Sequential and Repetitive Pattern Learning for Temporal Knowledge Graph Reasoning

Xuefei Li, Huiwei Zhou*, Weihong Yao, Wenchu Li, Yingyu Lin, Lei Du
Dalian University of Technology
No.2 Linggong Road, Ganjingzi District, Dalian, Liaoning, China
{lixuefei,GraceLee}@mail.dlut.edu.cn
{zhouhuiwei, weihongy, lyydut, dulei}@dlut.edu.cn

Abstract

Temporal Knowledge Graph (TKG) reasoning has received a growing interest recently, especially in forecasting the future facts based on the historical KG sequences. Existing studies typically utilize a recurrent neural network to learn the evolutional representations of entities for temporal reasoning. However, these methods are hard to capture the complex temporal evolutional patterns such as sequential and repetitive patterns accurately. To tackle this challenge, we propose a novel Sequential and Repetitive Pattern Learning (SRPL) method, which comprehensively captures both the sequential and repetitive patterns. Specifically, a Dependency-aware Sequential Pattern Learning (DSPL) component expresses the temporal dependencies of each historical timestamp as embeddings for accurately capturing the sequential patterns across temporally adjacent facts. A Time-interval guided Repetitive Pattern Learning (TRPL) component models the irregular time intervals between historical repetitive facts for capturing the repetitive patterns. Extensive experiments on four representative benchmarks demonstrate that our proposed method outperforms state-of-the-art methods in all metrics by an obvious margin, especially on GDELT dataset, where performance improvement of MRR reaches up to 18.84%.

Keywords: temporal knowledge graph, temporal dependency, sequential pattern, repetitive pattern

1. Introduction

Knowledge Graphs (KGs) have greatly promoted the development of practical applications, such as electronic health records analysis, disaster monitoring and financial forecasting (Zou, 2020). However, large scale KGs often suffer from incompleteness and therefore limit the performance of downstream applications. To address this issue, KG reasoning that predicts missing facts from known facts in the existing KGs is a crucial task in the field of natural language processing. Prior KG reasoning research traditionally focused on modeling static features based on static KGs (Yang et al., 2014). While the facts in KGs are dynamically evolving over time, how to take advantage of the historical information over Temporal Knowledge Graph (TKG) has become an urgent demand (Gottschalk and Demidova, 2018; Wang et al., 2023).

TKG can be regarded as a series of static KG snapshots in the chronological order. Each fact in the TKG is stored as quadruple (*subject, relation, object, timestamp*) (i.e. (s, r, o, t)), for example, (*Tokyo Olympic Games Women's Table Tennis Singles, Gold Medalist, Meng Chen, 2021/07/29*) indicates the fact that Meng Chen won the gold medal of Tokyo Olympic Games women's table tennis singles in 2021.

Given a TKG with timestamps varying from t_1 to t_T, reasoning over it can be classified into two settings, *interpolation* and *extrapolation* (Jin et al., 2019). *Interpolation* reasoning aims at predicting missing facts within $[t_1, t_T]$, while *extrapolation* reasoning attempts to forecast facts in future timestamp $t > t_T$.The *extrapolation* setting is of greater significance as it is helpful for many practical applications, such as crisis warning (Boschee et al., 2015) and disaster relief (Signorini et al., 2011). This paper focuses on entity prediction under the *extrapolation* setting over TKGs, which aims to predict the missing object entity of a query $(s, r, ?, t + 1)$(e.g., (*Paris Olympic Games Women's Table Tennis Singles, Gold Medalist, ?, 2024*)).

TKG reasoning is quite challenging due to the complex temporal dependencies, which is difficult to be captured from temporal sequences. Great efforts have been made on mining sequential patterns of dependencies in previous studies (Li et al., 2021b). They usually take advantage of recurrent encoder networks to model the historical dependency of events over timeline (Jin et al., 2019).

For historical dependencies of events, RE-NET (Jin et al., 2019) employs a recurrent event encoder to learn temporal dependencies from a temporal sequence of KGs. RE-GCN (Li et al., 2021b) applies gate recurrent components to capture sequential patterns across temporally adjacent facts. TITer (Sun et al., 2021) introduces a temporal-path-based reinforcement learning framework to travel on the historical snapshots and search for temporal patterns. To model the complex evolutional pattern, CEN (Li et al., 2022b) employs a length-aware Con-

*Corresponding author.

LREC-COLING 2024, pages 14744–14754
20-25 May, 2024.

volutional Neural Network (CNN) to mine evolutional patterns of different lengths.

All the above-mentioned models utilize recurrent structures, such as GRU (Cho et al., 2014) and LSTM (Hochreiter and Schmidhuber, 1997) to capture temporal dependences of facts implicitly, which is insufficient to model the long-term temporal dependencies.

Some other methods are proposed for modeling repetitive facts. Zhu et al. (2021) indicates that many facts exhibit a repetitive pattern along the timeline, such as economic crises happen periodically. CyGNet (Zhu et al., 2021) designs a copy mode to identify repetitive facts with an indicator vector of known facts in history. DA-Net (Liu et al., 2022b) models the dynamic distribution of repetitive facts by paying distributed attention to repetitive information at different historical timestamps. These models capture such repetitive patterns in history, which are proved to be of great help for TKG reasoning.

To capture both sequential and repetitive patterns, TiRGN (Li et al., 2022a) employs a local recurrent graph encoder to capture temporal dependencies at adjacent timestamps and designs a global history encoder to collect the historical global facts.

The existing repetitive pattern modeling methods only take into account whether historical facts occur or how often they occur, which are difficult to capture the historical variations of repetitive patterns over time.

To address the above challenges, we propose a novel Sequential and Repetitive Pattern Learning (SRPL)[1], a comprehensive method that captures both sequential patterns and repetitive patterns. Instead of capturing historical dependencies implicitly, we explicitly express the temporal dependencies of each historical timestamp as embeddings for capturing sequential patterns of facts. Concretely, SRPL first calculates the historical embeddings based on the temporal entity embeddings and temporal dependency embeddings from multiple previous timestamps. Then, the historical embeddings are used to generate hidden entity embeddings with a Gated Recurrent Unit (GRU) for capturing the sequential patterns. Finally, we design a time-interval guided attention mechanism to collect repetitive facts.Our contributions are three-fold:

- We propose a novel SRPL model to capture both sequential patterns and repetitive patterns for TKG reasoning. As far as we know, this is the first time to explicitly express the temporal dependencies of each historical timestamp as embeddings for TKG, which could better capture long-term dependencies from a sequence of KGs.
- We design a time-interval guided attention mechanism to model irregular time intervals between repetitive facts for capturing repetitive patterns.
- We conduct extensive evaluations on four representative datasets and the results show that our model is richly superior to state-of-the-art baselines by an obvious margin on all datasets.

[1]We make our code publicly available at https://github.com/Huiweizhou/SRPL.

2. Related Work

TKG reasoning can be divided into two settings, *interpolation* and *extrapolation* (Jin et al., 2019).

2.1. Interpolation Reasoning

The early works in temporal reasoning focus on *interpolation* reasoning, which predict missing facts at the historical timestamps. TTransE (Leblay and Chekol, 2018) extends TransE (Bordes et al., 2013) with the temporal constraints in the score function. TA-DistMult (García-Durán et al., 2018) learns time-aware representations of relations by a recursive neural network to incorporate temporal information, while using the same scoring function as DistMult (Yang et al., 2014). HyTE (Dasgupta et al., 2018) projects entities and relations to time-specific hyperplanes with a time-related projection instead of a normal projection in TransH (Wang et al., 2014).

These methods tend to integrate time information into the embeddings of different timestamps. However, they neglect the temporal dependencies of facts and could not capture temporal dynamics for future fact prediction.

2.2. Extrapolation Reasoning

Recent works concentrate on modeling historical evolution of facts to solve *extrapolation* problem of TKG reasoning. According to human cognition, temporally adjacent facts have sequential patterns. While according to historical recurrence theory (Trompf, 1979), historical facts have repetitive patterns.

To capture sequential patterns, query-specific methods (Jin et al., 2019) and entire graph-based methods (Li et al., 2021b) have been thoroughly investigated. Query-specific methods typically construct a subgraph sequence for each individual query with certain hop numbers, and then model the query-specific history with recurrent structures. RE-NET (Jin et al., 2019) learns temporal dependencies from the historical sequence related to the subject entity in each query. TITer (Sun et al., 2021)

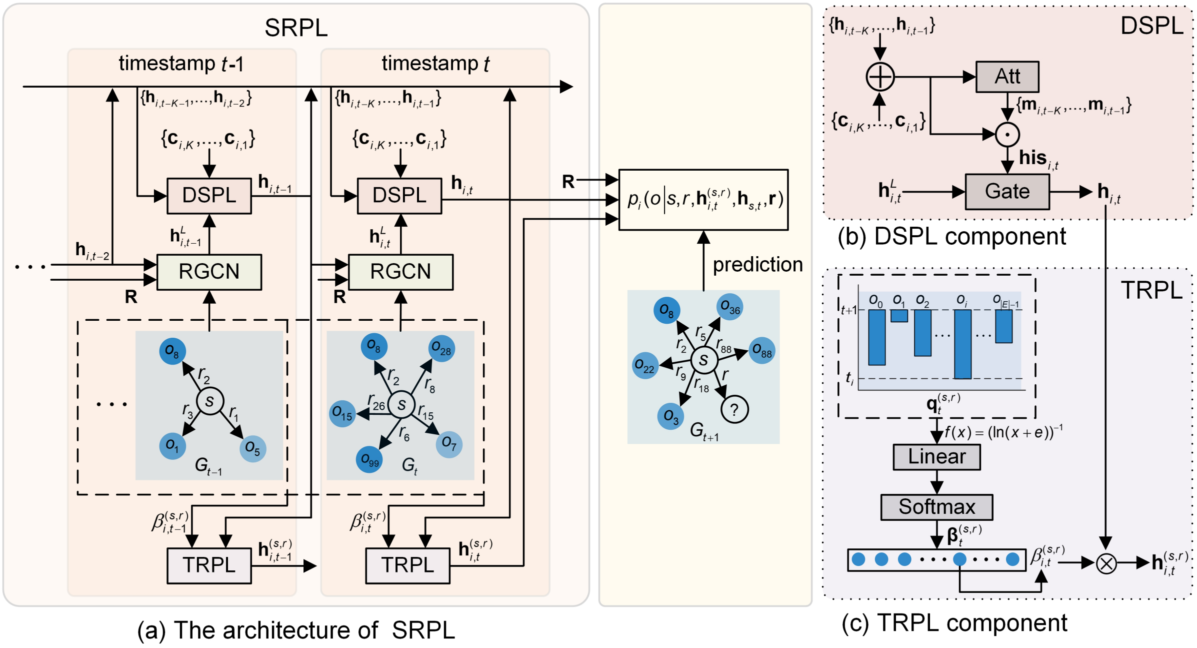

Figure 1: Overall Framework of our SRPL model.

and CluSTeR (Li et al., 2021a) capture sequential patterns in query-related temporal path with reinforcement learning.

Entire graph-based methods usually model the whole KG sequence uniformly to learn informative sequential patterns across timestamps. RE-GCN (Li et al., 2021b) captures the structural dependencies within each snapshot based on GCN and model the sequential patterns of all facts by gate recurrent networks. CEN (Li et al., 2022b) leverages a length-aware CNN to model the evolutional patterns of different lengths. HisMatch (Li et al., 2022c) combines both query-specific history and entire graph-based history into a unified framework to integrate two kinds of sequential patterns.

On the other hand, some methods are proposed to model repetitive facts and verify the effectiveness of repetitive patterns in TKG reasoning. CyGNet (Zhu et al., 2021) captures repetitive patterns by modeling the occurrence frequency of historical facts. DA-Net (Liu et al., 2022b) models dynamic distribution of repetitive facts by dual layers of attention, the first layer learns attention of the repetitive facts and the second layer adjusts the first layer attention based on frequency changes of the repetitive facts. TiRGN (Li et al., 2022a) simultaneously considers sequential patterns and repetitive patterns, which designs a double recurrent mechanism to encode the adjacent subgraph sequences and constructs a candidate entity matrix to record the occurrence of facts for collecting the repetitive facts.

Great progress has been made on *extrapolation* reasoning over TKG. However, these methods mainly suffer from two issues. First, the sequential pattern modeling methods only rely on recurrent structures, which may lead to important information oblivion in modeling long-term temporal dependencies. Second, historical facts have both sequential patterns and repetitive patterns. However, only few researchers have investigated how to comprehensively model both patterns for TKG reasoning.

To address these issues, we propose Sequential and Repetitive Pattern Learning (SRPL), a novel comprehensive method that mines both sequential patterns and repetitive patterns. SRPL models long-term sequential patterns with explicit temporal dependency embeddings across multiple historical timestamps. Meanwhile SRPL captures repetitive patterns by modeling the irregular time intervals between repetitive facts.

3. Method

3.1. Problem Definition

We formalize a TKG as a sequential static KG snapshots $G = \{G_1, G_2, ..., G_t, ...\}$, where $G_t = \{E, R, F_t\}$ at timestamp t contains the sets of entities, relations and facts. A fact in F_t is formalized as a quadruple (s, r, o, t), where $s, o \in E$, $r \in R$. The entity prediction under the *extrapolation* setting aims to predict the missing entity of a query $(s, r, ?, t+1)$ or $(?, r, o, t+1)$ with the previous snapshot sequence $\{G_1, G_2, ..., G_t\}$. In order to simplify the illustration, we use $(s, r, ?, t+1)$ to represent entity prediction task uniformly.

3.2. Model Overview

The overall framework of SRPL is shown in Figure1(a), which consists of three components. RGCN, Sequential Pattern Learning (DSPL) shown in Figure1(b) and Repetitive Pattern Learning (TRPL) shown in Figure1(c). RGCN (Schlichtkrull et al., 2018) is employed to learn the spatial embeddings of entities at each timestamp. DSPL is used to capture the sequential patterns with a sequence of dependency embeddings $(\mathbf{c}_{i,K}, ..., \mathbf{c}_{i,1})$ of each entity i. TRPL mines repetitive patterns with a time-interval guided attention mechanism. At timestamp t for each entity i, the entity embedding at timestamp $t-1$ $\mathbf{h}_{i,t-1}$ and relations embeddings $\mathbf{R}$ are fed to RGCN to get the spatial embedding at timestamp t $\mathbf{h}^L_{i,t}$ firstly. Then, historical entity embeddings $(\mathbf{h}_{i,t-K}, ..., \mathbf{h}_{i,t-1})$, $(\mathbf{c}_{i,K}, ..., \mathbf{c}_{i,1})$ and $\mathbf{h}^L_{i,t}$ are used as the input of DSPL to generate the entity embedding at timestamp t $\mathbf{h}_{i,t}$. Finally, $\mathbf{h}_{i,t}$ and the time-interval attention weight $\beta^{(s,r)}_{i,t}$ are fed to TRPL to obtain the query-induced temporal embedding $\mathbf{h}^{(s,r)}_{i,t}$.

3.3. Dependency-aware Sequential Pattern Learning (DSPL)

Traditional TKG reasoning approaches typically use recurrent structures to learn the temporal patterns from a sequence of temporal graph. However, these recurrent structures are hard to capture long-term temporal dependencies due to the vanishing gradient problem.

For better capturing long-range dependencies in traffic network, MSDR (Liu et al., 2022a) involves the hidden states and dependencies from multiple previous time-steps into the computation of the current state. MSDR (Liu et al., 2022a) applies a layer-wise attention to calculate the duration of influence caused by multiple previous time-steps in a graph convolution module.

To capture long-term temporal dependencies, we propose DSPL, which explicitly models temporal dependencies as embeddings and updates current entity embeddings by historical entity and dependency embeddings from multiple previous timestamps. Different from MSDR (Liu et al., 2022a), we think that the influence of the hidden states and dependencies from different timestamps should be different for the current state in TKG reasoning. Therefore, we propose a timestamp-wise attention to accurately learn the duration of influence caused by multiple previous timestamps.

Specifically, for each entity i, we represent the dependency between the current timestamp and the k-th previous timestamp as embedding $\mathbf{c}_{i,k} \in \mathbb{R}^d$, which can be trained globally. Thus, the influence of k-th previous timestamp on the current state can be expressed as $\mathbf{h}_{i,t-k} + \mathbf{c}_{i,k}$, where $\mathbf{h}_{i,t-k} \in \mathbb{R}^d$ is the entity embedding of the k-th previous timestamp.

The duration of influence by multiple previous timestamps, namely the historical context embedding of each entity i at timestamp t is calculated with attention over K previous timestamps as follows:

$$\mathbf{his}_{i,t} = \sum_{k=1}^{K} \mathbf{m}_{i,t-k} \odot (\mathbf{h}_{i,t-k} + \mathbf{c}_{i,k}) \tag{1}$$

where $\mathbf{h}_{i,t-k}$ is the entity embedding of entity i at timestamp $t-k$, $\mathbf{c}_{i,k}$ is the corresponding temporal dependency embedding of entity i between the historical timestamp $t-k$ and the current timestamp t, $\mathbf{m}_{i,t-k}$ is the attention weight vector and $\odot$ indicates element-wise multiplication.

The weight vector for each previous timestamp is computed by a novel timestamp-wise attention:

$$\mathbf{g}_{i,t-k} = \mathbf{W}_k(\mathbf{h}_{i,t-k} + \mathbf{c}_{i,k}) + \mathbf{b}_k \tag{2}$$

$$\boldsymbol{\lambda}_i[;,j] = softmax(\mathbf{g}_i[;,j]) \tag{3}$$

$$\mathbf{m}_{i,t-k} = \boldsymbol{\lambda}_i[K-k,;] \tag{4}$$

where $\mathbf{W}_k \in \mathbb{R}^{d\times d}$ and $\mathbf{b}_k \in \mathbb{R}^d$ are the trainable parameters at timestamp $t-k$, $\mathbf{g}_i = [\mathbf{g}_{i,t-K}||...||\mathbf{g}_{i,t-1}] \in \mathbb{R}^{K\times d}$ is the concatenation of all K $\mathbf{g}_{i,t-k}$, $\mathbf{g}_i[;,j]$ is the j-th column of $\mathbf{g}_i$, $\boldsymbol{\lambda}_i[;,j]$ is the j-th column of $\boldsymbol{\lambda}_i$ and $\boldsymbol{\lambda}_i[K-k,;]$ is the $(K-k)$-th row of $\boldsymbol{\lambda}_i$.

The historical context embeddings explicitly capture the duration of influence over K previous timestamps. With rich historical information, the embedding of entity i at timestamp t is updated by a GRU unit:

$$\mathbf{h}_{i,t} = u_{i,t}\,\mathbf{h}^L_{i,t} + (1-u_{i,t})\,\mathbf{his}_{i,t} \tag{5}$$

where $\mathbf{h}^L_{i,t}$ is the spatial embedding of entity i at each timestamp t. We calculate it with a shared $RGCN$ layer over the snapshot G_t:

$$\mathbf{h}^L_{i,t} = RGCN(\mathbf{h}_{i,t-1}, \mathbf{R}, G_t) \tag{6}$$

where L is the number of layers of RGCN and $\mathbf{R}$ are the embeddings of relations R. The relation embeddings $\mathbf{R}$ are randomly initialized and tuned during training.

The time gate $u_{i,t}$ can be computed as:

$$u_{i,t} = \sigma(\mathbf{W}_u \mathbf{his}_{i,t} + \mathbf{b}_u) \tag{7}$$

where $\mathbf{W}_u \in \mathbb{R}^{d\times d}$ and $\mathbf{b}_u \in \mathbb{R}^d$ are the trainable parameters. Then, we obtain the temporal embeddings of all entities $\mathbf{H}_t \in \mathbb{R}^{|E|\times d}$, where $|E|$ is the number of entities.

Dataset	$\|E\|$	$\|R\|$	$\|F_{train}\|$	$\|F_{valid}\|$	$\|F_{test}\|$	Time Granularity
ICEWS14	6,869	230	74,845	8,514	7,371	1 day
ICEWS18	23,033	256	373,018	45,995	49,545	1 day
WIKI	12,554	24	539,286	67,538	63,110	1 year
GDELT	7,691	240	1,734,399	238,765	305,241	15 mins

Table 1: Statistics of the TKG datasets

3.4. Time-interval Guided Repetitive Pattern Learning (TRPL)

To model the irregular time intervals between historical repetitive facts, we propose TRPL for capturing repetitive patterns.

We first find the latest occurrence timestamp t_i of each candidate entity i according to temporal snapshot sequence $\{G_1, G_2, ..., G_t\}$. For a query $(s, r, ?, t+1)$, the time interval between the query timestamp $t+1$ and the latest occurrence timestamp $t_i^{(s,r)}$ for candidate entity i is calculated as follows:

$$q_{i,t}^{(s,r)} = t + 1 - t_i^{(s,r)} \tag{8}$$

We set $t_i^{(s,r)} = 0$ for the facts that never happend. Jiang et al. (2023) model the impact between two roads with their relative time interval by a decay function. Inspired by Jiang et al. (2023), we then adopt a decay function to process the time interval as follows:

$$v_{i,t}^{(s,r)} = \frac{1}{\ln(q_{i,t}^{(s,r)} + e)} \tag{9}$$

where $e \approx 2.718$ is a natural constant. In this way, $v_{i,t}^{(s,r)}$ decreases with the time interval $q_{i,t}^{(s,r)}$ increasing.

Sequentially, the processed $v_{i,t}^{(s,r)}$ of all candidate entities form an indicator vector $\mathbf{v}_t^{(s,r)} \in \mathbb{R}^{|E|}$.

Furthermore, the obtained indicator vector $\mathbf{v}_t^{(s,r)}$ is used to calculate the time-interval attention weight vector $\boldsymbol{\beta}_t^{(s,r)} \in \mathbb{R}^{|E|}$ of each candidate entity by using a fully connected layer followed by a *softmax* layer:

$$\boldsymbol{\beta}_t^{(s,r)} = softmax(\mathbf{W}_l \mathbf{v}_t^{(s,r)} + \mathbf{b}_l) \tag{10}$$

where $\mathbf{W}_l \in \mathbb{R}^{|E| \times |E|}$ and $\mathbf{b}_l \in \mathbb{R}^{|E|}$ are trainable parameters.

Finally, based on the above two components, the query-induced temporal embedding $\mathbf{h}_{i,t}^{(s,r)}$ is obtained based on the temporal entity embedding $\mathbf{h}_{i,t}$ and the corresponding weight $\beta_{i,t}^{(s,r)}$:

$$\mathbf{h}_{i,t}^{(s,r)} = \beta_{i,t}^{(s,r)} \times \mathbf{h}_{i,t} \tag{11}$$

where $\beta_{i,t}^{(s,r)}$ is the i-th dimension of $\boldsymbol{\beta}_t^{(s,r)}$. In this way, we obtain the query-induced temporal embeddings of all entities $\mathbf{H}_t^{(s,r)} \in \mathbb{R}^{|E| \times d}$.

3.5. Inference Procedure and Learning Objective

The TKG sequence from timestamp 1 to T are used to train our SRPL model. When testing, we predict the missing object entity $(s, r, ?, T+1)$ in G_{T+1}.

ConvTransE (Shang et al., 2019) is utilized as decoder to calculate the probability vector of all entities as follows:

$$\mathbf{p}(o \,\middle|\, s, r, \mathbf{H}_t^{(s,r)}, \mathbf{H}_t, \mathbf{R}) = \sigma(\mathbf{H}_t^{(s,r)} ConvTransE(\mathbf{s}_t, \mathbf{r})) \tag{12}$$

where $\sigma(\cdot)$ is the sigmoid function, $\mathbf{s}_t$ and $\mathbf{r}$ are the embeddings of subject entity s and relation r. The prediction loss for SRPL model is the sum of the cross-entropy loss:

$$L = -\sum_{t=1}^{T} \sum_{(s,r,o,t+1) \in F_{t+1}} \sum_{i=0}^{|E|-1} y_{i,t+1} \log p_i(o \,|\, s, r, \mathbf{H}_t^{(s,r)}, \mathbf{H}_t, \mathbf{R}) \tag{13}$$

where T is the number of timestamps, $y_{i,t+1}$ is the ground truth label. The value of $y_{i,t+1}$ is 1 if the fact $(s, r, i, t+1)$ happened, otherwise 0.

4. Experiments

4.1. Experimental Setup

Datasets and Evaluation Metrics: We compare the performance of our model against other typical baselines for entity prediction on four traditional TKG datasets, including ICEWS14 (García-Durán et al., 2018), ICEWS18 (Jin et al., 2019), WIKI (Leblay and Chekol, 2018) and GDELT (Jin et al., 2019). Detailed statistics of the datasets are summarized in Table 1. We report the Mean Reciprocal Ranks (MRR) and Hits@$\{1, 3, 10\}$ under the time-aware filtered setting (Han et al., 2020).

Implementation Details: The dimension of entity embeddings, relation embeddings and dependency embeddings are set to 200, respectively. The number of layers of the RGCN is set to 2 and the dropout rate is set to 0.3 for each layer. The sequence length K of historical contexts is set to 6, 4, 7 and 5 for ICEWS14, ICEWS18, WIKI and GDELT,

Model		ICEWS14				ICEWS18			
		MRR	Hits@1	Hits@3	Hits@10	MRR	Hits@1	Hits@3	Hits@10
1	TTransE	13.72	2.98	17.70	35.74	8.31	1.92	8.56	21.89
	TA-DistMult	25.80	16.94	29.74	42.99	16.75	8.61	18.41	33.59
	HyTE	11.48	5.64	13.04	22.51	7.31	3.10	7.50	14.95
2	RE-NET	39.86	30.11	44.02	58.21	29.78	19.73	32.55	48.46
	TITer	41.73	32.74	46.46	58.44	29.98	22.05	33.46	44.83
	RE-GCN	42.00	31.63	47.20	61.65	32.62	22.39	36.79	52.68
	CEN	42.20	32.08	47.46	61.31	31.50	21.70	35.44	50.59
	HisMatch	<u>46.42</u>	<u>35.91</u>	<u>51.63</u>	<u>66.84</u>	<u>33.99</u>	<u>23.91</u>	37.90	53.94
	CyGNet	37.65	27.43	42.63	57.90	27.12	17.21	30.97	46.85
	TiRGN	44.04	33.83	48.95	63.84	33.66	23.19	<u>37.99</u>	<u>54.22</u>
	SRPL	**56.19**	**50.12**	**59.02**	**67.43**	**47.58**	**40.93**	**50.72**	**59.22**

Table 2: Main results (in percentage) of entity prediction on ICEWS14 and ICEWS18.The best results are boldfaced, and the second best ones are underlined.

Model		WIKI				GDELT			
		MRR	Hits@1	Hits@3	Hits@10	MRR	Hits@1	Hits@3	Hits@10
1	TTransE	29.27	21.67	34.43	42.39	5.50	0.47	4.94	15.25
	TA-DistMult	44.53	39.92	48.73	51.71	12.00	5.76	12.94	23.54
	HyTE	43.02	-	45.12	49.49	6.37	0.00	6.72	18.63
2	RE-NET	58.32	50.01	61.23	73.57	19.55	12.38	20.80	34.00
	TITer	73.91	71.70	75.41	76.96	18.19	11.52	19.20	31.00
	RE-GCN	78.53	74.50	81.59	84.70	19.69	12.46	20.93	33.81
	CEN	78.93	75.05	81.90	84.90	-	-	-	-
	HisMatch	78.07	73.89	81.32	84.65	<u>22.01</u>	<u>14.45</u>	<u>23.80</u>	<u>36.61</u>
	CyGNet	58.78	47.89	66.44	78.70	20.22	12.35	21.66	35.82
	TiRGN	<u>81.65</u>	<u>77.77</u>	<u>85.12</u>	<u>87.08</u>	21.67	13.63	23.27	37.60
	SRPL	**86.96**	**85.92**	**87.70**	**88.17**	**40.85**	**29.96**	**46.57**	**62.28**

Table 3: Main results (in percentage) of entity prediction on WIKI and GDELT.

respectively. Adam optimizer is adopted for parameter with the learning rate of $1e$-3 on all datasets. The experiments are all carried out on RTX3090.

4.2. Main Results

Our method is compared with the current state-of-the-art TKG reasoning baselines. The comparison results are shown in Table 2 and Table 3. We divide these models into two categories: *interpolation* (category 1) and *extrapolation* (category 2) reasoning models.

In the *interpolation* category, we select TTransE (Leblay and Chekol, 2018), TA-DistMult (García-Durán et al., 2018) and HyTE (Dasgupta et al., 2018). From the results, we can see that the *interpolation* models (i.e., category 1) lag significantly behind *extrapolation* reasoning models (i.e., category 2). The main reason is the *interpolation* models do not capture the temporal dependencies of historical facts, hence they cannot predict future facts.

As for the *extrapolation* category, RE-NET (Jin et al., 2019), TITer (Sun et al., 2021), RE-GCN (Li et al., 2021b), CEN (Li et al., 2022b) and HisMatch (Li et al., 2022c) focus on capturing sequential patterns. Among them, RE-NET and TITer are query-specific models, while RE-GCN and CEN are entire graph-based models. From the results in Table 2 and Table 3, we can see that entire graph-based models outperform query-specific models in general. Entire graph-based methods model the TKG sequence as a whole, which can capture richer temporal dependencies than query-specific models. HisMatch performs much better than the other sequential pattern capturing models, because it combines both query-specific and entire graph-based histories.

CyGNet (Zhu et al., 2021) is centered on capturing repetitive patterns, which achieves slightly lower performance in comparison with the sequential pattern capturing models. It is perhaps that sequential patterns are more important than repetitive patterns for TKG reasoning.

TiRGN (Li et al., 2022a) and our SRPL take both sequential and repetitive patterns into considera-

Model	ICEWS14	ICEWS18	WIKI	GDELT
SRPL	**56.19**	**47.58**	**86.96**	**40.85**
w/o TRPL	41.63	32.35	72.03	19.69
w/o $\mathbf{c}_{i,k}$ in DSPL	55.87	46.32	86.50	39.46
w/o $\mathbf{his}_{i,t}$ w $\mathbf{h}_{i,t-1} + \mathbf{c}_{i,1}$ in DSPL	55.58	46.18	86.33	39.29
w/o $\mathbf{his}_{i,t}$ & $\mathbf{c}_{i,k}$ w $\mathbf{h}_{i,t-1}$ in DSPL	53.85	44.71	85.82	38.87
w/o $\mathbf{his}_{i,t}$ & $\mathbf{c}_{i,k}$ w $\mathbf{h}_{i,t-1}$ in DSPL and layer-wise attention in RGCN	55.54	45.72	86.19	39.14

Table 4: Results (in percentage) of ablation studies with time-aware MRR.

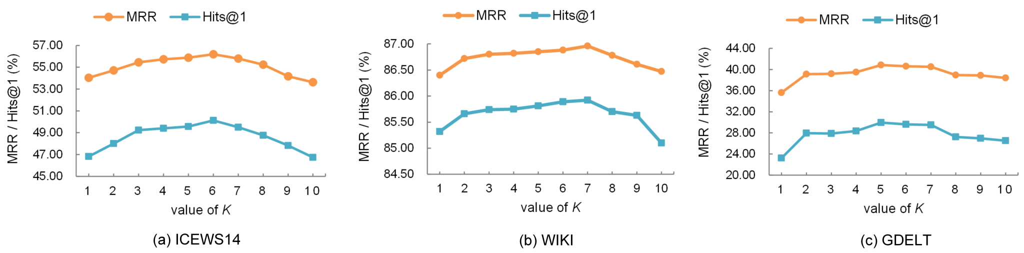

Figure 2: Impact of historical timestamp K on ICEWS14, WIKI and GDELT.

tion. TiRGN gets better performance than the other models except HisMatch on ICEWS14, ICEWS18 and GDELT. Our SRPL outperforms the other state-of-the-art models on the four datasets in terms of all the metrics. The results clearly demonstrate the necessity of capturing both sequential and repetitive patterns for temporal reasoning.

Our SRPL achieves up to 9.77%, 13.59%, 5.31% and 18.84% improvements in MRR comparing to the sub-optimal models on ICEWS14, ICEWS18, WIKI and GDELT datasets, respectively. By comprehensively capturing sequential patterns with explicit temporal dependency embeddings and modeling repetitive patterns with irregular time intervals between repetitive facts, SRPL gains the surprisingly excellent performance.

4.3. Ablation Studies

In order to further investigate the contribution of different components of SRPL, we conduct ablation studies on ICEWS14, ICEWS18, WIKI and GDELT. We report the MRR metric in Table 4.

Without TRPL, MRR drops by 14.56%, 15.23%, 14.93% and 21.16% on ICEWS14, ICEWS18, WIKI and GDELT, respectively. This indicates that capturing repetitive patterns are crucial for temporal reasoning, and TRPL can effectively collect historical repetitive facts.

Removing temporal dependency embeddings $\mathbf{c}_{i,k}$ in DSPL, the performance in all datasets drops significantly, which demonstrates that the necessity of temporal dependency embeddings for capturing sequential patterns.

Instead of using the historical context embedding $\mathbf{his}_{i,t}$ of K previous timestamps in DSPL, we use the $\mathbf{h}_{i,t-1} + \mathbf{c}_{i,1}$ from the previous timestamp $t-1$ to train the variant model, the performance also drops significantly. This suggests that DSPL could better capture sequential patterns with the help of historical contexts.

Removing both $\mathbf{his}_{i,t}$ and $\mathbf{c}_{i,k}$ in DSPL, we only use the entity embedding $\mathbf{h}_{i,t-1}$ from the previous timestamp $t-1$, which is the same as original GRU. The sharper drop of performance can be observed on all datasets, which demonstrates both historical contexts and dependencies are effective for modeling the sequential patterns.

Furthermore, instead of using the timestamp-wise attention in DSPL, we use a layer-wise attention as the same as MSDR (Liu et al., 2022a) to learn the historical information. That is, we make the following changes: replacing $\mathbf{his}_{i,t}$ and $\mathbf{c}_{i,k}$ with the entity embedding $\mathbf{h}_{i,t-1}$ in DSPL, and using a layer-wise attention in RGCN to learn the historical context embeddings. The performance on all datasets drops, which proves that our timestamp-wise attention is more conducive to learn the duration of influence caused by multiple timestamps.

4.4. Impact of Historical Timestamp K

To investigate the impact of historical timestamps, we vary the number of previous timestamps K when training SRPL models. The MRR and Hits@1 performance on ICEWS14, WIKI and GDELT are shown in Figure 2. From the figure we can see that the best performance can be achieved when K is

Decay Function	MRR	Hits@1	Hits@3	Hits@10
$f(x) = (\ln(x + e))^{-1}$	**47.58**	**40.93**	**50.72**	<u>59.22</u>
$f(x) = (\log_2(x + 2))^{-1}$	<u>45.56</u>	<u>38.65</u>	48.94	57.43
$f(x) = (\lg(x + 10))^{-1}$	38.44	31.83	41.61	50.42
$f(x) = (330 - x)/330$	43.42	33.87	<u>49.59</u>	**59.67**
$f(x) = e^{-3x/500}$	41.63	37.45	43.48	48.31
$f(x) = \cos(\pi x/600)$	25.11	21.51	27.54	30.40

Table 5: Results (in percentage) of the different decay functions on ICEWS18.

Metric	Model	Subset					
		Repetitive Facts					Unobserved Facts
		a	b	c	d	e	f
		10.4%	17.8%	20.4%	19.9%	8.4%	23.1%
Hits@1	SRPL	**54.84**	**54.71**	**52.03**	**50.07**	**40.09**	**18.39**
	CyGNet	31.49	25.84	20.49	18.84	10.02	0.39
	TiRGN	<u>32.96</u>	<u>26.44</u>	<u>23.50</u>	<u>22.33</u>	<u>10.45</u>	<u>9.47</u>
Hits@3	SRPL	**78.55**	**77.00**	**75.40**	**72.35**	**67.59**	**29.38**
	CyGNet	42.91	41.64	37.46	31.43	19.83	3.04
	TiRGN	<u>46.02</u>	<u>41.95</u>	<u>37.99</u>	<u>36.64</u>	<u>21.54</u>	<u>17.93</u>

Table 6: Results (in percentage) of different subsets on GDELT.

6, 7 and 5 on ICEWS14, WIKI and GDELT, respectively. And the MRR and Hits@1 on each dataset keep improving with the the number of previous timestamps K increasing to the optimum value, and then gradually decline. This indicates that learning hidden states in GRU from relatively long-term timestamps is helpful to avoid information forgetting. However, too long historical timestamps would introduce noises and affect the accuracy of SRPL models.

4.5. Effects of Different Decay Functions

We replace the decay function of formula (9) used in TRPL with other decay functions. Table 5 compares the effects of the different decay functions, including the log functions (the first three rows), the linear function (the forth row), the exponential function (the fifth row) and the trigonometric function (the sixth row). We set the average decay rate of the linear function, exponential and trigonometric functions to be the same as that of our log function (the first row).

As shown in Table 5, we can find that: (1) Our decay function outperforms the linear function, exponential and trigonometric functions, which suggests log function is more suitable as delay function. Even so, except the trigonometric function, the MRR and Hits@1 of the models with the linear function and exponential function achieve better performance than state-of-the-art models. This demonstrates the effectiveness of the proposed TRPL for capturing repetitive patterns. (2) Removing the base e of our log function with base 2 or 10 can lead to a drop of 2.02% and 9.14% in MRR, respectively, which shows that the average decay rate with different bases greatly influence the reasoning performance.

4.6. Detailed Analysis on Repetitive Fact Collection

To get insight into the performance of SRPL on repetitive facts with different time intervals, we conduct more detailed analysis on the test data of GDELT. We split the facts in the test data according to the time interval between the test timestamp and the nearest repetitive facts. The time intervals are (a) 1-3, (b) 4-12, (c) 13-40, (d) 41-500, (e) 501-2304 and (f) unobserved facts, respectively.

Table 6 shows the comparison results of our SRPL, CyGNet (Zhu et al., 2021) and TiRGN (Li et al., 2022a) with Hits@1 and Hits@3 on each subset. CyGNet models the occurrence frequency of historical facts to each query. Instead of the frequency, TiRGN only considers whether the fact has happened before. Similar to ours, TiRGN captures sequential patterns and repetitive patterns simultaneously.

From the results, we observe that (1) SRPL significantly outperforms the other two models on Hits@1 and Hits@3 in all intervals. This demonstrates the effectiveness of our time-interval guided attention mechanism in modeling the repetitive parttens of different lengths. (2) Both Hits@1 and Hits@3

Model	1-hop			2-hop			multi-hop		
	MRR	Hits@1	Hits@3	MRR	Hits@1	Hits@3	MRR	Hits@1	Hits@3
RE-GCN	48.57	35.36	56.19	49.72	40.55	54.25	18.32	12.71	20.17
SDPL	55.94	40.84	67.15	57.84	47.63	64.73	26.56	22.09	28.69
Improvement	15.17%	15.50%	19.51%	16.33%	17.50%	19.32%	44.98%	73.80%	42.24%

Table 7: The percentage improvement of SDPL over RE-GCN in different sequential patterns on ICEWS14.

History	Query at T+1	Answer	Model	Prediction
Violent group,Conduct boycott,Government,T-6 Government,Make statement,Judiciary,T-2 Judiciary,Accuse,Citizen,T	Judiciary, Charge with legal action, ?	Violent group	SRPL	Violent group
			RE-GCN	Citizen
Police,Use conventional military,Militant,T-13 Police,Arrest,Militant,T-10 Police,Use conventional military,Militant,T-3 Police,Detain,Criminal,T-1	Police, Arrest, ?	Militant	SRPL	Militant
			TiRGN	Criminal

Table 8: Case study.

decrease as the time interval gets longer for all models. This indicates that the longer the time interval between repetitive facts, the harder it is to be collected. However, compared with the other two models, the performance of our SRPL in each interval is relatively stable. (3) For unobserved facts, SRPL also achieves the improvements of 8.92%/18.00% in Hits@1 and 11.45%/26.34% in Hits@3 over TiRGN and CyGNet. It is more likely to benefit from DSPL in SRPL, which could capture informative sequential patterns. (4) TiRGN is consistently better than CyGNet in all intervals, espacially in the interval (f) unobserved facts. This indicates that it is necessary to capture sequential patterns, which is helpful to capture repetitive patterns.

4.7. Effects of DSPL on Capturing Temporal Sequential Patterns

To demonstrate the effectiveness of DSPL in capturing temporal sequential patterns, we compare pure SDPL (SRPL w/o TRPL) with RE-GCN (Li et al., 2021b) on ICEWS14. We divide the test data of ICEWS14 into the following scenarios: (1) 1-hop sequential patterns, that is, 1-hop reasoning can lead to answer, for example, the sequential pattern (*A*,R_1,*B*,t-*3*), (*B*,R_2,*C*,t-*1*) helps the object entity prediction (*A*,R_3,*C*,t+*1*);(2) 2-hop sequential patterns; (3) multi-hop (more than 2-hop) sequential patterns.

Table 7 shows the comparison results on each scenario. RE-GCN focuses on capturing sequential patterns and performs quite well. From the results we can conclude that DSPL is obviously superior to RE-GCN in all the scenarios, especially in the multi-hop scenario. This shows that DSPL is capable of capturing long-term temporal dependencies.

4.8. Case Study

In order to show the sequential patterns and repetitive patterns across temporally adjacent facts learned by SRPL, Table 8 illustrates two cases that SRPL gives the answers top 1 scores from the test set of ICEWS18.

The first case shows that the sequential pattern (*A,Conduct boycott,B,*T*-6*), (*B,Make statement,C,*T*-2*) can lead to (*C,Charge with legal action,A,*T*+1*). While RE-GCN incorrectly predicts "*Citizen*" according to the similar recent fact (*C,Accuse,Citizen,*T). From the result, we find that SRPL could capture long-term sequential patterns.

The second case shows that the sequential pattern (*A,Use conventional military,B,*T*-13*), (*A,Arrest,B,*T*-10*) and the repetitive context fact (*A,Use conventional military,B,*T*-3*) lead to the repetitive fact (*A,Arrest,B,*T*+1*). By contrast, TiRGN mainly focuses on predicting sequential facts with temporally adjacent facts, therefore incorrectly predicts "*Criminal*" according to the similar recent fact (*A,Detain,Criminal,*T*-1*). We can see that SRPL is able to accurately capture repetitive facts with long time intervals.

5. Conclusion

In this paper, we propose a novel SRPL model, which captures both sequential and repetitive patterns for *extrapolation* TKG reasoning. SRPL captures long-term sequential patterns with DSPL, and meanwhile, collects repetitive facts with TRPL. Experimental results on the four benchmarks reveal the superiority of DSPL and the effectiveness of TRPL. The qualitative analyses also show the necessity and significance of incorporating the two patterns of historical facts.

6. Bibliographical References

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

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

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.

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

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

Simon Gottschalk and Elena Demidova. 2018. Eventkg: A multilingual event-centric temporal knowledge graph. In *The Semantic Web: 15th International Conference, ESWC 2018, Heraklion, Crete, Greece, June 3–7, 2018, Proceedings 15*, pages 272–287. Springer.

Zhen Han, Yunpu Ma, Yuyi Wang, Stephan Günnemann, and Volker Tresp. 2020. Graph hawkes neural network for forecasting on temporal knowledge graphs. *arXiv preprint arXiv:2003.13432*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Jiawei Jiang, Dayan Pan, Houxing Ren, Xiaohan Jiang, Chao Li, and Jingyuan Wang. 2023. Self-supervised trajectory representation learning with temporal regularities and travel semantics. In *2023 IEEE 39th international conference on data engineering (ICDE)*, pages 843–855. IEEE.

Woojeong Jin, Meng Qu, Xisen Jin, and Xiang Ren. 2019. Recurrent event network: Autoregressive structure inference over temporal knowledge graphs. *arXiv preprint arXiv:1904.05530*.

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

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

Zixuan Li, Saiping Guan, Xiaolong Jin, Weihua Peng, Yajuan Lyu, Yong Zhu, Long Bai, Wei Li, Jiafeng Guo, and Xueqi Cheng. 2022b. Complex evolutional pattern learning for temporal knowledge graph reasoning. *arXiv preprint arXiv:2203.07782*.

Zixuan Li, Zhongni Hou, Saiping Guan, Xiaolong Jin, Weihua Peng, Long Bai, Yajuan Lyu, Wei Li, Jiafeng Guo, and Xueqi Cheng. 2022c. Hismatch: Historical structure matching based temporal knowledge graph reasoning. *arXiv preprint arXiv:2210.09708*.

Zixuan Li, Xiaolong Jin, Saiping Guan, Wei Li, Jiafeng Guo, Yuanzhuo Wang, and Xueqi Cheng. 2021a. Search from history and reason for future: Two-stage reasoning on temporal knowledge graphs. *arXiv preprint arXiv:2106.00327*.

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

Dachuan Liu, Jin Wang, Shuo Shang, and Peng Han. 2022a. Msdr: Multi-step dependency relation networks for spatial temporal forecasting. In *Proceedings of the 28th ACM SIGKDD conference on knowledge discovery and data mining*, pages 1042–1050.

Kangzheng Liu, Feng Zhao, Hongxu Chen, Yicong Li, Guandong Xu, and Hai Jin. 2022b. Da-net: Distributed attention network for temporal knowledge graph reasoning. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, pages 1289–1298.

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

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

Alessio Signorini, Alberto Maria Segre, and Philip M Polgreen. 2011. The use of twitter to track levels of disease activity and public concern in the us during the influenza a h1n1 pandemic. *PloS one*, 6(5):e19467.

Haohai Sun, Jialun Zhong, Yunpu Ma, Zhen Han, and Kun He. 2021. Timetraveler: Reinforcement learning for temporal knowledge graph forecasting. *arXiv preprint arXiv:2109.04101*.

Garry W Trompf. 1979. *The idea of historical recurrence in Western thought: from antiquity to the Reformation*, volume 1. Univ of California Press.

Jiapu Wang, Boyue Wang, Meikang Qiu, Shirui Pan, Bo Xiong, Heng Liu, Linhao Luo, Tengfei Liu, Yongli Hu, Baocai Yin, et al. 2023. A survey on temporal knowledge graph completion: Taxonomy, progress, and prospects. *arXiv preprint arXiv:2308.02457*.

Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014. Knowledge graph embedding by translating on hyperplanes. In *Proceedings of the AAAI conference on artificial intelligence*, volume 28.

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

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

Xiaohan Zou. 2020. A survey on application of knowledge graph. In *Journal of Physics: Conference Series*, volume 1487, page 012016. IOP Publishing.