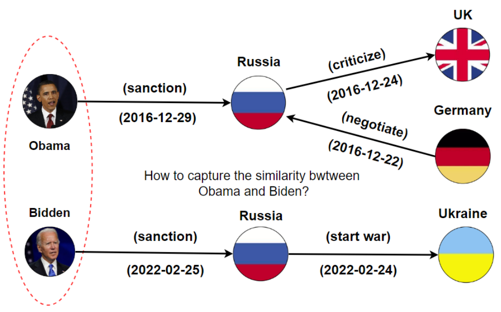
Multi-Granularity History and Entity Similarity Learning for Temporal Knowledge Graph Reasoning

Shi Mingcong[1], Chunjiang Zhu[2], Detian Zhang[1]*, Shiting Wen[3], Qing Li[4]

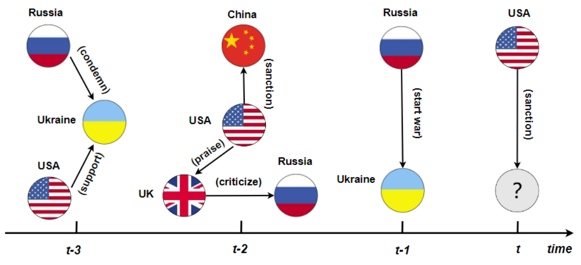
[1]School of Computer Science and Technology, Soochow University, Suzhou, China
[2]Department of Computer Science, University of North Carolina Greensboro, USA
[3]School of Computer and Data Engineering, NingboTech University, Ningbo, China
[4]Department of Computing, the Hong Kong Polytechnic University, Hong Kong
mcshi@stu.suda.edu.cn, chunjiang.zhu@uncg.edu, detian@suda.edu.cn
wensht@nbt.edu.cn, qing-prof.li@polyu.edu.hk

Abstract

Temporal Knowledge Graph (TKG) reasoning, aiming to predict future unknown facts based on historical information, has attracted considerable attention due to its great practical value. Insight into history is the key to predict the future. However, most existing TKG reasoning models singly capture repetitive history, ignoring the entity's multi-hop neighbour history which can provide valuable background knowledge for TKG reasoning. In this paper, we propose **M**ulti-**G**ranularity History and **E**ntity **S**imilarity **L**earning (MGESL) model for Temporal Knowledge Graph Reasoning, which models historical information from both coarse-grained and fine-grained history. Since similar entities tend to exhibit similar behavioural patterns, we also design a hypergraph convolution aggregator to capture the similarity between entities. Furthermore, we introduce a more realistic setting for the TKG reasoning, where candidate entities are already known at the timestamp to be predicted. Extensive experiments on three benchmark datasets demonstrate the effectiveness of our proposed model.



(a) An example of similarity learning problem



(b) An example of history of different granularities of entity

Figure 1: Illustration of the two problems of TKG reasoning task.

1 Introduction

Temporal Knowledge Graphs (TKGs), served as a way to represent and store dynamic knowledge, have shown great value in many applications, such as event prediction (Deng et al., 2020), question answering (Mavromatis et al., 2022) and recommendation (Liu et al., 2023b). In TKGs, each fact is represented as a quadruple, e.g., (Obama, sanction, Russia, 2016-12-29) in Figure 1(a).

Reasoning over TKGs can be performed under two primary settings, i.e., interpolation and extrapolation (Jin et al., 2020). Given a TKG with timestamps from t_0 to t_n, interpolation mainly aims at inferring missing facts that occur at time t ($t_0 \leq t \leq t_n$), while extrapolation attempts to predict facts that occur at time t ($t > t_n$). In this paper, we mainly focus on TKG extrapolation. Most of existing extrapolation models (Jin et al., 2020; Li et al., 2021b, 2022b; Liu et al., 2023a) assume the candidate entities are unknown during the reasoning. However, there are cases that we already know the candidate entities, e.g., suspects are often identified beforehand in criminal investigations and candidates are usually already determined before the presidential election. In these cases, existing extrapolation models (Jin et al., 2020; Li et al., 2021b, 2022b; Liu et al., 2023a) cannot effectively utilize the information of those candidate entities because they treat all entities equally during the reasoning. Therefore, we introduce a new setting called the candidate entity known setting, where all the entities at t are known in advance. In contrast, if the

* Corresponding author.

candidate entities at t are unknown during the reasoning, we call this the candidate entity unknown setting. In this paper, both candidate entity known and unknown settings will be discussed.

To predict what will happen in the future, we found that (1) searching for similar entities, observing and understanding the evolutionary pattern of the actions of similar entities, and (2) delving into the entity historical context from multi-granularity are crucial. Figure 1(a) shows an example of TKG similarity learning problem, where Obama and Biden both sanction Russia. However, since Obama and Biden are not connected directly in this example, vanilla graph convolution is unable to capture the interaction between them. To address this issue, we realize that both Obama and Biden share the same relation of sanction. Since hypergraph convolution can enable information interaction among entities under the same relation, we therefore design a hypergraph convolutional aggregator to capture similarity information between them. Additionally, existing models (Jin et al., 2020; Li et al., 2021b) mainly focus on utilising the available temporal and structural information in the TKG for inference, ignoring the history information. Even though some recent studies (Zhu et al., 2021; Li et al., 2022a; Xu et al., 2023) tried to find the correct answer from long-term global repeated history (i.e., fine-grained history), but they ignore the more generalised history. For instance, Figure 1(b) illustrates a temporal knowledge graph with several timestamps, where the task is to predict the answer to the query (USA, sanction, ?, t). Most models (Zhu et al., 2021; Xu et al., 2023) prioritize repeated history, and return China as the answer. However the correct answer to the question is Russia which is a multi-hop neighbour of USA. To overcome this limitations, we further consider multi-hop neighbour entities (i.e., coarse-grained history) in TKG reasoning.

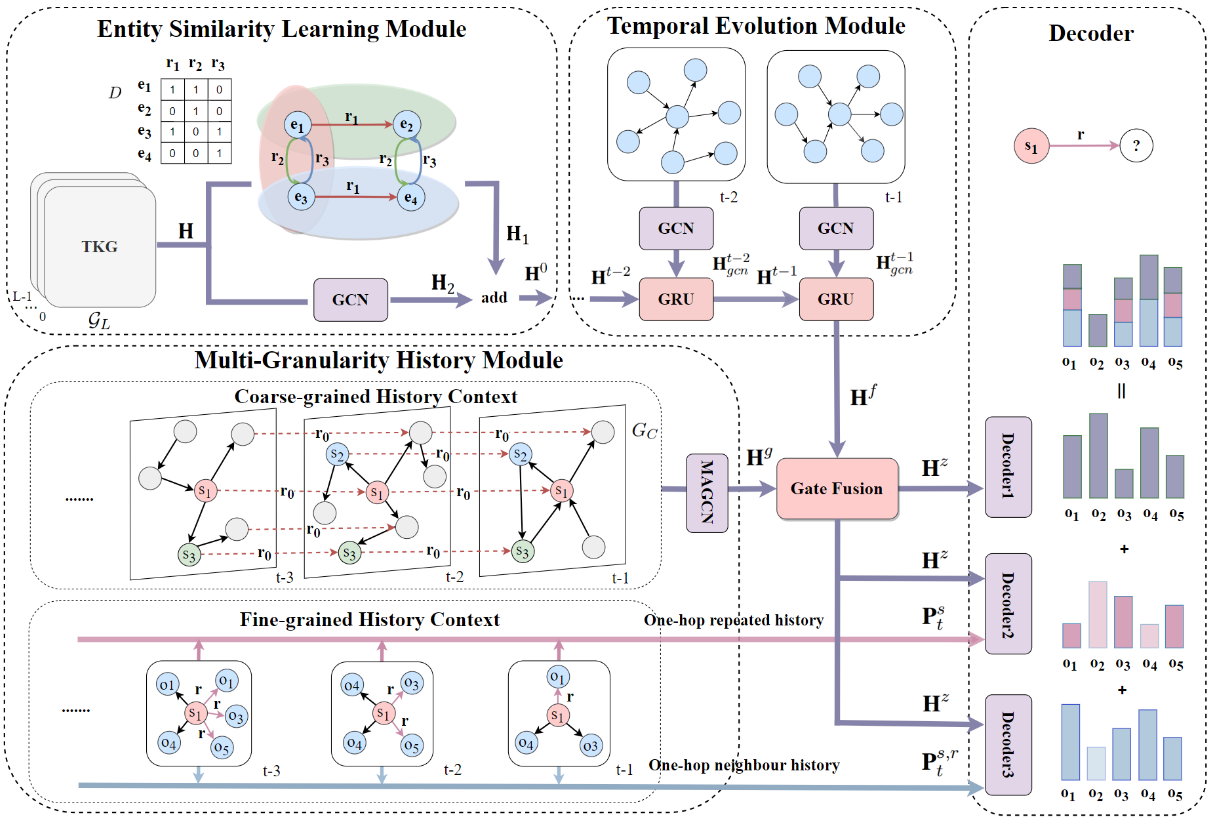
To this end, we consider history at two levels of granularity (i.e., fine and coarse-grained history) and entity similarity learning simultaneously, and propose the **M**ulti-**G**ranularity History and **E**ntity **S**imilarity **L**earning (MGESL) model for Temporal Knowledge Graph Reasoning. Specifically, MGESL consists of three modules, i.e., (1) Entity Similarity Learning Module, which is used to capture the similarity between entities that share the same relation; (2) Temporal Evolution Module, which is used to aggregate and transfer the KG information from spatial and temporal views, respectively; (3) Multi-Granularity History Module, which is used to capture history from both coarse and fine granularities. Our main contributions are summarized as follows:

- We propose a TKG reasoning model MGESL, which can simultaneously consider entity similarity learning, coarse-grained and fine-grained history. To the best of our knowledge, we are the first to consider these features together.
- We design a novel hypergraph convolutional aggregator to capture similarities between entities, and utilize the coarse-grained history to capture multi-hop historical contextual information and fine-grained history for decoding to make full use of historical information.
- Besides the candidate entity unknown setting, we also propose another realistic TKG reasoning setting, i.e., the candidate entities are already known. Extensive experiments on three benchmark datasets show that our proposed MGESL model outperforms existing TKG reasoning methods under both settings.

The remaining sections of this paper are structured as follows: Section 2 discusses related work on TKG reasoning models on the extrapolation setting. Section 3 presents the problem definition. Section 4 provides a detailed representation of the MGESL model. Section 5 contains the experimental analyses, followed by the conclusion in Section 6.

2 Related Work

Since TKG interpolation is outside the scope of our study, we mainly review the existing TKG reasoning models under the extrapolation setting. Many extrapolation models utilise the available temporal and structural information in TKG for inference. RE-Net (Jin et al., 2020) utilizes heterogeneous graph convolution (RGCN) (Schlichtkrull et al., 2018) to capture the structural information within the same timestamp and employs a recurrent neural network (RNN) to model the temporal information between different timestamps. RE-GCN (Li et al., 2021b) further constrains the evolution of entities by incorporating additional static attributes. However, they do not consider the history information. CyGNet (Zhu et al., 2021) and CENET (Xu et al., 2023) propose a copy mechanism to find the correct answer among long-term global



Entity Similarity Learning Module
Temporal Evolution Module
Decoder
TKG
H
H_1
H_2
H^0
GCN
add
GRU
Multi-Granularity History Module
Coarse-grained History Context
Fine-grained History Context
MAGCN
Gate Fusion
H^f
H^g
H^z
P_t^s
$P_t^{s,r}$
One-hop repeated history
One-hop neighbour history
Decoder1
Decoder2
Decoder3

Figure 2: Illustration of the proposed MGESL model. Entity Similarity Learning Module captures the similarities between entities that share the same relation. Temporal Evolution Module aggregates and transfers the KG information from spatial and temporal views, respectively. Multi-Granularity History Module models history from both coarse and fine granularity.

history, i.e., the fine-grained history. TiRGN (Li et al., 2022a) considers the sequential, repetitive and cyclical patterns of historical facts. However, they ignore the multi-hop neighbour history, i.e., the coarse-grained history. xERTE (Han et al., 2021) employs a subgraph sampling technique to construct interpretable reasoning graphs. CluSTeR (Li et al., 2021a) and TITer (Sun et al., 2021) both utilize reinforcement learning to search for a series of historical facts for reasoning. HGLS (Zhang et al., 2023) captures the long and short history of an entity by constructing global graphs. However, all the above models do not consider the importance of entity similarity learning in TKG reasoning.

3 Preliminaries

A temporal knowledge graph can be defined as $\mathcal{G} = \{\mathcal{G}_1, \mathcal{G}_2, ..., \mathcal{G}_T\}$, and T is the number of timestamps. The subgraph $\mathcal{G}_t = (\mathcal{E}, \mathcal{R}, \mathcal{F}_t)$ at t is a directed multi-relational graph, where $\mathcal{E}$ is the set of entities, $\mathcal{R}$ is the set of relations, and $\mathcal{F}_t$ is the set of facts at t. A fact in $\mathcal{F}_t$ can be formalized as a quadruple (s, r, o, t), where $s, o \in \mathcal{E}$ and $r \in \mathcal{R}$. It describes that a fact of relation type r occurs between subject entity s and object entity o at time t.

The extrapolation reasoning task aims to predict the missing object entity o or subject s via answering query like $(s, r, ?, t_q)$ or $(?, r, o, t_q)$ based on the historical facts $\{(s, r, o, t_i)|t_i < t_q\}$. For each quadruple (s, r, o, t), an inverse relation quadruple (o, r^{-1}, s, t) is often added to the dataset (Vashishth et al., 2020). Therefore, when predicting the missing subject of a query $(?, r, o, t_q)$, we can convert it into predicting $(o, r^{-1}, ?, t_q)$. Based on this, the model in this paper only aims to predict the missing object entity. We use **bold** items to denote vector embeddings. For example, $\mathbf{H} \in \mathbb{R}^{|\mathcal{E}|\times d}$ and $\mathbf{R} \in \mathbb{R}^{2|\mathcal{R}|\times d}$ are used to represent the randomly initial embedding of entities and relations respectively, where d denotes the embedding dimension.

4 Methodology

4.1 Model Overview

The framework of MGESL is shown in Figure 2, comprising three modules: (1) the Entity Similarity

Learning Module, (2) the Temporal Evolution Module, and (3) the Multi-Granularity History Module. First, the Entity Similarity Learning Module learns the representation of entity with similarity information. Next, the learned entity representation is fed to the Temporal Evolution Module, where it further learns about the structural and sequential characteristics of recent facts. Then, it combines with historical context information learnt from the coarse-grained history in the Multi-Granularity History Module. Finally, the entity representation is decoded under the guidance of the fine-grained history.

4.2 Entity Similarity Learning

4.2.1 Pre-Learning Graph

Inspired by the pre-training model (Devlin et al., 2019), we first construct a pre-learning graph and initially learn the representation of entities on the pre-learning graph. Entity similarity information is also learnt on this graph. For a TKG $\mathcal{G}$, we ignore the time factor to merge the subgraphs of the first L timestamps to form a pre-learning graph $\mathcal{G}_L$, i.e., $\mathcal{G}_L = (\mathcal{E}, \mathcal{R}, \mathcal{F}_L)$, where $\mathcal{F}_L = \{(s, r, o) \,|\, (s, r, o, t) \in \mathcal{F}_t, 0 < t < L\}$ is a set of facts.

4.2.2 Hypergraph Convolution

To effectively capture the similarity between entities in the pre-learning graph, we design a hypergraph convolutional network. First, we construct a hypergraph neighbourhood matrix $D \in \mathbb{R}^{|\mathcal{E}| \times 2|\mathcal{R}|}$, where $D_{i,j} = 1$ means the i^{th} entity is the subject entity of the j^{th} relation, otherwise it equals 0. Please note that for simplicity, we have omitted the inverse relation in Figure 2. As stated in Section 3, for each relation, we only aggregate messages from its subject entity through employing an inverse relation.

First, messages from the subject entity are passed into the relation:

$$\mathbf{X} = \frac{1}{2}\mathbf{W}_1 D^{-1}\mathbf{H} + \frac{1}{2}\mathbf{W}_2\mathbf{R} \tag{1}$$

where $\mathbf{W}_1$, $\mathbf{W}_2$ are the learnable weights. The result $\mathbf{X} \in \mathbb{R}^{|2\mathcal{R}| \times d}$ contains messages from the subject entities and the relation itself. Next, the relation message is passed into the subject entity:

$$\mathbf{H}_1 = \sigma(\frac{1}{2}\mathbf{W}_3 D\mathbf{X} + \frac{1}{2}\mathbf{W}_4\mathbf{H}) \tag{2}$$

where $\mathbf{W}_3$, $\mathbf{W}_4$ are the learnable weights and σ is the ReLU activation function. Through the above steps, we can initially learn the representation of entities $\mathbf{H}_1$, which incorporates the similarity information between entities.

4.2.3 Structural Encoder

Hypergraph convolution on the pre-learning graph mainly captures the similarity information between entities, but it cannot capture the inherent graph structure information of the pre-learning graph. Therefore, we utilize a heterogeneous graph convolution network (Vashishth et al., 2020) as a structural encoder to aggregate information from multiple relations and multi-hop neighbour entities on the pre-learning graph, which is defined as follows:

$$\mathbf{h}_s^{l+1} = \sigma\left(\sum\nolimits_{(s,r,o)\in\mathcal{F}_L} \frac{1}{c_s}\mathbf{W}_0^l(\mathbf{h}_o^l + \mathbf{r}) + \mathbf{W}_1^l\mathbf{h}_s^l\right) \tag{3}$$

where $\mathbf{h}_s^l$, $\mathbf{h}_o^l$ denote the l^{th} layer embeddings of entities s, o respectively, $\mathbf{r}$ denotes the embedding of relation r, c_s is a normalizing factor equal to the number of neighbours of s, $\mathbf{W}_0^l$ and $\mathbf{W}_1^l$ denote the learnable weights of the l^{th} layer, and σ is the ReLU activation function. We denote the entity embedding of the output of the last layer as $\mathbf{H}_2$. For convenience, we denote Equation (3) as GCN.

Given that the meaning of relation r remains consistent over time and the observation that updating the relation embeddings did not enhance the model's performance during our experiments, we do not directly update relation embedding in this paper to maintain its semantic stability. Finally, we combine $\mathbf{H}_1$ and $\mathbf{H}_2$ to get the entity representation $\mathbf{H}^0$,

$$\mathbf{H}^0 = \alpha\mathbf{H}_1 + (1-\alpha)\mathbf{H}_2 \tag{4}$$

where $\alpha \in [0, 1]$ denotes hyperparameter, $\mathbf{H}^0$ denotes entity embedding obtained by learning on the pre-learning graph, incorporating similarity and structural information between entities.

4.3 Temporal Evolution

Future facts are usually closely related to recent facts, and our temporal evolution module aims to model recent facts. KGs naturally have graph structure information, while TKGs have the additional dimension of time compared to KGs. Therefore, we aggregate and transfer the most h recent timestamps of the timestamp t to be predicted in TKG from both spatial and temporal views. To capture the structural information between entities, we also utilize the heterogeneous graph convolutional network in Equation (3) for each timestamp,

$$\mathbf{H}_{gcn}^{t-1} = \mathrm{GCN}(\mathbf{H}^{t-1}, \mathbf{R}) \tag{5}$$

where $\mathbf{H}^{t-1}$ denotes the entity embedding at time $t-1$ and the initial value of $\mathbf{H}^{t-h}$ at time $t-h$ is the output of the similarity learning module $\mathbf{H}^0$. $\mathbf{H}_{gcn}^{t-1}$ denotes the entity embedding after aggregation by GCN Encoder. In order to include the sequential dependencies of subgraphs at the previous timestamps, we utilize the gated recurrent unit (GRU) to update the representations of entities,

$$\mathbf{H}^t = \mathrm{GRU}(\mathbf{H}_{gcn}^{t-1}, \mathbf{H}^{t-1}). \tag{6}$$

We denote the output of the last timestamp as $\mathbf{H}^f$.

4.4 Multi-Granularity History Learning

4.4.1 Background Graph

In order to more accurately model the representation of entities and the connections between them, we construct a background graph G_C based on the most recent C timestamps, similar to HGLS (Zhang et al., 2023). Specifically, when the candidate entities are known, the steps to construct the background graph are as follows: (1) identify the position where each candidate entity appears in the recent C timestamps. (2) conduct breadth-first search from each candidate entity to extract their n-hop neighbours. (3) merge the common neighbours of candidate entities and add temporal edge $\mathbf{r}_0$ (a randomly initial vector) between identical entities across different timestamps. With the steps above, we have established a background graph for more accurate entity representation learning. When the candidate entities are unknown, we take all entities in TKG as candidates and then execute the above three steps to construct the background graph.

4.4.2 Multi-head Attention GCN (MAGCN)

We employ a heterogeneous graph convolution network that incorporates the multi-head attention mechanism to effectively capture entity representation in the background graphs. First, all entities in the background graph are initialised by $\mathbf{H}$ for their initial embedding. Next, we combine the embeddings of the subject entity, the relation, and the object entity to calculate their attention scores,

$$\beta_{s,r,o} = LeakyRelU(\mathbf{W}_5[\mathbf{h}^s \oplus \mathbf{r} \oplus \mathbf{h}^o]) \tag{7}$$

where $\mathbf{h}^s$, $\mathbf{h}^o$ and $\mathbf{r}$ denote the embeddings of entities s, o and relation r, respectively, $\mathbf{W}_5$ denotes learnable weight, and $\oplus$ is the concatenation operation. After that, we further calculate their coefficients based on the scores of each triple,

$$\alpha_{s,r,o} = \frac{exp(\beta_{s,r,o})}{\sum_{(s,r^i,o^i)\in N_s} exp(\beta_{s,r^i,o^i})} \tag{8}$$

where N_s denotes the set of all triples with s as subject entity. After that, we can attentively aggregate message from all neighbours of entity s in the background graph. The utilization of the multi-head attention mechanism can enhance the stability of the convolution. Formally, the aggregator is defned as follows:

$$\mathbf{h}_s^{l+1,c} = \|_{m=1}^{M} \, \sigma \left(\sum_{(s,r,o)\in N_s} \alpha_{s,r,o}^{m} \mathbf{W}_6^{l,m} (\mathbf{h}_o^l + \mathbf{r}) + \mathbf{W}_7^{l,m} \mathbf{h}_s^l \right) \tag{9}$$

$$\mathbf{h}_s^{l+1} = \mathbf{W}^c \mathbf{h}_s^{l+1,c} \tag{10}$$

where M denotes the number of attention heads, $\|$ represents concatenation, $\mathbf{h}_s^l$ and $\mathbf{h}_o^l$ denote the embedding of entity s and o after the l^{th} layer aggregation, $\mathbf{r}$ denotes the embedding of relation r, $\mathbf{W}_6^l$ and $\mathbf{W}_7^l$ are learnable weights, and σ is the ReLU activation function. $\mathbf{W}^c \in \mathbb{R}^{d\times dM}$ reduces the dimension of $h_s^{l+1,c}$ from dM to d. We denote the entity embedding of the last layer as $\mathbf{H}^g$.

Finally, we use a gate mechanism to fuse the entity embedding learnt from the temporal evolution module with the entity embedding learnt from the background graph,

$$\mathbf{H}^z = \sigma(\mathbf{U}) \odot \mathbf{H}^f + (1 - \sigma(\mathbf{U})) \odot \mathbf{H}^g \tag{11}$$

where $\mathbf{U} \in \mathbb{R}^{|\mathcal{E}|\times d}$ denotes the gate vector, $\odot$ denotes element-wise dot option, σ denotes $sigmoid$ function to map values to the range of 0 to 1. Finally, we obtain a representation of the entity $\mathbf{H}^z$, which incorporates the similarity information between entities, the entity's recent temporal information and contextual information.

4.4.3 Fine-grained History

Based on human experience in predicting future facts, the answer to a query is often an entity that is closely related to the current entity. Therefore, we extract two kinds of fine-grained histories, i.e., one-hop history neighbours and repeated history answers (Li et al., 2022a). Specifically, for a query $(s, r, ?, t)$ the indicator vector $\mathbf{P}_t^s$ of one-hop history neighbours for the entity s at t can be defined as follows:

$$\mathbf{P}_t^s = \mathbf{p}_0^s \vee \mathbf{p}_1^s \vee \mathbf{p}_2^s \vee ... \vee \mathbf{p}_{t-1}^s \tag{12}$$

where $\mathbf{p}_t^s$ denotes a vector where each element represents an entity. If the corresponding element of an entity is 1, it means that the entity is a one-hop neighbour of s at t, otherwise it is 0. The symbol $\vee$ represents the bitwise OR operation. Similarly, we can calculate the repeated history answers indicator vector $\mathbf{P}_t^{s,r}$,

$$\mathbf{P}_t^{s,r} = \mathbf{p}_0^{s,r} \vee \mathbf{p}_1^{s,r} \vee \mathbf{p}_2^{s,r} \vee ... \vee \mathbf{p}_{t-1}^{s,r} \tag{13}$$

where $\mathbf{p}_t^{s,r}$ denotes a vector where each element indicates whether a corresponding entity is an answer to the query $(s, r, ?, t)$;it is 1 if the entity is an answer and 0 otherwise.

4.5 Fine-grained History Guided Decoder

4.5.1 Scoring Function

We utilize ConvTransE (Shang et al., 2019) as decoder to fuse the semantic information of s and r in query $(s, r, ?, t)$. Since $\mathbf{H}^z$ already incorporates information of the coarse-grained history, the scores caculated based on coarse-grained history can be defined as follows:

$$\mathbf{p}^{coarse} = softmax(\text{ConvTransE}(\mathbf{h}_t^s, \mathbf{r})\mathbf{H}^z) \tag{14}$$

where $\mathbf{h}_t^s$ and $\mathbf{r}$ denote the embedding of subject entity s and relation r, respectively. For the fine-grained history (i.e., one-hop neighbour history and repeated history), we use these two vectors ($\mathbf{P}_t^s$ and $\mathbf{P}_t^{s,r}$) generated in section 4.4.3 to guide the decoder in scoring, i.e.,

$$\mathbf{p}^{local} = softmax(\text{ConvTransE}(\mathbf{h}_t^s, \mathbf{r})\mathbf{H}^z\mathbf{P}_t^s) \tag{15}$$

$$\mathbf{p}^{history} = softmax(\text{ConvTransE}(\mathbf{h}_t^s, \mathbf{r})\mathbf{H}^z\mathbf{P}_t^{s,r}) \tag{16}$$

where $\mathbf{p}^{local}$ and $\mathbf{p}^{history}$ denote the scores guided by one-hop neighbour history and repeated history respectively. The final score is calculated as follows:

$$\mathbf{p} = \mu_1\mathbf{p}^{coarse} + \mu_2\mathbf{p}^{local} + \mu_3\mathbf{p}^{history} \tag{17}$$

where $\mu_1, \mu_2, \mu_3 \in [0, 1]$ are hyperparameters and $\mu_1 + \mu_2 + \mu_3 = 1$.

4.5.2 Training Objective

Predicting the object entity based on a given query $(s, r, ?, t)$ can be viewed as a multi-class classification task (Jin et al., 2020), where each class corresponds to one entity. The learning objective is to minimize the following cross-entropy loss $\mathcal{L}$ during training:

$$\mathcal{L} = -\sum\nolimits_{(s,r,o,t)\in\mathcal{G}} \mathbf{y}_t^e \log \mathbf{p}(o \mid s, r, t) \tag{18}$$

where $\mathbf{p}(o \mid s, r, t)$ is the final probability score of entity, $\mathbf{y}_t^e \in \mathbb{R}^{|\mathcal{E}|}$ is the label vector, of which the element is 1 if the fact occurs, otherwise is 0.

5 Experiments

5.1 Setup

5.1.1 Datasets

We use three typical TKG datasets in our experiments: ICEWS14 (Riloff et al., 2018), ICEWS18 (Jin et al., 2020), and ICEWS05-15 (Riloff et al., 2018). We divide them into training, validation, and test sets with a proportion of 80%, 10%, and 10% by timestamps following (Li et al., 2021b, 2022a; Xu et al., 2023). The details of datasets statistics are shown in Appendix A.

5.1.2 Baselines

Under the candidate entity unknown setting, we compare our proposed MGESL model with three kinds of baselines: (1) Static KG reasoning models, (2) Interpolated TKG reasoning models, and (3) Current state-of-the-art extrapolated TKG reasoning model. Under the candidate entity known setting, we mainly focus on comparing to the extrapolated TKG reasoning models. For the details of baselines under both candidate entity known and unknown settings, please refer to Appendix B.

5.1.3 Training Settings and Evaluation Metrics

We report a widely used time-aware filtered version (Sun et al., 2021; Li et al., 2022a,b) of Mean Reciprocal Ranks (MRR) and Hits@1/3/10. For implementation details and parameter sensitivity analysis experiments of MGESL, please refer to Appendix C and D, respectively.

5.2 Results

Table 1 presents the MRR and Hits@1/3/10 results of entity prediction on three TKGs under the candidate entity unknown setting. Specifically, our proposed MGESL significantly outperforms all the

Model	ICEWS14				ICEWS18				ICEWS05-15			
	MRR	Hit@1	Hit@3	Hit@10	MRR	Hit@1	Hit@3	Hit@10	MRR	Hit@1	Hit@3	Hit@10
DistMult (Yang et al., 2015)	15.44	10.19	17.24	23.92	11.51	7.03	12.87	20.86	17.95	13.12	20.71	29.32
ConvE (Dettmers et al., 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
ComplEx (Trouillon et al., 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
ConvTransE (Shang et al., 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 (Sun et al., 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 (Jiang et al., 2016)	13.43	3.11	17.32	34.55	8.31	1.92	8.56	21.89	15.57	4.80	19.24	38.29
DE-SimplE (Goel et al., 2020)	32.67	24.43	35.69	49.11	19.30	11.53	21.86	34.80	35.02	25.91	38.99	52.75
TA-DistMult (Riloff et al., 2018)	26.47	17.09	30.22	45.41	16.75	8.61	18.41	33.59	24.31	14.58	27.92	44.21
RE-NET (Jin et al., 2020)	39.86	30.11	44.02	58.21	29.78	19.73	32.55	48.46	43.67	33.55	48.83	62.72
GyGNet (Zhu et al., 2021)	37.65	27.43	42.63	57.90	27.12	17.21	30.97	46.85	40.42	29.44	46.06	61.60
xERTE (Han et al., 2021)	40.79	32.70	45.67	57.30	29.31	21.03	33.51	46.48	46.62	37.84	52.31	63.92
RE-GCN (Li et al., 2021b)	39.42	30.13	43.80	57.08	27.51	17.82	31.17	46.55	38.27	27.43	43.06	59.93
TITER (Sun et al., 2021)	41.73	32.74	—	58.44	29.98	22.05	—	44.83	47.60	38.29	—	64.86
TLogic (Liu et al., 2022)	40.90	32.10	45.50	57.60	30.00	22.10	33.50	44.80	47.70	38.00	52.90	65.80
CEN (Li et al., 2022b)	42.20	32.08	47.46	61.31	31.50	21.70	35.44	50.59	45.27	34.18	—	66.46
TiRGN (Li et al., 2022a)	41.52	32.04	46.20	59.62	31.70	21.82	35.90	51.15	48.52	37.55	53.54	68.74
CENET (Xu et al., 2023)	41.30	32.58	—	58.22	29.65	19.98	—	48.23	47.13	37.25	—	67.61
DaeMon (Dong et al., 2023)	—	—	—	—	31.85	22.67	35.92	49.80	—	—	—	—
HGLS (Zhang et al., 2023)	40.28	30.39	44.95	59.56	31.36	21.27	35.25	51.23	50.08	39.32	56.03	70.49
RETIA (Liu et al., 2023a)	41.61	31.66	46.36	60.61	31.23	21.55	35.07	50.17	>20Days	>20Days	>20Days	>20Days
MGESL (ours)	**45.88**	**35.43**	**51.54**	**65.70**	**34.18**	**23.66**	**38.64**	**54.89**	**53.78**	**42.52**	**60.40**	**75.04**

Table 1: Performance on three datasets in terms of MRR (%), Hit@1 (%), Hit@3 (%) and Hit@10 (%) under the candidate entity unknown setting. The best is highlighted in boldface, and the second is underlined.

Model	ICEWS14				ICEWS18				ICEWS05-15			
	MRR	Hit@1	Hit@3	Hit@10	MRR	Hit@1	Hit@3	Hit@10	MRR	Hit@1	Hit@3	Hit@10
RE-GCN (Li et al., 2021b)	46.19	34.97	51.79	67.97	33.90	23.20	38.06	55.11	54.98	43.50	61.52	76.49
TiRGN (Li et al., 2022a)	47.46	36.50	52.68	68.65	34.88	23.96	39.33	56.48	55.87	44.44	62.31	77.45
HGLS (Zhang et al., 2023)	47.00	35.06	—	70.41	29.32	19.21	—	49.83	46.21	35.32	—	67.12
MGESL (ours)	**51.86**	**40.49**	**58.26**	**73.41**	**37.57**	**26.10**	**42.63**	**60.16**	**58.06**	**46.84**	**64.47**	**79.63**

Table 2: Performance on three datasets in terms of MRR (%), Hit@1 (%), Hit@3 (%) and Hit@10 (%) under the candidate entity known setting. The best is highlighted in boldface, and the second is underlined.

static models (i.e., the first block in Table 1) because they ignore the time dimension of the facts in TKGs. MGESL also performs much better than the temporal models for the interpolation setting (i.e., the second block in Table 1) because MGESL additionally captures temporally sequential patterns by temporal evolution module. In comparison to the current sate-of-the-art temporal models under the extrapolation setting (i.e., the third block in Table 1), our model also achieves notable improvements. Specifically, MGESL improves approximately 8.72%, 8.22%, 8.60%, and 7.16% on ICEWS14 for MRR, Hit@1, Hit@3, and Hit@10, respectively. This is because our model can effectively capture the similarity information between entities by hypergraph convolution and model the representation of entities more accurately from multiple granularities.

Table 2 shows that MGESL also significantly outperforms other TKG extrapolation models under the candidate known setting. Specifically, MGESL improves approximately 9.27%, 10.93%, 10.59%, and 4.26% on ICEWS14 for MRR, Hit@1, Hit@3, and Hit@10, respectively. These improvements mainly arises from the background graph constructed by the candidate entities which captures the coarse-grained history and the two kinds of fine-grained histories we extracted. The background graph allows us to comprehensively understand and analyze the connections between these entities and effectively find the correct answer. The fine-grained history can guide the model to converge quickly and make more precise predictions.

5.3 Ablation Study

The ablation studies are performed on ICEWS14 with all four evaluation metrics. Seven submodels are compared, including (1) MGESL without similarity learning module (MGESL w/o SLM), (2) MGESL without temporal evolution module (MGESL w/o TEM), (3) MGESL without fine-grained history (MGESL w/o Fine), (4) MGESL without coarse-grained history (MGESL w/o Coarse), (5) MGESL without repeated history

Model	ICEWS14			
	MRR	Hit@1	Hit@3	Hit@10
MGESL w/o SLM	44.10	33.89	49.37	63.35
MGESL w/o TEM	43.71	33.14	49.17	64.32
MGESL w/o Fine	42.20	32.14	46.78	61.93
MGESL w/o Coarse	42.94	33.07	48.08	61.59
MGESL w/o Fine-his	43.96	33.56	49.18	64.01
MGESL w/o Fine-loc	44.27	33.97	49.48	64.21
MGESL	**45.88**	**35.43**	**51.54**	**65.70**

Table 3: Ablation results under the candidate unknown setting. The best performance is highlighted in boldface.

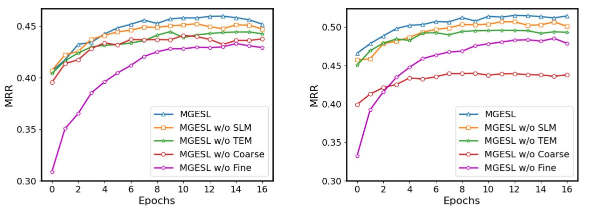
Model	ICEWS14			
	MRR	Hit@1	Hit@3	Hit@10
MGESL w/o SLM	50.21	39.05	56.38	71.30
MGESL w/o TEM	49.69	38.57	55.91	70.58
MGESL w/o Fine	46.61	35.37	52.50	68.51
MGESL w/o Coarse	42.75	32.26	47.82	61.59
MGESL w/o Fine-his	50.00	38.77	56.03	71.83
MGESL w/o Fine-loc	49.97	38.68	56.35	71.98
MGESL	**51.86**	**40.49**	**58.26**	**73.41**

Table 4: Ablation results under the candidate known setting. The best performance is highlighted in boldface.

(MGESL w/o Fine-his), (6) MGESL without one-hop history neighbours (MGESL w/o Fine-loc), (7) the original MGESL model (MGESL).

Table 3 shows the ablation results under the candidate entity unknown setting. When the similarity learning module (SLM) and temporal evolution module (TEM) are removed, the performance of the model decreased by 3.87% and 4.73% for MRR respectively, which indicates the effectiveness of these two modules. We can notice that removing the fine-grained history module (Fine) degrades the performance of the model more severely compared to removing the coarse-grained history module (Coarse), which causes a 8.02% performance degradation for MRR compared with MGESL. This is because coarse-grained history may contain more noisy information compared to fine-grained history under candidate unknown setting. When either repeated history or one-hop history neighbours is removed, the performance of the model declined by 4.18% or 3.51%, respectively.

Table 4 shows the ablation results under the candidate entity known setting. Performance declined when either the entity similarity module (SLM) or the temporal evolution module (TEM) is removed. In contrast to the candidate unknown setting, the candidate known setting demonstrates that removing coarse-grained history has a more significant impact on model performance compared to removing fine-grained history, causing a 17.2% performance degradation for MRR compared with MGESL. This is because when we have knowledge of the candidate entities, the background graph that we build using these entities can serve as an effective means to understand and learn the relationships between them. Also, after removing repeated history or one-hop history neighbours, the performance of the model declined by 3.59% and 3.64%, respectively.



(a) candidate unknown setting (b) candidate known setting

Figure 3: Convergence analysis results on ICEWS14 in MRR.

5.4 Convergence Analysis

Figure 3 presents the convergence analysis results of our study on ICEWS14 dataset. Obviously, after the initial training epoch, "MGESL w/o Fine" falls noticeably behind the other models in terms of MRR metrics, and requires more epochs to attain the optimal performance compared to the other models as shown in Figure 3(a). This demonstrates that fine-grained history can serve as a good guide for the model to learn during the training process.

Similarly, as shown in Figure 3(b), we notice that after the initial epoch of training, the results of "MGESL w/o Fine" are still the lowest. Besides, the results of "MGESL w/o Coarse" no longer remain almost the same with other models as in Figure 3(a). This phenomenon indicates that both coarse-grained and fine-grained histories are crucial in facilitating the model's convergence during training, particularly when the candidate entities are known. The fine-grained history can make the model converges faster, while the coarse-grained history can improve the accuracy of the model to a great extent. These findings further validate the effectiveness of our capturing historical information from various granularities.

6 Conclusion

In this paper, we introduce the MGESL model for TKG extrapolation. The model considers entity similarity, coarse-grained history and fine-grained history simultaneously. To capture entity similarities, we design a hypergraph convolutional aggregator. We construct the background graph to capture the coarse-grained history and extract two kinds of fine-grained histories to guide the model reasoning. Moreover, we introduce a more realistic setting for TKG extrapolation, i.e., candidate entities are known. Extensive experiments on three datasets demonstrate the effectiveness of our model.

Limitations

Under the candidate entity known setting, we need to know all of the candidate entities in advance, which is not always realistic. Therefore, in our future work, we will focus on how to accurately predict candidate entities.

Acknowledgements

Detian Zhang was supported by the Collaborative Innovation Center of Novel Software Technology and Industrialization, the Priority Academic Program Development of Jiangsu Higher Education Institutions. Shiting Wen was supported by Zhejiang Provincial Natural Science Foundation of China under Grant No. LY24F020021, and the Ningbo Science and Technology Special Projects under Grant No. 2022Z235 and 20224Z263. Chunjiang Zhu was supported by NSF grant CNS-2349369. Qing Li was supported by a grant from HK RGC Theme-based Research Scheme (PolyU No.: T43-513/23-N).

References

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*, page 1585–1595, New York, NY, USA. Association for Computing Machinery.

Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. 2018. Convolutional 2d knowledge graph embeddings. In *Proceedings of the 32th AAAI Conference on Artificial Intelligence*, pages 1811–1818, New Orleans, Louisiana, USA. AAAI Press.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 17th Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4171–4186, Minneapolis, MN, USA. Association for Computational Linguistics.

Hao Dong, Zhiyuan Ning, Pengyang Wang, Ziyue Qiao, Pengfei Wang, Yuanchun Zhou, and Yanjie Fu. 2023. Adaptive path-memory network for temporal knowledge graph reasoning. In *Proceedings of the 32th International Joint Conference on Artificial Intelligence*, pages 2086–2094, Macao, SAR, China. ijcai.org.

Rishab Goel, Seyed Mehran Kazemi, Marcus A. Brubaker, and Pascal Poupart. 2020. Diachronic embedding for temporal knowledge graph completion. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence*, pages 3988–3995, New York, NY, USA. AAAI Press.

Zhen Han, Peng Chen, Yunpu Ma, and Volker Tresp. 2021. Explainable subgraph reasoning for forecasting on temporal knowledge graphs. In *Proceedings of the 9th International Conference on Learning Representations*, Austria. OpenReview.net.

Tingsong Jiang, Tianyu Liu, Tao Ge, Lei Sha, Baobao Chang, Sujian Li, and Zhifang Sui. 2016. Towards time-aware knowledge graph completion. In *Proceedings of the 26th International Conference on Computational Linguistics*, pages 1715–1724, Osaka, Japan. ACL.

Woojeong Jin, Meng Qu, Xisen Jin, and Xiang Ren. 2020. Recurrent event network: Autoregressive structure inference over 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.

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 *Proceedings of the 31th International Joint Conference on Artificial Intelligence*, pages 2152–2158, Vienna, Austria. ijcai.org.

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*, pages 290–296, Dublin, Ireland. Association for Computational Linguistics.

Zixuan Li, Xiaolong Jin, Saiping Guan, Wei Li, Jiafeng Guo, Yuanzhuo Wang, and Xueqi Cheng. 2021a. Search from history and reason for future: Two-stage reasoning on temporal knowledge graphs. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics*, pages 4732–4743. Association for Computational Linguistics.

Zixuan Li, Xiaolong Jin, Wei Li, Saiping Guan, Jiafeng Guo, Huawei Shen, Yuanzhuo Wang, and Xueqi Cheng. 2021b. 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, Canada. ACM.

Kangzheng Liu, Feng Zhao, Guandong Xu, Xianzhi Wang, and Hai Jin. 2023a. RETIA: relation-entity twin-interact aggregation for temporal knowledge graph extrapolation. In *Proceedings of the 39th IEEE International Conference on Data Engineering*, pages 1761–1774, Anaheim, CA, USA. IEEE.

Xiaoqian Liu, Xiuyun Li, Yuan Cao, Fan Zhang, Xiongnan Jin, and Jinpeng Chen. 2023b. Mandari: Multimodal temporal knowledge graph-aware sub-graph embedding for next-poi recommendation. In *IEEE International Conference on Multimedia and Expo, ICME 2023, Brisbane, Australia, July 10-14, 2023*, pages 1529–1534. 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 36th AAAI Conference on Artificial Intelligence*, pages 4120–4127. AAAI Press.

Costas Mavromatis, Prasanna Lakkur Subramanyam, Vassilis N. Ioannidis, Adesoji Adeshina, Phillip R. Howard, Tetiana Grinberg, Nagib Hakim, and George Karypis. 2022. Tempoqr: Temporal question reasoning over knowledge graphs. In *Thirty-Sixth AAAI Conference on Artificial Intelligence*, pages 5825–5833. AAAI Press.

Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii. 2018. Learning sequence encoders for temporal knowledge graph completion. In *Proceedings of the 22th Conference on Empirical Methods in Natural Language Processing*, pages 4816–4821, Brussels, Belgium. Association for Computational Linguistics.

Michael Sejr Schlichtkrull, Thomas N. Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *Proceedings of the 15th Semantic Web International Conference*, volume 10843, pages 593–607, Heraklion, Crete, Greece. 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 33th AAAI Conference on Artificial Intelligence*, pages 3060–3067, Honolulu, Hawaii, USA. AAAI Press.

Haohai Sun, Jialun Zhong, Yunpu Ma, Zhen Han, and Kun He. 2021. Timetraveler: Reinforcement learning for temporal knowledge graph forecasting. In *Proceedings of the 25th Conference on Empirical Methods in Natural Language Processing*, pages 8306–8319, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. 2019. Rotate: Knowledge graph embedding by relational rotation in complex space. In *Proceedings of the 7th International Conference on Learning Representations*, New Orleans, LA, USA. OpenReview.net.

Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex embeddings for simple link prediction. In *Proceedings of the 33nd International Conference on Machine Learning*, pages 2071–2080, New York City, NY, USA. JMLR.org.

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

Yi Xu, Junjie Ou, Hui Xu, and Luoyi Fu. 2023. Temporal knowledge graph reasoning with historical contrastive learning. In *Proceedings of the 37th AAAI Conference on Artificial Intelligence*, pages 4765–4773, Washington, DC, USA. AAAI Press.

Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2015. Embedding entities and relations for learning and inference in knowledge bases. In *Proceedings of the 3rd International Conference on Learning Representations*, San Diego, CA, USA.

Mengqi Zhang, Yuwei Xia, Qiang Liu, Shu Wu, and Liang Wang. 2023. Learning long- and short-term representations for temporal knowledge graph reasoning. In *Proceedings of the ACM Web Conference 2023*, pages 2412–2422, Austin, TX, USA. ACM.

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 35th AAAI Conference on Artificial Intelligence*, pages 4732–4740. AAAI Press.

A Datasets

The statistics of the three TKG datasets used in our experiments are summarized in Table 5.

B Baselines

Under the candidate entity unknown setting, we compare our proposed MGESL model with three kinds of baselines, i.e., (1) Static KG reasoning

# Datasets	ICEWS14	ICEWS18	ICEWS05-15
# Entities	7128	23033	10488
# Relations	230	256	251
# Training	74845	373018	368868
# Validation	8514	45995	46302
# Test	7371	49545	46159
# Granularity	24 hours	24hours	24hours

Table 5: The statistics of the datasets. Granularity represents time granularity between temporally adjacent facts.

models, i.e., DistMult (Yang et al., 2015), ConvE (Dettmers et al., 2018), ComplEx (Trouillon et al., 2016), ConvTransE (Shang et al., 2019) and RotatE (Sun et al., 2019). (2) Interpolated TKG reasoning models, i.e., TTransE (Jiang et al., 2016), DE-SimplE (Goel et al., 2020), and TA-DistMult (Riloff et al., 2018). (3) Current state-of-the-art extrapolated TKG reasoning models, i.e., RE-NET (Jin et al., 2020), CyGNet (Zhu et al., 2021), xERTE (Han et al., 2021), RE-GCN (Li et al., 2021b), TITER (Sun et al., 2021), TLogic (Liu et al., 2022), CEN (Li et al., 2022b), TiRGN (Li et al., 2022a), CENET (Xu et al., 2023), HGLS (Zhang et al., 2023), RETIA (Liu et al., 2023a) and DaeMon (Dong et al., 2023). For RE-GCN (Li et al., 2021b) and TiRGN (Li et al., 2022a). we remove the static information from the model to ensure the fairness of comparisons between all baselines.

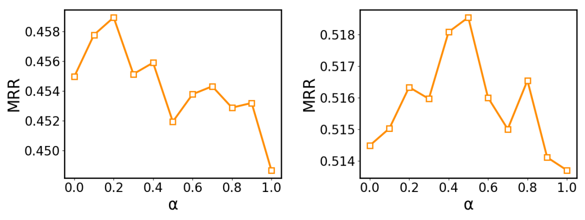
Under the candidate entity known setting, we mainly focus on comparing to the extrapolated TKG reasoning models, including RE-GCN (Li et al., 2021b), TiRGN (Li et al., 2022a) and HGLS (Zhang et al., 2023). In this setting, we assume that the entities in the timestamp to be predicted are all known. We propose this setting for the following two reasons: (1) There are scenarios in reality where we already know the candidate entities and all we need to do is to find out the exact answer from these entities, such as presidential elections where president is often chosen from multiple known candidates. (2) When entities are given to predict the relationship between them, the entities are also known. As the previous TKG extrapolation models were conducted under the candidate entity unknown setting, we intentionally revealed all the entities of the timestamp to be predicted. This means that these models only need to score and find the correct answer from the revealed candidate entities, not from all entities in the TKG.

C Implementation Details

We employed a random search algorithm to sample a fixed number of combinations within the hyperparameter space. Specifically, the embedding dimension d ranges from 100, 200, and 300. The length of timestamps for pre-learning graph L ranges from 30, 50, 80, and 100, while the length of timestamps for background graph C ranges from 10, 20, and 30. The number of GCN convolutional layers and the hops of neighbours n were selected from 1, 2, and 3. The hyperparameter α ranges from 0.1 to 0.9. The length of historical timestamps h is set to 9 and the number of attention heads M is set to 5. Additionally, the parameters μ_1, μ_2 and μ_3 ranges from 0.1 to 0.9 with a step size of 0.1, ensuring that their sum equals to 1. As to the best model configurations, we set the embedding dimension d to 200, L is 30 for candidate unknown setting and 50 for candidate known setting, α is 0.2 for candidate unknown setting and 0.5 for candidate known setting, C is 20 for the candidate unknown setting and 10 for the candidate known setting, n is 2, the layer of structural encoder and multi-head attention GCN are both 2. μ_1, μ_2 and μ_3 are 0.3, 0.5 and 0.2, respectively. Adam is used for parameter learning, and the learning rate is set to 0.001. All experiments are conducted on NVIDIA Tesla A100 (40G) and Intel Xeon 6248R.

D Sensitivity Analysis

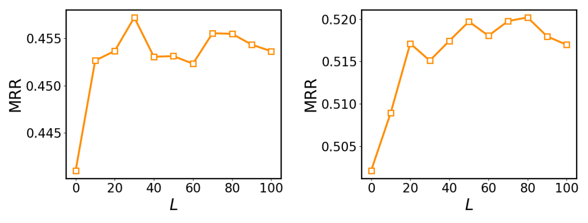
After determining the optimal hyperparameters for each setting by means of the random search algorithm, we fix the other hyperparameters to analyze the following specific hyperparameters.



(a) candidate unknown setting (b) candidate known setting

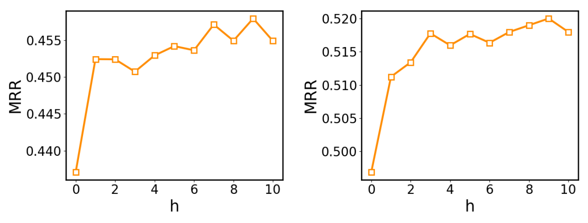
Figure 4: Performance of MGESL under different α-values on ICEWS14 in MRR.

The value of α determines the weight of the Hypergraph Convolution in the SLM module. Figure 4 demonstrates the performance of MGESL for different α-values under different settings. When α increases, the performance improves, indicating



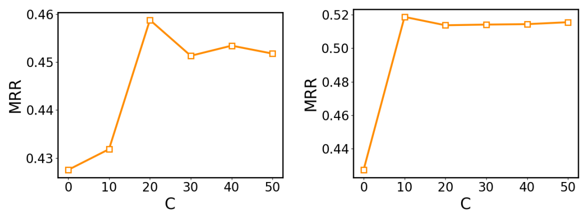
(a) candidate unknown setting (b) candidate known setting

Figure 5: Performance of MGESL under different L-values on ICEWS14 in MRR.



(a) candidate unknown setting (b) candidate known setting

Figure 6: Performance of MGESL under different h-values on ICEWS14 in MRR.



(a) candidate unknown setting (b) candidate known setting

Figure 7: Performance of MGESL under different C-values on ICEWS14 in MRR.

that learning the similarity between entities through the Hypergraph Convolution can improve the performance of our model. However, as α continues to increase, the performance declines, indicating that inherent graph structure information of the pre-trained graph is also significant.

The value of L determines the numbers of timestamps for pre-learning graph in the SLM module. As shown in Figure 5, with L-values increasing, the model performance first improves and then declines. This suggests that an optimal number of timestamps for the pre-learning graph can improve the model's performance, whereas an excessive amount may have adverse effects. This could be due to the fact that when we predict the facts in the n^{th} timestamp, information from that timestamp might have already been assimilated through pre-learning, potentially diminishing the model's generalization ability.

Model	ICEWS14	ICEWS18	ICEWS05-15
TiRGN	4.5 minutes	22 minutes	58 minutes
HGLS	5 minutes	25 minutes	1 hour 5 minutes
RETIA	21 minutes	2 hours	32 hours
MGESL(ours)	6 minutes	31 minutes	1 hour 22 minutes

Table 6: The time consumed for one training epoch on three datasets under candidate entity unknown setting.

The value of h determines the length of the historical timestamps in TEM module. According to Figure 6, an increase in length results in a gradual improvement in the model's performance under both settings. This suggests that more history timestamps are beneficial to the model. For efficiency considerations, we opted for a history timestamp length of 9 in our experiments under both settings.

The value of C determines the length of timestamps of background graph. As shown in Figure 7, the performance of the model intially improves but later declines with the increase of C-values under both settings. This phenomenon may be attributed to the fact that excessively large background graph incorporates more additional noisy data, hindering the accurate modeling of entity representations.

E Efficiency Comparison

Table 6 shows the time consumed of several state-of-the-art models for one training epoch on three datasets under candidate entity unknown setting on NVIDIA Tesla A100 (40G) and Intel Xeon 6248R.

Specifically, compared to RETIA (Liu et al., 2023a), our model consumes relatively less time. However, it takes more time than TiRGN (Li et al., 2022a) and HGLS (Zhang et al., 2023) primarily due to the additional time required for modeling the background graph. In terms of model performance, our results show significant improvements over these models.