Evolving Temporal Knowledge Graphs by Iterative Spatio-Temporal Walks

Hao Tang, Donghong Liu, Xinhai Xu(✉), and Feng Zhang(✉)

Academy of Military Science, Beijing, China
fengz.cs@qq.com

Abstract. Predicting facts that occur in the future is a challenging task in temporal knowledge graphs (TKGs). TKGs represent temporal facts about entities and their relations, where each fact is associated with a timestamp. Inspired from the human inference process that predictions are usually made by analyzing relevant historical clues, in this paper, we propose a model based on temporal evolution and temporal graph attention mechanism to infer future facts. Specifically, we construct a node pool to keep the importance of all nodes encountered in the historical search. We learn temporal evolution features and sub-graph structures based on temporal random walks and graph attention networks. Moreover, these sub-graphs are sets of objects with the same subjects and relations as the query. Experiments on five temporal datasets demonstrate the effectiveness of the model compared with the state-of-the-art methods. Codes are available at https://github.com/lendie/SWGAT.

Keywords: Temporal knowledge graph · Spatio-Temporal walks · Future facts

1 Introduction

Knowledge graphs (KGs) are directed graphs which excel at organizing relational facts. They represent factual entities as nodes and semantic relations as edges. Each fact is presented as a triple of the form (*subject, relation, object*), e.g., (*Donald Trump, PresidentOf, USA*). Large-scale knowledge graphs have been used in various artificial intelligence applications including recommender system [11] and information retrieval [12]. Knowledge graph reasoning [1] refers to inferring missing facts from existing facts, but they treat a knowledge graph as a static graph, meaning that the entities and relationships do not change over time. However, in reality, many facts are only true at a certain point in time or period of time. For example, (*Donald Trump, PresidentOf, USA*) is valid only from January 2017 to January 2021. To this end, temporal knowledge graphs (TKGs) were introduced. A TKG represents a temporal fact as a quadruple (*subject, relation, object, timestamp*), describing that this fact is valid at this timestamp. Recently, research on temporal knowledge graph reasoning has received increasingly more attentions, which can be categorized into interpolation and extrapolation tasks [27]. The former studies reasoning facts within

H. Tang and D. Liu—Equal contribution.

M. Tanveer et al. (Eds.): ICONIP 2022, CCIS 1791, pp. 501–512, 2023.
https://doi.org/10.1007/978-981-99-1639-9_42

a known time range, while the latter studies predicting facts in the future and is more challenging. In this paper, we focus on the extrapolation tasks.

To solve extrapolation problems, different TKG embedding approaches have been developed. These approaches maps entities, relations and time information into a continuous vector space to capture the semantic meanings of a temporal knowledge graph. However, there exist two main challenges for current TKG embedding approaches. Firstly, how to model the evolutionary patterns of historical facts to accurately predict future facts. Secondly, how to model the topological structure of a TKG. Latest top-tier works focus on either side but not take both of them into consideration.

Inspired from the process of human reasoning, in this paper, we propose to tackle extrapolation problems by iterative spatio-temporal random walks followed by a temporal relation graph attention layer. In spatio-temporal random module, we select the one-hop neighbors that are close to the subject, and then calculate their importance scores by the relationship between these neighbors and the subject. Then, after assigning an importance score to each one-hop neighbor, we iteratively walk from the neighbors with the top-n importance scores, and select the two-hop neighbors of the subject. After that, TRGAT is used to capture the topological structure which select sub-graphs that are sets of objects with the same subject and relation as the query. Similar to CyGNet [28], this layer is mainly used to capture repetitive facts related to the query, except that we use graph attention network to capture such repetitive patterns. The key contributions of this paper can be summarized as follows:

- The reasoning idea of temporal knowledge graph is derived from the human cognitive process, consisting of iterative spatio-temporal walks and temporal graph attention mechanism.
- We resort to graph attention networks to capture repetitive patterns.
- Our model achieves state-of-the-art performance in five temporal datasets.

2 Related Work

2.1 Static KG Representation Learning

There is a growing interest in knowledge graph embedding methods. This type of method is broadly classified into three paradigms: (1) Translational distance-based models [1,25]. (2) Tensor factorization-based models [14,15]. (3) Neural network-based models [4,13]. Translation-based models consider the translation operation between entity and relation embedding, such as TransE [1] and TransH [25]. Factorization-based models assume KG as a third-order tensor matrix, and the triple score can be carried out through matrix decomposition, including RESCAL [15], HOLE [14]. Other models use convolutional neural networks or graph neural networks to model scoring functions, like ConvE [4], KBGAT [13]. However, all the above methods cannot predict future events, as all entities and relations are treated as static.

2.2 Temporal KG Representation Learning

In order to better capture the dynamic changes of information, the temporal knowledge graph embedding(TKGE) model encodes temporal information into entities or relationships. A number of recent works have attempted to model the changing facts in TKGs. Temporal knowledge graph inferring can be divided into interpolation [6] and extrapolation problems [28]. The former attempts to reason about facts in known time, while the latter, which is the focus of this paper, attempts to predict future facts. On the interpolation task, DE-SimplE [6] defines a function that takes entities and relations as input and then produces time-specific embeddings. But this approach ignores the dynamic changes of entities and relationships. On the extrapolation task, some models estimate the conditional probability to predict future facts via temporal point process taking all historical events into consideration. RE-NET [9] is used to capture the evolutionary patterns of fixed-length subgraphs specific to a query. CyGNet [28] models repeated facts with sequential copy-generation networks. xERTE [8] learns to find the query-related subgraphs with a fixed number of hops. Glean [3] enriches entity information by introducing time-dependent descriptions. EvoKG [17] performs temporal graph inference by jointly modeling event time and network structure.

3 Our Model

We describe our model in a top-down fashion where we provide an overview in Sect. 3.1, and then in Sect. 3.2 and 3.3 we explain each module separately.

3.1 Model Overview

Our model performs the inference process from walking sequences obtained by dynamic sampling on temporal knowledge graph and temporal relation graph attention layer (TRGAT). Our model contains two parts, spatio-temporal walks and TRGAT. Specifically, part 1 focuses on sampling dynamic node sequences whose semantic and temporal information is closely related to the given query. Afterwards, the sampled node sequences are provided to the temporal inference cell, which focuses on modeling the node sequence information and then assigning importance scores to each node related to the query. In the TRGAT module, we sample objects with the same subject and relationship as the query from the history information. And we note that different objects and subjects should also have different importance under the same relationship (Fig. 1).

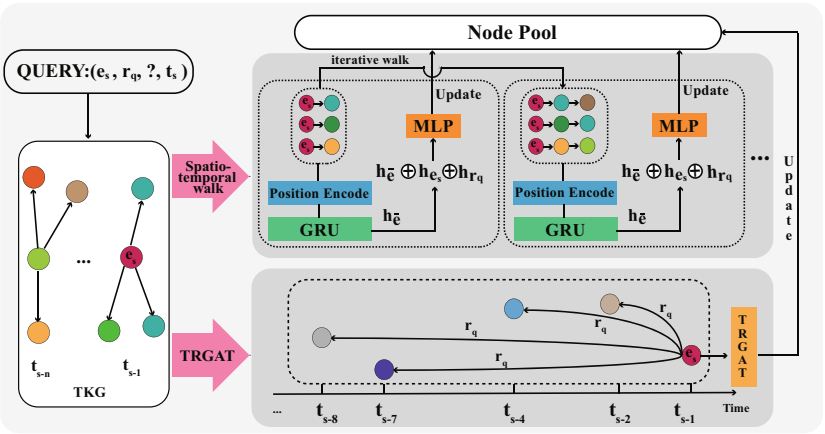

Fig. 1. Overview of model architecture.

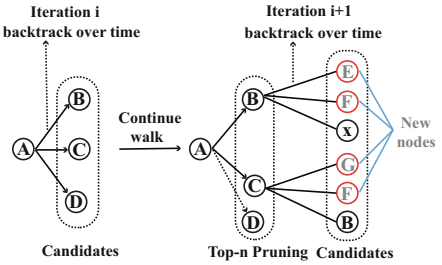

Fig. 2. Three 2-step walks (x is default node id, which we set to -1 when no historical links can be found).

3.2 Spatio-temporal Walking

We assume that the older the event is, the less impact on the inference. So instead of using all the historical event information, we choose the set consisting of nodes $\mathcal{V}$ that are closer to the time of the query, where $\mathcal{V}$ denotes the subset of all entities in the historical event that are directly linked to the query subject entity. We sample the connected links by backtracking over time to extract the potential time-evolving relations of temporal knowledge graph. According to our hypothesis, more recent events may contain more information and thus we use time-aware weighted sampling $\mathbb{P}(q_v = (s_i, r_k, o_j, t')) = \exp(t' - t)$, where t is previous sampled timestamp before t'. We show a toy example in Fig. 2. Given a query $(e_s, r_p, ?, t_s)$, we use A to denote node e_s. We first find the most recent moment of node A from historical events, such as $t - 1$. Since we use time

backtracking to search for historical information, in the next step, we will search for nodes that have direct link with node A from facts with less than or equal to $t-1$. As shown in Fig. 2, we obtain three walks, $\{ (A \rightarrow B), (A \rightarrow C), (A \rightarrow D) \}$, and here we omit the relationship and timestamp for simplicity. Then we put these walks into the time unit to calculate the relevance score of nodes B, C and D to the query. After, the walk continues, which we call iterative walk. To reduce the path selection time, we use the Top-n pruning method to continue the walking only from the n neighboring nodes with highest relevance scores.

Sampled Walks. We define sampled edges $\mathcal{S}_{v,t} = \{(e, t') \mid e \in \mathcal{G}, t' \leq t, v \in e\}$ to include the links contained before node v. The walking sequence of the temporal knowledge graph can be expressed as:

$$E = \left((w_0, t_0), (w_1, t_1), \ldots, (w_m, t_m)\right), t_0 \geq \cdots \geq t_m, \quad (w_i, t_i) \in \mathcal{S} \tag{1}$$

where (w_i, t_i) denotes quadruples (e_i, r_i, o_i, t_i).

Position Encoder. Inspired by CAW [23], in order to make the model inductive, we use the method mentioned in CAW [23] to remove the node identifiers to encode the relative position information. Let the set of walks sampled from node e_s be S_e. Each node from S_e is replaced by a vector that encode a positional frequency of the node in each corresponding positions of walks in S_e. For node e_s, the vector of position frequencies relative to all walks in S_e is defined as:

$$PE(e_s, S_e) = \{|\{W \mid W \in S_e, e_s = W_m, m \in \{1, \ldots, m\}\}|\} \tag{2}$$

This equation simply expresses that the position of node e_s is encoded as a vector, so that the m_{th} component of this vector is equal to the number of occurrences of node e_s at the m_{th} position of all walks in S_e.

Finally, we will encode the relative positions of the nodes in each walk:

$$\widehat{E} = \left(PE\left(w_0\right), PE\left(w_1\right), \ldots, PE\left(w_m\right)\right) \tag{3}$$

Representation of each position of each walk, i.e., $\widehat{E}$, is passed through a multi-layered perceptron (MLP) to obtain the corresponding position embedding:

$$f_0(\widehat{E}) = \mathrm{MLP}(\widehat{E}) \tag{4}$$

Iterative Update. Just like the human learning process [24], humans update their existing knowledge base when they gain new observations. In our condition, the existing knowledge base is the node scores to be discovered, which we call the node pool. When new nodes are reached, our spatio-temporal random walk module updates the importance scores in the node pool, including known nodes and new nodes. As shown in the Fig. 2, the query node is found in the historical

information, and then the nearest spatio-temporal neighbors are selected starting from the query node, called one-hop spatio-temporal neighbors. The node sequences are then fed into the GRU model to calculate their node importance. Subsequently, the spatio-temporal walking is performed again, starting from its one-hop neighbors. As the walking continues, the model's knowledge of the query subject node is constantly updated, and finally we make predictions using the node pool. We obtain the encoding of E as follows:

$$Encode(E) = GRU(\{F_1(h_i, f_1(t_i)) \oplus h_r \oplus f_0(\widehat{E})\}_{i=0,1,\ldots,m}) \tag{5}$$

where F_1 is the time-aware encoding function, h_i and h_r are the hidden representation of node and relation, respectively, f_1 is the time embedding function, $\widehat{E}$ is the relative position embeddings, $\oplus$ is the concatenation operation. The F_1 conducts nonlinear transformation as:

$$F_1(h_i, f_1(t_i)) = W_1(h_i \oplus f_1(t_i)) \tag{6}$$

where W_1 is the time-aware trainable parameter. For f_1, we adopt Bochner Time Embedding [23].

$$f_1(t) = [\cos(w_1 t), \sin(w_2 t), \ldots, \cos(w_d t), \sin(w_d t)] \tag{7}$$

where w_i's are learnable parameters.

To get the relevance of the discovered nodes to our query, we consider the node and relation information of the query and then update the seen entity scores in the node pool by computing the query-related attention scores using:

$$Att(E, q) = f(W_\lambda(h_s \oplus Encode(E) \oplus h_r)) \tag{8}$$

where Att is the attention score of the seen nodes regarding the query $q = (e_s, r_p, ?, t_s)$, W_λ is the weight matric for aggregating features from evolving node sequences and query, h_s and h_p denotes the embeddings of entity e_s and relation e_p related to the query, respectively, and $f(\cdot)$ is an activation function.

3.3 TRGAT

As for the TRGAT module, we only consider those objects in history that have a connection to a given subject under same relation r_q. We define $\mathbf{O}_t(e_{s_i}, e_{p_i})$ to represent the set of objects that have a relation r_p with the subject entity at a history timestamp $t(0 \leq t \leq t_s)$. The TRGAT module can be considered as a special kind of neighborhood aggregation. We assume that there are differences between different entities under the same relationship. We assign different weights to each edge by computing the attention.

$$a_{e_s, \mathbf{O}_t} = f(W_2(F_1(h_{e_s}, f_1(t)) \oplus h_{p_i} \oplus h_{\mathbf{O_t}})) \tag{9}$$

where $a_{e_s, \mathbf{O}_t}$ is the attention of $\mathbf{O}$ and the subject entity, W_2 is the relation-aware transformation matric, h_{p_i} and $h_{\mathbf{O}_t}$ is the embeddings of relation e_{p_i} and $\mathbf{O}_t$, respectively.

To get the relative attention values, a softmax function is applied over $a_{e_s,\mathbf{O}_t}$:

$$\alpha_{e_s,\mathbf{O}_i} = \text{softmax}(a_{e_s,\mathbf{O}_i}) = \frac{\exp(a_{e_s,\mathbf{O}_i})}{\sum_{n\in\mathcal{O}_t}\exp(a_{e_s,\mathbf{O}_n})} \tag{10}$$

We aggregate the representations of prior neighbors and weight them using the normalized attention scores, which is written as

$$\widetilde{h_{e_s}} = \sum_{n\in\mathbf{O}_t} \alpha_{e_s,n} h_n \tag{11}$$

After, we use the updated subject entity and the object entity in set $\mathbf{O}_t$ to update the scores in the node pool.

$$Att(e_s, \mathbf{O}_t) = f(W_\mu(\widetilde{h_{e_s}}, h_{\mathbf{O}_t})) \tag{12}$$

4 Experiments

In this section, we demonstrate the effectiveness of our model using five public datasets. We first explain the experimental setup, including datasets, implementation details, benchmark methods and evaluation metrics. After that, we discuss the experimental results. In particular, we also conduct several ablation studies to analyze the impact of entity/relationship embedding and various components of SWGAT.

4.1 Experimental Setup

Datasets. In previous studies, there are five typical TKGs commonly used, namely, ICEWS14, ICEWS18, WIKI, YAGO and GDELT. Integrated Crisis Early Warning System(ICEWS) dataset contains political events annotated with specific time, e.g. (Barack Obama, visit, Malaysia, 2014-02-19). ICEWS14 and ICEWS18 are subsets of ICEWS, corresponding to facts from 2014 and facts from 2018, respectively. It is worth noting that all time annotations in the ICEWS dataset are time points. WIKI and YAGO are knowledge bases with temporally associated facts. Global Database of Events, Language, and Tone(GDELT) dataset is an initiative to construct a global dataset of all events, connecting people, organizations, and news sources.

Baseline Methods. Our model is compared with two categories of models: static KG reasoning models and TKG reasoning models. DistMult [26], RGCN [18], ConvE [4] and RotateE [20] are selected as static models. Temporal methods include TA-DistMult [5], R-GCRN [19], HyTE [2], dyngraph2vecAE [7], EvolveGCN [16], know-Evolve [21], know-Evolve+MLP [21], DyRep [22], CyGNet [28], RE-Net [9], xERTE [8] and EvoKG [17]. We note that both T-GAP [10] and xERTE [8] use subgraph extraction and attention flow walks, but the former is used for the interpolation problem.

Table 1. Experiments results for the extrapolation task on five temporal datasets. Hits@N values are in percentage. The best score is in **Bold** and the second is underlined. The results of all the baseline methods are taken from [17].

	Method	ICEWS14			ICEWS18			WIKI			YAGO			GDELT		
		MRR	Hits@3	Hits@10	MRR	Hits@3	Hits@10	MRR	Hits@3	Hits@10	MRR	Hits@3	Hits@10	MRR	Hits@3	Hits@10
Static	DistMult [26]	9.72	10.09	22.53	13.86	15.22	31.26	27.96	32.45	39.51	44.05	49.70	59.94	8.61	8.27	17.04
	R-GCN [18]	15.03	16.12	31.47	15.05	16.49	29.00	13.96	15.75	22.05	27.43	31.24	44.75	12.17	12.37	20.63
	ConvE [4]	21.64	23.16	38.37	22.56	25.41	41.67	26.41	30.36	39.41	41.31	47.10	59.67	18.43	19.57	32.25
	RotateE [20]	9.79	9.37	22.24	11.63	12.31	28.03	26.08	31.63	38.51	42.08	46.77	59.39	3.62	2.26	8.37
Temporal	TA-DistMult [5]	11.29	11.60	23.71	15.62	17.09	32.21	26.44	31.36	38.97	44.98	50.64	61.11	10.34	10.44	21.63
	HyTE [2]	7.72	7.94	20.16	7.41	7.33	16.01	25.40	29.16	37.54	14.42	39.73	46.98	6.69	7.57	19.06
	dyngraph2vecAE [7]	6.95	8.17	12.18	1.36	1.54	1.61	2.67	2.75	3.00	0.81	0.74	0.76	4.53	1.87	1.87
	EvolveGCN [16]	8.32	7.64	18.81	10.31	10.52	23.65	27.19	31.35	38.13	40.50	45.78	55.29	6.54	5.64	15.22
	Know-Evovle [21]	0.05	0.00	0.10	0.11	0.00	0.47	0.03	0.00	0.04	0.02	0.00	0.01	0.11	0.02	0.10
	Know-Evolve+MLP [21]	16.81	18.63	29.20	7.41	7.87	14.76	10.54	13.08	20.21	5.23	5.63	10.23	15.88	15.69	22.28
	DyRep+MLP [22]	17.54	19.87	30.34	7.82	7.73	16.33	10.41	12.06	20.93	4.98	5.54	10.19	16.25	16.45	23.86
	R-GCRN+MLP [19]	21.39	23.60	38.96	23.46	26.62	41.96	28.68	31.44	38.58	43.71	48.53	56.98	18.63	19.80	32.42
	CyGNet [28]	22.83	25.36	39.97	25.43	28.95	43.86	33.89	36.10	41.86	52.07	56.12	63.77	18.05	19.11	31.50
	RE-Net [9]	23.91	26.63	42.70	26.81	30.58	45.92	31.55	34.45	42.26	46.37	51.95	61.59	19.44	20.73	33.81
	xERTE [8]	23.92	27.30	39.54	27.14	31.08	43.31	70.52	75.01	76.46	84.12	88.62	89.90	21.21	22.93	26.60
	EvoKG [17]	27.18	30.84	**47.67**	29.28	33.94	**50.09**	68.03	79.60	85.91	68.59	81.13	**92.73**	19.28	20.55	34.44
	SWGAT (our)	**27.81**	**31.75**	43.84	**30.15**	**35.12**	47.54	**82.76**	**84.44**	**86.83**	**87.63**	**90.52**	91.25	**22.10**	**24.54**	**36.49**

4.2 Results on TKG Reasoning

Table 1 summarizes the time-aware filtered results of link prediction task on the ICEWS14, ICEWS18, WIKI, YAGO,and GDELT datasets. The benchmark results are obtained from [17]. Our model outperforms basically all baseline methods on different datasets. Compared with the best benchmark model EvoKG [17], our model achieves 2.3% and 2.8% improvement in MRR and Hits@3 on the ICEWS14 dataset, 3% and 3.5% improvement in MRR and Hits@1 on the ICEWS18 dataset, 21.6%, 6% and 1.1% improvement in MRR, Hits3 and Hits10 on the WIKI dataset, 27.7% and 11.6% improvement in MRR and Hits3 on the YAGO dataset, 14.6%, 19.5% and 5.9% improvement in MRR, Hits3 and Hits10 on the GDELT dataset, respectively. And our model significantly outperforms all other benchmark methods on all metrics, indicating that learning time series and directly related spatio-temporal neighbors can help the model find correct target nodes. In particular, on the YAGO dataset, it is 4.2% and 27.7% higher MRR than xERTE [8] and EvoKG [17], respectively, probably due to the fact that the YAGO dataset contains events that occur with relative regularity and have a small number of neighbouring entities, which allow SWGAT to find target entities quickly and accurately. Among the benchmark methods, the static methods have relatively poor performance because they do not consider temporal information.

4.3 Ablation Study

To evaluate the effectiveness of SWGAT, we conduct ablation studies on dataset ICEWS18.

Impact of Two Components. To verify the importance of each component of SWGAT, we mask the saptio-temporal random walk and the TRGAT module, respectively. The experimental results are shown in Fig. 3. We find that the spatio-temporal walk component has a considerable impact on the performance of the model. From the results, we can obtain that SWGAT performs better than when the two components act alone, which suggests that SWGAT can integrate the properties of the two components, i.e., exploration of temporal evolution and structural features.

Embedding Size. We set the embedding size (both temporal and structural embedding) to 100, 200, 300, and 400. As shown in the Fig. 4, the best results on ICEWS18 dataset are achieved with embedding size 200. However, a larger embedding size, such as 400, will hurt the performance, probably because too large dimensions can lead to overfitting.

Number of Walks. Figure 5 shows the performance of three evaluation metrics on the ICEWS18 dataset with different number of walks. We observe that the performance increases as the number of walks increases. However, the performance is close to the saturation state when the number of walks reaches 20, i.e., only a small improvement in performance can be obtained regardless of the increase in the number of walks.

Inductive Link Prediction. As time evolves, new nodes may appear, such as new users or new posts. Therefore, a good model should have good inductive representation capability to cope with unseen entities. For example, in the test set of ICEWS14, we have a quadruple *(Mehdi Hasan, Make an appeal or request, Citizen(India), 2014-11-12)*. The entity *Mehdi Hasan* does not appear in the training set, which means that the quadruple contains an entity that the model does not observe in the training phase. Specifically, we divide the test dataset

Table 2. Experiments results for inductive link prediction on ICEWS18 datasets. Hits@N values are in percentage. The best score is in **Bold** and the second is underlined.

Methods	**ICEWS18 test set (Mixed entities)**				**ICEWS18 test set (seen entities)**				**ICEWS18 test set (unseen entities)**			
Model	MRR	Hits@1	Hits@3	Hits@10	MRR	Hits@1	Hits@3	Hits@10	MRR	Hits@1	Hits@3	Hits@10
CyGNet [28]	25.43	16.09	28.95	43.86	26.37	16.68	30.05	45.51	2.08	1.18	1.99	3.45
RE-Net [9]	26.81	17.6	30.58	45.92	27.22	17.29	31.04	46.61	2.19	1.15	2.28	3.84
xERTE [8]	27.14	19.64	31.08	43.31	28.01	19.93	32.12	44.92	6.96	4.82	6.45	8.57
EvoKG [17]	29.28	19.02	33.94	**50.09**	30.39	19.58	35.16	**51.83**	4.13	2.71	4.45	6.34
SWGAT	**30.15**	**21.52**	**35.12**	47.54	**31.24**	**22.14**	**36.33**	49.67	**7.52**	**6.01**	**8.52**	**10.68**

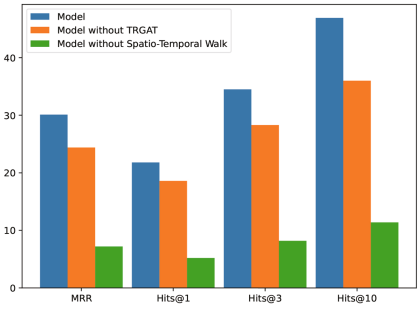

Fig. 3. Time-aware filtered metrics of SWGAT with or without the TRGAT module on ICEWS18.

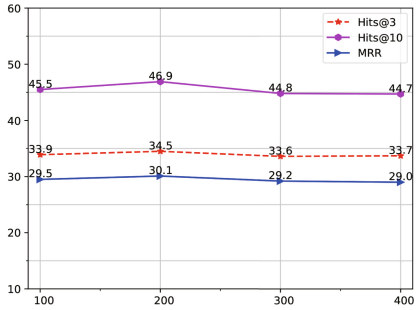

Fig. 4. Embedding Size.

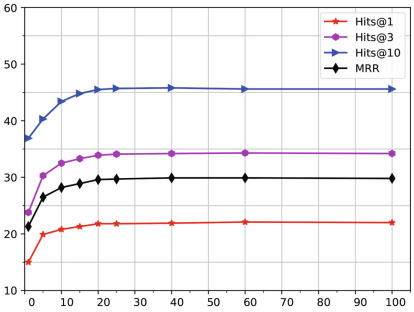

Fig. 5. Number Walks.

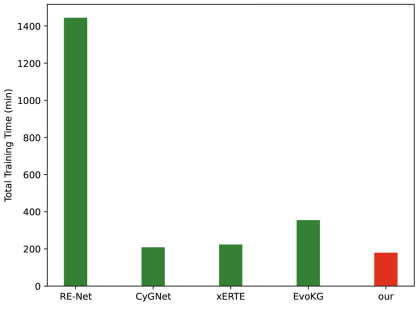

Fig. 6. Training Time Cost.

into three categories, seen entities, unseen entities and mixed entities (containing both seen and unseen entities), and the results are shown in Table 2. We find that our proposed method SWGAT achieves optimal performance on all evaluation metrics, showing the good performance of our model SWGAT on inductive link prediction.

Time Cost. It is important to get strong performance while keeping the training time short on the model. To investigate the balance between accuracy and efficiency of SWGAT, we report the total training time for convergence of our model and four benchmarks on Fig. 6. We find that although Re-Net [9] is one of the strongest performance baselines, it takes almost 13 times longer to train compared with CyGNet [28] and xERTE [8]. Whereas our model ensures shorter training time while maintaining state-of-the-art performance for the extrapolation problem, which shows the superiority of our model.

5 Conclusions

Representing and reasoning about temporal knowledge is a challenging problem. In this paper, we propose a model for temporal graph prediction that learns the evolution patterns of entities and relations over time and spatio-temporal subgraph specific to the query entities and relations, respectively. Compared with state-of-the-art methods, extensive experiments on five benchmark datasets demonstrate the effectiveness of the model on the extrapolation problem. It is necessary to study more efficient node/edge sampling strategies, because the efficiency of the model is limited by the choice of nodes when the model is extended to real-life temporal graph with tens of billions of nodes. Interesting future work includes developing fast and efficient versions and applications in streaming scenarios.

Acknowledgment. This work is supported by grants from Shengze Li's National Natural Science Foundation of China (No. 11901578).

References

1. Bordes, A., Usunier, N., Garcia-Duran, A., Weston, J., Yakhnenko, O.: Translating embeddings for modeling multi-relational data. In: Advances in Neural Information Processing Systems, vol. 26 (2013)
2. Dasgupta, S.S., Ray, S.N., Talukdar, P.: Hyte: hyperplane-based temporally aware knowledge graph embedding. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pp. 2001–2011 (2018)
3. Deng, S., Rangwala, H., Ning, Y.: Dynamic knowledge graph based multi-event forecasting. In: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 1585–1595 (2020)
4. Dettmers, T., Minervini, P., Stenetorp, P., Riedel, S.: Convolutional 2d knowledge graph embeddings. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 32 (2018)
5. García-Durán, A., Dumančić, S., Niepert, M.: Learning sequence encoders for temporal knowledge graph completion. arXiv preprint arXiv:1809.03202 (2018)
6. Goel, R., Kazemi, S.M., Brubaker, M., Poupart, P.: Diachronic embedding for temporal knowledge graph completion. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, pp. 3988–3995 (2020)
7. Goyal, P., Chhetri, S.R., Canedo, A.: dyngraph2vec: capturing network dynamics using dynamic graph representation learning. Knowl. Based Syst. **187**, 104816 (2020)
8. Han, Z., Chen, P., Ma, Y., Tresp, V.: Explainable subgraph reasoning for forecasting on temporal knowledge graphs. In: International Conference on Learning Representations (2020)
9. Jin, W., Qu, M., Jin, X., Ren, X.: Recurrent event network: autoregressive structure inference over temporal knowledge graphs. arXiv preprint arXiv:1904.05530 (2019)
10. Jung, J., Jung, J., Kang, U.: T-gap: Learning to walk across time for temporal knowledge graph completion. arXiv preprint arXiv:2012.10595 (2020)
11. Koren, Y., Bell, R., Volinsky, C.: Matrix factorization techniques for recommender systems. Computer **42**(8), 30–37 (2009)

12. Liu, Z., Xiong, C., Sun, M., Liu, Z.: Entity-duet neural ranking: Understanding the role of knowledge graph semantics in neural information retrieval. arXiv preprint arXiv:1805.07591 (2018)
13. Nathani, D., Chauhan, J., Sharma, C., Kaul, M.: Learning attention-based embeddings for relation prediction in knowledge graphs. arXiv preprint arXiv:1906.01195 (2019)
14. Nickel, M., Rosasco, L., Poggio, T.: Holographic embeddings of knowledge graphs. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 30 (2016)
15. Nickel, M., Tresp, V., Kriegel, H.P.: A three-way model for collective learning on multi-relational data. In: ICML (2011)
16. Pareja, A., et al.: EvolveGCN: evolving graph convolutional networks for dynamic graphs. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, pp. 5363–5370 (2020)
17. Park, N., Liu, F., Mehta, P., Cristofor, D., Faloutsos, C., Dong, Y.: 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 (2022)
18. Schlichtkrull, M., Kipf, T.N., Bloem, P., van den Berg, R., Titov, I., Welling, M.: Modeling relational data with graph convolutional networks. In: Gangemi, A., Navigli, R., Vidal, M.-E., Hitzler, P., Troncy, R., Hollink, L., Tordai, A., Alam, M. (eds.) ESWC 2018. LNCS, vol. 10843, pp. 593–607. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-93417-4_38
19. Seo, Y., Defferrard, M., Vandergheynst, P., Bresson, X.: Structured sequence modeling with graph convolutional recurrent networks. In: Cheng, L., Leung, A.C.S., Ozawa, S. (eds.) ICONIP 2018. LNCS, vol. 11301, pp. 362–373. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-04167-0_33
20. Sun, Z., Deng, Z.H., Nie, J.Y., Tang, J.: Rotate: knowledge graph embedding by relational rotation in complex space. arXiv preprint arXiv:1902.10197 (2019)
21. Trivedi, R., Dai, H., Wang, Y., Song, L.: Know-evolve: deep temporal reasoning for dynamic knowledge graphs. In: international Conference on Machine Learning, pp. 3462–3471. PMLR (2017)
22. Trivedi, R., Farajtabar, M., Biswal, P., Zha, H.: DYREP: learning representations over dynamic graphs. In: International Conference on Learning Representations (2019)
23. Wang, Y., Chang, Y.Y., Liu, Y., Leskovec, J., Li, P.: Inductive representation learning in temporal networks via causal anonymous walks. arXiv preprint arXiv:2101.05974 (2021)
24. Wang, Y., Chiew, V.: On the cognitive process of human problem solving. Cogn. Syst. Res. **11**(1), 81–92 (2010)
25. Wang, Z., Zhang, J., Feng, J., Chen, Z.: Knowledge graph embedding by translating on hyperplanes. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 28 (2014)
26. Yang, B., Yih, W.t., He, X., Gao, J., Deng, L.: Embedding entities and relations for learning and inference in knowledge bases. arXiv preprint arXiv:1412.6575 (2014)
27. Zhao, M., Zhang, L., Kong, Y., Yin, B.: Temporal knowledge graph reasoning triggered by memories. arXiv preprint arXiv:2110.08765 (2021)
28. Zhu, C., Chen, M., Fan, C., Cheng, G., Zhan, Y.: Learning from history: modeling temporal knowledge graphs with sequential copy-generation networks. arXiv preprint arXiv:2012.08492 (2020)