HIP Network: Historical Information Passing Network for Extrapolation Reasoning on Temporal Knowledge Graph

Yongquan He[1,2] , **Peng Zhang**[1*] , **Luchen Liu**[1,2] , **Qi Liang**[1] , **Wenyuan Zhang**[1,2] and **Chuang Zhang**[1]

[1]Institute of Information Engineering, Chinese Academy of Sciences
[2]School of Cyber Security, University of Chinese Academy of Sciences
{heyongquan, pengzhang, liuluchen, liangqi, zhangwenyuan, zhangchuang}@iie.ac.cn

Abstract

In recent years, temporal knowledge graph (TKG) reasoning has received significant attention. Most existing methods assume that all timestamps and corresponding graphs are available during training, which makes it difficult to predict future events. To address this issue, recent works learn to infer future events based on historical information. However, these methods do not comprehensively consider the latent patterns behind temporal changes, to pass historical information selectively, update representations appropriately and predict events accurately. In this paper, we propose the **H**istorical **I**nformation **P**assing (HIP) network to predict future events. HIP network passes information from temporal, structural and repetitive perspectives, which are used to model the temporal evolution of events, the interactions of events at the same time step, and the known events respectively. In particular, our method considers the updating of relation representations and adopts three scoring functions corresponding to the above dimensions. Experimental results on five benchmark datasets show the superiority of HIP network, and the significant improvements on Hits@1 prove that our method can more accurately predict what is going to happen.

1 Introduction

Knowledge graphs (KGs) are multi-relational graphs, where nodes and various types of edges reflect entities and relations respectively. Each edge is presented as a triple of the form $(subject, relation, object)$, e.g., $(Obama, visit, China)$. Due to most of the KGs are far from complete, knowledge graph reasoning is proposed to infer missing facts [Dai *et al.*, 2020]. And this problem has been extensively studied for static KGs. The common way is to embed entities and relations into the continuous vector spaces, and computing a score for each triple by applying a scoring function to these embeddings [Bordes *et al.*, 2013; Yang *et al.*, 2015; Trouillon *et al.*, 2016]. However, facts may not always be true in the real world, which introduces the concept of the temporal knowledge graph (TKG). And each fact can be seen as a quadruple which takes the timestamp into consideration, i.e., $(subject, relation, object, timestamp)$. Reasoning on TKGs is more complex because of its native dynamic nature.

[*]Corresponding Author

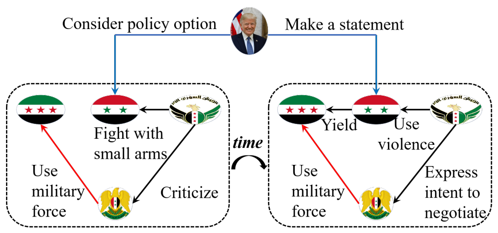


Figure 1: Temporal knowledge subgraphs about the evolution of political events in Syria and the Trump government's responses.

To solve this problem, previous works attempt to extend the static KG embedding methods which score the likelihood of missing facts with timestamps [Jiang *et al.*, 2016; Dasgupta *et al.*, 2018; García-Durán *et al.*, 2018; Goel *et al.*, 2020]. But these methods neglect the structure during learning the representations. So methods like TeMP [Wu *et al.*, 2020] are proposed recently. They focus on the evolving graph snapshots at multiple time steps for the dynamic representation and inference. However, they are not suitable for predicting future events, as the timestamps and the corresponding graphs are unknown. This problem is called **extrapolation reasoning**, which aims to predict new facts over future timestamps [Jin *et al.*, 2020]. Extrapolation reasoning over TKGs is less studied but significant, since forecasting emerging events is useful for real-world applications.

Recently, RE-NET [Jin *et al.*, 2020] and CyGNet [Zhu *et al.*, 2021] are proposed to tackle the extrapolation reasoning problem. RE-NET uses RNNs [Cho *et al.*, 2014] and RGCN-based aggregator [Schlichtkrull *et al.*, 2018] to encode historical information for future predictions. And CyGNet proposes a copy-generation mechanism, which generates future events based on the historical distribution. However, there are still several challenges not fully addressed. As the patterns associated with the occurrence of events are sophisticated, how to disentangle and model these patterns is useful and challenging. As shown in Figure 1, we can see the temporal evolution of events (blue edges), as policy options are often considered before making a statement. And at the same time

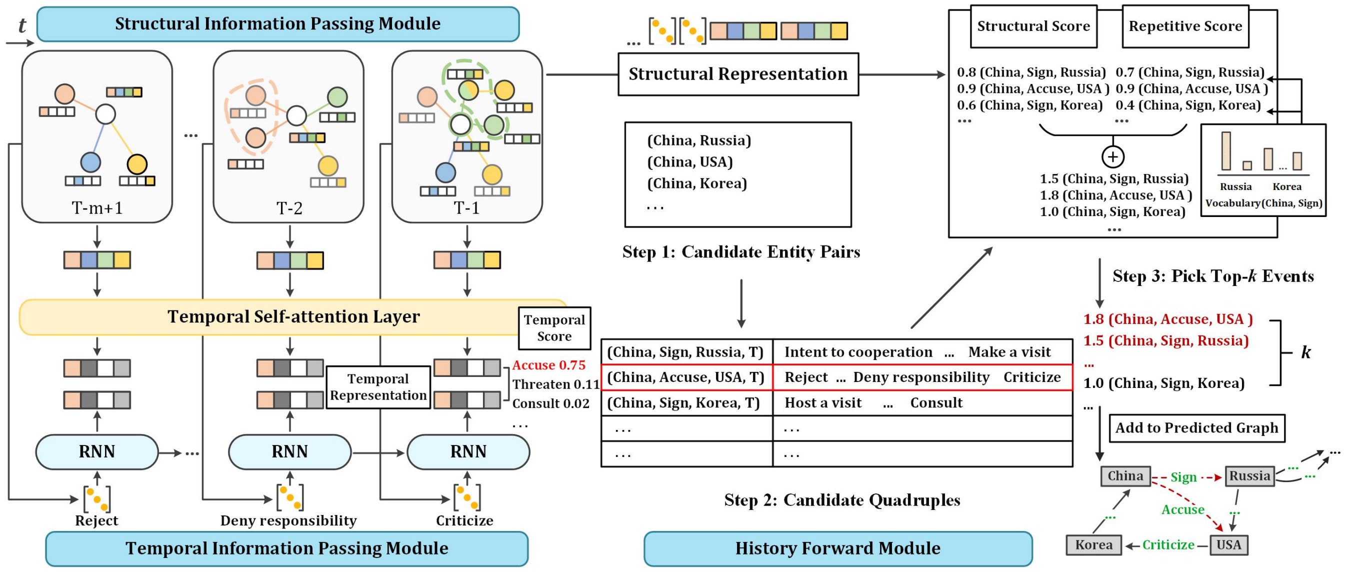

Structural Information Passing Module
T-m+1
T-2
T-1
Temporal Self-attention Layer
Temporal Score
Temporal Representation
Accuse 0.75
Threaten 0.11
Consult 0.02
RNN
Reject
Deny responsibility
Criticize
Temporal Information Passing Module
Structural Representation
(China, Russia)
(China, USA)
(China, Korea)
Step 1: Candidate Entity Pairs
(China, Sign, Russia, T) Intent to cooperation ... Make a visit
(China, Accuse, USA, T) Reject ... Deny responsibility Criticize
(China, Sign, Korea, T) Host a visit ... Consult
Step 2: Candidate Quadruples
History Forward Module
Structural Score
Repetitive Score
0.8 (China, Sign, Russia) 0.7 (China, Sign, Russia)
0.9 (China, Accuse, USA) 0.9 (China, Accuse, USA)
0.6 (China, Sign, Korea) 0.4 (China, Sign, Korea)
1.5 (China, Sign, Russia)
1.8 (China, Accuse, USA)
1.0 (China, Sign, Korea)
Russia Korea
Vocabulary(China, Sign)
Step 3: Pick Top-k Events
1.8 (China, Accuse, USA)
1.5 (China, Sign, Russia)
1.0 (China, Sign, Korea)
k
Add to Predicted Graph
China
Sign
Russia
Accuse
Korea
Criticize
USA

Figure 2: The framework of our HIP network. HIP network contains three components. The structural information passing module to capture neighborhood interactions. The temporal information passing module to model evolution patterns of events. And the HIP module to calculate the plausibility of events from the repetitive, structural and temporal perspectives.

step, the co-occurring events in a neighborhood structure also have dependencies and interactions (dotted lines). Moreover, events may occur repeatedly along the history (red edges). These phenomenons highlight the importance of learning different patterns to access more reference information. Besides, due to the dynamic nature of TKGs, only a part of entities are active at each time step. And the relations between entity pairs also change. Previous methods usually update the embeddings as static KG methods or only concentrate on the representations of entities, which can not fully learn the time-sensitive features for both entities and relations.

In this paper, we propose **H**istorical **I**nformation **P**assing (HIP) network, a novel framework for extrapolation reasoning on TKGs. In the structural information passing part, our method uses CompGCN [Vashishth *et al.*, 2020] to update the structural representations in a disentangled manner, which utilizes the relation embeddings and aggregates neighborhood information selectively. As for the temporal part, we generate the relation embeddings for specific entity pairs to capture the temporal evolution, and get the current embeddings for entities with the temporal self-attention mechanism. We also select object entities from history, to make reference to repetitive facts. Then HIP network adopts a multi-step reasoning algorithm with three score functions, to generate predictions in a sequential manner.

Our contributions are summarized as follows: 1)We propose a novel learning framework to handle the extrapolation reasoning problem on TKGs, which passes historical information from the temporal, structural and repetitive perspectives. 2)Our method specifically considers the updating of relation embeddings in the information passing process and proposes a multi-step reasoning algorithm with three scoring functions, which can infer future events sequentially. 3) The experimental results on five TKG datasets achieve state-of-the-art performance, and the significant improvements on Hits@1 prove the effectiveness of our method.

2 Related Work

Static KG embeddings. There are increasing interests in knowledge graph embedding (KGE) methods, which aim to embed entities (nodes) and relations (edges) into the continuous vector spaces, such as the translating models [Bordes *et al.*, 2013; Wang *et al.*, 2014] and the semantic matching models [Yang *et al.*, 2015; Trouillon *et al.*, 2016; Sun *et al.*, 2019]. And some other models score facts based on the deep neural network with feed-forward or convolutional layers [Dettmers *et al.*, 2018; Schlichtkrull *et al.*, 2018; Wang *et al.*, 2019; Vashishth *et al.*, 2020]. There are also works that use external information, such as typing information [Zhang *et al.*, 2018] and logic rules [He *et al.*, 2020]. More details can be found in recent surveys [Dai *et al.*, 2020]. However, these methods are not able to predict future events, as all entities and relations are treated as static.

Temporal KG reasoning. Recent works attempt to extend the static KG embeddings methods with temporal information. HyTE [Dasgupta *et al.*, 2018] projects the entities and the relations onto timestamp specific hyperplanes. DE-SimplE [Goel *et al.*, 2020] defines a function that takes an entity and a timestamp as input, to generate time-specific representations. And there are many other methods consider the timestamp to do the temporal reasoning [Jiang *et al.*, 2016; García-Durán *et al.*, 2018]. Another line of works try to capture changes in neighborhood structure, which use temporal recurrence and graph neural network together [Wu *et al.*, 2020; Sankar *et al.*, 2020]. However, the above methods are not suitable for extrapolation reasoning on TKGs, because the structure and timestamp of the future graph are unknown. To solve this problem, RE-NET [Jin *et al.*, 2020] and

CyGNet [Zhu *et al.*, 2021] are proposed, which are the most relevant methods to our work. RE-NET defines the joint probability distribution of all events in an autoregressive fashion. And CyGNet proposes a copy-generation mechanism, which makes predictions with the historical vocabulary.

3 Method

This section introduces the proposed model, named HIP network, for extrapolation reasoning on TKGs. We first define the task and notations, and give an overview of the model architecture. Then we detail the individual components in the following sections.

3.1 Task Definition and Model Architecture

Notations and task definition. A temporal knowledge graph (TKG) is composed of a sequence of time-stamped subgraphs, i.e., $\mathcal{G} = \{\mathcal{G}^{(1)}, \mathcal{G}^{(2)}, ..., \mathcal{G}^{(T)}\}$, where $\mathcal{G}^{(t)} = \{\mathcal{E}, \mathcal{R}, \mathcal{O}^t\}$. Here, $\mathcal{E}$ and $\mathcal{R}$ are known entities and relations across all time steps. And $\mathcal{O}^t$ is the events (edges) set at time t, each event is denoted as (s, r, o, t), where $s, o \in \mathcal{E}$ and $r \in \mathcal{R}$. Our task is to predict missing events about an object query $(s, r, ?, t + \Delta t)$ (or a subject query $(?, r, o, t + \Delta t)$), through using the historical information in $\{\mathcal{G}^{(t')}\}(t' \leq t)$, even if the events in time period Δt are unknown.

Model architecture. As shown in Figure 2, our model has three main components, namely structural information passing (SIP) module, temporal information passing (TIP) module, and history forward (HF) module. In the SIP module, we consider the interactions of co-occurring events in the latest graph through the CompGCN-based aggregator. We divide the representations into multiple independent components, and update structural embeddings for the entities and relations according to the attention weight of each component. And in the TIP module, we focus on capturing the temporal evolution patterns of the events between specific entity pairs and integrating structural embeddings across time. We selectively incorporate historical representations for entities with a temporal self-attention layer, and use RNN to model evolution process of events to generate the temporal embeddings for relations. Finally, in the HF module, we need to predict what events are likely to occur. We adopt three scoring functions to evaluate the events from different perspectives. Then we generate the predicted graph according to the reasoning results and move to the next time step.

3.2 Structural Information Passing Module

To capture the neighborhood interactions, we update the structural embeddings of entities and relations based on the recently historical graph $\mathcal{G}^{(t-1)}$. However, due to the streaming nature of the temporal knowledge graph, when new neighbors come or disappear, we need to update the related parts of embeddings while retaining useful information. So our method uses CompGCN [Vashishth *et al.*, 2020] in a disentangled manner to selectively pass structural information.

In this module, we aim to learn a disentangled embedding $\boldsymbol{x}_{s,t}$ for entity s. Firstly, we project $\boldsymbol{x}_{s,t-1}$ onto K spaces as follows:

$$\boldsymbol{h}_{s,k} = \boldsymbol{U}_k^T \boldsymbol{x}_{s,t-1}, \tag{1}$$

where $\boldsymbol{U}_k \in \mathrm{R}^{d_{in} \times \frac{d_{in}}{K}}$ is the parameter of channel k and K is the hyperparameter. Then we can obtain the node embeddings of K different components, i.e., $\boldsymbol{h}_s = \{\boldsymbol{h}_{s,1}, \boldsymbol{h}_{s,2}, ..., \boldsymbol{h}_{s,k}, ..., \boldsymbol{h}_{s,K}\}$.

Unlike most graph neural networks which embed only nodes in the graph, CompGCN uses embeddings of relations instead of matrices [Schlichtkrull *et al.*, 2018; Vashishth *et al.*, 2020]. So for each quadruple $(s, r, o, t-1) \in \mathcal{G}^{(t-1)}$, we also need to project the results of entity-relation composition operation onto the K spaces as follows:

$$\boldsymbol{c}_{o,k} = \boldsymbol{V}_k^T \phi(\boldsymbol{x}_{r,t-1}, \boldsymbol{x}_{o,t-1}), \tag{2}$$

where $\boldsymbol{V}_k \in \mathrm{R}^{d_{in} \times \frac{d_{in}}{K}}$ is the parameter of channel k. And $\phi(\boldsymbol{x}_r, \boldsymbol{x}_o)$ is the composition operation, such as subtraction, multiplication and circular-correlation. Then we can set K attention values, and the k-th attention value α_k indicates how related the message $\phi(\boldsymbol{x}_r, \boldsymbol{x}_o)$ is to the k-th component $\boldsymbol{h}_{s,k}$. The attention weight α_k is computed as:

$$\alpha_k = \frac{\exp(\mathrm{ReLU}(\boldsymbol{W}[\boldsymbol{c}_{o,k}; \boldsymbol{h}_{s,k}]))}{\sum_{k'=1}^{K} \exp(\mathrm{ReLU}(\boldsymbol{W}[\boldsymbol{c}_{o,k'}; \boldsymbol{h}_{s,k'}]))}, \tag{3}$$

where $\boldsymbol{W} \in \mathrm{R}^{1 \times \frac{2d_{in}}{K}}$ is a transformation matrix that can be trained. Then the update equation of the structural aggregator is given as:

$$\boldsymbol{x}_{s,k}^{l} = \sum_{(r,o') \in \mathcal{N}(s)} \alpha_k \boldsymbol{W}_{\lambda(r)}^{l,k} \boldsymbol{c}_{o',k}^{l-1}. \tag{4}$$

Here, $\boldsymbol{W}_{\lambda(r)}^{l,k} \in \mathrm{R}^{\frac{out}{K} \times \frac{in}{K}}$ is a relation-type parameter about the k-th component for the original, inverse and self-loop relations. Finally, We tile the K output embeddings $\boldsymbol{x}_{s,k}^{L}$ to obtain the structural representation $\boldsymbol{x}_{s,t}$ of entity s at time t, where L is the last layer. And as the CompGCN can consider the embeddings of relations in the aggregating process, we can also get the refined embedding $\boldsymbol{x}_{r,t}$ for relation r.

3.3 Temporal Information Passing Module

This module seeks to model the events evolution patterns between entity pairs and integrate information across time, to generate the temporal embeddings for entities and relations.

We assume that the embeddings from the SIP module sufficiently capture local structural information at each time step, which enables a disentangled modeling of structural and temporal information. For each entity s, the input is $\{\boldsymbol{x}_{s,t-m+1}, \boldsymbol{x}_{s,t-m+2}, ..., \boldsymbol{x}_{s,t}\}$, where m is the size of time window. We use a temporal self-attention layer to integrate the representations of entities in the temporal dimension, and the output embedding sequence for entity s is defined as $\{\boldsymbol{z}_{s,t-m+1}, \boldsymbol{z}_{s,t-m+2}, ..., \boldsymbol{z}_{s,t}\}$. We perform the scaled dot-product form of attention [Vaswani *et al.*, 2017] over the input embeddings at each time step, to generate the time-dependent embedding $\boldsymbol{z}_{s,t}$:

$$e_{ij} = \frac{((\boldsymbol{X}\boldsymbol{W}_q)(\boldsymbol{X}\boldsymbol{W}_k)^T)_{ij}}{\sqrt{d}} + \boldsymbol{M}_{ij}, \tag{5}$$

$$\beta_{ij} = \frac{\exp(e_{ij})}{\sum_{j'=0}^{m} \exp(e_{ij'})}, \tag{6}$$

$$\boldsymbol{Z} = \boldsymbol{\beta}(\boldsymbol{X}\boldsymbol{W}_v), \tag{7}$$

where $\boldsymbol{X} \in \mathrm{R}^{m\times d}$ and $\boldsymbol{Z} \in \mathrm{R}^{m\times d}$ are the input and output representations packed together across time. $\boldsymbol{\beta} \in \mathrm{R}^{m\times m}$ is the attention weight matrix obtained by the multiplicative attention function. And $\boldsymbol{W}_q \in \mathrm{R}^{d\times d}$, $\boldsymbol{W}_k \in \mathrm{R}^{d\times d}$ and $\boldsymbol{W}_v \in \mathrm{R}^{d\times d}$ are the linear projection matrices corresponding to the queries, keys, and values. To ensure that future information is not exposed, we use the mask matrix $\boldsymbol{M} \in \mathrm{R}^{m\times m}$ as

$$\boldsymbol{M}_{ij} = \begin{cases} 0, & i \leq j, \\ -\infty, & otherwise. \end{cases} \tag{8}$$

In this way, when $i > j$, the attention weight $\beta_{ij} \to 0$, which enables that only representations in the past are assigned non-zero weights.

As the SIP module is used to model the neighborhood interactions, the TIP module focuses more on fine-grained changes, i.e., the events (relations) that occur between specific entity pairs across the time steps. For each entity pair (s, o), we aim to predict what events may happen. So we use the sequence of relations between s and o, i.e., $\{r_{so}^0, r_{so}^1, ..., r_{so}^n, ..., r_{so}^N\}(1 \leq n \leq N)$, where N is the number of the events that occur in the time window. And relations are sorted by occurrence time. We use a gated recurrent unit (GRU) [Cho *et al.*, 2014] to get the relation transition representation for entity pair (s, o) as follows:

$$\boldsymbol{z}_{so,n} = \mathrm{GRU}(\boldsymbol{x}_{so,n}, \boldsymbol{z}_{so,n-1}), \tag{9}$$

where $\boldsymbol{x}_{so,n} \in \mathrm{R}^d$ is the structural representation of relation r_{so}^n at corresponding time. Noting that an entity pair can have multiple events at the same time step, we simply follow the order in the raw data set. And we take the output of the last hidden layer as the temporal relation representation $\boldsymbol{z}_{so,t}$.

3.4 History Forward Module

In this section, we introduce our scoring functions, which can evaluate an event from the structural, temporal and repetitive perspectives. And we proposed a multi-step inference algorithm, which can generate new graphs sequentially for future time step to handle the extrapolation reasoning problem.

The temporal scoring function is used to predict what events may occur. So we use the output from the TIP module as:

$$I_t(s, r, o, t) = \mathrm{softmax}(\boldsymbol{W}_t[\boldsymbol{z}_{s,t}; \boldsymbol{z}_{so,t}; \boldsymbol{z}_{o,t}])_r, \tag{10}$$

where $\boldsymbol{W}_t \in \mathrm{R}^{p\times 3d}$ and p is the number of relation types. In this way, we can focus on the temporal evolution of events between specific entity pairs to predict the next event.

The structural scoring function evaluates events based on the structural representations, to consider the interactions between them at the same time step. And borrowing scoring function from static KGE methods, we use DistMult [Yang *et al.*, 2015] to assign scores for the quadruples as follows:

$$I_s(s, r, o, t) = \sigma(\langle \boldsymbol{x}_{s,t}, \boldsymbol{x}_{r,t}, \boldsymbol{x}_{o,t}\rangle), \tag{11}$$

where σ is the sigmoid function. And $\langle\cdot\rangle$ denotes the tri-linear dot product. Through the above two scoring functions,

Algorithm 1 Reasoning algorithm of HIP network

Input: Historical graph sequence $\{\mathcal{G}^{(1)}, \mathcal{G}^{(2)}, ..., \mathcal{G}^{(t)}\}$, Query set $\mathcal{S}_{query}$ with object entities missing at time $t + \Delta t$.
Output: The reasoning results for each query in descending order of scores.

```
1:  $t' = t + 1$.
2:  while $t' < t + \Delta t$ do
3:      Structural embeddings for entities and relations. ▷SIP
4:      Temporal embeddings for entities. ▷TIP
5:      for each $(s, r, ?, t + \Delta t) \in \mathcal{S}_{query}$ do
6:          Generate candidate entity pair set $\mathcal{S}_{ep}^{t'}(s)$.
7:          Generate candidate quadruple set $\mathcal{S}_q^{t'}(s)$. ▷Eq. 10
8:          Pick top-$k$ quadruples in $\mathcal{S}_q^{t'}(s)$ and add them to
            $\mathcal{G}^{(t')}$. ▷Eq. 11 and 12
9:      end for
10:     Add $\mathcal{G}^{(t')}$ to the graph sequence and update the histor-
        ical vocabulary.
11: end while
12: Update structural embeddings.
13: Replace the missing part with all entities for each query
    and compute scores for them. ▷Eq. 11 and 12
```

our method can consider the evolution patterns of events between entity pairs across time and neighborhood interactions at the same time step.

However, the existence of entities and events is time-sensitive, which may lead to less information available for some entities in the time window. And based on the observation that many facts occur repeatedly along the history [Zhu *et al.*, 2021], we can generate new events from the historical events selectively to solve this problem. In our method, we use historical vocabulary for each entity and relation. For a query $(s, r, ?, t)$, we compute the repetitive history score as follows:

$$I_h(s, r, o, t) = \mathrm{softmax}(\boldsymbol{W}_h[\boldsymbol{e}_{s,t}; \boldsymbol{e}_{r,t}] + \boldsymbol{V}_t^{(s,r)})_o, \tag{12}$$

$$\boldsymbol{V}_t^{(s,r)} = \boldsymbol{v}_1^{(s,r)} + \boldsymbol{v}_2^{(s,r)} + ... + \boldsymbol{v}_{t-1}^{(s,r)}, \tag{13}$$

where $\boldsymbol{W}_h \in \mathrm{R}^{q\times 2d}$ and q is the number of entities. $\boldsymbol{e}_{s,t} \in \mathrm{R}^d$ and $\boldsymbol{e}_{r,t} \in \mathrm{R}^d$ are representations of entities and relations to model the preference of the history, which are independent of time and structure. And $\boldsymbol{v}_{t^-}^{(s,r)}$ is an q-dimensional multi-hot indicator vector for subject s and relation r, and position o is 1 represents that (s, r, o, t^-) exists in graph $\mathcal{G}^{(t^-)}$.

Now our goal is to handle the extrapolation reasoning problem with these scoring functions. And the reasoning process during testing is shown in Algorithm 1 and Figure 2. At each time step, the first thing is to update representations using the SIP and TIP module (line 3-4). But we compute the temporal embeddings for relations later. Then the reasoning for queries can be divided into three main steps (line 5-10).

Step 1: When given a query $(s, r, ?, t + \Delta t)$ and current time is t', we don't take all entities in $\mathcal{E}$. We generate the **candidate entity pair set** $\mathcal{S}_{ep}^{t'}(s)$ by searching for all entities that had events with s, to reduce noise and computed space.

Dataset	Entities	Relations	Training	Validation	Test	Time gap
YAGO	10,623	10	161,540	19,523	20,026	1 year
WIKI	12,554	24	539,286	67,538	63,110	1 year
ICEWS14	12,498	260	323,895	-	341,409	1 day
ICEWS18	23,033	256	373,018	45,995	49,545	1 day
GDELT	7,691	240	1,734,399	238,765	305,241	15 mins

Table 1: Statistics of five datasets.

Step 2: Our method focuses on temporal information in this step. We get the temporal embeddings for relations and use equation 10 to find the most likely event for each entity pair in $\mathcal{S}_{ep}^{t'}(s)$. We put the event with the highest score of each entity pair into **candidate quadruple** set $\mathcal{S}_{q}^{t'}(s)$. This step enables our approach to incorporate temporal evolution, which can avoid predicting the same result at every step.

Step 3: We use the sum of equation 11 and 12 to evaluate the plausibility of quadruples in $\mathcal{S}_{q}^{t'}(s)$, which consider the preferences of current structure and history. We choose the top-k quadruples for each query, add them to the predicted graph $\mathcal{G}^{(t')}$, and then move on to the next time step $t'+1$.

To answer the queries, we use the sum of equation 11 and 12 to assign scores for each query with the missing part completed at target time step $t+\Delta t$ (line 13), and rank them in descending order.

3.5 Training Objective

As the query $(s,r,?,t)$ for an object entity can be seen as a multi-class classification problem, where each class corresponds to each object entity. And if we need to predict what events will happen between entity pairs, we can also regard this as a multi-classification problem for relation types. So the loss function in each time step can be defined as follows:

$$\mathcal{L} = \sum_{(s,r,o,t)\in\mathcal{G}^{(t)}} \sum_{*} -\log(I_*(s,r,o,t)). \tag{14}$$

Here, $I_*(\cdot)$ are the scoring functions defined in the previous section, i.e., equation 10, 11 and 12.

4 Experiments

In this section, we evaluate our proposed framework, HIP network, on five public datasets.

4.1 Experimental Setup

Datasets. We use five TKG datasets, namely ICEWS14 [Trivedi *et al.*, 2017], ICEWS18 [Boschee *et al.*, 2015], GDELT [Leetaru and Schrodt, 2013], WIKI [Leblay and Chekol, 2018] and YAGO [Mahdisoltani *et al.*, 2015]. ICEWS14, ICEWS18 and GDELT are event-based TKGs which record events with timestamps. WIKI and YAGO are subsets of Wikipedia history and YAGO3 respectively. And we preprocess these datasets for extrapolation reasoning task as prior works [Jin *et al.*, 2020; Zhu *et al.*, 2021]: we split them into three subsets by timestamps except ICEWS14, i.e., train(80%)/valid(10%)/test(10%). Thus, (timestamps of train) $<$ (timestamps of valid) $<$ (timestamps of test). More details about the five datasets can be found in Table 1.

Evaluation metrics. We evaluate our method on the link prediction task which evaluates whether the ground-truth entity is ranked ahead of other entities. We report the results of mean reciprocal rank (MRR), hits at 1/3/10 (Hits@1/3/10) in our experiments. And we remove corrupted entities as other baselines during evaluation which is called $filtered$ setting.

Baselines. We mainly focus on comparing to the methods of static KGs and temporal graphs as prior works. Static KG learning methods include TransE [Bordes *et al.*, 2013], DistMult [Yang *et al.*, 2015], ComplEx [Trouillon *et al.*, 2016], ConvE [Dettmers *et al.*, 2018], RotatE [Sun *et al.*, 2019], RGCN-DistMult [Schlichtkrull *et al.*, 2018], and CompGCN-DistMult [Vashishth *et al.*, 2020]. And temporal reasoning methods include TTransE [Jiang *et al.*, 2016], HyTE [Dasgupta *et al.*, 2018], TA-DistMult [García-Durán *et al.*, 2018], DySAT [Sankar *et al.*, 2020], TeMP [Wu *et al.*, 2020], RE-NET [Jin *et al.*, 2020] and CyGNet [Zhu *et al.*, 2021]. Since RE-NET and CyGNet are the most relevant methods for extrapolation reasoning, we will do more analysis on them.

4.2 Results on TKGs

In this task, we finetune the hyperparameters according to the MRR performance on each validation set. Since ICEWS14 is not provided with a validation set, we use the same settings in ICEWS18. And to be consistent with the baselines, most hyperparameters are the same on all datasets. We use the ADAM optimizer with a learning rate of 0.001 to minimize the global loss. The embedding dimension is 200. The dropout rate is 0.5. The batch size is 1024. And the time window size is set to 10. Additionally, in our SIP module, we use multiplication as the composition operation and adopt 4 channels. For the static KG methods, we simply remove all timestamps in datasets.

As Table 2 shows, our HIP network outperforms all the baselines. All static methods perform worse than our method since they do not consider temporal factors. And TTransE, HyTE, TA-DistMult and DySAT don't even work as well as some static methods. RE-NET, TeMP, CyGNet and our method outperform other methods by a large margin. TeMP is not primarily proposed to solve the extrapolation reasoning problem, but it's also trained along history, which proves that modeling historical evolution patterns can better deal with this problem. And compared to the best baseline CyGNet, our improvements on MRR and Hits@1 are more significant than Hits@10. This may be because CyGNet generates predictions based on the historical distribution, which can narrow the scope of predictions. And some entities never appear in the training set, which also makes it difficult to improve Hits@10. The improvements on Hits@1 and MRR fully demonstrate that our method is able to utilize historical information more effectively, to predict exactly what is going to happen.

4.3 Ablation Study

We conduct ablation experiments on the WIKI and ICEWS14 datasets. Firstly, we only use the SIP module and score the events through equation 11. Then we consider historical vocabulary, i.e., equation 12. In the third setting, we combine historical vocabulary with structural representations to

	YAGO				WIKI				ICEWS14				ICEWS18				GDELT			
Method	MRR	Hits@1	Hits@3	Hits@10	MRR	Hits@1	Hits@3	Hits@10	MRR	Hits@1	Hits@3	Hits@10	MRR	Hits@1	Hits@3	Hits@10	MRR	Hits@1	Hits@3	Hits@10
TransE	48.97	46.23	62.45	66.05	46.68	36.19	49.71	51.71	18.65	1.12	31.34	47.07	17.56	2.48	26.95	43.87	16.05	0.00	26.10	42.29
DistMult	59.47	52.97	60.91	65.26	46.12	37.24	49.81	51.38	19.06	10.09	22.00	36.41	22.16	12.13	26.00	42.18	18.71	11.59	20.05	32.55
ComplEx	61.29	54.88	62.28	66.82	47.84	38.15	50.08	51.39	24.47	16.13	27.49	41.09	30.09	21.88	34.15	45.96	22.77	15.77	24.05	36.33
ConvE	62.32	56.19	63.97	65.60	47.57	38.76	50.10	50.53	40.73	33.20	43.92	54.35	36.67	28.51	39.80	50.69	35.99	27.05	39.32	49.44
RotatE	65.09	57.13	65.67	66.16	50.67	40.88	50.71	50.88	29.56	22.14	32.92	42.68	23.10	14.33	27.61	38.72	22.33	16.68	23.89	32.29
RGCN	41.30	32.56	44.44	52.68	37.57	28.15	39.66	41.90	26.31	18.23	30.43	45.34	23.19	16.36	25.34	36.48	23.31	17.24	24.96	34.36
CompGCN	41.42	32.63	44.59	52.81	37.64	28.33	39.87	42.03	26.46	18.38	30.64	45.61	23.31	16.52	25.37	36.61	23.46	16.65	25.54	34.58
TTransE	32.57	27.94	43.39	53.37	31.74	22.57	36.25	43.45	6.35	1.23	5.80	16.65	8.36	1.94	8.71	21.93	5.52	0.47	5.01	15.27
HyTE	23.16	12.85	45.74	51.94	43.02	34.29	45.12	49.49	11.48	5.64	13.04	22.51	7.31	3.10	7.50	14.95	6.37	0.00	6.72	18.63
TA-DistMult	61.72	52.98	63.32	65.19	48.09	38.71	49.51	51.70	20.78	13.43	22.80	35.26	28.53	20.30	31.57	44.96	29.35	22.11	31.56	41.39
DySAT	43.43	31.87	43.67	46.49	31.82	22.07	26.59	35.59	18.74	12.23	19.65	21.17	19.95	14.42	23.67	26.67	23.34	14.96	22.57	27.83
TeMP	62.25	55.39	64.63	66.12	49.61	46.96	50.24	51.81	43.13	35.67	45.79	56.12	40.48	33.97	42.63	52.38	37.56	29.82	40.15	48.60
RE-NET	65.16	63.29	65.63	68.08	51.97	48.01	52.07	53.91	45.71	38.42	49.06	59.12	42.93	36.19	45.47	55.80	40.12	32.43	43.40	53.80
CyGNet	66.58	64.26	67.98	70.16	52.60	50.48	53.26	55.82	49.89	43.15	53.68	61.18	47.83	42.02	50.71	57.72	51.06	44.66	54.74	61.32
HIP network	**67.55**	**66.32**	**68.49**	**70.37**	**54.71**	**53.82**	**54.73**	**56.46**	**50.57**	**45.73**	**54.28**	**61.65**	**48.37**	**43.51**	**51.32**	**58.49**	**52.76**	**46.35**	**55.31**	**61.87**

Table 2: Performance comparison on temporal link prediction.

	WIKI				ICEWS14			
Metric	MRR	Hits@1	Hits@3	Hits@10	MRR	Hits@1	Hits@3	Hits@10
SIP only	48.25	39.17	50.36	52.11	31.81	26.17	33.65	48.59
Vocabulary only	52.89	50.62	53.39	55.96	49.14	44.23	52.27	58.92
Vocabulary and SIP	54.48	53.64	53.81	56.03	49.47	45.02	52.42	59.13
HIP network	**54.71**	**53.82**	**54.73**	**56.46**	**50.57**	**45.73**	**54.28**	**61.65**

Table 3: Effectiveness of each component on WIKI and ICEWS14.

measure the plausibility of events. Finally, we evaluate our method with all components according to Algorithm 1.

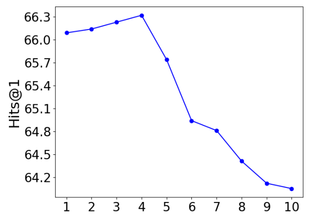
As shown in Table 3, under the first setting, our method still performs better than most static KGE models, especially CompGCN. This proves the usefulness of our SIP module which updates representations in a disentangled manner, as changes often occur in the part of the neighborhood structure on TKGs. And in the second setting, scoring with equation 12, our model can achieve better results, which demonstrates the importance of selectively incorporating historical information for each query. When we combine the two settings together, the result on all metrics still improves. As we consider the evolution patterns about events between entity pairs, HIP network achieves the best results by using the multi-step reasoning process. In particular, the improvements in the purely event-based dataset (ICEWS14) are more significant, this may be because events in the ICEWS14 have more evolution patterns (the time gap in ICEWS14 is only 24 hours), which proves the importance of considering the temporal evolution of events.

4.4 Sensitivity Analysis

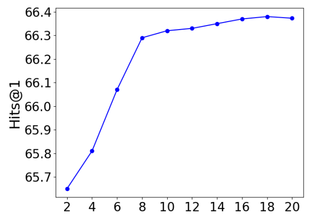
In this section, we investigate the influence of hyperparameters on YAGO.

Number of channels in the structural aggregator. To update the related parts of embeddings in the aggregated process, we project the embeddings and the results of composition operation onto K subspaces. We consider the value of K from 1 to 10. As Figure 3(a) shows, our model achieves the best result when K is set to 4. And our model is sensitive to K especially when K is larger than 4. Since the weight of each subspace is calculated in a similar way to the multi-head attention mechanism, large K results in a smaller subspace dimension. So we need to choose K more carefully according to the embedding dimension.

Length of history in the temporal part. The input of the TIP module is the information in the time window. In detail,



(a) Number of channels



(b) Length of history

Figure 3: Parameter sensitivity on HIP network.

we take the sequence of events up to m between each entity pair. Note that there may be multiple events between each entity pair at the same time step, we take the m events that have recently occurred in the order of the original dataset. And we simply remove the above events that do not occur in the time window. So m is the limit on the number and scope of events. And in the temporal self-attention layer, we only consider the embeddings within the time window. Figure 3(b) shows that the longer the history is considered, the higher the Hits@1. When the history length is around 8, Hits@1 starts to be relatively stable. And a large time window may result in a long sequence of events in RNN. Therefore, to simplify the computation and save time, we don't need to set m too large.

5 Conclusions

In this paper, we propose HIP network to solve the extrapolation reasoning problem on TKGs. We pass historical information from the temporal, structural and repetitive perspectives to make future predictions. And the proposed HIP network not only considers the important role of relations in the information passing process, but also evaluates the plausibility of an event from above perspectives, which can effectively model the evolution patterns, neighborhood interactions, and historical repetition. Experimental results show that HIP network achieves improvements over state-of-the-art baselines.

Acknowledgments

This work was supported by the National Key R&D Program with No.2016QY03D0503,2016YFB081304, and Strategic Priority Research Program of Chinese Academy of Sciences, Grant No.XDC02040400.

References

[Bordes *et al.*, 2013] Antoine Bordes, Nicolas Usunier, Alberto García-Durán, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. In *NIPS*, pages 2787–2795, 2013.

[Boschee *et al.*, 2015] Elizabeth Boschee, Jennifer Lautenschlager, Sean O'Brien, Steve Shellman, James Starz, and Michael Ward. Icews coded event data. *Harvard Dataverse*, 12, 2015.

[Cho *et al.*, 2014] Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *EMNLP*, pages 1724–1734, 2014.

[Dai *et al.*, 2020] Yuanfei Dai, Shiping Wang, Neal N. Xiong, and Wenzhong Guo. A survey on knowledge graph embedding: Approaches, applications and benchmarks. *Electronics*, 9(5):750, 2020.

[Dasgupta *et al.*, 2018] Shib Sankar Dasgupta, Swayambhu Nath Ray, and Partha P. Talukdar. Hyte: Hyperplane-based temporally aware knowledge graph embedding. In *EMNLP*, pages 2001–2011, 2018.

[Dettmers *et al.*, 2018] Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. Convolutional 2d knowledge graph embeddings. In *AAAI*, pages 1811–1818, 2018.

[García-Durán *et al.*, 2018] Alberto García-Durán, Sebastijan Dumancic, and Mathias Niepert. Learning sequence encoders for temporal knowledge graph completion. In *EMNLP*, pages 4816–4821, 2018.

[Goel *et al.*, 2020] Rishab Goel, Seyed Mehran Kazemi, Marcus Brubaker, and Pascal Poupart. Diachronic embedding for temporal knowledge graph completion. In *AAAI*, pages 3988–3995, 2020.

[He *et al.*, 2020] Yongquan He, Zhihan Wang, Peng Zhang, Zhaopeng Tu, and Zhaochun Ren. VN network: Embedding newly emerging entities with virtual neighbors. In *CIKM*, pages 505–514, 2020.

[Jiang *et al.*, 2016] Tingsong Jiang, Tianyu Liu, Tao Ge, Lei Sha, Baobao Chang, Sujian Li, and Zhifang Sui. Towards time-aware knowledge graph completion. In *COLING*, pages 1715–1724, 2016.

[Jin *et al.*, 2020] Woojeong Jin, Meng Qu, Xisen Jin, and Xiang Ren. Recurrent event network: Autoregressive structure inferenceover temporal knowledge graphs. In *EMNLP*, pages 6669–6683, 2020.

[Leblay and Chekol, 2018] Julien Leblay and Melisachew Wudage Chekol. Deriving validity time in knowledge graph. In *WWW*, pages 1771–1776, 2018.

[Leetaru and Schrodt, 2013] Kalev Leetaru and Philip A Schrodt. Gdelt: Global data on events, location, and tone. In *ISA Annual Convention*, 2013.

[Mahdisoltani *et al.*, 2015] Farzaneh Mahdisoltani, Joanna Biega, and Fabian M. Suchanek. YAGO3: A knowledge base from multilingual wikipedias. In *CIDR*, 2015.

[Sankar *et al.*, 2020] Aravind Sankar, Yanhong Wu, Liang Gou, Wei Zhang, and Hao Yang. Dysat: Deep neural representation learning on dynamic graphs via self-attention networks. In *WSDM*, pages 519–527, 2020.

[Schlichtkrull *et al.*, 2018] Michael Sejr Schlichtkrull, Thomas N. Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. Modeling relational data with graph convolutional networks. In *ESWC*, volume 10843, pages 593–607, 2018.

[Sun *et al.*, 2019] Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. Rotate: Knowledge graph embedding by relational rotation in complex space. In *ICLR*, 2019.

[Trivedi *et al.*, 2017] Rakshit Trivedi, Hanjun Dai, Yichen Wang, and Le Song. Know-evolve: Deep temporal reasoning for dynamic knowledge graphs. In *ICML*, volume 70, pages 3462–3471, 2017.

[Trouillon *et al.*, 2016] Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. Complex embeddings for simple link prediction. In *ICML*, volume 48, pages 2071–2080, 2016.

[Vashishth *et al.*, 2020] Shikhar Vashishth, Soumya Sanyal, Vikram Nitin, and Partha P. Talukdar. Composition-based multi-relational graph convolutional networks. In *ICLR*, 2020.

[Vaswani *et al.*, 2017] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NIPS*, pages 5998–6008, 2017.

[Wang *et al.*, 2014] Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. Knowledge graph embedding by translating on hyperplanes. In *AAAI*, pages 1112–1119, 2014.

[Wang *et al.*, 2019] Zihan Wang, Zhaochun Ren, Chunyu He, Peng Zhang, and Yue Hu. Robust embedding with multi-level structures for link prediction. In *IJCAI*, pages 5240–5246, 2019.

[Wu *et al.*, 2020] Jiapeng Wu, Meng Cao, Jackie Chi Kit Cheung, and William L. Hamilton. Temp: Temporal message passing for temporal knowledge graph completion. In *EMNLP*, pages 5730–5746, 2020.

[Yang *et al.*, 2015] Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. Embedding entities and relations for learning and inference in knowledge bases. In *ICLR*, 2015.

[Zhang *et al.*, 2018] Zhao Zhang, Fuzhen Zhuang, Meng Qu, Fen Lin, and Qing He. Knowledge graph embedding with hierarchical relation structure. In *EMNLP*, pages 3198–3207, 2018.

[Zhu *et al.*, 2021] Cunchao Zhu, Muhao Chen, Changjun Fan, Guangquan Cheng, and Yan Zhan. Learning from history: Modeling temporal knowledge graphs with sequential copy-generation networks. In *AAAI*, 2021.