Applied Intelligence
https://doi.org/10.1007/s10489-024-05767-6

# Temporal knowledge graph reasoning based on evolutional representation and contrastive learning

Qiuying Ma[1] · Xuan Zhang[1,2] · ZiShuo Ding[7] · Chen Gao[4] · Weiyi Shang[3] · Qiong Nong[1] · Yubin Ma[1] · Zhi Jin[5,6]

Accepted: 11 August 2024


**Abstract**
Temporal knowledge graphs (TKGs) are a form of knowledge representation constructed based on the evolution of events at different time points. It provides an additional perspective by extending the temporal dimension for a range of downstream tasks. Given the evolving nature of events, it is essential for TKGs to reason about non-existent or future events. Most of the existing models divide the graph into multiple time snapshots and predict future events by modeling information within and between snapshots. However, since the knowledge graph inherently suffers from missing data and uneven data distribution, this time-based division leads to a drastic reduction in available data within each snapshot, which makes it difficult to learn high-quality representations of entities and relationships. In addition, the contribution of historical information changes over time, distinguishing its importance to the final results when capturing information that evolves over time. In this paper, we introduce CH-TKG (Contrastive Learning and Historical Information Learning for TKG Reasoning) to addresses issues related to data sparseness and the ambiguity of historical information weights. Firstly, we obtain embedding representations of entities and relationships with evolutionary dependencies by R-GCN and GRU. On this foundation, we introduce a novel contrastive learning method to optimize the representation of entities and relationships within individual snapshots of sparse data. Then we utilize self-attention and copy mechanisms to learn the effects of different historical data on the final inference results. We conduct extensive experiments on four datasets, and the experimental results demonstrate the effectiveness of our proposed model with sparse data.

**Keywords** Temporal knowledge graph reasoning · Contrastive learning · Graph convolutional network · Knowledge weight learning

## 1 Introduction

Knowledge graph, as a structured form of human knowledge [1], plays a significant role in supporting semantic interoperability between massive multi-source heterogeneous data. Knowledge graph is widely used in fields such as medical health [2], question-and-answer systems [3], and recommender systems [4]. The traditional knowledge graph is usually a static knowledge base with a graph structure, which stores events in the real world in the form of triples *(s,r,o)*, where *s* and *o* represent the head entity and the tail entity respectively, and *r* represents the relationship. Knowledge graphs commonly face integrity issues due to the high cost of manual fact annotation and the incompleteness of automatically generated facts. These limitations severely restrict the performance and scope of applications based on knowledge graphs. To alleviate this problem, researchers have proposed predicting missing facts by representing entities and relationships in low-dimensional vector spaces [5] and performing reasoning over the knowledge graph. Current inference research mainly focuses on static knowledge graphs without considering temporal properties. In real-world application scenarios, time is an important attribute of most facts, as the validity of triple facts is often constrained to specific time ranges. For example, the fact *(Trump, served as, President of the United States, 2017/1/20-2021/1/20)* is only correct within the stated time period. To address this, temporal knowledge graphs incorporate additional temporal information for each triplet in the form of a quadruple with timestamps, that is *(Head Entity, Relationship, Tail Entity, Time)*. Ignoring temporal information at this stage would inevitably result in incorrect inferences on the graph. As a result, there is growing research interest in knowledge graph reasoning that takes into account temporal attributes.

Extended author information available on the last page of the article

Published online: 31 August 2024

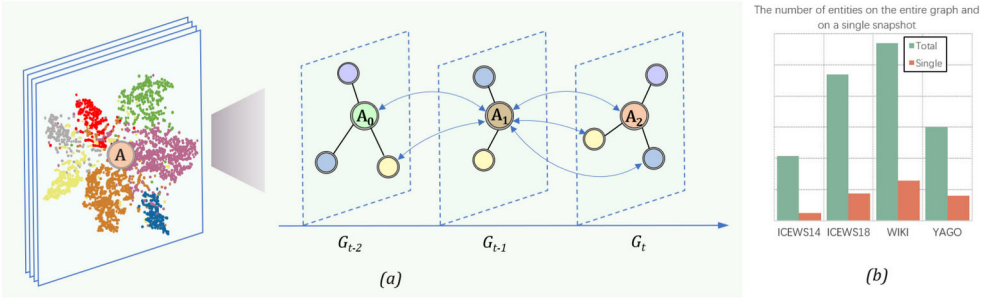



**Fig. 1** Data sparsity problem for temporal knowledge graph snapshots

Traditional static knowledge graphs commonly suffer from serious data missing issues, which are also more widely and severely present in temporal knowledge graphs. Furthermore, the evolution of time introduces complex temporal dependency properties between entities and relationships [6]. Therefore, effectively incorporating time information into models and enhancing the representation of entities and relations presents a key challenge. Doing so would allow us to appropriately capture the dynamic evolutionary characteristics of entities, relations, and graph structures and leverage these temporal features for reasoning tasks. As shown in Fig. 1(a), most existing temporal knowledge graph reasoning methods[6–8] split all graph events into $t$ sub-knowledge graph snapshots based on time. They then learn entity and relation embeddings from the information in each snapshot to capture semantic and structural features as they evolve over time. These types of methods have the following limitations: segmenting the knowledge graph by time results in a dramatic decrease in data for each graph as time slices become more granular, leading to the data sparsity issue. Figure 1(b) shows a comparison of the total number of entities (represented by green bars) and the number of entities on a single snapshot (represented by orange bars) of the four public datasets. Table 1 shows the comparison of the number of entities, the time granularity, and the number of entities on a single snapshot. It can be seen that the amount of quadruple data in each time slice is extremely small. This extreme sparsity makes it difficult for the model to learn high-quality representations of the entities and relationships.

On the other hand, from a human cognition perspective, events at the current time are influenced not only by the current state of the knowledge graph but also by historical information. The different times and frequencies of the relevant historical events play different roles in prediction. Hence, it is necessary to learn the weights of historical information on inference results.

**Table 1** Total number of entities in the 4 public temporal datasets and average number of entities on individual timestamps

| Data | ICEWS14 | ICEWS18 | WIKI | YAGO |
|---|---|---|---|---|
| Total | 90730 | 468558 | 669937 | 201092 |
| Granularity | 365 | 304 | 232 | 189 |
| Single | 248 | 1541 | 2888 | 1063 |

To address the above problems, we propose a novel temporal knowledge graph reasoning model, CH-TKG based on evolutionary representation [8] and contrastive learning. This model can learn enhanced entity and relationship representations from the sparse snapshot information, while also identifying important information from the full graph. To better model the structural and semantic information at each time slice, we first capture the structural dependencies within the knowledge graph of each slice using a relation-aware graph convolutional network (R-GCN) and obtain the embedding of the graph on each time slice through R-GCN. On the foundation of this, inspired by contrastive learning in the field of recommendation systems [9], we mine potential neighborhood relationships from node-level contrastive representations. In other words, we identify the semantically similar nodes between entities. These potential semantic relationships based on entities are optimized for entity representation under sparse data. Specifically, positive sample distances are minimized while negative sample distances are maximized through the prototype contrastive learning framework. At the same time, GRU is used to model relationships and capture the temporal dependencies between historical KG sequences. Finally, considering the importance of historical information, the most recent historical subgraph plays the biggest role in prediction. Different levels of attention are applied to different historical information based on the self-attention mechanism and the copy mechanism. This allows selective attention to important information in the representation of the current time slice to obtain better prediction results. Experimental results on four publicly available datasets demonstrate the effectiveness of the proposed model for temporal knowledge graph reasoning. The contributions of this paper are summarized as follows:

- We propose a new temporal knowledge graph reasoning model CH-TKG based on evolutionary representation

and contrastive learning. It mines semantically similar neighborhood relationships from node-level contrastive representations and optimizes entity and relationship representations obtained under sparse data;
- To better learn the weight of historical information, CH-TKG uses the self-attention mechanism and the copy mechanism to learn the impact of different events on inference results from the temporal sequence and frequency of event occurrences;
- We conducted comprehensive experiments on four public benchmark datasets. The experimental results show that CH-TKG outperforms existing temporal knowledge graph reasoning methods in the case of sparser data, proving the effectiveness of the proposed model.

## 2 Related work

In this section, we introduce related work on static knowledge graph reasoning, temporal knowledge graph reasoning, and contrastive learning.

### 2.1 Static knowledge graph reasoning

In recent years, researchers have proposed representation-based static knowledge graph reasoning methods, which can be divided into four categories: distance-based models, tensor decomposition models, neural network models, and graph neural network-based models. Distance-based models use distance-based scoring functions to calculate the distance between entities to measure the reasonableness of facts, for example, TransE [5], TransR [11], etc. Tensor decomposition models represent entities and relationships as tensor representations in low-dimensional vector space, such as, DistMult [13], and ComplEx [14]. The neural network models take the embedding of entities and relationships as input, and output the probability of fact triples through the neural network, such as MLP [16]. The GNN-based models capture the structural characteristics of KGS through graph neural networks, Such as R-GCN [18], CompGCN [19], etc. However, these methods are designed for static knowledge graphs and cannot accurately learn the representation of temporal knowledge. During reasoning, issues can arise such as semantic similarity causing event confusion and excessive interference items. For example, in the triplets *(person, born in, place)*, *(person, died in, place)*, the head entity types for both *"born in"* and *"died in"* relationships are person names, while the tail entity types are place names, which produces confusing candidate results. Too many interference items mean that traditional static reasoning methods cannot capture the impact of temporal information on the evolution of entities and relationships. Queries without time constraints can produce many candidate answers. The excessive number of interfering items means that traditional static reasoning methods are unable to capture the effect of temporal information on the evolution of entities and relationships, and the query will have many candidate answers in the absence of time constraints. For example *(?, serve, the president of the United States)* can have multiple candidates such as *"Obama"*, *"Trump"*, *"Biden"*, etc. Due to the lack of temporal information constraints, it is difficult to determine the exact Answers.

### 2.2 Temporal knowledge graph reasoning

Due to the shortcomings of traditional knowledge graph reasoning, researchers have proposed temporal knowledge graph reasoning to alleviate the above problems. There are usually two different settings for temporal knowledge graph reasoning: interpolation and extrapolation [20].

When given a sequence of knowledge graphs with timestamps from $t_0$ to $t_1$, the interpolation setup predicts missing events occurring in the timestamp range $[t_0, t_T]$, also known as TKG completion. For example, Ta-Distmult [21] and TTransE [22]. The extrapolation setting predicts future events in the timestamp range $[t_T, +\infty]$, which is predicting the future based on the past and is usually more challenging than the interpolation problem [23]. The work of this paper mainly focuses on fact prediction on future timestamps and predicts new facts on future timestamps based on historical subsequences. CyGNet [24] uses a copy-generation network to model the frequency of historical facts associated with a given query when all historical facts are considered. RE-NET [6] uses a neighborhood aggregator to encode structural information about entities and relationships in snapshots, and an RNN-based recurrent event encoder to encode the time sequence information between snapshots. RE-GCN [8] models the information of historical subgraphs through recurrent R-GCN and time gate units. CluSTeR [25] is designed based on reinforcement learning, limiting its applicability to event-based TKGs. GLANet [26] propose the Global and Local Information-Aware Network to capture both global and local information. Several other methods have been designed to model temporal point processes. RLAT [29] addresses the shortcomings of multi-hop inference methods that lack path memory by modeling the combination of reinforcement learning and attention mechanisms. TFSC [30] use the attention mechanism to model the neighbor entities of the task entity with timestamp information and generate expressive time-aware entity pair representations through the Transformer encoder, which compares the input query to the given few-shot references to make predictions. CDRGN-SDE [31] is a cross-dimensional recurrent graph network with a neural stochastic differential equation framework that resolves

the continuity problem of dynamic features in the continuous time domain. [32] preserves the continuity of dynamic multi-relational graph data by extending the idea of divine frequent differential equations (ODEs) to multi-relational graph convolutional networks and encoding temporal and structural information as continuous-time dynamic embeddings. While the aforementioned extrapolation models attempt to improve performance by capturing links or additional information between snapshots, we design models that address the problems of data sparsity on snapshots as well as assigning weights to historical information.

### 2.3 Contrastive learning

Contrastive learning [33] is a self-supervised learning paradigm that aims to capture the latent representations of a dataset by learning the similarities and differences between data samples. The structure and features between data samples are deduced without explicit labeling. Contrastive learning was first used in the field of CV [34]. Over time, it has been increasingly adopted in NLP [35], graph data mining [36], and recommendation systems [37]. Existing research methods can be categorized into node-level contrastive learning [38], graph-level contrastive learning [39], and prototype contrastive learning [40]. Inspired by these areas, some researchers have applied contrastive learning to knowledge graph inference tasks. SimKGC [41] introduces three types of negatives: in-batch negatives, pre-batch negatives, and self-negatives which act as a simple form of hard negatives to improve the contrastive learning efficiency. SimRE [42] introduces a self-supervised framework that leverages the input rule bodies to predict the corresponding rule heads through a contrastive objective. These methods are applications of contrastive learning on static graphs. CENET [43] is the first model to apply contrastive learning to TKG reasoning. It identifies highly relevant entities in historical information by training contrastive representations of queries, jointly investigating historical and non-historical information to predict potential new events. However, CENET has to consider the entire entity set information to capture historical and non-historical dependencies and does not sufficiently consider the potential semantic relationships between entities on a single snapshot. In this paper, we propose to explicitly model the potential semantic neighbor information on a single timestamp through contrastive learning to alleviate the data sparsity problem on a single temporal snapshot, which is able to obtain better results in a lesser range of time as compared to CENET, and further improve the effect by learning the weights of historical information.

## 3 Method

### 3.1 Notations

Temporal knowledge graphs are multi-relational knowledge graphs with temporal information on the relations and can represent knowledge graph sequences $G = \{G_1, G_2, ..., G_t\}$, where the graph $G_t = (E, R, F_t)$ at each timestamp $t$ can be represented as a directed multi-relationship graph, where $E$ is the set of entities, $R$ is the set of relationships, $F_t$ is the set of facts on timestamp $t$, $|E|$ and $|R|$ denote the number of entities and relationships. Any fact $F_t$ can be expressed in the form of a quaternion $(s, r, o, t)$, where $s, o \in E, r \in R$, it represents the fact that at timestamp $t$, the head entity $s$ and the tail entity $o$ have a relationship $r$. For each quadruple $(s, r, o, t)$, we construct its inverse relation quadruple $(o, r^{-1}, s, t)$ and append it to the dataset, so that the prediction of both head and tail entities can be realized. Also, we set the embedding dimension of entities and relations to $d$, $\mathbf{E} \in \mathbb{R}^{|E|\times d}$, $\mathbf{R} \in \mathbb{R}^{|R|\times d}$ are respectively expressed as the embedding matrices of all entities and relationships. For each temporal subgraph $G_t$, we define the entity embedding matrix as $\mathbf{E}_t$ and the relation embedding matrix as $\mathbf{R}_t$, and randomly initialize the input embeddings of entities and relations of the first historical sequence as $\mathbf{E}_{init}$, $\mathbf{R}_{init}$ respectively. Table 2 lists the relevant mathematical symbols.

Temporal knowledge graphs reasoning refers to the prediction of missing tail entities in the query $(s, r, ?, t)$given the set of historical events $G_{0:t-1}$ before time $t$. Without loss of generality, the model proposed in this paper can be easily extended to predict the missing head entity in query $(o, r^{-1}, ?, t)$. Given a sequence of historical graphs, the probability of calculating the predicted entity at the current time can be expressed mathematically as:

$$p(o|s, r, G_{t-k:t}) = p(o|s, r, \mathbf{E}_t, \mathbf{R}_t) \tag{1}$$

where $G_{t-k:t}$ represents the given $k$ historical temporal knowledge graphs, $\mathbf{E_t} \in \mathbb{R}^{|E|\times d}$, denote the entity and the relationship embedding matrices, respectively, and $\mathbf{R_t} \in \mathbb{R}^{2|R|\times d}$ is the dimension of the embedding.

**Table 2** Important mathematical symbols and related descriptions

| Notations | Descriptions |
|---|---|
| $G, G_t$ | TKG, KG at time t |
| $E, R, F_t$ | Set of entities, relations and facts on TKG |
| $\|E\|, \|R\|$ | The number of entities and relationships |
| $\mathbf{E}_t$,$\mathbf{R}_t$ | Entity and relationship embedding matrices at time t |
| $G_{0:t-1}$ | The set of knowledge graphs before time t |

### 3.2 Model overview

As shown in Fig. 2, our proposed CH-TKG model consists of three modules, namely the evolutionary representation module, the contrastive learning module, and the historical information weighting module. The evolutionary representation learning module first models historical graph sequences using R-GCN and GRU to obtain entity and relation embeddings with evolutionary dependencies. On this basis, the contrastive learning module then optimizes the entity and relation representations based on the prototype contrastive framework. Finally, for the historical information evolution module, the self-attention mechanism and copy mechanism are used to learn the importance of different historical information to the final inference result.

### 3.3 Evolutional representation module

We first model each temporal subgraph using R-GCN to obtain entity embedding for the relevant knowledge subgraph. For a temporal knowledge graph $G$, each temporal subgraph $\{G_i|0 \leqslant i \leqslant t-1\}$ can be viewed as a static multi-relationship graph. Since GCNs are powerful for modeling graph-structured data, they can capture the structural dependencies within subgraphs. Specifically, for $G_i$ on timestamp $i$, we follow the processing strategy of RE-GCN, using R-GCN to model each subgraph. For each node (entity) in the graph, its adjacent information is aggregated through a message-passing architecture. Specifically, each entity in layer $l$ ($l \in [0, 1]$) aggregates the neighboring relationship embeddings and entity embeddings according to the dif-

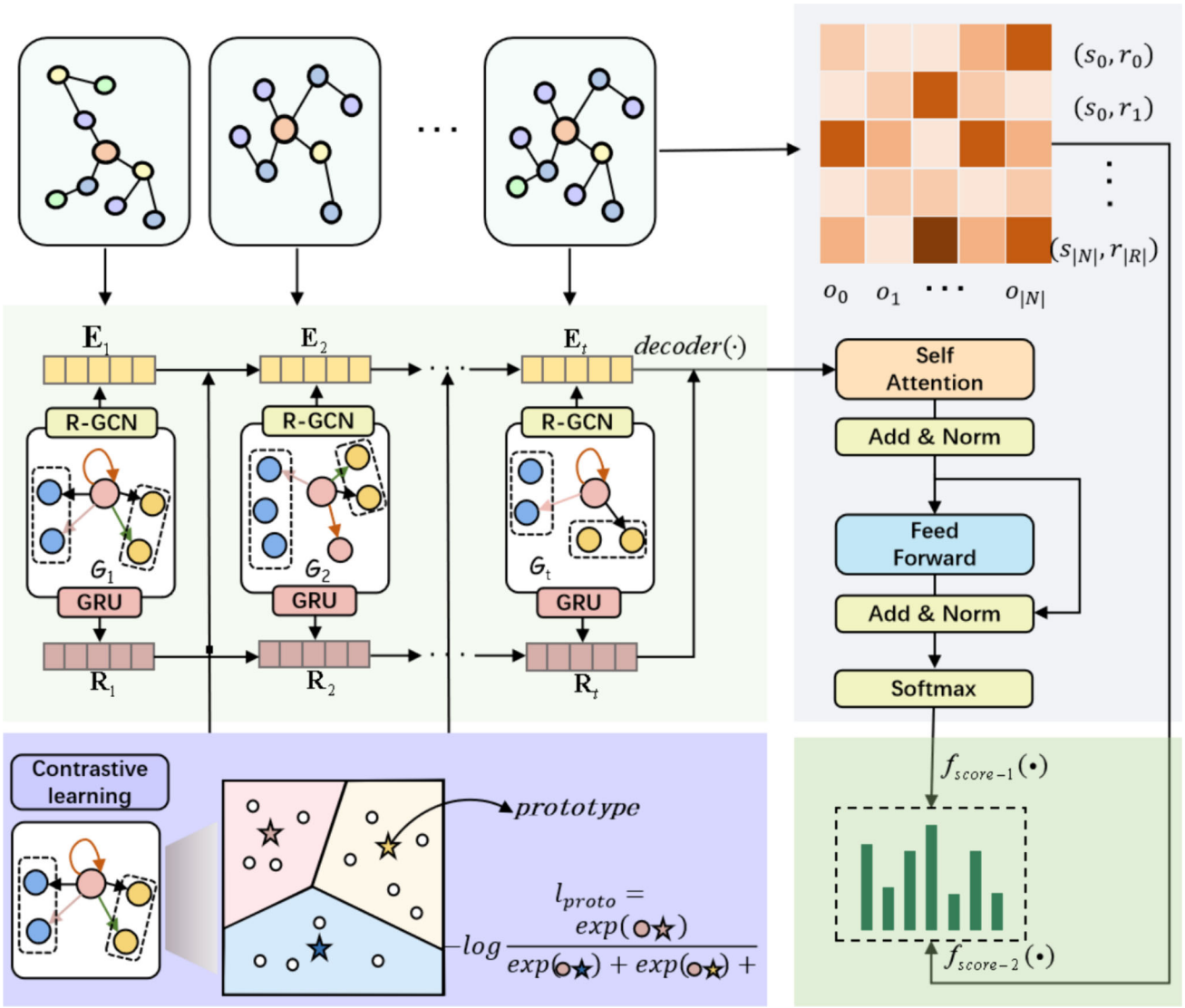



**Fig. 2** An illustration of the architecture of the model. Different colors represent different modules: light green marks the Evolutional Representation module, purple marks the Contrastive Learning module, and grey is the Historical Information Learning module

ferent information of the incidence edges (relationships) to obtain the entity embeddings in layer $l+1$. Experiments in Section 4.2.4 show that two R-GCN aggregation layers are sufficient to capture enough structural information for entity embedding. Since we create quadruples with inverse relations, a temporal subgraph can be viewed as an undirected graph, with each relation of an entity viewed as an in-degree edge:

$$\boldsymbol{h}_{o,t}^{l+1} = f\left(\frac{1}{c_o}\sum_{(s,r),\exists(s,r,o)\in e_t} W_1^l\left(\boldsymbol{h}_{s,t}^l + \boldsymbol{r}_t^l\right) + W_2^l \boldsymbol{h}_{\mathrm{o},t}^{l}\right) \quad (2)$$

where $\boldsymbol{h}_{s,t}^l$, $\boldsymbol{h}_{o,t}^l$, and $\boldsymbol{r}_t^l$ represent the embedding representation of entities $s$, $o$ and relations $r$ in R-GCN at layer $l$ at timestamp $t$, $W_1^l$ indicates the parameter of the structural characteristics of all entities aggregated at layer $l$ according to the connected edges of the aggregation, $W_2^l$ is the self-recycling parameter of the characteristics of the aggregated entities themselves, $c_o$ is a normalization constant representing the degree of incidence of the current node $o$, and $f(\cdot)$ is the ReLU activation function. By (2), its entity embedding $\boldsymbol{h}_{o,t}^{l+1}$ at timestamp $t$ can be obtained, which is also expressed as $\mathbf{E}_t$.

To model time dependencies between temporal subgraphs, we use the embedding output from the previous subgraph as input to the R-GCN model for the next timestamp:

$$\boldsymbol{h}_{o,t}^1 = \mathbf{E}_{t-1} \quad (3)$$

where $\mathbf{E}_{t-1} \in \mathbb{R}^{|E|\times d}$ denotes the output entity embedding representation of the $(t-1)^{th}$ time subgraph (i.e., the output of layer 2 of the $(t-1)^{th}$ R-GCN). $\boldsymbol{h}_{o,t}^1$ denotes the input entity embedding representation of layer 1 of the R-GCN at time t (i.e., the $t^{th}$ time subgraph). For the $t^{th}$ timestamped query, we can obtain a representation $\{\mathbf{E}_{t-k}, \cdots, \mathbf{E}_{t-2}, \mathbf{E}_{t-1}\}$ of the sequence of time sub-graphs with history length $k$.

In addition, the dependencies between relationships at different times are closely related to the involved entities. In order to aggregate structural information when learning relationship embeddings, we concatenate the mean of entity embeddings from the previous subgraph associated with the current relationship, with the mean of the edge embeddings of the initial subgraph as inputs to the GRU unit at moment $t$:

$$\mathbf{R}_{input}^t = [\mathbf{R}; mean(\mathbf{E}_{t-1}, \mathbf{E}_p^{t-1})] \quad (4)$$

where $\mathbf{R}_{input}^t \in \mathbb{R}^{2|R|\times d}$ denotes the input matrix of the GRU unit at the $t^{th}$ timestamp, $\mathbf{R} \in \mathbb{R}^{2|R|\times d}$, $\mathbf{R}_{init}$ indicates the relationship embedding in the input matrix of the first history subgraph, [; ] means the vector linking operation, which represents the entity embedding of the temporal subgraph at the $(t-1)^{th}$ timestamp, and $\mathbf{E}_{\mathrm{r}}^{t-1}$ is the set of entities associated with the particular relationship recorded at the $(t-1)^{th}$ timestamp.

In the next step, we further model the temporal dependence of the relation by taking the hidden unit output of the GRU of the previous temporal subgraph as the hidden unit input of the GRU of the next temporal subgraph:

$$\mathbf{R}_{hidden}^t = GRU\left(\mathbf{R}_{input}^t, \mathbf{R}_{hidden}^{t-1}\right) \quad (5)$$

where $\mathbf{R}_{hidden}^{t-1} \in \mathbb{R}^{2|R|\times d}$ indicates the relation embedding representation (including the inverse relation) at the $(t-1)^{th}$ timestamp, and $t^{th}$ represents the relation embedding representation at the $k$ timestamp. Specifically, for a history subgraph of length, the hidden unit input of the GRU unit at the 1st history timestamp $\mathbf{R}_{\mathrm{init}}^t$ is $\mathbf{R}_{hidden}^{t-k}$.

### 3.4 Contrastive learning module

To mitigate data sparsity, we consider enhancing the embedding representation by merging semantic neighbors, which refer to relation or entity embeddings unreachable on the graph but with similar characteristics. Inspired by prototype graph contrastive learning [9, 40], we can identify semantic neighbors by learning potential prototypes of each entity and relation. Based on this idea, we further propose the prototype-contrastive objective to explore potential semantic neighbors, aiming to better capture the semantic characteristics of entities and relationships through collaborative filtering. Typically, similar entities and relations tend to fall in the adjacent embedding space, with a prototype representing the center of a semantically similar cluster. Therefore, we apply the clustering algorithm to the embedding representation of entity nodes after GNN initialization. For convenience, we use the classical $k-mean$ clustering algorithm. In the $k-mean$ clustering algorithm, there is a clustering center, called a prototype, for each category that is classified. Within a category, the cluster centers are positive samples and the other cluster centers are negative samples of the nodes, which is used to cluster all entity nodes into different classes, allowing us to apply contrastive learning well. The contrastive learning loss function is shown in (6):

$$l_p^E = \sum_{u\in E} -\log\frac{\exp\left(\mathbf{e}_u\cdot\mathbf{c}_i/\tau\right)}{\sum_{c_j\neq i\in\mathbf{c}}\exp\left(\mathbf{e}_u\cdot\mathbf{c}_j/\tau\right)+\exp\left(\mathbf{e}_u\cdot\mathbf{c}_i/\tau\right)} \quad (6)$$

where $\mathbf{c}_i$ is the prototype of entity $\mathbf{e}-u$, which is obtained by clustering all entity embeddings using the $k-mean$ algorithm, and $k$ clusters are obtained by covering all entity nodes using the clustering algorithm, $\mathbf{c}_j$ is the prototype of the other

categories, and $\tau$ is the temperature hyperparameter of softmax. The contrastive learning on relations is similar:

$$l_p^R = \sum_{u \in R} -\log \frac{\exp\left(\mathbf{r}_u \cdot \mathbf{c}_j/\tau\right)}{\sum\limits_{\mathbf{c}_t \neq j \in \mathbf{c}} \exp\left(\mathbf{r}_u \cdot \mathbf{c}_t/\tau\right) + \exp\left(\mathbf{r}_u \cdot \mathbf{c}_j/\tau\right)} \quad (7)$$

where $\mathbf{c}_j$ is the prototype of relation $\mathbf{r}_u$ .

The final prototype comparison target is the weighted sum of the entity target and the relation target:

$$l_P = l_p^E + \alpha l_p^R \quad (8)$$

In this way, we explicitly merge the semantic neighbors of entities and relations into contrastive learning to mitigate the sparsity of the data.

## 3.5 Historical information learning module

### 3.5.1 Temporal attention module

After obtaining the distributional embeddings of temporal subgraphs with $k$-length histories, considering that the most recent history timestamps play the largest role in prediction, we apply learnable attention to all history timestamps. This allows learning the importance of different history subgraphs to the final inference result through the self-attention mechanism. For the query task $(s, r, ?, t)$, the embeddings $\mathbf{s}$ and $\mathbf{r}$ of entities and relations at the $t^{th}$ timestamp are first obtained through the embedding matrices $\mathbf{E}_t$ and $\mathbf{R}_t$ of the entities and relations. Probability scores for candidate triples are then calculated using a scoring function (i.e., the decoder) to model the conditional probabilities in (1). The decoder component employs Conv-TransE, derived from ConvE with the preservation of TransE's translation property. Conv-TransE is one of the most commonly used decoders for calculating the probability of candidate entities. It models the interaction between input entities and relations through convolutional and fully connected layers. First, it transforms the embeddings of the head and tail entities into a two-dimensional tensor, and then applies standard convolution operations on this tensor to calculate the triple score. For each pair $(s, r)$ in a given quaternion $(s, r, o, t)$, Conv-TransE computes candidate entities as follows: the representations $\mathbf{e}_t$ and $\mathbf{r}_t$ corresponding to the entity $s$ and the relationship $r$ that are obtained by the comparison learning module are spliced into a $2\star n$ matrix. Next, a convolution operation $\mathrm{M}(\cdot,\cdot)$ is performed on this matrix to obtain the feature maps, which are then flattened into a vector by the flattening operation $vec\,(\cdot)$ and the dimension is reduced through a fully connected layer. The intermediate vector $\boldsymbol{y}_{temp}$ fusing $\mathbf{s}$ and $\mathbf{t}$ at the current time $t$ is obtained:

$$\boldsymbol{y}_{temp} = vec(\mathrm{M}([\mathbf{e}_t; \mathbf{r}_t] * \omega))W \quad (9)$$

For a history subgraph sequence query $(s, r, o, t)$ of length $k$, a sequence of $k$ intermediate vectors $\boldsymbol{y}_{temp:t-k}$, ..., $\boldsymbol{y}_{temp:t-2}$, $\boldsymbol{y}_{temp:t-1}$ can be obtained, and since the influence of distant history information on the outcome diminishes with the passage of time, we let $\boldsymbol{y}_{t-1}$ impose learnable attention on all history timestamps, including itself. Then, the query vector $\boldsymbol{Q}$ is generated by $\boldsymbol{y}_{t-1}$ , and the key-value vectors $\boldsymbol{K}$ and $\boldsymbol{V}$ are generated by $\boldsymbol{y}_{t-k}$, ..., $\boldsymbol{y}_{t-2}$, $\boldsymbol{y}_{t-1}$ :

$$\begin{aligned} &\boldsymbol{Q} = W_q \boldsymbol{y}_{t-1}, \boldsymbol{K} = W_k[\boldsymbol{y}_{t-k}, ..., \boldsymbol{y}_{t-2}, \boldsymbol{y}_{t-1}], \\ &\boldsymbol{V} = W_v[\boldsymbol{y}_{t-k}, ..., \boldsymbol{y}_{t-2}, \boldsymbol{y}_{t-1}] \end{aligned} \quad (10)$$

where, $W_q \in \mathbb{R}^{d_q \times d}$, $W_k \in \mathbb{R}^{d_k \times d}$, $W_v \in \mathbb{R}^{d_v \times d}$, $\boldsymbol{y}_{t-1} \in \mathbb{R}^d$, $[\boldsymbol{y}_{t-k}, ..., \boldsymbol{y}_{t-2}, \boldsymbol{y}_{t-1}] \in \mathbb{R}^{k \times d}$

The self-attention operation can be expressed as :

$$\text{self\_attention}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) = \text{softmax}\left(\frac{\boldsymbol{Q}\boldsymbol{K}^T}{\sqrt{d_k}}\boldsymbol{V}\right) \quad (11)$$

Among them $\boldsymbol{Q} \in \mathbb{R}^{d_q}$ , $\boldsymbol{K} \in \mathbb{R}^{d_k \times d}$, $\boldsymbol{V} \in \mathbb{R}^{d_v \times d}$ . $\sqrt{d_k}$ is used to prevent the gradient vanishing problem. $W_q$, $W_k$, $W_v$ are learnable parameters that assign learnable attention weights to each historical timestamp. In practice, we introduce multi-head attention and then utilize feed-forward neural network FFN to introduce deep semantic information:

$$\mathrm{FFN}(z) = W_2(\mathrm{ReLU}(W_3 z)) \quad (12)$$

where $z \in \mathbb{R}^d$ is the output of the multi-head attention mechanism. $W_2 \in \mathbb{R}^{d_{ff} \times d}$, $W_3 \in \mathbb{R}^{d_{ff} \times d}$ are parameters, $d_{ff}$ is the number of hidden units in the FFN.

In addition, we add residual concatenation and layer normalization to the output of the multi-head attention mechanism and FFN, respectively. Then, for query $(s, r, ?, t)$, after Conv-TransE, we perform matrix multiplication on the output $\boldsymbol{g}$ and entity embedding $\mathbf{E}_{t-1}$ at the most recent history timestamp:

$$S_D = mm(\boldsymbol{g}, \mathbf{E}_{t-1}) \quad (13)$$

$$P_1(o|(s, r, t)) = S_D \quad (14)$$

where $\boldsymbol{g} \in \mathbb{R}^d$, $\mathbf{E}_{t-1} \in \mathbb{R}^{d \times N}$, $S_D \in \mathbb{R}^N$ are the final scores obtained in this module.

#### 3.5.2 Frequency statistics module

In terms of the importance of historical information, not only does the timing of an event greatly influence the ultimate inference result, but the frequency of event occurrences also wields a similarly substantial impact on the final inference result. In this module, we count the occurrences of historical events that have appeared, and utilize the copy mechanism to assign a higher probability to historical events with greater frequencies in the prediction. Specifically, for a query $(s, r, ?, t)$ at time $t$, obtain the set of tail entities associated with $(s, r)$ at each timestamp in the training set, denoted as $\{o_i|(s, r, o_i) \in G_{1,2,...,t-1}\}$. It is expressed as follows in (15):

$$\boldsymbol{O}_t^{(s,r)} = \boldsymbol{O}_0^{(s,r)} + \boldsymbol{O}_1^{(s,r)} + ... + \boldsymbol{O}_{t-1}^{(s,r)} \tag{15}$$

where $\boldsymbol{O}_t^{(s,r)}$ is an $N$-dimensional multi-hot indication vector, each dimension represents the frequency of occurrence of the corresponding historical entity.

If the query $(s, r, ?, t)$ has a historical entity $\boldsymbol{O}_t^{(s,r)}$ associated with $(s, r)$ at the timestamp $t$, the model will increase the estimated probability that the associated historical entity is selected as the final result. Specifically, we use a sparse matrix of size $|E| \cdot |R| \times |E|$ to hold all $(s, r, o)$ triples, and each row of the matrix represents a set of $(s, r)$, When $(s, r, o)$ has appeared in a historical event $(s, r)$, the sparse matrix assigns the value of 1 to the $(s, r)$ row where the index of the column of the $(s, r)$ row is $o$. In order to minimize the probability of certain entities that are not historically associated with $s$ and $r$ (i.e., entities that are not interested in the replication model), we first modify $\boldsymbol{H}_t^{(s,r)}$. The index value of the entity with column index 0 in $\boldsymbol{O}_t^{(s,r)}$ is changed to a small negative number, and the probability of the tail entity in the historical vocabulary is estimated using the softmax function to minimize the probability of the uninterested entity, and the final output score of the replication mechanism is as follows:

$$P_2(o|(s, r, t)) = \text{softmax}\left(\boldsymbol{O}_t^{(s,r)}\right) \tag{16}$$

### 3.6 Parameter learning

For a query $(s, r, ?, t)$ , we merge $P_1$ and $P_2$ in the final prediction:

$$P = P_1 + P_2 \tag{17}$$

Therefore, by simultaneously considering the evolutionary, contrastive learning and historical information of TKGs, the prediction process can be viewed as an n-labeled classification problem, and we use the cross-entropy loss to accomplish the task:

$$l_m = -\sum_{t\in T}\sum_{i\in E}\sum_{j\in E} o_i^t \ln P_t\left(y_j^t|s, r, \mathbf{E}_t, \mathbf{R}_t\right) \tag{18}$$

where $o_i^t$ denotes the $i^{th}$ ground truth-tailed entity in the $t^{th}$ time subgraph, and $P_t\left(y_j^t|s, r, \mathbf{E}_t, \mathbf{R}_t\right)$ denotes the probability that the $j^{th}$ entity is predicted to be the inference entity at the $t^{th}$ timestamp.

In summary, our model can be described as a multi-task learning approach, where the total loss of the modeled tasks is defined in (19):

$$l = l_m + l_p \tag{19}$$

## 4 Experiments

In this section, we present and analyze the experimental results of the proposed CH-TKG framework on inference tasks on four datasets. Firstly, we introduce the parameter settings and the experimental environment in detail. Secondly, the experimental results obtained are analyzed. Then we perform data sparsity experiments to demonstrate the effectiveness of the proposed algorithm. Finally, we conduct ablation experiments and parameter sensitivity analysis to explore the importance of each module.

### 4.1 Experimental setup

#### 4.1.1 Datasets

We use four typical TKG data sets commonly used in previous work for evaluation, namely ICEWS14 [27], ICEWS18 [6], WIKI and YAGO. The first two datasets are from the Integrated Crisis Early Warning System (ICEWS) [3], which contains political events with precise timestamps. The former dataset contains events from the year 2014, and the latter contains events from the year 2018. YAGO is a traditional large-scale knowledge graph, whereas YAGO3 [44] is a subset that incorporates temporal information. WIKI encompasses timing information extracted from Wikipedia [22]. We evaluated the model on these datasets. Following previous work [6], the dataset is divided into training, validation, and testing sets with an 80%, 10%, and 10% split respectively for comparison. The time interval represents the granularity between each timestamp. Since the datasets have different time spans and forms, we follow previous work [24]

**Table 3** Statistics of the dataset

| Data | Entities | Relation | Training | Validiation | Test | Time Interval | Time Granules |
|---|---|---|---|---|---|---|---|
| ICEWS14 | 7128 | 230 | 74845 | 8514 | 7371 | 24 hours | 365 |
| ICEWS18 | 23033 | 256 | 373018 | 45995 | 49545 | 24 hours | 304 |
| WIKI | 12554 | 24 | 539286 | 67538 | 63110 | 1 year | 232 |
| YAGO | 10623 | 10 | 161540 | 19523 | 20026 | 1 year | 189 |

and discretize the time facts into yearly snapshots. Table 3 summarizes the detailed statistics of these datasets.

#### 4.1.2 Evaluation metrics

We evaluate the model through the entity prediction task, which aims to predict missing knowledge for a future timestamp t during training (formally, predicting missing entities for $(s, r, ?, t)$ or $(?, r, o, t)$, where $t \notin T_{train}$ ). For each dataset, we follow the strategy of TransE [5] to measure the model performance. Specifically, for each test quadruple $(s, r, o, t)$, we replace the head and tail entities with all possible entities and relationships in turn to obtain a candidate knowledge list. Then, candidates are ranked by descending similarity score along with original facts.

We measure the performance using four commonly used metrics: *MRR* (the average inverse rank of correct entities), $Hits@1$ , $Hits@3$, and $Hits@10$ (the accuracy of the correct entities in the top 1/3/10), the formulas for calculating each of them are as follows:

$$MRR = \frac{1}{|S|}\sum_{i=1}^{|S|}\frac{1}{rank_i} = \frac{1}{|S|}\left(\frac{1}{rank_1} + \frac{1}{rank_2} + \cdots + \frac{1}{rank_{|S|}}\right) \tag{20}$$

where $S$ is the set of triples, $|S|$ is the number of sets of triples, and $rank_i$ is the predicted ranking of links for the $i^{th}$ triple, the larger the better for this metric.

$$Hits@n = \frac{1}{|S|}\sum_{i=1}^{|S|}\mathbb{I}(rank_i \leqslant n) \tag{21}$$

The symbols are the same as those involved in the formula for $MRR$, where $\mathbb{I}$ is the INDICATOR function (the value of the function is 1 if the condition is true, 0 otherwise), and the larger the indicator, the better.

#### 4.1.3 Baselines

We compare our model with static and temporal knowledge graph reasoning methods respectively to verify its superiority.

For static knowledge graph representation learning methods, we chose several typical models.

- ComplEx [14]: the first method that introduces the complex space into knowledge graph embedding;
- ConvE [45]: the first model that uses two-dimensional convolution for link prediction;
- Conv-TransE [46]: a decoder that captures graph structural information while preserving translation properties;
- RotatE [47]: can efficiently models and inferring various symmetric and asymmetric relationship patterns;
- R-GCN [18]: a graph convolutional neural network that handles multiple relationships.

For temporal knowledge graphs reasoning methods, we compare some interpolation models.

- TTransE [22]: adds a temporal dimension to TransE;
- HyTE [48]: models temporal information as a hyperplane;
- TA-DistMult [21], a model that integrates temporal information about facts into embedding representations of relations.

We also compare the extrapolation models in recent years that are more similar to our working task.

- RE-NET [6]: effectively extracts temporal and structural information from temporal knowledge graphs to predict the occurrence of events at future moments.
- RE-GCN [8]: efficiently models all the historical information in the TKG into evolutional representations.
- EvoKG [7]: utilizes recurrent neural networks to capture temporal information on temporal graphs, and graph neural networks on graphs of the same time to capture structural information;
- CyGNet [24]: is the first model that applies the copy-generation mechanism to temporal knowledge graphs reasoning;
- CluSTeR [25]: a two-stage search inference model based on reinforcement learning and GCN;
- CEN [49]: a model for learning information evolution patterns using online settings;
- GLANet [26]: solves information sparsity through global and local information;
- CENET [43]: the first model to apply contrastive learning to temporal knowledge graphs reasoning.

#### 4.1.4 Implementation Details

In this task, we optimize the Adam optimizer with a learning rate of 0.001 to minimize the loss function. The vector embedding dimension of entities and relations is 100, the number of R-GCN layers is set to 2, the dropout rate is set to 0.2, the temperature coefficient $\tau$ is 0.005, and the prototypical $k$ is 15. More detailed experiments related to the parameter settings we described in detail in Section 4.2.4. The number of cores in the decoder Conv-TransE is set to 50, the core size is set to 2×3, and the dropout rate is set to 0.2.

### 4.2 Results and analysis

#### 4.2.1 Entity prediction

In this section, we compare the performance of CH-TKG with TKG-based static and dynamic inference methods. Tables 4 and 5 show our results versus baselines. Since static models ignore the temporal information of TKGs and cannot accurately model event evolution, CH-TKG fully considers the historical sequence of events and significantly outperforms static inference methods. The performance of TTransE and HyTE is even worse than static methods because they are designed for interpolation tasks instead of extrapolation tasks. Compared with history-based methods such as RE-NET, CyGNet, RE-GCN, CluSTeR, etc., CH-TKG still performs better. Our analysis suggests that although these methods utilize the linkage on and between historical series, they all ignore the problem of data sparsity on individual time slices, making them less effective. The CH-TKG model outperforms the other baseline models on most of the metrics on the ICEWS14 and ICEWS18 datasets. Compared to CENET, the first temporal knowledge reasoning contrastive learning model, CH-TKG achieves suboptimal values for $Hits@1$ on the ICEWS18 dataset, while achieves an improvement of 1.65%, 8.23% and 24.26% on the *MRR*, $Hits@3$ and $Hits@10$ metrics, respectively.

On YAGO and WIKI datasets, our model slightly underperforms the CENET model on *MRR* and $Hits@3$, while improving $Hits@10$ by 15.17% and 8.79% respectively. For the reason that the effect is lower than the CENET model on WIKI and YAGO datasets, we analyze that it may be because of the larger amount of data on these two datasets, and the CENET model learns more history-related information, while our model is biased to solve the data-sparse type of inference. To further analyze the reason, we conduct experiments on data sparsity in Section 4.2.2. $Hits@1$ focuses on the correctness of the model predictions, while $Hits@10$ accounts for multiple candidates, suggesting our proposed model provides better overall performance and robustness,

**Table 4** Entity prediction results on ICEWS14 and ICEWS18

| Model | ICEWS14 | | | | ICEWS18 | | | |
|---|---|---|---|---|---|---|---|---|
| | *MRR* | $Hits@1$ | $Hits@3$ | $Hits@10$ | *MRR* | $Hits@1$ | $Hits@3$ | $Hits@10$ |
| ComplEx* | 22.61 | 9.88 | 28.93 | 47.57 | 15.45 | 8.04 | 17.19 | 30.73 |
| ConvE* | 30.30 | 21.30 | 34.42 | 47.89 | 22.81 | 13.63 | 25.83 | 41.43 |
| Conv-TransE* | 31.50 | 22.46 | 34.98 | 50.03 | 23.22 | 14.26 | 26.13 | 41.34 |
| RotatE* | 25.71 | 16.41 | 29.01 | 45.16 | 14.53 | 6.47 | 15.78 | 31.86 |
| R-GCN* | 28.03 | 19.42 | 31.95 | 44.83 | 15.05 | 8.13 | 16.49 | 29.00 |
| TTransE* | 12.86 | 3.14 | 15.72 | 33.65 | 8.44 | 1.85 | 8.95 | 22.38 |
| HyTE* | 16.78 | 2.13 | 24.84 | 43.94 | 7.41 | 3.10 | 7.33 | 16.01 |
| TA-DistMult* | 26.22 | 16.83 | 29.72 | 45.23 | 16.42 | 8.60 | 18.13 | 32.51 |
| RE-NET* | 35.77 | 25.99 | 40.10 | 54.87 | 26.17 | 16.43 | 29.89 | 44.37 |
| CyGNet* | 34.68 | 25.35 | 38.88 | 53.16 | 24.98 | 15.54 | 28.58 | 43.54 |
| RE-GCN* | 41.25 | 30.46 | 46.26 | 62.05 | 30.79 | 20.06 | 35.22 | 51.77 |
| CluSTeR* | 46.00 | 33.80 | – | 71.20 | 32.30 | 20.60 | – | 55.90 |
| CEN* | 41.64 | 31.22 | 46.55 | 61.59 | 29.70 | 19.38 | 33.91 | 49.90 |
| EvoKG* | 27.18 | – | 30.84 | 47.67 | 29.28 | – | 33.94 | 50.09 |
| GLANet* | 44.06 | 33.92 | 49.23 | 63.35 | 32.07 | 21.59 | 36.60 | 52.46 |
| CENET | 46.88 | 43.10 | 48.30 | 54.62 | 46.20 | **43.07** | 47.09 | 52.42 |
| CH-TKG | **62.31** | **48.27** | **72.19** | **88.52** | **47.85** | 33.61 | **55.32** | **76.68** |
| Improv. | 15.43 ↑ | 5.17 ↑ | 23.89 ↑ | 33.90 ↑ | 1.65 ↑ | 9.46 ↓ | 8.23 ↑ | 24.26 ↑ |

(The best results are shown in boldface type, and suboptimal results are underlined. ↑ indicates an increase in results, ↓ indicates a decrease in results. * indicates the data in the original text)

**Table 5** Entity prediction results on WIKI and YAGO

| Model | WIKI | | | YAGO | | |
|---|---|---|---|---|---|---|
| | *MRR* | *Hits*@3 | *Hits*@10 | *MRR* | *Hits*@3 | *Hits*@10 |
| ComplEx* | 27.69 | 31.99 | 38.61 | 44.09 | 49.57 | 59.64 |
| ConvE* | 26.03 | 30.51 | 39.18 | 41.22 | 47.03 | 59.90 |
| Conv-TransE* | 30.89 | 34.30 | 41.45 | 46.67 | 52.22 | 62.52 |
| RotatE* | 26.08 | 31.63 | 38.51 | 42.08 | 46.77 | 59.39 |
| R-GCN* | 13.96 | 15.75 | 22.05 | 20.25 | 24.01 | 37.30 |
| TTransE* | 20.66 | 23.88 | 33.04 | 26.10 | 36.28 | 47.73 |
| HyTE* | 25.40 | 29.16 | 37.54 | 14.42 | 39.73 | 46.98 |
| TA-DistMult* | 26.44 | 31.36 | 38.97 | 44.98 | 50.64 | 61.11 |
| RE-NET* | 30.87 | 33.55 | 41.27 | 46.81 | 52.71 | 61.93 |
| CyGNet* | 30.77 | 33.83 | 41.19 | 46.72 | 52.48 | 61.52 |
| RE-GCN* | 50.99 | 57.34 | 68.50 | 62.50 | 70.24 | 81.55 |
| CEN* | 51.98 | 58.96 | 70.61 | 63.39 | 71.68 | 83.16 |
| EvoKG* | 50.66 | 63.84 | – | 55.11 | 81.13 | – |
| GLANet* | 53.18 | 61.16 | 71.52 | 65.05 | 74.86 | 87.51 |
| CENET | **67.06** | **67.36** | 67.52 | **83.80** | **83.94** | 84.10 |
| CH-TKG | 56.38 | 64.78 | **82.69** | 67.78 | 77.36 | **92.89** |
| Improv. | 10.68 ↓ | 2.58 ↓ | 15.17 ↑ | 16.02 ↓ | 6.58 ↓ | 8.79 ↑ |

(The best results are shown in boldface type, and suboptimal results are underlined. ↑ indicates an increase in results, ↓ indicates a decrease in results.* indicates the data in the original text)

across a wider range, which is more important for the case of multiple candidates.

### 4.2.2 Experiments on data sparsity

In the entity prediction experiments on the WIKI and YAGO datasets, our model gives lower experimental results than CENET on *MRR* and *Hits*@3, but better results on *Hits*@10. To further analyze this, we conduct experiments using randomly sampled percentages of the datasets. Specifically, we randomly selected 3%, 5%, 10%, and 20% of data from the WIKI and YAGO datasets respectively, and evaluated the performance of the model. The experimental results are shown in Fig. 3, where our model significantly outperforms the CENET model on *MRR* and *Hits*@3 metrics with less data, indicating that our model is more effective in dealing with the data sparsity problem. With the increase of data volume, the metrics gap between the two models gradually decreases. Regarding the choice of data sparsity for the two datasets, we take the data volume in Table 1 as a reference and

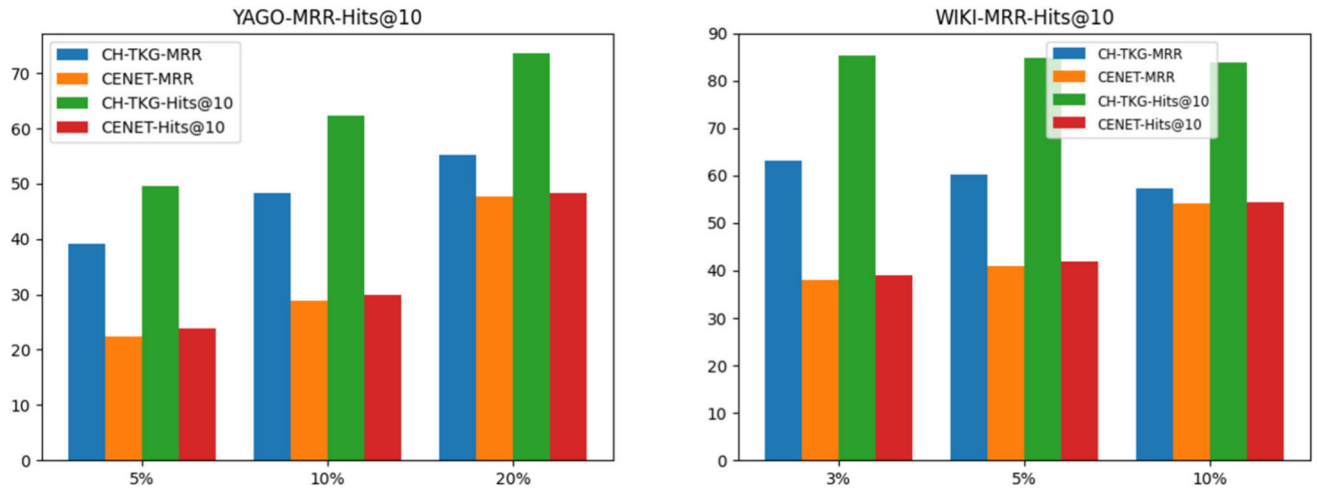



**Fig. 3** Impact of data sparsity on inference performance on YAGO and WIKI datasets

**Table 6** Results of ablation experiments on ICEWS14 and YAGO

| Method | ICEWS14 | | | YAGO | | |
|---|---|---|---|---|---|---|
| | *MRR* | *Hits*@3 | *Hits*@10 | *MRR* | *Hits*@3 | *Hits*@10 |
| w/o CL Module | 61.79 | 71.53 | 88.08 | 67.19 | 77.01 | 92.41 |
| w/o SA Mechanism | 62.03 | 71.60 | 88.14 | 67.24 | 77.17 | 92.56 |
| w/o CM Mechanism | 62.15 | 71.64 | 88.12 | 67.37 | 76.82 | 92.69 |
| w/o HIL Module | 61.85 | 71.49 | 88.01 | 67.22 | 76.95 | 92.53 |
| CH-TKG | 62.31 | 72.19 | 88.52 | 67.78 | 77.36 | 92.89 |

choose 3%, 5%, and 10% of data for experiments on the WIKI dataset, which has a larger volume of data, and 5%, 10%, and 20% of data on the YAGO dataset, which has a smaller volume of data. The datasets that support this experiment are openly available at https://github.com/mqygit/TKGs.

### 4.2.3 Ablation experiment

In order to understand the contribution of each module of CH-TKG, we choose to conduct ablation experiments on ICEWS14 and YAGO datasets. We experimented by removing the contrastive learning module, the self-attention mechanism module and the copy mechanism module of historical information weight learning, and the entire historical information weight learning module, respectively. The results are shown in Table 6. It can be seen that the contrastive learning module and the historical information weight learning module have different degrees of improvement for each metric, which demonstrates the effectiveness of our model. “w/o CL module" means that the contrastive learning module is disregarded and only the embedding representation obtained by the evolutionary representation learning module is used for inference. It can be seen that on both datasets, after removing the contrastive learning module that captures the latent semantics, the MRR decreases by 0.5% and 0.6% on ICEWS14 and YAGO respectively. Contrastive learning is slightly more effective on the YAGO dataset than on ICEWS14, which may be due to the fact that the data on the YAGO dataset is denser and contains a wide range of entities and relationships, whereas in ICEWS14, which mainly consists of information about international events and crises, certain events or relationships may be very sparse, making contrastive learning difficult to capture information effectively. With the exclusion of the self-attention(SA) model, copy mechanism(CM) model, and the historical information learning(HIL) module, respectively, we can observe that the experimental results of the ICEWS14 dataset are reduced to a lesser extent compared to the YAGO dataset. This may again relate to data sparsity: with less data, our model shows a significant performance advantage over other models, but with more data, the model is still able to learn more information from more data. The complete model achieves optimal performance on both datasets, thus validating the effectiveness of the individual constituent units and the generalizability of the model across datasets.

### 4.2.4 Parameter Sensitivity

In order to evaluate the effects of different parameter variations on the performance of CH-TKG, we change the settings of the following parameters: (a) the size of the embedding dimension, (b) the number of layers of the R-GCN, (c) the magnitude of the temperature coefficient $\tau$, and (d) the number of clustering archetypes $k$.

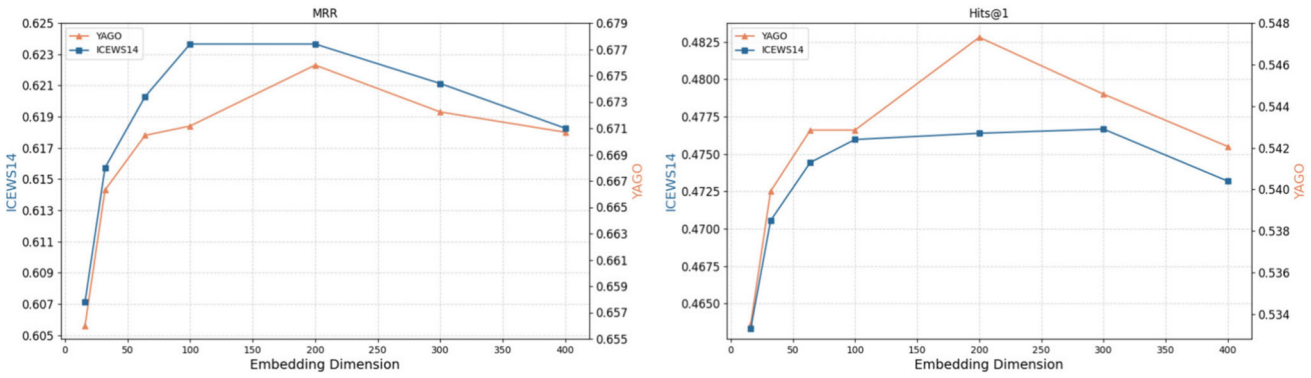



**Fig. 4** Impact of embedding dimension on reasoning performance on ICEWS14 and YAGO dataset

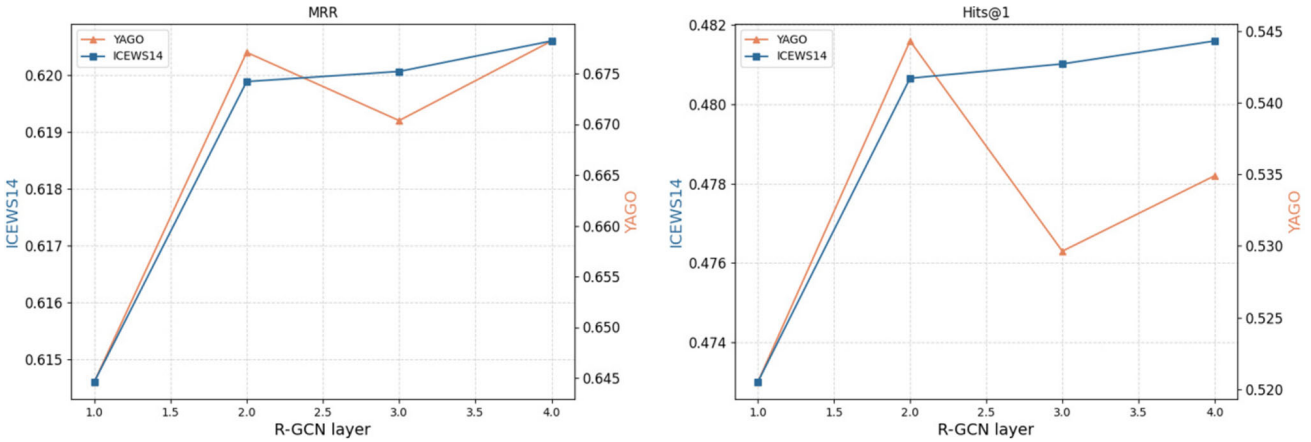



**Fig. 5** Effect of the number of R-GCN layers on the inference Performance of ICEWS14 and YAGO datasets

(1) Embedding size

To evaluate the impact of embedding dimensions on the model, we set the embedding sizes of entities and relationships to 16,32,64,100,200,300 and 400, respectively. Figure 4 shows the performance trend of $MRR$ and $Hits@10$ in terms of embedding dimensions. The results show that when the dimensions are small, the performance of the system continues to improve as the embedding dimension grows until it reaches a relatively stable state. When the embedding dimension increases to 300 and above, the performance is instead affected, which is due to the fact that too high a dimension not only takes a longer time to train but also leads to overfitting. Therefore, considering the performance and efficiency, we set the embedding dimension to 200.

(2) Layers of R-GCN

R-GCN is an important component of the evolutionary representation learning module. We examined the

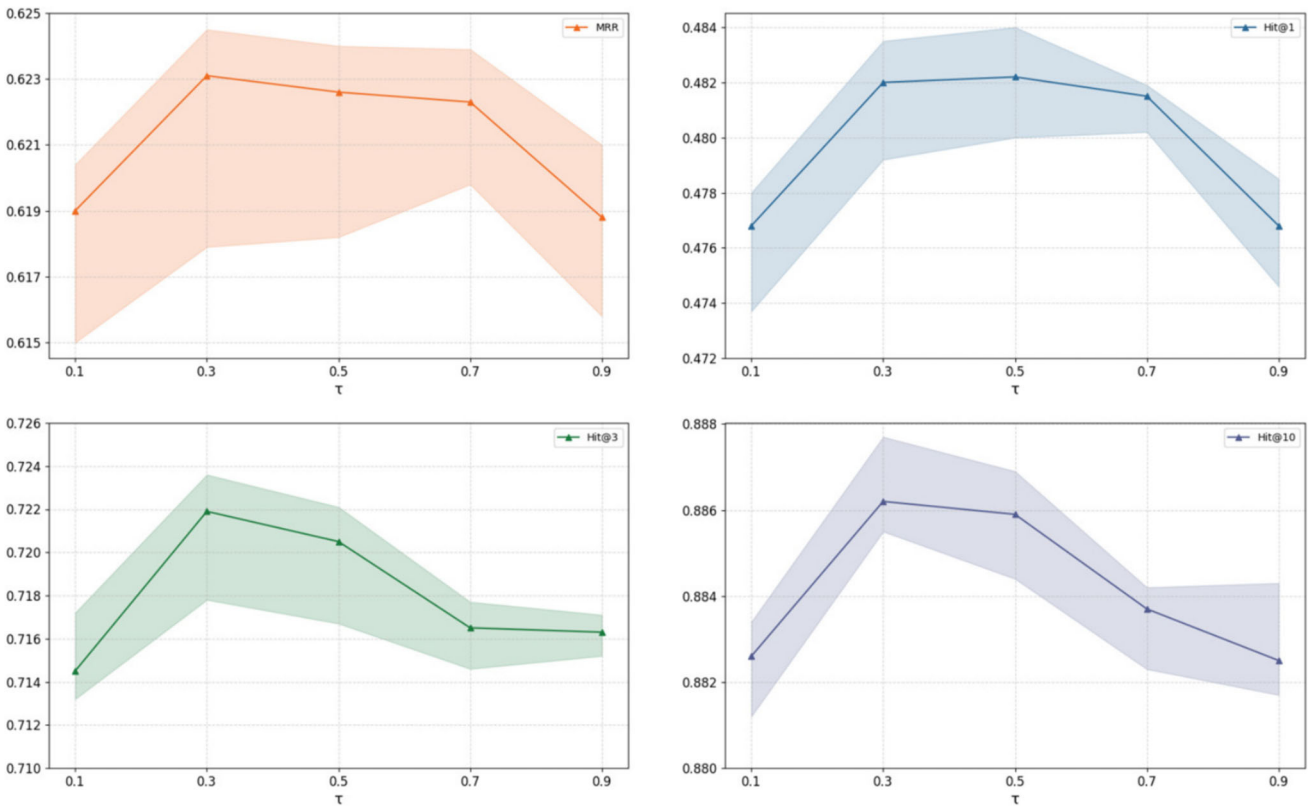



**Fig. 6** Effect of temperature coefficient $\tau$ on inference performance of ICEWS14 data set

performance of the model at different layer depths and the results are shown in Fig. 5. The results show that on the ICEWS14 dataset, the model performance increases with depth, but gains gradually diminish. On the YAGO dataset, it can be seen that the effect is better when the number of network layers is 2, and the experimental results are getting worse as the number of network layers increases. We analyze that it is mainly because the number of entities and relationships on the snapshots of the ICEWS14 dataset is very small during the experiment, and more information can be learned with a higher number of network layers. On the other hand, the information on a single snapshot of the YAGO dataset is richer, and the number of entities and relations is roughly four times the number of entities and relations on the ICEWS14 dataset, so fewer network layers can learn enough rich information, and too many network layers will lead to information overfitting instead, which is not conducive to the final inference results.

(3) Temperature coefficient $\tau$
The temperature parameter mainly regulates the degree of attention given to difficult samples. In the two more extreme cases, as it tends to 0, the contrast loss function degenerates into a loss function that focuses only on the difficult negative samples, and as it tends to infinity, all negative samples are processed under the loss of contrast, and the focusing characteristics of the difficult negative samples are lost. To verify which $\tau$ is more appropriate, we choose $\tau$ = 0.1,0.3,0.5,0.7,0.9; the experimental results are shown in Fig. 6, and the model works best at $\tau = 0.3$.

(4) Prototype $k$
In order to test the effect of aggregation category number $k$ on the model, we conducted experiments using different aggregation numbers. The experimental results are shown in Fig. 7. The results show that the model works best when = 20 and decreases to different degrees when $k$ is greater or less than 20. We analyze that this may be because too many aggregation categories increase the noise of the model, while too few aggregation categories are not enough to explore the potential semantic relationships in TKG.

(5) Relationship between temperature coefficient and number of prototypes
Typically, appropriate adjustment of the temperature coefficient can help balance the sensitivity of the model to positive and negative samples, while appropriate selection of the number of clustered prototypes can affect the modeling ability of the contrast loss function. As an important parameter in contrastive learning, we further analyzed its correlation on the ICEWS18 dataset, and the experimental results are shown in Fig. 8.

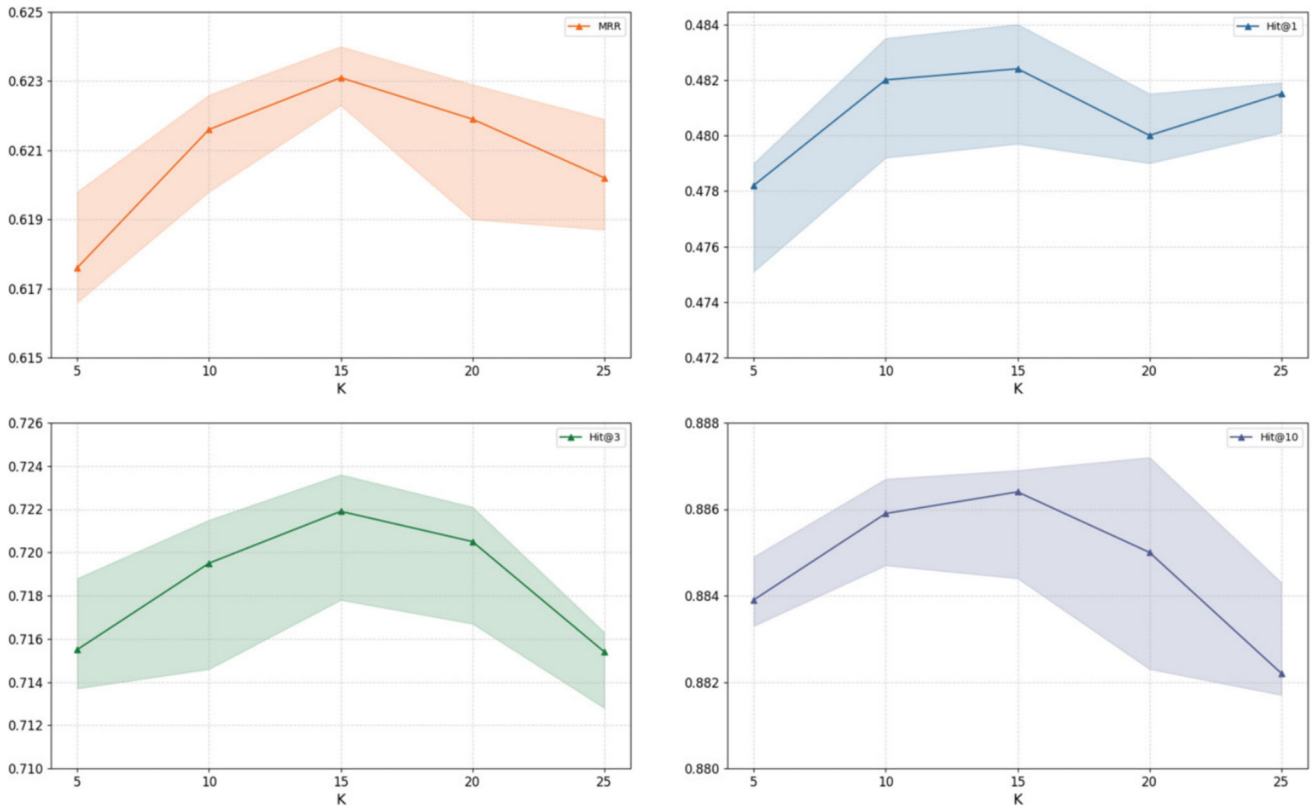



**Fig. 7** Impact of clustering prototype $k$ on inference performance of ICEWS14 dataset

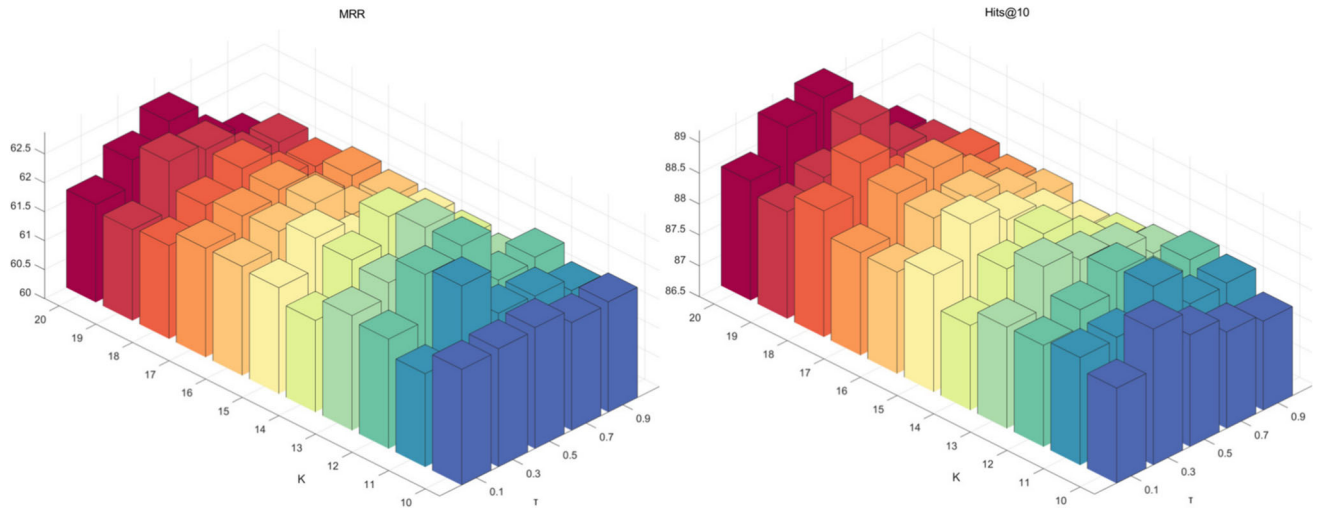



**Fig. 8** Impact of temperature coefficient and clustering prototype on inference performance

It can be seen that on this dataset, the variation of temperature coefficients has a greater impact on the experimental results, while the clustering prototypes have a smaller impact on the experimental results. We analyze the possible reason that the temperature coefficient focuses on the similarity or difference between the samples, and too much focus on the similarity or difference is not conducive to the experimental results, so the temperature coefficient of 0.5 has the best effect, while there are more entity and relationship categories on this dataset, and the number of clustering prototypes ranges from 10 to 20, which is relatively small in the magnitude of the change, and has little effect on the experimental results.

## 5 Conclusion and future work

In this paper, we focus on the temporal knowledge graph reasoning task. We propose a model called CH-TKG based on evolutionary representation and contrastive learning. The model first utilizes the embedding representation of entities and relations with evolutionary dependence in the evolutionary representation learning module. It then optimizes the representation of entities and relations under sparse data by using contrastive learning techniques. Finally, it uses the self-attention mechanism and the copy mechanism to learn the weights of historical information. The model achieves better results on the ICESW14 and ICEWS18 datasets with sparser data. However, it performs less well on the YAGO and WIKI datasets, which have sufficient data. On larger datasets, the computational complexity is higher, which leads to overfitting and poor inference. In addition, the data in real-world scenarios suffers from problems such as noise and incompleteness, which affects the modeling results. In future work, we will consider generating effective data augmentation for each temporal slice through data augmentation such as cropping, masking, and reordering to further enhance the representation of entities and relationships, and to improve the inference and prediction capabilities of temporal knowledge graphs.

**Acknowledgements** National Natural Science Foundation of China under Grant Nos. 62192731; Science Foundation of Young and Middle-aged Academic and Technical Leaders of Yunnan under Grant No. 202205AC160040; Science Foundation of Yunnan Jinzhi Expert Workstation under Grant No. 202205AF150006; Major Project of Yunnan Natural Science Foundation under Grant No. 202302AE09002003; Knowledge-driven Smart Energy Science and Technology Innovation Team of Yunnan Provincial Department of Education; Open Foundation of Yunnan Key Laboratory of Software Engineering under Grant No. 2023SE101.

**Author Contributions** • Conceptualization: [Qiuying Ma]; • Methodology: [Qiuying Ma]; • Formal analysis and investigation: [Qiuying Ma], [Xuan Zhang]; • Writing - original draft preparation: [Qiuying Ma]; • Writing - review and editing: [Xuan Zhang],[Chen Gao],[ZiShuo Ding],[Qiong Nong],[Yubin Ma]; • Funding acquisition: [Xuan Zhang]; • Resources: [Xuan Zhang],[Weiyi Shang],[Zhi Jin]; • Supervision: [Xuan Zhang]

**Data Availability** The data that support the findings of this study are openly available at https://github.com/mqygit/TKGs

## Declarations

**Competing of interest** The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## References


1. Chen X, Jia S, Xiang Y (2020) A review: Knowledge reasoning over knowledge graph. Expert Syst Appl 141:112948
2. Chen IY, Agrawal M, Horng S, Sontag D (2019) Robustly extracting medical knowledge from ehrs: a case study of learning a health knowledge graph. In: Pacific symposium on biocomputing 2020, pp 19–30. World Scientific
3. Jiang Z, Chi C, Zhan Y (2021) Research on medical question answering system based on knowledge graph. IEEE Access 9:21094–21101
4. Ahmed IA, AL-Aswadi FN, Noaman KM et al (2022) Arabic knowledge graph construction: A close look in the present and into the future. J King Saud University-Comput Inf Sci 34(9):6505–6523
5. Bordes A, Usunier N, Garcia-Duran A, Weston J, Yakhnenko O (2013) Translating embeddings for modeling multi-relational data. Advances in Neural Information Processing Systems 26
6. Jin W, Qu M, Jin X, Ren X (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)
7. Park N, Liu F, Mehta P, Cristofor D, Faloutsos C, Dong Y (2022) Evokg: Jointly modeling event time and network structure for reasoning over temporal knowledge graphs. In: Proceedings of the fifteenth ACM international conference on web search and data mining, pp 794–803
8. Li Z, Jin X, Li W, Guan S, Guo J, Shen H, Wang Y, Cheng X (2021) Temporal knowledge graph reasoning based on evolutional representation learning. In: Proceedings of the 44th international ACM SIGIR conference on research and development in information retrieval, pp 408–417
9. Lin Z, Tian C, Hou Y, Zhao WX (2022) Improving graph collaborative filtering with neighborhood-enriched contrastive learning. Proceedings of the ACM web conference 2022:2320–2329
10. Wang Z, Zhang J, Feng J, Chen Z (2014) Knowledge graph embedding by translating on hyperplanes. In: Proceedings of the AAAI conference on artificial intelligence, vol 28
11. Lin Y, Liu Z, Sun M, Liu Y, Zhu X (2015) Learning entity and relation embeddings for knowledge graph completion. In: Proceedings of the AAAI conference on artificial intelligence, vol 29
12. Nickel M, Tresp V, Kriegel H-P et al (2011) A three-way model for collective learning on multi-relational data. Icml 11:3104482–3104584
13. Bishan Yang and Wen-tau Yih and Xiaodong He and Jianfeng Gao and Li Deng (2014) embedding entities and relations for learning and inference in knowledge bases. In: International conference on learning representations
14. Trouillon T, Welbl J, Riedel S, Gaussier É, Bouchard G (2016) Complex embeddings for simple link prediction. In: International conference on machine learning, pp 2071–2080. PMLR
15. Bordes A, Glorot X, Weston J, Bengio Y (2014) A semantic matching energy function for learning with multi-relational data: Application to word-sense disambiguation. Mach Learn 94:233–259
16. Zhang S, Liu Y, Sun Y, Shah N (2021) Graph-less neural networks: Teaching old mlps new tricks via distillation. In: International conference on learning representations
17. Socher R, Chen D, Manning CD, Ng A (2013) Reasoning with neural tensor networks for knowledge base completion. Adv Neural Inf Process Syst 26
18. Schlichtkrull M, Kipf TN, Bloem P, Van Den Berg R, Titov I, Welling M (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, pp 593–607. Springer
19. Vashishth S, Sanyal S, Nitin V, Talukdar P (2019) Composition-based multi-relational graph convolutional networks. In: International conference on learning representations
20. Zhang M, Xia Y, Liu Q, Wu S, Wang L (2023) Learning long-and short-term representations for temporal knowledge graph reasoning. In: Proceedings of the ACM web conference vol 2023, pp 2412–2422
21. Garcia-Duran A, Dumančić S, Niepert M (2018) Learning sequence encoders for temporal knowledge graph completion. In: Proceedings of the 2018 conference on empirical methods in natural language processing, pp 4816–4821
22. Leblay J, Chekol MW (2018) Deriving validity time in knowledge graph. In: Companion proceedings of the the web conference vol 2018, pp 1771–1776
23. Kazemi SM, Goel R, Jain K, Kobyzev I, Sethi A, Forsyth P, Poupart P (2020) Representation learning for dynamic graphs: A survey. J Mach Learn Res 21(1):2648–2720
24. Zhu C, Chen M, Fan C, Cheng G, Zhang Y (2021) Learning from history: Modeling temporal knowledge graphs with sequential copy-generation networks. In: Proceedings of the AAAI conference on artificial intelligence vol 35, pp 4732–4740
25. Li Z, Jin X, Guan S, Li W, Guo J, Wang Y, Cheng X (2021) 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 and the 11th international joint conference on natural language processing (vol 1: Long Papers), pp 4732–4743
26. Wang J, Lin X, Huang H, Ke X, Wu R, You C, Guo K (2023) Glanet: temporal knowledge graph completion based on global and local information-aware network. Appl Intell, pp 1–17
27. Trivedi R, Dai H, Wang Y, Song L (2017) Know-evolve: deep temporal reasoning for dynamic knowledge graphs. In: International conference on machine learning, pp 3462–3471. PMLR
28. Sun H, Geng S, Zhong J, Hu H, He K (2022) Graph hawkes transformer for extrapolated reasoning on temporal knowledge graphs. In: Proceedings of the 2022 conference on empirical methods in natural language processing, pp 7481–7493
29. Bai L, Chai D, Zhu L (2023) Rlat: Multi-hop temporal knowledge graph reasoning based on reinforcement learning and attention mechanism. Knowl-Based Syst 269:110514
30. Zhang H, Bai L (2023) Few-shot link prediction for temporal knowledge graphs based on time-aware translation and attention mechanism. Neural Netw 161:371–381
31. Zhang D, Feng W, Wu Z, Li G, Ning B (2024) Cdrgn-sde: Cross-dimensional recurrent graph network with neural stochastic differential equation for temporal knowledge graph embedding. Expert Syst Appl 247:123295
32. Han Z, Ding Z, Ma Y, Gu Y, Tresp V (2021) Learning neural ordinary equations for forecasting future links on temporal knowledge graphs. In: Proceedings of the 2021 conference on empirical methods in natural language processing
33. You Y, Chen T, Sui Y, Chen T, Wang Z, Shen Y (2020) Graph contrastive learning with augmentations. Adv Neural Info Process Syst 33:5812–5823
34. Chen X, Fan H, Girshick R, He K (2020) Improved baselines with momentum contrastive learning. arXiv:2003.04297
35. Gao T, Yao X, Chen D (2021) Simcse: Simple contrastive learning of sentence embeddings. In: Proceedings of the 2021 conference on empirical methods in natural language processing
36. You Y, Chen T, Wang Z, Shen Y (2022) Bringing your own view: Graph contrastive learning without prefabricated data augmentations. In: Proceedings of the fifteenth ACM international conference on web search and data mining, pp 1300–1309

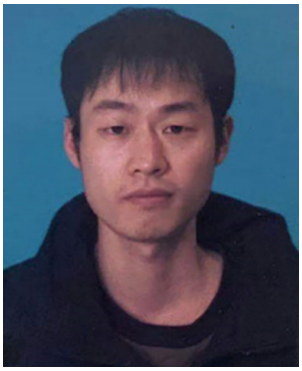
37. Chen Y, Liu Z, Li J, McAuley J, Xiong C (2022) Intent contrastive learning for sequential recommendation. In: Proceedings of the ACM web conference vol 2022, pp 2172–2182
38. Zhu Y, Xu Y, Yu F, Liu Q, Wu S, Wang L (2021) Graph contrastive learning with adaptive augmentation. In: Proceedings of the web conference vol 2021, pp 2069–2080
39. Sun F-Y, Hoffman J, Verma V, Tang J (2019) Infograph: Unsupervised and semi-supervised graph-level representation learning via mutual information maximization. In: International conference on learning representations
40. Xu M, Wang H, Ni B, Guo H, Tang J (2021) Self-supervised graph-level representation learning with local and global structure. In: International conference on machine learning, pp 11548–11558. PMLR
41. Wang L, Zhao W, Wei Z, Liu J (2022) Simkgc: Simple contrastive knowledge graph completion with pre-trained language models. In: Proceedings of the 60th annual meeting of the association for computational linguistics (vol 1: Long Papers). https://aclanthology.org/2022.acl-long.295
42. Zhang D, Rong Z, Xue C, Li G (2024) Simre: Simple contrastive learning with soft logical rule for knowledge graph embedding
43. Xu Y, Ou J, Xu H, Fu L (2023) Temporal knowledge graph reasoning with historical contrastive learning. In: Proceedings of the AAAI conference on artificial intelligence vol 37, pp 4765–4773
44. Mahdisoltani F, Biega J, Suchanek FM (2013) Yago3: A knowledge base from multilingual wikipedias. In: CIDR
45. Dettmers T, Minervini P, Stenetorp P, Riedel S (2018) Convolutional 2d knowledge graph embeddings. In: Proceedings of the AAAI conference on artificial intelligence, vol 32
46. Shang C, Tang Y, Huang J, Bi J, He X, Zhou B (2019) End-to-end structure-aware convolutional networks for knowledge base completion. In: Proceedings of the AAAI conference on artificial intelligence vol 33, pp 3060–3067
47. Sun Z, Deng Z-H, Nie J-Y, Tang J (2019) Rotate: Knowledge graph embedding by relational rotation in complex space. In: International conference on learning representations. https://openreview.net/forum?id=HkgEQnRqYQ
48. Dasgupta SS, Ray SN, Talukdar P (2018) Hyte: Hyperplane-based temporally aware knowledge graph embedding. In: Proceedings of the 2018 conference on empirical methods in natural language processing, pp 2001–2011
49. Li Z, Guan S, Jin X, Peng W, Lyu Y, Zhu Y, Bai L, Li W, Guo J, Cheng X (2022) Complex evolutional pattern learning for temporal knowledge graph reasoning. In: Proceedings of the 60th annual meeting of the association for computational linguistics (vol 2: Short Papers), pp 290–296




**Qiuying Ma** graduated from the School of Software of Yunnan University and obtained a master's degree in Cyber security in June 2024. Her research interests include Temporal Knowledge Graph Reasoning, Threat Intelligence Reasoning.

**Xuan Zhang** received the B.S. and M.S. degrees in computer science, and the Ph.D. degree in system analysis and integration from Yunnan University, Kunming, China. She is a professor with the School of Software, Yunnan University, Kunming, China. She is author of 4 books and more than 120 articles. She has been principal investigator for more than 30 national, provincial, and private grants and contracts. She is the core scientist of Yunnan Key Laboratory of Software Engineering and Yunnan Software Engineering Academic Team. Her research interests include knowledge graph, natural language processing, recommendation system, and computer vision. More information at http://www.sei.ynu.edu.cn/info/1023/1480.htm.

**ZiShuo Ding** is an Assistant Professor at the Hong Kong University of Science and Technology (Guangzhou). His research primarily revolves around the intersection of natural language processing and software engineering, with a focus on areas such as software performance engineering, software log analytics, and code representation learning. His work has been published in flagship conferences and journals including ICSE, FSE, ASE, TOSEM, and EMSE, and recognized with the SIGSOFT Distinguished Paper Award at ICSE 2020. More information at https://ece.uwaterloo.ca/~z8ding/.



**Chen Gao** received the B.S. and M.S. degrees in School of Software from East China University of Science and Yunnan University, China, in 2017 and 2021, respectively. He is currently pursuing the Ph.D. degree with the School of Information Science and Engineering, Yunnan University. His research interests include knowledge graphs, information extraction, and few-shot learning.

**Weiyi Shang** is an Associate Professor at the University of Waterloo. His research interests include AIOps, big data software engineering, software log analytics and software performance engineering. He serves as a Steering committee member of the SPEC Research Group. He is ranked top worldwide SE research stars in a recent bibliometrics assessment of software engineering scholars. He is a recipient of various premium awards, including the SIGSOFT Distinguished paper award at ICSE 2013 and ICSE 2020, best paper award at WCRE 2011 and the Distinguished reviewer award for the Empirical Software Engineering journal. His research has been adopted by industrial collaborators (e.g., BlackBerry and Ericsson) to improve the quality and performance of their software systems that are used by millions of users worldwide. Contact him at wshang@uwaterloo.ca https://ece.uwaterloo.ca/~wshang/

**Qiong Nong** graduated from the School of Software of Yunnan University and obtained a master's degree in software engineering in June 2024. Her research interests include natural language processing, Abstractive Summarization.

**Yubin Ma** graduated from the School of Software of Yunnan University and obtained a master's degree in software engineering at Yunnan University. His current research interests include recommender systems, graph neural network and knowledge graphs.

**Zhi Jin** (Fellow, IEEE) received the Ph.D. degree in computer science from the Changsha Institute of Technology, China, in 1992. She is a Full Professor of Computer Science with Peking University, where she is the Deputy Director of the Key Laboratory of High Confidence Software Technologies (Ministry of Education). She is/was the Principal Investigator of more than 15 national competitive grants. Her research interests include software engineering, requirements engineering, knowledge engineering, and machine learning. She is a Standing Board Member of China Computer Federation (CCF) and the Chair of CCF Technical Committee of System Software. She was elected as a fellow of CCF in 2012.

## Authors and Affiliations

**Qiuying Ma[1] · Xuan Zhang[1,2] · ZiShuo Ding[7] · Chen Gao[4] · Weiyi Shang[3] · Qiong Nong[1] · Yubin Ma[1] · Zhi Jin[5,6]**

✉ Xuan Zhang
zhxuan@ynu.edu.cn

1 School of Software, Yunnan University, Yunnan 650091, China

2 Yunnan Key Laboratory of Software Engineering, Yunnan 650091, China

3 Department of Electrical and Computer Engineering, University of Waterloo, Ontario N2L 3G1, Canada

4 School of Information Science and Engineering, Yunnan University, Yunnan 650091, China

5 School of Computer Science, Peking University, Beijing 10087, China

6 Key Lab. of High-Confidence Software Technologies (PKU), Ministry of Education, Beijing 10087, China

7 Data Science and Anaytics Thrust, Hong Kong University of Science and Technology (Guangzhou), Guangzhou, Guangdong 511442, China