Relation Logical Reasoning and Relation-aware Entity Encoding for Temporal Knowledge Graph Reasoning

Longzhou Liu[1], **Chenglong Xiao**[1], **Shanshan Wang** [1*] , **Tingwen Liu**[2*]
[1]Department of Computer Science, Shantou University, Shantou, China
[2]Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
{22lzliu, chlxiao, sswang}@stu.edu.cn
liutingwen@iie.ac.cn

Abstract

Temporal Knowledge Graph Reasoning (TKGR) aims to predict future facts based on historical data. Current mainstream models primarily use embedding techniques, which predict missing facts by representing entities and relations as low-dimensional vectors. However, these models often consider only the structural information of individual entities and relations, overlooking the broader structure of the entire TKG. To address these limitations, we propose a novel model called **R**elation **L**ogical Reasoning and Relation-aware **E**ntity **E**ncoding (RLEE), drawing inspiration from attention mechanisms and logical rule-based techniques. RLEE introduces a two-layer representation of the TKG: an entity layer and a relation layer. At the relation layer, we extract relation paths to mine potential logical correlations between different relations, learning relation embeddings through a process of relation logical reasoning. At the entity layer, we use the relation-aware attention mechanism to learn the entity embeddings specific to the predicted query relations. These learned relation and entity embeddings are then used to predict facts at future timestamps. When evaluated on five commonly used public datasets, RLEE consistently outperforms state-of-the-art baselines.

1 Introduction

Temporal Knowledge Graphs (TKGs) integrate temporal information into traditional knowledge graphs, representing real-world facts (events) as quadruples (subject, relation, object, timestamp). TKGs consist of static subgraphs divided by the temporal dimension, with each subgraph containing all the facts that occurred at a specific corresponding timestamp. TKGR under the extrapolation setting focuses on inferring unfinished events

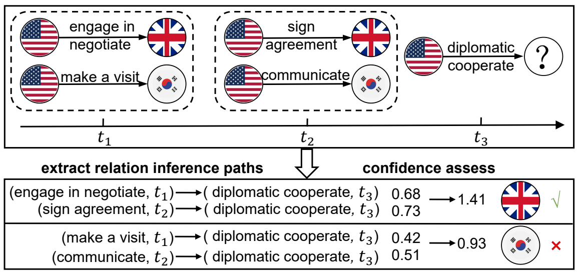

engage in negotiate
make a visit
sign agreement
communicate
diplomatic cooperate
?
t_1
t_2
t_3
extract relation inference paths
confidence assess
(engage in negotiate, t_1)⟶(diplomatic cooperate, t_3) 0.68
(sign agreement, t_2)⟶(diplomatic cooperate, t_3) 0.73
⟶1.41
√
(make a visit, t_1)⟶(diplomatic cooperate, t_3) 0.42
(communicate, t_2)⟶(diplomatic cooperate, t_3) 0.51
⟶0.93
×

Figure 1: An example of reasoning using relational logic correlation between pairs of entities.

in future subgraphs based on the information contained in historical subgraphs. Given its practical significance, TKGR under extrapolation settings has been widely used in areas such as financial early warning and behavioral prediction. To predict future events more accurately, more and more researchers are focusing on mining information from historical subgraphs in TKG to learn embedding representations of relations and entities. This is a representation learning problem in a low-dimensional space, aiming at efficient encoding of temporal information in the knowledge graph by capturing the dynamics of entity-to-entity connections. However, there are still two challenges that need to be addressed.

Neglecting the logical correlation of relations at the temporal level. Some graph-structure-based TKGR methods CyGNet (Zhu et al. (2021)), CENET (Xu et al. (2023)), HGLS (Zhang et al. (2023)), EvoExplore (Zhang et al. (2022)) do not focus on the changes of relations of the same entity pairs at different timestamps, ignoring the logical correlations that are implied between these relations. The predicted query (USA, diplomatic cooperate, ?, t_3) in Figure 1 is an example from the ICEWS dataset, and we choose Britain and South Korea as candidates. First, we extract the relation inference paths between the two entity pairs (USA, Britain) and (USA, South Korea). We then eval-

*Corresponding authors.

uate the confidence that a relation inference path holds by mining the logical correlations between historical and query relations. Finally, we summarize the confidence scores of the relation inference paths and use the highest-scoring candidate entity, Britain, as the inference result.

Neglecting the dependence of entities on relations. Existing methods aggregate subgraphs of entities to obtain a common entity representation, which remains unchanged when predicting different future events. However, an entity plays different roles under different relations, and its specific entity representation should vary with the relations. For example, when answering queries (Obama, criticize, ?) and (Obama, endorse,?), the entity "Obama" should learn two embeddings with different semantic information, focusing on information related to the relation "criticize" and the relation "endorse", respectively. When learning entity embeddings, we should pay attention to the association between entities and relations and learn entity embeddings specific to the current query relation when facing different queries.

To address the aforementioned challenges, we propose a model called RLEE, which has two main modules that model relation information and entity information. To address Challenge 1, we introduce the Relation Logical Reasoning module, which mines the potential logical correlations between different relations by extracting the relation inference paths between pairs of entities and in this way learns the relation embeddings, enabling different relations to determine the degree of logical correlations between the relations by their distance in the embedding space. To address Challenge 2, we introduce the Relation-aware Entity Encoding module, which aims to learn the encoding of entities located under different relations. Specifically, we use a relation-aware attention mechanism to assign different weights to the subgraph neighbors of an entity, and finally aggregate the subgraphs based on the weights to obtain the embedding of an entity that is specific to a particular relation.

In summary, the contribution of our work is as follows:

1) We mine the underlying logical correlations between relations by exploring the relation inference paths between pairs of entities.

2) We learn relation-specific entity embeddings, which can effectively avoid the interference of irrelevant information in subgraphs when reasoning about predictions.

3) Extensive experiments indicate that our model substantially outperforms existing methods.

2 Related Work

Depending on the type of historical information a model focuses on, existing models can be divided into two categories: models based on historical entity information and models based on historical relation information.

Models based on historical entity information focus on modeling information about the entity. For instance, CyGNet (Zhu et al. (2021)) counts the frequency of entities occurring repeatedly in history and uses a copy mechanism to select prediction results from the entities that appear frequently. CENET (Xu et al. (2023)) adopts a comparative learning approach to capture the dependency of queries on both historical and non-historical entities. EvoExplore (Zhang et al. (2022)) implements a hierarchical attention mechanism to model the intricate local and global structures of entities.

Models based on historical relation information are completely independent of entities and focus on modeling the temporal path of relations. For instance, CluSTeR (Li et al. (2021a)) utilizes reinforcement learning to develop cluster search strategies that identify explicit and reliable relation clues for predicting future facts. DaeMon (Dong et al. (2023)) introduces a novel architecture that leverages timeline relations to adaptively capture temporal path information between query topics and candidate objects. ALRE-IR (Mei et al. (2022)) extracts relation paths from historical subgraphs, aligns these paths with current events to formulate rules, and then uses these rules to predict missing entities.

3 Method

3.1 Preliminaries

Let ε,R,T denote the finite set of entities, relations, and timestamps, respectively. In the temporal knowledge graph, each fact is represented by a quaternion (s,r,o,t), where $s \in \varepsilon$ is the subject entity, $o \in \varepsilon$ is the object entity, and $r \in R$ is the relation between s and o that occurs at timestamp $t \in T$. Specifically, given a query $q = (s, r_q, ?, t_q)$, we take the candidate object $o_i \in \varepsilon_c$ as an example, where the subscript c of ε_c is the initial letter of the

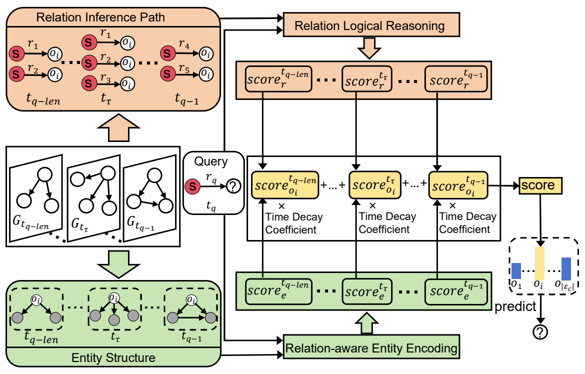


Figure 2: The overall architecture of RLEE. The orange portion is to get the relation level contribution score and the green portion is to get the entity level contribution score.

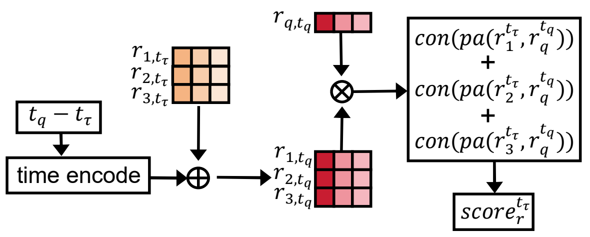
candidate, and ε_c is denoted as the set of all entities connected in the history of the query subject s,which we take as the set of candidate entities.

3.2 Model Overview

For prediction queries, we perform a two-step process. First, from a relational perspective: The subject s and the candidate object o_i have relations r_1 and r_2 under timestamps t and $t+\triangle t$, which form an inference path $pa\left(r_1^t, r_2^{t+\triangle t}\right) = (r_1, t) \rightarrow (r_2, t+\triangle t)$ suggesting that any pair of entities that have a relation r_1 under timestamp t, that pair will have relation r_2 after the time interval $\triangle t$. We use the training data to obtain confidence that different relation inference paths hold, and the relation embeddings learned in this way effectively capture logical correlations with other relations. In predicting query $q = (s, r_q, o_i, t_q)$, we aggregate the confidence scores of all relation inference paths between subject s and candidate o_i at historical timestamp t_τ and use this score as the contribution score of the historical subgraph at timestamp t_τ to support the construction of query $q = (s, r_q, o_i, t_q)$ at the relational level.

Next, from the perspective of entities: Based on the learned relation embeddings, we use the relation-aware attention mechanism to obtain a representation of entity embeddings specific to the query relation. We then use the DistMult function to capture semantic associations between entities and relations. This approach can obtain a score for the contribution of the history subgraph of the timestamp t_τ to the establishment of the query $q = (s, r_q, o_i, t_q)$ at the level of the entity structure.

We integrate the contribution scores from both



$$t_q - t_\tau$$

$$con(pa(r_1^{t_\tau}, r_q^{t_q})) + con(pa(r_2^{t_\tau}, r_q^{t_q})) + con(pa(r_3^{t_\tau}, r_q^{t_q}))$$

$$score_r^{t_\tau}$$

Figure 3: A framework for relation logic reasoning that obtains contribution scores at the relational level under timestamp t_τ.

the relational level and the entity structural level using the product method to determine the subgraph contribution score for a specific timestamp. Subsequently, we calculate the probability score that query $q = (s, r_q, o_i, t_q)$ holds, by performing a weighted aggregation of the subgraph contribution scores throughout the historical time frame from t_{q-len} to t_{q-1} using a time decay function. The overall methodology of our proposed model is shown in Figure 2.

3.3 Relation Logical Reasoning

The workflow of this module is shown in Figure 3. To discern the logical correlations across time between the query subject s and the candidate object o_i entity pairs, we begin by encoding the temporal information. Temporal information captures both the periodic and non-periodic nature of event occurrences—for instance, presidential elections occur periodically. In contrast, the lifespan of an individual, from birth to death, occurs non-periodically. Recognizing the significance of these temporal patterns, we specifically design separate vectors to represent periodic and non-periodic time characteristics:

$$\mathbf{v}_{\triangle t}^{p} = cos(\beta_t \triangle t + \phi_c) \tag{1}$$

$$\mathbf{v}_{\triangle t}^{np} = tanh(\gamma_t \triangle t + \phi_t) \tag{2}$$

$$\mathbf{T}_{\triangle t} = \mathbf{v}_{\triangle t}^{p} + \mathbf{v}_{\triangle t}^{np} \tag{3}$$

$\mathbf{v}_t^p$ and $\mathbf{v}_t^{np}$ are d-dimensional periodic and non-periodic vectors, respectively. β_t, γ_t, ϕ_c and ϕ_t are learnable parameters, $\triangle t = t_q - t_\tau$. After encoding the temporal information, we add the temporal encoding to the relational encoding $\mathbf{r}_{j,t_\tau}$:

$$\mathbf{r}_{j,t_q} = \mathbf{r}_{j,t_\tau} + \mathbf{T}_{\Delta t} \tag{4}$$

Next, we obtain the relation inference path $pa\left(r_j^{t_\tau}, r_q^{t_q}\right) = (r_j, t_\tau) \rightarrow (r_q, t_q)$ from the relation r_j between the entity pairs s and o_i to the relation

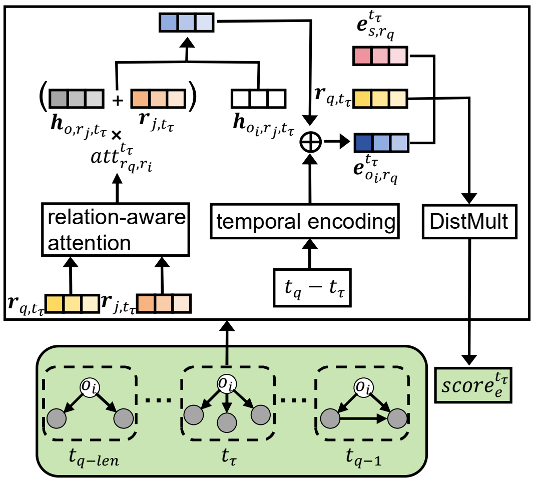


Figure 4: A framework for relation-aware entity encoding, which obtains contribution scores at the entity level under timestamp t_τ.

r_q. We consider r_j as the cause and r_q as the effect. Finally, we assess the confidence that the relation inference path $pa(r_j^{t_\tau}, r_q^{t_q})$ holds by capturing the logical correlation between r_j and r_q. To compute this, we directly use the dot product method:

$$con(pa(r_j^{t_\tau}, r_q^{t_q})) = \mathbf{r}_{j,t_q} * \mathbf{r}_{q,t_q} \tag{5}$$

We aggregate the confidence scores of all relation inference paths for subject s and candidate o_i at timestamp t_τ to obtain the contribution score of the history subgraph at timestamp t_τ to get the contribution scores at the relational level.

$$socre_r^{t_\tau} = \sum_{j=1}^{|R^{t_\tau}_{s\to o_i}|} con(pa(r_j^{t_\tau}, r_q^{t_q})) \tag{6}$$

Where $r_j^{t_\tau} \in R^{t_\tau}_{s\to o_i}$, $R^{t_\tau}_{s\to o_i}$ denote the set of all relations connected by s and o_i at timestamp t_τ.

3.4 Relation-aware Entity Encoding

The workflow of this module is shown in Figure 4. Current methodologies often overlook the impact of relations on entities whose attributes should not be static but vary according to the relation. After learning the relation embeddings, we delve deeper into learning entity embeddings tailored to specific query relations. Initially, we assign varying weights to the neighbors of the subgraphs, based on the degree of logical correlations between the relation r_q and the adjacency r_j of the entity o_i at timestamp t_τ. Subsequently, we aggregate the subgraphs, using the weighted neighbors to derive the entity embeddings. This embedding framework is inspired by RGCN, and we employ a ω-layer RGCN for encoding, which is defined as follows:

$$\mathbf{h}^l_{o_i,r_q,t_\tau} = f\Big(\frac{1}{|N_{o_i,t}|}\sum_{(r_j,o)\in N_{o_i,t_\tau}} att^{t_\tau}_{r_q,r_j} W_1^l(\mathbf{h}^l_{o,r_j,t_\tau} + \mathbf{r}_{j,t_\tau}) + W_2^l \mathbf{h}^{l-1}_{o_i,r_q,t_\tau}\Big) \tag{7}$$

Where $f(\cdot)$ is the RReLU activation function, $N_{o_i,t}$ is the set of neighbors of entity o_i in the static subgraph that is at timestamp t, att^t_{r,r_i} is the relation-aware attentional weight, $W_1^l \in \mathbb{R}^{d\times d}$ and $W_2^l \in \mathbb{R}^{d\times d}$ are the weight parameters.

In Equation 7 we use the normalized attention mechanism att^t_{r,r_j} to assign the attentional weights, att^t_{r,r_j} is computed as follows:

$$att^{t_\tau}_{r_q,r_j} = \frac{exp(cos(\mathbf{r}_{j,t_\tau}, \mathbf{r}_{q,t_\tau}))}{\sum_{r_k\in R_{o_i,t_\tau}} exp(cos(\mathbf{r}_{k,t_\tau}, \mathbf{r}_{q,t_\tau}))} \tag{8}$$

Where R_{o_i,t_τ} is the set of all relations to which entity o_i is connected.

Similarly, we add time coding to the entity embedding to obtain dynamic entity coding:

$$\mathbf{e}^{t_\tau}_{o_i,r_q} = \mathbf{h}^{\omega}_{o_i,r_q,t_\tau} + \mathbf{T}_{\triangle t} \tag{9}$$

Since the DistMult function uses simple mathematical operations to represent the semantic associations between entities and relations with high computational efficiency and good interpretability, here we use this function as the scoring function for the entity structure part as follows:

$$socre_e^{t_\tau} = \sigma(< \mathbf{e}^{t_\tau}_{s,r_q}, \mathbf{r}_{q,t_\tau}, \mathbf{e}^{t_\tau}_{o_i,r_q} >) \tag{10}$$

Where $\sigma(\cdot)$ is a sigmoid function and $< \cdot >$ denotes the trilinear dot product. Eventually we obtain the score of the contribution of the history subgraph of the timestamp t_τ to the establishment of query $q = (s, r_q, o_i, t_q)$ at the level of entity structure.

3.5 Result Prediction

In the ablation experiments in Section 4.3 below, we found a strong dependence between the relation inference path scores and the entity structure scores. Here, we use multiplication to combine relation and entity-level scores to obtain the predicted score at time t_τ.

$$score^{t_\tau}_{o_i} = socre^{t_\tau}_r \cdot socre^{t_\tau}_e \tag{11}$$

After obtaining the candidate entity o_i scores at each timestamp in the time range of $[t_{q-len}, t_{q-1}]$

through Equation 11, we aggregate these scores. Considering that the impact of historical events varies with the proximity of their occurrence, we design a power function based time decay coefficient:

$$W_d(t_q, t_\tau) = (t_q - t_\tau)^{-\gamma} \tag{12}$$

The larger the value of γ in the above equation, the faster the rate at which W_d decays over time. The time decay coefficient W_d ensures that relation inference paths closer in time to the query time t_q are assigned higher weights. We then weighted the predicted scores at each timestamp together to get the final score:

$$score(o_i|s, r_q, t_q) = \sum_{\tau=q-len}^{q-1} W_d(t_q, t_\tau) score_{o_i}^{t_\tau} \tag{13}$$

Finally, we take the candidate entity with the highest score as the final prediction:

$$\hat{o} = argmax_{o \in \varepsilon_c} score(o|s, r_q, t_q) \tag{14}$$

Where $score(o|s, r_q, t_q)$ denotes the predicted probability of all candidate object entities $o \in \varepsilon$.

3.6 Train

We use positive and negative sample comparison learning for training. First, we negatively sample and generate the error quaternion. Specifically, given a correct quaternion $pos = (s, r, o, t)$, we randomly sample an object entity from historical events and disrupt the quaternion to generate an incorrect quaternion neg that satisfies the condition $neg = \{(s, r, o', t)|o' \in \varepsilon - o\}$. We ensure that the correct quaternions (positive samples) receive higher scores and the incorrect quaternions (negative samples) receive lower scores by using the $SoftMarginLoss$ function, expressed as follows:

$$L = \sum_{(s,r,o,t) \in P \bigcup N} log(1 + exp(-y \cdot score(s, r, o, t))) \tag{15}$$

$$y = \begin{cases} 1, & (s, r, o, t) \in P \\ -1, & (s, r, o, t) \in N \end{cases} \tag{16}$$

where P is the set of correct quaternions and N is the set of error quaternions.

From the level of the embedding space of relations, the loss function's task is to bring the historical relation embedding of the relation inference path in the positive examples close to the query relation embedding, and at the same time to move the historical relation embedding of the relational inference path in the negative examples away from the query relation embedding. The learned relation embeddings by this method can reflect the logical correlations between relations at the level of the embedding space. From the entity structure level, the task of the loss function is to learn an entity embedding specific to the query relation that can focus more on features related to the query relation to answer a specific query more efficiently, avoiding the interference of irrelevant information in the subgraphs in predicting the query.

4 Experiment

4.1 Experimental Setup

4.1.1 Datasets

We use five benchmark datasets (ICEWS14 (Li et al. (2022b)), ICEWS0515 (Ren et al. (2023)), ICEWS18 (Boschee et al. (2015)), WIKI (Vrandečić and Krötzsch (2014)), and YAGO (Suchanek et al. (2007))) to evaluate the performance of the model on the link prediction task. Table 1 below provides statistics for these datasets. All datasets are categorized chronologically into training, validation, and test sets.

4.1.2 Baselines

Our RLEE model is compared with TKGC models under the extrapolation setting. We chose DistMult (Yang et al. (2014)), ComplEX (Trouillon et al. (2016)) and R-GCN (Schlichtkrull et al. (2018)) as static models for comparison. TTransE (Leblay and Chekol (2018)), HyTE (Dasgupta et al. (2018)) and TA-DistMult (García-Durán et al. (2018)) as interpolated TKGR models for comparison. CyGNet (Zhu et al. (2021)), xERTE (Han et al. (2020)), TiTer (Sun et al. (2021)), RE-GCN (Li et al. (2021b)), CluSTeR (Li et al. (2021a)), HiSMatch (Li et al. (2022b)), CEN (Li et al. (2022a)), EvoExplore (Zhang et al. (2022)), TECHS (Lin et al. (2023)), DaeMon (Dong et al. (2023)), CENET (Xu et al. (2023)), RPC (Liang et al. (2023)), TiPNN (Dong et al. (2024)), and DLGR (Xiao et al. (2024)) as extrapolated TKGR models for comparison.

4.1.3 Evaluation Metrics

We employ widely used evaluation metrics, namely mean reversed rank (MRR), hits@1, hits@3, and hits@10. For a fair comparison, we perform time-aware filtering where all correct entities at the query timestamp except for the true query object are filtered out from the answers. In comparison to the

Datasets	Entities	Relations	Training	Validation	Test	Time Granules
ICEWS14	6869	230	74845	8514	7371	365
ICEWS0515	10488	251	368868	46302	46159	4017
ICEWS18	23033	256	373018	45995	49545	304
WIKI	12554	24	539286	67538	63110	232
YAGO	10623	10	161540	19523	20026	189

Table 1: Statistical data for the datasets.

Model	ICEWS14				ICEWS18				ICEWS0515			
	MRR	H@1	H@3	H@10	MRR	H@1	H@3	H@10	MRR	H@1	H@3	H@10
DistMult(2015)	20.32	6.13	27.59	46.61	13.86	5.61	15.22	31.26	19.91	5.63	27.22	47.33
ComplEX(2016)	22.61	9.88	28.93	47.57	15.45	8.04	17.19	30.73	20.26	6.66	26.43	47.31
R-GCN(2018)	28.03	19.42	31.95	44.83	15.05	8.13	16.49	29.00	27.13	18.83	30.41	43.16
TTransE(2016)	12.86	3.14	15.72	33.65	8.44	1.85	8.95	22.38	16.53	5.51	20.77	39.26
HyTE(2018)	16.78	2.13	24.84	43.94	7.41	3.10	7.33	16.01	16.05	6.53	20.20	34.72
TA-DistMult(2018)	26.22	16.83	29.72	45.23	16.42	8.60	18.13	32.51	27.51	17.57	31.46	47.32
CyGNet(2021)	32.73	23.69	36.31	50.67	24.93	15.90	28.28	42.61	34.97	25.67	39.09	52.94
xERTE(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
TiTer(2021)	41.23	32.54	46.10	58.44	29.98	22.05	33.46	44.83	46.35	37.06	52.42	66.13
RE-GCN*(2021)	41.99	32.93	46.60	62.47	30.55	20.00	34.73	51.46	46.41	37.17	52.76	67.64
CluSTeR(2021)	46.00	33.80	-	71.20	32.30	20.60	-	55.90	44.60	34.90	-	63.00
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
CEN*(2022)	41.52	31.38	46.02	61.36	30.85	20.53	34.28	49.86	49.21	37.52	56.74	71.68
EvoExplore(2022)	43.60	32.10	50.60	64.70	31.50	21.70	35.90	51.00	50.00	39.40	56.10	69.60
TECHS(2023)	43.88	34.59	49.36	61.95	30.85	21.81	35.39	49.82	48.38	38.34	54.69	68.92
DaeMon(2023)	45.32	34.96	50.07	63.72	31.23	22.51	35.65	48.76	47.85	37.90	52.61	68.57
CENET(2023)	41.30	32.58	-	58.22	29.65	19.98	-	48.23	47.13	37.25	-	67.61
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
TiPNN(2024)	-	-	-	-	32.17	22.74	36.24	50.72	-	-	-	-
DLGR(2024)	46.72	36.67	51.61	-	35.48	25.11	40.03	-	-	-	-	-
RLEE	**52.63**	**39.53**	**58.70**	**78.35**	**36.71**	**25.73**	**41.35**	**58.42**	**56.84**	**44.37**	**63.08**	**80.23**
Absolute Boost	5.91	2.86	7.07	7.15	1.23	0.62	1.32	2.52	3.99	2.36	4.03	6.95
Relative Boost	12.65	7.80	13.69	10.04	3.46	2.47	3.30	4.51	7.55	5.62	6.82	9.48

Table 2: Performance (in percentage) on ICEWS14, ICEWS18, ICEWS0515. The best performance is highlighted in boldface, and the second-best is underlined. * indicates that we remove the static information from the model to ensure the fairness of comparisons between all baselines.

alternative setting that filters out all other objects that appear together with the query subject and relation at any timestamp, time-aware filtering yields a more realistic performance estimate. Our experiments report average results over four runs.

4.1.4 Implementation Details

Referring to previous research, we use random initialization to generate entity and relation embeddings of dimension 200. To optimize all model parameters, we used the Adam optimizer and set the initialized learning rate to 0.001. The number of layers w of the RGCN is set to 2; for each layer of the RGCN, the dropout rate is set to 0.2 and the history length parameter len is set to 10. The value of the parameter γ of the time decay coefficient is set to 0.8. Specifically, we train the model for 100 epochs, and stop the training if the verification loss does not decrease for 10 consecutive epochs. All experiments were conducted on a single Tesla T4 GPU with 16GB of memory. For the static reasoning methods, the time dimension is removed from all the TKG datasets. Some of the baseline results are adopted from RE-GCN. For the important CENET, DaeMon, EvoExplore, HiSMatch, RE-GCN, TiTer, xERTE, and CyGNet baseline works, we use their default parameters and replicate the results obtained under the original setup using their open codes. For CEN, we report the results obtained in the online setting. For DLGR, TiPNN, RPC, TECHS, and CluSTeR baseline works, we report the results presented in their papers since the model is not open source.

4.2 Experimental Result

Table 2 and Table 3 presents the performance comparison of all baseline models. On the ICEWS14, ICEWS18, and ICEWS0515 datasets, our proposed RLEE model outperforms other baselines on all assessment metrics, which validates the effectiveness of our model. Specifically, RLEE significantly outperforms static models, which demonstrates the importance of modeling temporal information in TKGR. However, some dynamic approaches, such as TransE and HyTE, perform even worse than static approaches because adding temporal representation to the scoring function destroys the transformation between entities. This illustrates the importance of modeling temporal information in a sensible way. RLEE still performs higher compared to the embedded models of CyGNet, xERTE,

Model	WIKI				YAGO			
	MRR	H@1	H@3	H@10	MRR	H@1	H@3	H@10
DistMult(2015)	27.96	-	32.45	39.51	44.05	-	49.70	59.94
ComplEX(2016)	27.69	-	31.99	38.61	44.09	-	49.57	59.64
R-GCN(2018)	13.96	-	15.75	22.05	20.25	-	24.01	37.30
TTransE(2016)	20.66	-	23.88	33.04	26.10	-	36.28	47.73
HyTE(2018)	25.40	-	29.16	37.54	14.42	-	39.73	46.98
TA-DistMult(2018)	26.44	-	31.36	38.51	44.98	-	50.64	61.11
CyGNet(2021)	58.78	47.89	66.44	78.70	68.98	58.97	76.80	86.98
xERTE(2021)	73.60	69.05	78.03	79.73	84.19	80.09	88.02	89.78
TiTer(2021)	73.91	71.70	75.41	76.96	87.47	80.09	89.96	90.27
RE-GCN*(2021)	51.53	-	58.29	69.53	63.07	-	71.17	82.07
HiSMatch(2022)	78.07	73.89	81.32	84.65	87.21	84.10	90.64	91.83
CEN*(2022)	78.93	75.05	81.90	84.90	82.37	79.52	85.93	88.64
TECHS(2023)	75.98	-	-	82.39	89.24	-	-	92.39
DaeMon(2023)	82.38	78.26	86.03	88.01	91.59	90.03	93.00	93.34
RPC(2023)	81.18	76.28	85.43	88.71	88.87	85.10	92.57	94.04
TiPNN(2024)	83.04	79.04	86.45	88.54	92.06	90.79	93.15	93.58
DLGR(2024)	82.14	80.14	84.04	-	88.87	84.60	92.35	-
RLEE	**85.53**	**81.65**	**88.22**	**89.95**	**92.43**	**91.02**	**94.17**	**95.21**
Absolute Boost	2.49	1.51	1.77	1.24	0.37	0.23	1.02	1.17

Table 3: Performance (in percentage) on WIKI, YAGO. The best performance is highlighted in boldface, and the second-best is underlined. * indicates that we remove the static information from the model to ensure the fairness of comparisons between all baselines.

RE-GCN, HiSMatch, CEN, EvoExplore, RPC, and DLGR, because these methods ignore the association between entities and relations. TiTer, CluSTeR, TECHS, Daemon, and TiPNN are logic rule-based models that extract potential logic rules from graphs by path searching. However, these methods are constrained by existing paths, limiting the scope of their searches and impairing their performance.

Further, we find that most models achieve good predictions on the ICEWS0515 and ICEWS14 datasets, but perform much worse on the ICEWS18 dataset. Upon observing Table 1, we note that the ICEWS18 dataset contains a large number of entities that introduce many relation inference paths with low confidence, making it difficult to learn valid relational logical associations.

On the WIKI and YAGO datasets, most of the models achieve high prediction performance, mainly because these two datasets contain a small number of relations, and the structure of the knowledge graphs they constitute is simple and easy to analyze. In particular, the YAGO dataset has only 10 relations, and we found through careful data analysis that the relations “isMarriedTo”, “owns” and “isAffiliatedTo” in the YAGO dataset occur more than 80% of the time. This results in a very simple knowledge graph constructed from the YAGO dataset, which does not need to take into account the complex connections between entities in the reasoning process.

4.3 Ablation Study

To further analyze the contribution that each part of the model makes to the final prediction results, we report in Table 4 above the results of the MRR metrics for the six sub-models on the test sets of the three datasets.

	ICEWS14	ICEWS18	YAGO
RLEE	52.63	36.71	92.43
RLEE w/o R	49.16	33.26	87.32
RLEE w/o E	50.81	34.05	85.65
RLEE-Add	48.81	33.17	86.02
RLEE w/o relation-attention	50.13	34.95	83.72
RLEE w/o (temporal encoding)	47.22	34.16	82.37

Table 4: Results (in percentage) by different variants of our model on three datasets.

The several sub-models in the table are:1. RLEE, the complete model. 2. RLEE w/o R represents the model that does not use the Relational Logic Reasoning module. 3. RLEE w/o E represents the model that does not use the Relation-aware Entity Encoding module. 4. RLEE-Add represents the use of addition to combine the relation inference path scores and the entity structure scores. 5. RLEE w/o relation-attention represents models that do not use the Relation-aware Attention mechanism during entity embedding learning. 6. RLEE w/o (temporal encoding) represents models that do not use temporal encoding in the model.

From the experimental data presented in the table above, it is clear that both the Relation Logical Reasoning module and the Relation-aware Entity Encoding module are critical. Further to explore the extent to which the relation inference path score and entity structure score contribute to the final prediction results, we combine the scores at both the relation and entity levels by addition to obtain the RLEE-Add model. Compared to the RLEE model, which integrates scores through multiplication, the

Datasets	$\lvert\mathcal{E}\rvert$	$\lvert R\rvert$	Training	Validation	Test
YAGO	10623	10	161540	19523	20026
YAGOs	10038	10	51205	10973	10973

Table 5: Statistical data for YAGO and YAGOs.

	MRR	H@1	H@3	H@10
YAGOs	51.71	46.93	57.02	59.36
YAGO→YAGOs	49.35	44.02	53.26	56.94
LIMP	95.44	93.80	93.41	95.92
YAGO	92.43	91.02	94.17	95.21
YAGOs→YAGO	88.03	85.78	90.23	92.42
LIMP	95.25	94.24	95.82	97.07

Table 6: Logical Inference migration Performance(→ denotes the cross-dataset inference result).

RLEE-Add model's performance is substantially lower. This significant disparity suggests a strong interdependence between the relational inference path score and the entity structure score, indicating that high scores in both categories are imperative for achieving accurate inference outcomes.

The performance of RLEE w/o relation-attention drops significantly compared to RLEE, suggesting that learning relation-specific entity embeddings can be more beneficial in answering the query at hand. RLEE w/o temporal encoding also does not perform as well as RLEE, demonstrating that temporal numerical information is essential in learning embedded representations of entities and relations.

4.4 Validation of the Effectiveness of the Relation Logical Reasoning

The relation logical reasoning module learns the temporal logical correlations of relations that exist only between relations and are entity-independent, which means that the logical correlations can be migrated to other datasets with the same set of relations. In other words, the RLEE model is trained on one dataset to learn the logical correlations between relations, which can then be applied to different datasets with the same set of relations for inference. To demonstrate the effectiveness of the relation logical reasoning module, we conducted an experimental analysis.

We first select a target dataset "A" and another homologous dataset "B", which means that "A" and "B" have the same set of relation types. Secondly, we train the relation logical reasoning module using the training data of "A" and test the performance with the testing data of "A", and we can obtain the direct result of the relation logical reasoning module on the target dataset "A". Then, we use the training data of "B" to train the relation logical reasoning module and use the testing data of "A" to test the performance, and we can get the cross-dataset inference result of the relation logical reasoning module on the target dataset "A" using the logical correlations learned from dataset "B". Finally, we evaluate the ability of the relation logical reasoning module to capture logical correlations between relations by looking at the logical inference migration performance(*LIMP*), which is calculated by the percentage ratio of the cross-dataset inference result divided by the direct result.

More specifically, YAGO and YAGOs are homologous datasets (compared as shown in Table 5), and there is no intersection between their entity identifiers. Therefore, we use YAGO and YAGOs as target datasets in turn. Table 6 shows the results of the logical inference migration performance evaluation of the relation logical reasoning module on YAGO and YAGOs datasets. We can observe that all the logical inference migration performance of the relation logical reasoning module is above 90%. Even when learning relation logical correlations from the smaller dataset YAGOs and testing on the larger dataset YAGO, the relation logical reasoning module achieves effective performance on each of the TKG reasoning evaluation metrics. Thus, the experiments demonstrate that the relation logical reasoning module can effectively capture the logical correlations of different relations at the temporal level and that the learned logical correlations can be effectively applied to different datasets.

5 Conclusion

How to learn effective relation embeddings and entity embeddings is a problem that current models have been studying. In terms of relational embedding learning, this paper extracts relation inference paths between entity pairs and learns relational embeddings by evaluating whether these relation inference paths hold in the reasoning process so that the learned relation embeddings can reflect the logical correlations of different relations on the temporal level in the embedding space. In terms of entity embedding learning, we use the relation-aware attention mechanism to learn relation-specific entity embeddings, which enables the learned entity embeddings to pay more attention to the structural information related to the query relation and avoids the interference of irrelevant information. Experiments on five benchmark datasets demonstrate the effectiveness of our model in temporal knowledge graph extrapolation tasks.

Acknowledgements

This work was supported by the National Key Research and Development Program of China (Grant No.2021YFB3100600), Scientific Research Project of Colleges and Universities in Guangdong Province (2021ZDZX1027), Guangdong Basic and Applied Basic Research Foundation (2022A1515110712 and 2023A1515010077), and STU Scientific Research Foundation for Talents (NTF20016 and NTF20017).

References

Elizabeth Boschee, Jennifer Lautenschlager, Sean O'Brien, Steve Shellman, James Starz, and Michael Ward. 2015. Icews coded event data. *Harvard Dataverse*, 12:2. Doi:10.7910/DVN/28075.

Shib Sankar Dasgupta, Swayambhu Nath Ray, and Partha Talukdar. 2018. Hyte: Hyperplane-based temporally aware knowledge graph embedding. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, pages 2001–2011.

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 Thirty-Second International Joint Conference on Artificial Intelligence*, pages 2086–2094.

Hao Dong, Pengyang Wang, Meng Xiao, Zhiyuan Ning, Pengfei Wang, and Yuanchun Zhou. 2024. Temporal inductive path neural network for temporal knowledge graph reasoning. *Artificial Intelligence*, 329:104085.

Alberto García-Durán, Sebastijan Dumančić, and Mathias Niepert. 2018. Learning sequence encoders for temporal knowledge graph completion. *arXiv preprint arXiv:1809.03202*.

Zhen Han, Peng Chen, Yunpu Ma, and Volker Tresp. 2020. Explainable subgraph reasoning for forecasting on temporal knowledge graphs. 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.

Zixuan Li, Saiping Guan, Xiaolong Jin, Weihua Peng, Yajuan Lyu, Yong Zhu, Long Bai, Wei Li, Jiafeng Guo, and Xueqi Cheng. 2022a. 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. 2022b. Hismatch: Historical structure matching based temporal knowledge graph reasoning. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 7328–7338. Doi:10.48550/arXiv.2210.09708.

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. *arXiv preprint arXiv:2106.00327*.

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.

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.

Qika Lin, Jun Liu, Rui Mao, Fangzhi Xu, and Erik Cambria. 2023. Techs: Temporal logical graph networks for explainable extrapolation reasoning. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1281–1293.

Xin Mei, Libin Yang, Xiaoyan Cai, and Zuowei Jiang. 2022. An adaptive logical rule embedding model for inductive reasoning over temporal knowledge graphs. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 7304–7316.

Xin Ren, Luyi Bai, Qianwen Xiao, and Xiangxi Meng. 2023. Hierarchical self-attention embedding for temporal knowledge graph completion. In *Proceedings of the ACM Web Conference 2023*, pages 2539–2547. Doi:10.1145/3543507.358339.

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.

Fabian M Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. Yago: a core of semantic knowledge. In *Proceedings of the 16th international conference on World Wide Web*, pages 697–706.

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

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.

Denny Vrandečić and Markus Krötzsch. 2014. Wikidata: a free collaborative knowledgebase. *Communications of the ACM*, 57(10):78–85.

Yao Xiao, Guangyou Zhou, Zhiwen Xie, Jin Liu, and Jimmy Xiangji Huang. 2024. Learning dual disentangled representation with self-supervision for temporal knowledge graph reasoning. *Information Processing & Management*, 61(3):103618.

Yi Xu, Junjie Ou, Hui Xu, and Luoyi Fu. 2023. Temporal knowledge graph reasoning with historical contrastive learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 4765–4773. Doi:10.1609/aaai.v37i4.25601.

Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2014. Embedding entities and relations for learning and inference in knowledge bases. *arXiv preprint arXiv:1412.6575*.

Jiasheng Zhang, Shuang Liang, Yongpan Sheng, and Jie Shao. 2022. Temporal knowledge graph representation learning with local and global evolutions. *Knowledge-Based Systems*, 251:109234. Doi:10.1016/j.knosys.2022.109234.

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. Doi:10.1145/3543507.3583242.

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. Doi:10.1609/aaai.v35i5.16604.