun Kai, Linhao Luo, Wei Wei, Yongli Hu, Alan Wee-Chung Liew, Shirui Pan, and Baocai Yin. 2024. Large language models-guided dynamic adaptation for temporal knowledge graph reasoning. *Advances in Neural Information Processing Systems*, 37:8384–8410.

Shengzhe Zhang, Wei Wei, Rikui Huang, Wenfeng Xie, and Dangyang Chen. 2024. Modeling historical relevant and local frequency context for representation-based temporal knowledge graph forecasting. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 7675–7686.

Xiaojuan Zhao, Yan Jia, Aiping Li, Rong Jiang, Kai Chen, and Ye Wang. 2021. Target relational attention-oriented knowledge graph reasoning. *Neurocomputing*, 461:577–586.

Cunchao Zhu, Muhao Chen, Changjun Fan, Guangquan Cheng, and Yan Zhang. 2021. Learning from history: Modeling temporal knowledge graphs with sequential copy-generation networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pages 4732–4740.

A Semantic association between relations

A.1 Explanation of five types of association

To mine the latent semantic associations between relations in TKGs, we first use LLMs to analyze the latent types of association among relations present in each dataset. The ICEWS series datasets contain rich and fine-grained relations, so we specifically identify five types of association: (1) Causal Relationship, (2) Synonymous Relationship, (3) Oppositional Relationship, (4) Inclusion Relationship and (5) Progressive Relationship. These five types cover the most common dynamic interaction patterns between events (e.g., causal logic chains, opposing conflicts, hierarchical semantics), which are widely regarded as core dimensions of complex event associations in linguistics and social sciences. Unlike the ICEWS series datasets, the WIKI dataset contains fewer relations that are mainly knowledge-based and relatively coarse-grained, while the GDELT dataset covers a broader range of relation types. For these two datasets, we do not identify specific association types; instead, we directly use LLMs to identify the relations that are relatively associated with each relation.

Causal Relationship: One event (or action) serves as the cause of another event (or action), where the latter is the result of the former.

Synonymous Relationship: Two or more events (or actions) are semantically similar or interchangeable.

Oppositional Relationship: Two events (or actions) oppose each other in terms of goals, intentions, or outcomes.

Inclusion Relationship: One event (or action) is a subset or a specific manifestation of another event (or action).

Progressive Relationship: There is a progressive advancement between events (or actions), emphasizing logical, stage-wise development.

A.2 Specific instances

For the five types of association between relations, some specific instances mined by the LLM are provided in Table 8.

Relationship Type (p_{r_i})	Examples
Causal Relationship (p_{r_0})	[88.Demand military aid, p_{r_0}, 59.Provide military aid] [185.Threaten to reduce or stop aid, p_{r_0}, 151.Reduce or stop aid] ... [57.Express intent to cooperate economically, p_{r_0}, 47.Cooperate economically]
Synonymous Relationship (p_{r_1})	[29.Appeal for diplomatic cooperation, p_{r_1}, 15.Appeal for policy support] [55.Express intent to provide military aid, p_{r_1}, 127.Express intent to cooperate militarily] ... [30.Return, release persons, p_{r_1}, 139.Return, release property]
Oppositional Relationship (p_{r_2})	[7.Praise or endorse, p_{r_2}, 8.Criticize or denounce] [22.Make optimistic comment, p_{r_2}, 26.Make pessimistic comment] ... [41.Ease administrative sanctions, p_{r_2}, 43.Impose administrative sanctions]
Inclusion Relationship (p_{r_3})	[9.Accuse, p_{r_3}, 74.Accuse of crime, corruption] [13.Express intent to cooperate p_{r_3}, 127.Express intent to cooperate militarily] ... [40.Provide aid, p_{r_3}, 32.Provide humanitarian aid]
Progressive Relationship (p_{r_4})	[87.Give ultimatum, p_{r_4}, 58.Threaten with military force] [18.Investigate, p_{r_4}, 42.Bring lawsuit against] ... [17.Sign formal agreement, p_{r_4}, 60.Engage in material cooperation]

Table 8: Examples of different types of association between relations in ICEWS14.

B A prompt example for mining the connections between relations

Taking the subgraph $\mathcal{G}_0$ corresponding to the 0 timestamp in the ICEWS14 dataset as an example, we construct its corresponding relational association subgraph $\mathcal{RG}_0$. First, we list all the relations contained in $\mathcal{G}_0$. Then, for each of the five types of relationships, we identify which relations correspond to each type. The results are output in the format $[r_i, p_r, r_j]$. The prompt example is shown in Table 9.

C Temporal logical rule

A cyclic temporal logical rule R of length $l \in \mathbb{N}$ is defined as

$$(E_1, r_h, E_{l+1}, T_{l+1}) \leftarrow \wedge_{i=1}^{l} (E_i, r_i, E_{i+1}, T_i)$$

with the temporal constraints

$$T_1 \leq T_2 \leq \cdots \leq T_l < T_{l+1}.$$

The left-hand side of R is referred to as the rule head, with r_h representing the head relation, while the right-hand side is referred to as the rule body, with r_i representing the body relation. E_i and T_i are replaceable variables that represent entities and timestamps. The rule head and rule body form two different paths that connect the same two entities E_1 and E_{l+1}. A rule head can be supported by several rule bodies, each representing a distinct rule denoted as $\mathcal{TR}$. A $\mathcal{TR}$ indicates that if the rule body holds, the rule head will be true at a future timestamp T_{l+1}.

Dataset	ICEWS14	ICEWS18	ICEWS05-15	WIKI	GDELT
Entities	7,128	23,033	10,488	12,554	7,691
Relations	230	256	251	24	240
Train	74,845	373,018	368,868	539,286	1,734,399
Valid	8,514	45,995	46,302	67,538	238,765
Test	7,371	49,545	46,159	63,110	305,241
Time gap	24 hours	24 hours	24 hours	1 year	15 mins
Timestamps	365	365	4,017	232	2,975

Table 10: The statistics of the datasets.

D Experimental details

D.1 Datasets

We select five representative datasets in TKG reasoning domain to evaluate our method. They are GDELT (Leetaru and Schrodt, 2013), WIKI (Leblay and Chekol, 2018), ICEWS14 (Garcia-Duran et al., 2018), ICEWS18 (Jin et al., 2020b) and ICEWS05-15 (Garcia-Duran et al., 2018). The YAGO dataset are supplemented with time information based on the traditional static KG YAGO3. GDELT is from the Global Database of Events, Language and Tone. The ICEWS series are sourced from the Integrated Crisis Early Warning System (Boschee et al., 2015). Following extensive previous work, We divide the five datasets into training, validation, and test sets based on timestamps, with corresponding ratios of 80%, 10% and 10%. Table 10 presents the dataset statistics.

Prompt Example: [0. Make_statement, 1. Consult, 10. Use_unconventional_violence, 5. Host_a_visit, 23. Threaten, 7. Praise_or_endorse, 3. Express_intent_to_meet_or_negotiate, 4. Make_a_visit, 6. Arrest,_detain,_or_charge_with_legal_action, 16. Engage_in_diplomatic_cooperation, 15. Express_intent_to_engage_in_diplomatic_cooperation(such_as_policy_support), 69. Appeal_for_intelligence, 2. Make_an_appeal_or_request, 8. Criticize_or_denounce, 26. Make_pessimistic_comment, 11. Use_conventional_military_force, 45. Expel_or_deport_individuals, 19. Discuss_by_telephone, 25. Abduct,_hijack,_or_take_hostage, 12. Engage_in_negotiation, 132. Appeal_for_policy_change, 22. Make_optimistic_comment, 27. Meet_at_a_'third'_location, 9. Accuse, 77. Acknowledge_or_claim_responsibility, 65. Kill_by_physical_assault, 46. Reduce_or_break_diplomatic_relations, 17. Sign_formal_agreement, 98. Apologize, 33. Make_empathetic_comment, 14. Demand, 13. Express_intent_to_cooperate, 38. Deny_responsibility, 29. Appeal_for_diplomatic_cooperation_(such_as_policy_support), 131. Attempt_to_assassinate, 18. Investigate, 30. Return,_release_person(s), 104. Torture, 68. Carry_out_suicide_bombing, 34. Use_tactics_of_violent_repression, 49. Complain_officially, 24. Demonstrate_or_rally, 57. Express_intent_to_cooperate_economically, 43. Impose_administrative_sanctions, 32. Provide_humanitarian_aid, 31. Protest_violently,_riot, 48. Sexually_assault, 36. Physically_assault, 44. Confiscate_property, 21. fight_with_small_arms_and_light_weapons, 20. Reject, 50. Rally_support_on_behalf_of, 81. Express_intent_to_settle_dispute, 116. Appeal_for_aid] Each item separated by a comma represents a type of relation, and the format for each relation is: id. relation. To better aggregate the connections between different relations, I want to use these relations as vertices to construct a graph. Considering the logic and process relationships between different relations, and based on the nature of each and relationship their possible interactions, I define the following five types of connections: [0. Causal Relationship, 1. Synonymous Relationship, 2. Oppositional Relationship, 3. Inclusion Relationship, 4. Progressive Relationship]. (1) Causal Relationship: One event (or action) serves as the cause of another event (or action), where the latter is the result of the former. (2) Synonymous Relationship: Two or more events (or actions) are semantically similar or interchangeable. (3) Oppositional Relationship: Two events (or actions) oppose each other in terms of goals, intentions, or outcomes. (4) Inclusion Relationship: One event (or action) is a subset or a specific manifestation of another event (or action). (5) Progressive Relationship: There is a progressive advancement between events (or actions), emphasizing logical, stage-wise development. Think step by step as follows: 1. Fully understand the meaning of each relationship in the text, and based on the five defined types of connections, construct edges between each relationship node, and try to fully explore the potential connections between as relationships much as possible. 2. The thinking process does not need to be output in the result. The final result should be returned in the following JSON format. EXAMPLE OUTPUT: [[10, 3, 11], [4, 2, 5], [15, 0, 16], [7, 2, 8], ...]
Output: [[10, 2, 11], [15, 0, 16], [3, 4, 4], [5, 1, 4], [13, 0, 16], [7, 2, 8], [9, 4, 77], [46, 2, 7], [23, 0, 9], [24, 4, 25], [13, 0, 4], [49, 2, 31], [132, 0, 14], [1, 4, 2], [5, 3, 17], [26, 2, 22], [81, 0, 16], [2, 0, 30], [43, 3, 44], [32, 3, 22], [57, 3, 16], [12, 0, 5], [49, 3, 31], [43, 0, 49], [36, 3, 48], [13, 4, 16], [31, 4, 24], [7, 4, 17], [11, 4, 10], [14, 2, 7]]

Table 9: A prompt example.

D.2 Evaluation metrics

In the experiments, we adopt the widely used metrics MRR and Hits@{1, 3, 10} to evaluate the effectiveness of TKG reasoning methods. MRR is the mean reciprocal rank of the correct facts for all queries, while Hits@k represents the proportion of times the true entity candidates appear among the top-k ranked candidates. Higher values for these metrics are better. We report the experimental results under the time-aware filtered setup, which is commonly used in recent works.

D.3 Baselines

To validate the effectiveness of our method, we compare it with recent SOTA methods. **Static KG reasoning methods** include: ComplEx (Trouillon et al., 2016), ConvE (Dettmers et al., 2018), Conv-TransE (Shang et al., 2019) and RotatE (Sun et al., 2019). **TKG reasoning (interpolation) methods** include: TTransE (Leblay and Chekol, 2018), DE-SimplE (Goel et al., 2020) and TNT-ComplEx (Lacroix et al., 2020). **TKG reasoning (extrapolation) methods** include: CyGNet (Zhu et al., 2021), xERTE (Han et al., 2021), RE-GCN (Li et al., 2021), CEN (Li et al., 2022b), TiRGN (Li et al., 2022a), HisMatch(Li et al., 2022c), RETIA(Liu et al., 2023), RPC (Liang et al., 2023), LogCL (Chen et al., 2024b), CRAFT (Zhang et al., 2024), HTCCN (Chen et al., 2024a), as well as **LLM-based method** such as GenTKG (Liao et al., 2024b) and LLM-DA(Wang et al., 2024).

D.4 Implementation details

For all datasets, the embedding size d is set to 200, the learning rate is set to 0.001, and the batch size is set to the number of quadruples at each timestamp. The number of layers for entity aggregation R-GCN and relation aggregation R-GCN is set to 2, with a dropout rate of 0.2 for each layer. The optimal local historical KG subgraph sequence lengths for ICEWS14, ICEWS18, ICEWS05-15, GDELT and WIKI are set to 7, 7, 9, 7 and 1, respectively. The parameters are optimized using Adam (Kingma and Ba, 2015) during training. We follow the work of adding static KG information to the three datasets (Li et al., 2021). In all datasets, the weight λ is set to 0.9. For the decoder on all datasets, the number of kernels is set to 50, the kernel size is set to 2×3, and the dropout rate is set to 0.2. For the mining of semantic associations between relations, we implement it using the gpt-3.5-turbo model.

E Ablation study

To further investigate the contribution of each component to the model's performance, we conduct additional ablation sutdy on ICEWS14 dataset. The results are reported in Table 11.

Model	ICEWS14			
	MRR	Hits@1	Hits@3	Hits@10
w/o L	44.19	33.20	49.38	66.19
w/o G	48.58	37.03	54.81	71.23
w/o SG	49.95	38.99	56.45	70.72
w/o TG	50.03	38.78	56.23	71.64
SRM-LLM	**50.84**	**39.80**	**56.82**	**71.92**

Table 11: The ablation study results on ICEWS14 dataset.

F Comparison between original and spare ICEWS14

Table 12 presents the statistics of ICEWS14 and Spare ICEWS14.

Dataset	ICEWS14	Spare ICEWS14
Entities	7,128	6,320
Relations	230	177
Train	74,845	10,274
Valid	8,514	2,016
Test	7,371	1,791

Table 12: The statistics of original and spare ICEWS14.

G Parameter analysis

(1) Analysis of the rule length l: In the rule-based historical relation retrieval module, the rule length l affects the extraction of global logical relation history for the query quadruple. Therefore, we choose different values of l to analyze its impact on prediction performance. As shown in Figure 3, compared to when l is set to 2, both $l = 1$ and $l = 3$ lead to a performance decline. When the rule length is too short, the extracted global logical relation history is insufficient, while when the rule length is too long, the relevance of the extracted global logical relation history to the query decreases, resulting in a performance drop.

(2) Analysis of the number of layers in R-GCN: We conduct experiments to analyze the impact of changing the number of layers in R-GCN for entity

aggregation and relation aggregation, with results shown in Figure 4. It can be observed that on the ICEWS14 dataset, the two-hop results are slightly better than the one-hop results. However, increasing the number of layers to three does not improve the performance on the ICEWS14 dataset; instead, it leads to a decrease in performance.

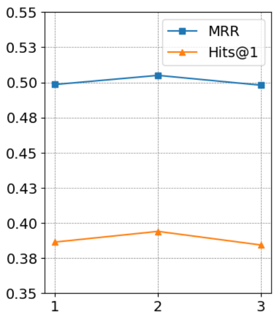


Figure 3: The results of different rule length l on the ICEWS14 dataset.

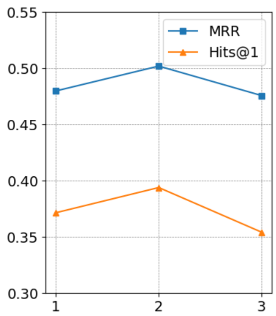


Figure 4: The results of different R-GCN layers on the ICEWS14 dataset.

(3) Analysis of the weight λ: To analyze the weight λ used for balancing global and local embedding representations of entities, we conduct experiments on the ICEWS14 dataset with different λ values ranging from 0 to 1, while keeping other hyperparameters fixed at their optimal values. The results for MRR and Hist@{1, 3, 10} are shown in Figure 5. By observing the results, it can be seen that the performance is relatively poor when λ is 0 or 1, indicating that considering only local or global historical information is not effective for the prediction task. As λ increases, the model performance shows an overall trend of first rising and then declining, suggesting that local historical information, which is closer to the query timestamp, plays a more significant role.

(4) Analysis of the parameter k: From Figure 6, we can observe that the model performance degrades relatively when the value of k is either smaller or larger. We believe that when k is smaller, the number of historical subgraphs is insufficient to capture enough historical information. As a result, the model cannot adequately learn the evolution patterns of entities and relations, thereby affecting its performance negatively. On the other hand, when k is larger, historical subgraphs that are distant from the query timestamp may contain irrelevant or noisy facts, which can interfere with the learning process of the model and lead to a relative decline in performance. This indicates that appropriately increasing the amount of historical information can enhance model performance, but an excessive amount of historical information may cause performance saturation or even degradation.

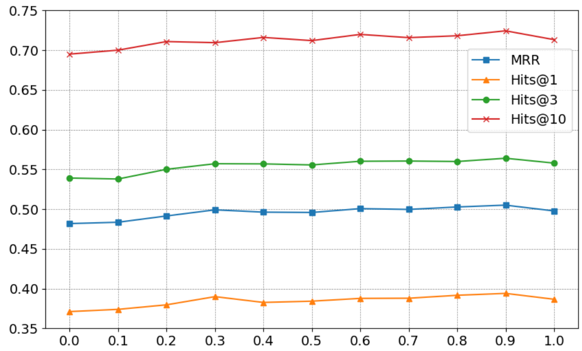


Figure 5: The results of different λ on the ICEWS14 dataset.

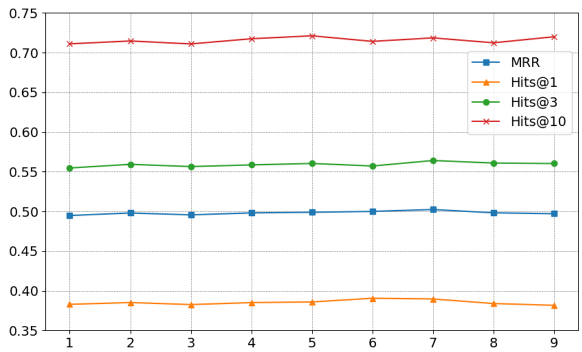


Figure 6: The results of different k on the ICEWS14 dataset.

H Time overhead

We compare the overheads of HTCCN, LogCL, RETIA and SRM-LLM in terms of training and testing time, as shown in Table 13.

Model	Train (per epoch)	Test (per epoch)
HTCCN	5.93 min	1.05 min
LogCL	3.33 min	0.18 min
RETIA	93.68 min	8.78 min
SRM-LLM	13.1 min	1.32 min

Table 13: Time overheads on ICEWS14.