


AUTHOR QUERY FORM

| | | |
|---|--|---|
|  | <p>Journal: NEUCOM</p> <p>Article Number: 126390</p> | <p>Please e-mail your responses and any corrections to:</p> <p>E-mail: corrections.esch@straive.com</p> |
|---|--|---|

Dear Author,

Please check your proof carefully and mark all corrections at the appropriate place in the proof. **It is crucial that you NOT make direct edits to the PDF using the editing tools as doing so could lead us to overlook your desired changes.** Rather, please request corrections by using the tools in the Comment pane to annotate the PDF and call out the changes you would like to see. To ensure fast publication of your paper please return your corrections within 48 hours.

For correction or revision of any artwork, please consult <http://www.elsevier.com/artworkinstructions>.

Any queries or remarks that have arisen during the processing of your manuscript are listed below and highlighted by flags in the proof.

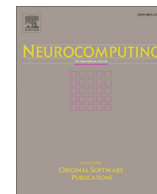
| Location in article | Query / Remark: Click on the Q link to find the query's location in text Please insert your reply or correction at the corresponding line in the proof |
|--|---|
| <p><u>Q1</u></p> <p><u>Q2</u></p> <p><u>Q3</u></p> | <p>Your article is registered as a regular item and is being processed for inclusion in a regular issue of the journal. If this is NOT correct and your article belongs to a Special Issue/Collection please contact c.samiullah@elsevier.com immediately prior to returning your corrections.</p> <p>The author names have been tagged as given names and surnames (surnames are highlighted in teal color). Please confirm if they have been identified correctly.</p> <p>The country name has been inserted for the affiliation 'b'. Please check, and correct if necessary.</p> <div data-bbox="416 1885 981 1987"> <p>Please check this box or indicate your approval if you have no corrections to make to the PDF file</p> <input data-bbox="868 1902 940 1966" type="checkbox"/> </div> |

Thank you for your assistance.



Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Hierarchical graph attention network for temporal knowledge graph reasoning

Pengpeng Shao^a, Jiayi He^a, Guanjin Li^a, Dawei Zhang^a, Jianhua Tao^{b,*}

^aThe State Key Laboratory of Multimodal Artificial Intelligence Systems, Institute of Automation, Chinese Academy of Sciences, Beijing, China

^bDepartment of Automation, Tsinghua University, China

ARTICLE INFO

Article history:

Received 27 August 2021

Revised 10 November 2022

Accepted 26 May 2023

Available online xxxx

Keywords:

Temporal knowledge graphs

Graph attention network

Reasoning

ABSTRACT

Temporal knowledge graphs (TKGs) reasoning has attracted increasing research interest in recent years. However, most of the existing TKGs reasoning models aim to learn a dynamic entity representation by binding timestamps information with the entities, neglecting to learn adaptive entity representation that is valuable to the query from relevant historical facts. To this end, we propose a Hierarchical Graph Attention neTwork (HGAT) for the TKGs reasoning task. Specifically, we design a hierarchical neighbor encoder to model the time-oriented and task-oriented roles of the entities. The time-aware mechanism is developed in the first layer to differentiate the contributions of query-relevant historical facts at different timestamps to the query. The designed relation-aware attention is used in the second layer to discern the contributions of the structural neighbors of an entity. Through this hierarchical encoder, our model can absorb valuable knowledge effectively from the relevant historical facts, and thus learn more expressive adaptive entity representation for the query. Finally, we evaluate our model performance on four TKGs datasets and justify its superiority against various state-of-the-art baselines.

© 2023 Elsevier B.V. All rights reserved.

1. Introduction

Knowledge graphs (KGs) are graph-structured representations of facts, which are represented as a collection of triples (s, r, o) , with subject s , relation r , and object o . Due to the high cost of annotating facts, most of the KGs are far from completion, thus KGs reasoning becomes an important task and attracts rising interest in recent years. As its powerful expressiveness over structured knowledge, KGs have widespread applications like recommendation system [1], question answering [2,3], and information retrieval [4].

With the continuous increase of structured data, however, knowledge presents dynamic and temporal gradually, and most of the facts are held in a specific period or a certain time point. Therefore, reasoning over the TKGs becomes an increasingly important task. On a TKGs with timestamps varying from t_0 to t_T , TKGs reasoning primarily has two settings - interpolation and extrapolation. In the interpolation reasoning, new facts are inferred out for time t ($t_0 \leq t \leq t_T$) by using historical information and future information. Corresponding extrapolation setting only employs historical knowledge to predict future new facts for time

t ($t > t_T$). In this work, we focus on the extrapolation TKGs reasoning.

Recently, many works have been accomplished on the TKGs reasoning task. Interpolation methods DE-Simple [5], TNTComplex [6], and TuckERTNT [7] aim to design a decoder that binds temporal information with entity or relation in score function. Extrapolation models Know-Evolve [8] and its extension DyRep [9] model the occurrence of the facts as a temporal point process, and predict future facts assuming ground truths of the preceding events are given at inference time. CyGNet [10] observes that many facts often show a repeated pattern along the timeline, then presents a copy mechanism that can learn from the known historical facts and a generation mechanism to collaboratively predict future facts. ReNET [11] models the occurrence of a fact as a probability distribution conditioned on temporal sequences of past knowledge graphs, and then designs an autoregressive architecture for predicting future facts. However, these models are devoted to developing a new decoder function to predict the answer, neglecting to score the relevant historical facts which have different contributions to the query and learn a powerful encoder from the relevant historical facts.

In this work, we design a Hierarchical Graph Attention neTwork (HGAT) for TKGs reasoning. Fig. 1 presents an example of a query and its relevant historical facts. The left part of Fig. 1 is the relevant

* Corresponding author.

E-mail address: jhtao@tsinghua.edu.cn (J. Tao).

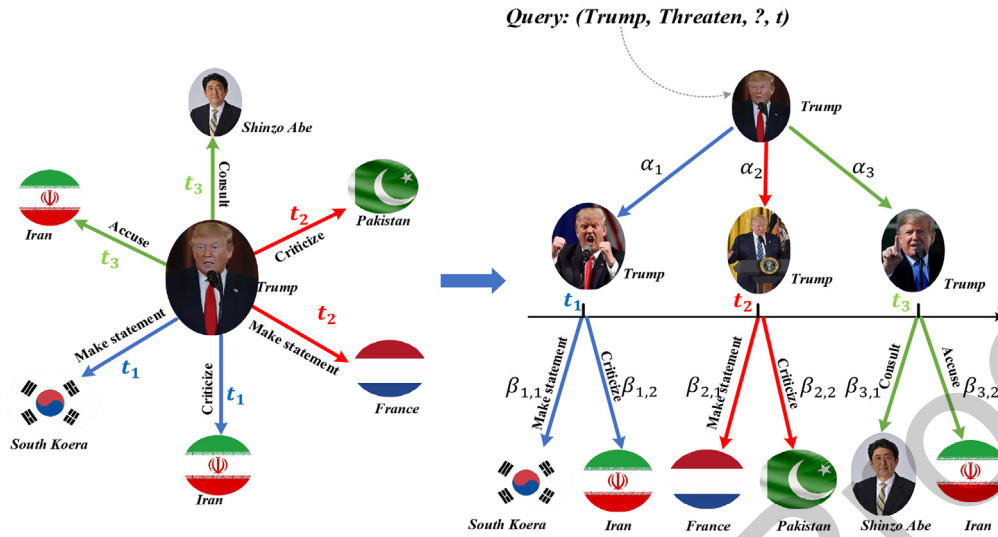


Fig. 1. The left is the query-relevant history subgraph. The right is the hierarchical structure of the subgraph.

historical facts about the query. We find that the entity *Trump* has six neighbors in three historical timestamps t_1, t_2 , and t_3 . Then we categorize the neighbors by timestamps and show them in the right part of Fig. 1. Based on this hierarchical structure, we propose an HGAT for TKGs reasoning. First, we present an assumption that the events that are closer to the query in time distance have a greater contribution to the query. Thus we design a graph convolutional network (GCN) with time-aware attention mechanism in the first layer to adaptively aggregate the neighbors *Trump* at historical time t_1, t_2 , and t_3 into the central entity *Trump* at time t , and enable $\alpha_3 > \alpha_2 > \alpha_1$. Second, the neighbors of an entity have different impacts associated with different task relations. Thus we present a relation graph convolutional network (RGCN) with relation-aware attention mechanism in the second layer to adaptively learn the representation of entity *Trump* at time t_1, t_2 , and t_3 from their respective structural neighbors. In this case, the task relation *Threaten* is more relevant with *Accuse* than *Consult* in semantic aspect at time t_3 , that is, $\beta_{3,2} > \beta_{3,1}$. Finally, we develop an information aggregator to integrate the two independent graph attention networks into a whole to learn an adaptive entity representation for the query. The extensive experimental results show that our method obtains considerable performance compared with existing approaches and demonstrate the effectiveness and superiority of our method. Overall, our contributions are as follows:

- To learn the valuable knowledge for the query from relevant historical facts, we propose an assumption from the perspective of time that the events that are closer to the query in time distance have a greater contribution to the query, thus designing a graph convolutional network with time-aware attention mechanism to model this assumption and aggregate the temporal neighbors into the central entity.
- From the perspective of semantics, we propose a second assumption that the neighbors of an entity have different impacts associated with different task relations, thus developing a relation graph convolutional network with relation-aware attention mechanism to learn the structural representation of the entity adaptively.
- We design an information aggregator to integrate the two independent graph attention networks into a whole to assign a relatively reasonable weight to each historical fact and learn an adaptive entity representation for the query.

- Experimental studies on four TKGs datasets demonstrate that our model achieves considerable performance compared with existing models.

2. Related work

2.1. Static KGs reasoning

Recent years have witnessed increasing attention on the static KGs reasoning. Overall, the static KGs reasoning methods can be roughly classified into three categories: KGs embedding-based approaches, path-based approaches, and logical rule-based approaches.

Among these approaches, the most popular one is KGs embedding-based approaches, its core idea is to project the entities and relations in KGs into vector space, then model the relation between the head and tail entity. This kind of method also broadly falls into three paradigms: (i) Translational distance-based models, the most representative method is TransE [12], which views relation as a translation operation from head entity to tail entity. Since then, many improved versions have been proposed, such as TransD [13], and TransH [14]. (ii) Tensor factorization-based models, which are popular with their high efficiency and powerful function, like RESCAL [15], DisMult [16], ComplEx [17], TuckER [18]. (iii) Neural network-based models, the representative ConvE [19] employs a deep neural network to enhance the interaction between entity and relation, R-GCN [20] utilizes a relational graph neural network to model KGs.

As for path-based approaches, they target at learning relational paths in KGs. For instance, PRA [21,22] uses random walk with restarts to conduct multiple bounded depth-first search processes to learn relational paths. Different from PRA, DeepPath [23], M-Walk [24], and MINERVA [25] frame the path-finding problem as Markov decision process (MDP) and utilize reinforcement learning to maximize the expected return. Moreover, DIVA [26] takes the KGs reasoning problem as an inference problem in a probabilistic graphical framework and resolves it from a variational inference perspective.

Logical rule-based approaches are able to leverage domain knowledge to boost reasoning performance, but the logical rules are very 'hard' in the reasoning process, which makes the test facts either right or wrong. Even worse, the logical rules may be contra-

dictory in some cases and have a certain amount of uncertainty. Markov Logic Network (MLN) [27] is able to learn the weights of logical rules in a probabilistic graphical framework to soften the rules, which can effectively model the uncertainty. PLogicNet [28] and ExpressGNN [29] which combines the advantages of KGs embedding and MLN, can be efficiently optimized with the variational EM algorithm in the reasoning process. RNNLogic [30] learns logical rules which are viewed as latent variables and simultaneously predicts the answer in the EM framework.

2.2. Temporal KGs reasoning

Most of the existing TKGs reasoning works primarily focus on extending static KGs reasoning to TKGs, and the key idea of TKGs reasoning is to model the dynamic nature of the facts. TKGs reasoning has two settings: interpolation [31,5–7] and extrapolation [11,10,32]. For interpolation reasoning, the model aims to infer new facts at history timestamps by employing historical information and future information. For instance, t-TransE [31] integrates the temporal constraints on relation into the object function of TransE. HyTE [33] maps entity and relation into the hyperplane space modeled by temporal information. TA-DistMult [34] leverages recurrent neural networks to learn time-aware relation representation to express the temporality of the fact. DE-Simple [5], TNTComplEx [6], and TuckERTNT [7] combine temporal information with entity or relation to model the dynamic nature of the facts. However, these methods model the facts at all timestamps and predict the new facts at history timestamps, they can not predict the new facts at future timestamps. Corresponding extrapolation reasoning aims to predict future new facts based on historical information. Know-Evolve [8] and its extension DyRep [9] model the occurrence of the facts as temporal point process to learn evolving entity representations. Re-NET [11] models the occurrence of a fact as a probability distribution conditioned on temporal sequences of past knowledge graphs and designs an autoregressive architecture for predicting future facts. CyGNet [10] which is most related to our work argues that a reasoning model is capable of learning much from the known historical facts based on the observation that many facts occur repeatedly along with history, and presents a copy-generation mechanism to predict future facts collaboratively. In essence, the copy mechanism provides a candidate answer space to the generation mechanism to predict future facts collaboratively. The answers in the space have an equal impact on predicting future facts, which is not consistent with human intuition. In this work, we design a hierarchical graph attention network to assign an adaptive weight for each query-relevant historical fact from human intuition.

3. Problem formulation

Let \mathcal{E} , \mathcal{R} , and \mathcal{T} denote a finite set of entities, relations, and timestamps, respectively. The facts in TKGs are defined by quadruples $(s, r, o, t) \subseteq \mathcal{E} \times \mathcal{R} \times \mathcal{E} \times \mathcal{T}$, and TKGs are built upon a sequence of quadruples based on their timestamps, $\mathcal{G} = \{\mathcal{G}_t\}_{t=1}^T = \{(s, r, o, t_i)\}_{t_i=1}^T$, where $s, o \in \mathcal{E}$, $r \in \mathcal{R}$. This represents a collection of event facts at different timestamps in ascending order of their timestamps. Given a query $(s_q, r_q, ?, t_q)$, the observed historical facts $\mathcal{O} = \{(s, r, o, t_i) \in \mathcal{G}_t | t_i < t_q\}$. The observed query-relevant historical facts $\mathcal{O}_r = \{(s_q, r, o, t_i) \in \mathcal{G}_t | t_i < t_q\}$, which have same head entity with query, and $\mathcal{O}_r \subseteq \mathcal{O}$. In addition, we denote the set of prior neighbours of query node s_{q,t_q} as $\mathcal{N}_p(s_{q,t_q}) = \{(s_{q,t_i}) | (s_{q,t_i}, r, o, t_i) \in \mathcal{G}_t, t_i < t_q\}$, $\mathcal{N}_s(s_{q,t_i}) = \{(o) | (s_{q,t_i}, r, o, t_i) \in \mathcal{G}_t, t_i < t_q\}$ represents the structural neighbours of s_{q,t_i} . Our TKGs reasoning task is

to predict the missing object o_q^* based on the observed query-relevant historical facts \mathcal{O}_r .

4. The proposed approach

In this section, we provide the technical details of our model. Fig. 2 presents the framework of single-layer HGAT, which follows the encoder-decoder pattern and can be easily generalized to multi-layer networks. The details of the encoder and decoder are presented as follows.

4.1. Encoder

The encoder includes two GCN with attention mechanisms and an information aggregator that is used to integrate these two GCN into a whole to learn the adaptive representation of entities.

GCN with Time-Aware Attention The designed GCN with time-aware attention is employed to aggregate prior neighbours of the query node into the central entity, and then incorporate the temporal information into GCN to differentiate the importance of prior neighbors. Specifically, given the query subject entity s_{q,t_q} , we aim to learn valuable information from its prior neighbors $\mathcal{N}_p(s_{q,t_q}) = \{s_{q,t_i}\}_{t_i=1}^{t_q-1}$ by GCN. Furthermore, we argue that the facts that are closer to the query in time distance have greater contributions to the query. That is, the facts at t_3 should have a larger impact on predicting the query at t than the facts at t_1 and t_2 in Fig. 1. Thus we incorporate the designed time-aware attention mechanism into GCN to absorb valuable neighboring information, and the time-aware attention is defined as follows,

$$\alpha_{t_i} = \text{softmax}(a_{t_i}) = \frac{e^{(t_i - t_q)/\tau_t}}{\sum_{t_i=1}^{t_q-1} e^{(t_i - t_q)/\tau_t}} \quad (1)$$

where τ_t is the temperature hyperparameter, t_i represents the timestamps of one of the relevant historical facts, and $t_i < t_q$. The GCN with time-aware attention is described as follows,

$$\mathbf{s}_{q,t_q}^{(k+1)} = \sigma \left(\sum_{t_i=1}^{t_q-1} \alpha_{t_i} \mathbf{s}_{q,t_i}^{(k)} + \mathbf{s}_{q,t_q}^{(k)} \right) = \sigma \left(\sum_{t_i=1}^{t_q} \alpha_{t_i} \mathbf{s}_{q,t_i}^{(k)} \right) \quad (2)$$

where $\alpha_{t_q} = 1$, $\sigma(\cdot)$ is an activation function, $\mathbf{s}_{q,t_q}^{(k)}$ denotes the hidden state of the query subject entity in the k -th layer of the neural network, \mathbf{s}_{q,t_i} represents the prior neighbours of query node s_{q,t_q} . Next, we will utilize the structural neighbours information of s_{q,t_i} to learn the representation of s_{q,t_i} , respectively.

RGCN with Relation-Aware Attention In this part, we present RGCN with relation-aware attention to learn the prior neighbors s_{q,t_i} of query node s_{q,t_q} . As the relation of an entity reflects the roles of the entity, the structural neighbour of the s_{q,t_i} may make different contributions to s_{q,t_i} , thus making different contributions to the queries. Therefore, we design a relation-aware attention mechanism to differentiate their contributions by task-relation r_q . The RGCN with relation-aware attention is illustrated as follows,

$$\mathbf{s}_{q,t_i}^{(k+1)} = \sum_{i=1}^{|\mathcal{N}_s(s_{q,t_i})|} \beta_{t,i} \phi(r_{t,i}^{(k)}, \mathbf{o}_{t,i}^{(k)}) \quad (3)$$

where $\phi: \mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R}^d$ is a composition operator, which is used to combine object and relation to obtain the subject, we use a simple but effective multiplication operator here. $r_{t,i}$ and $\mathbf{o}_{t,i}$ are i -th neighboring relation and object of s_{q,t_i} .

$$\phi(r_{t,i}^{(k)}, \mathbf{o}_{t,i}^{(k)}) = \mathbf{r}_{t,i}^{(k)} * \mathbf{o}_{t,i}^{(k)} \quad (4)$$

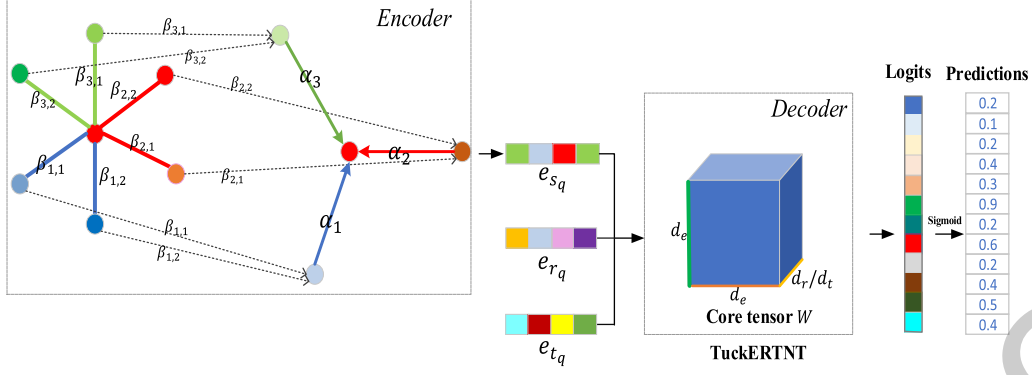


Fig. 2. The framework of our end-to-end Hierarchical Graph Attention Network. For the encoder, a stack of multiple two-layer GCN with attention mechanism layers builds an entity embedding. For the decoder, the learned entity embedding, initial relation embedding, and time embedding are fed into TuckERTNT to predict the missing object.

$$\beta_{t,i}^{(k)} = \frac{\exp(\psi(r_q, r_{t,i})/\tau_r)}{\sum_{i=1}^{|\mathcal{N}(s_{q,t_i})|} \exp(\psi(r_q, r_{t,i})/\tau_r)} \quad (5)$$

here, τ_r is the temperature hyperparameter, and the metric function $\psi(\cdot)$ is employed to compute the relevance score by a bilinear dot product. That means, the relation is more related to the task relation, and the corresponding neighboring information would contribute more to s_{q,t_i} .

$$\psi(r_q, r_{t,i}) = \mathbf{r}_q^T \mathbf{W}_s^{(k)} \mathbf{r}_{t,i} \quad (6)$$

where $\mathbf{W}_s \in \mathbb{R}^{d \times d}$ are learnable parameters.

Information Aggregator RGCN with relation-aware attention first aggregates the structural neighbor information into central entity s_{q,t_i} , then GCN with time-aware attention differentiates the contribution of s_{q,t_i} to query node based on timestamps information, and incorporates prior neighborhood information s_{q,t_i} into s_{q,t_q} . The information aggregator integrates the two independent GCN into a whole to learn an adaptive representation for query entity s_{q,t_q} .

$$\mathbf{s}_{q,t_q}^{(k+1)} = \sigma \left(\sum_{t_i=1}^{t_q} \alpha_{t_i} \sum_{i=1}^{|\mathcal{N}_s(s_{q,t_i})|} \beta_{t_i,i} \phi(r_{t_i,i}^{(k)}, o_{t_i,i}^{(k)}) \right) \quad (7)$$

The total contribution of i -th relevant fact (s_{q,t_i}, r, o, t_i) at time t_i to query is $\mu_{t,i} = \alpha_{t_i} * \beta_{t,i}$. It is worth noting that this information aggregator may omit valuable information from relevant facts in the distance. For instance, we assume the relevant-query fact $(s_{t_0}, r_{t_0}, o_{t_0}, t_0)$ in the distance have much impact on answering the query $(s_q, r_q, ?, t_q)$, but the α_{t_0} would pull down the contribution of the fact to the query. To alleviate this problem, we present an improved version of the information aggregator as follow,

$$\mathbf{s}_{q,t_q}^{(k+1)} = \sigma \left(\sum_{t_i=1}^{t_q} \sum_{i=1}^{|\mathcal{N}_s(s_{q,t_i})|} (\alpha_{t_i} + \beta_{t,i}) \phi(r_{t_i,i}^{(k)}, o_{t_i,i}^{(k)}) \right) \quad (8)$$

This information aggregator can be viewed as a composition operation of the two independent graph attention networks. To a certain extent, the information aggregator mitigates the impact of the time-aware attention mechanism on the valuable information in the distance. From another perspective, Eq. 8 can be viewed as a graph attention network with a voting mechanism, that is, the time-aware mechanism and the relation-aware attention mechanism assign a weight to the query-relevant historical fact respectively according to their own criteria, then the information aggregator combines the two attention score and is more consistent with our

real practice of scoring relevant historical facts from multiple perspectives. Intuitively, the importance of the two attention mechanisms may be different, thus we add the importance factor to Eq. 8 and obtain Eq. 9 as follows,

$$\mathbf{s}_{q,t_q}^{(k+1)} = \sigma \left(\sum_{t_i=1}^{t_q} \sum_{i=1}^{|\mathcal{N}_s(s_{q,t_i})|} (\gamma * \alpha_{t_i} + (1 - \gamma) * \beta_{t,i}) \phi(r_{t_i,i}^{(k)}, o_{t_i,i}^{(k)}) \right) \quad (9)$$

where γ controls the importance of each attention mechanism.

4.2. Decoder

Most of the existing TKGs reasoning models can be employed as our decoder. In this work, we use TuckERTNT [7] as our decoder model due to its powerful expressive ability. TuckERTNT is a Tucker decomposition model of order-4 tensor for TKGs reasoning task, the core tensor that contains a large number of parameters contributes to increasing the interactions between input entities and relations and fitting the large-scale discrete temporal data. The scoring function of TuckERTNT is presented as follows,

$$f(\mathbf{e}_{s_q}, \mathbf{e}_{r_q}, \mathbf{e}_{o_q}, \mathbf{e}_{t_q}) = \langle \mathbf{W}; \mathbf{e}_{s_q}, \mathbf{e}_{r_q}^t \odot \mathbf{e}_{t_q} + \mathbf{e}_{r_q}, \mathbf{e}_{o_q} \rangle \quad (10)$$

where \odot denotes dot product, $\mathbf{W} \in \mathbb{R}^{d_e \times d_r \times d_e}$ is core tensor. Given the query $(s_q, r_q, ?, t_q)$, predicting the missing object can be viewed as a multi-class classification task, where each class corresponds to one object entity. Then we can minimize the following multi-class loss function to train the proposed model,

$$\mathcal{L} = - \sum_{t \in \mathcal{T}} \sum_{i \in \mathcal{Y}} \sum_{n=1}^N \mathbf{o}_{it} \ln \mathbf{y}_{in} \quad (11)$$

where \mathbf{y} is the set of node indices that have labels, \mathbf{y}_{in} is the n -th entry of the decoder output for the i -th labeled node, \mathbf{o}_{it} denotes its respective ground truth label in the snapshot \mathcal{G}_t . In practice, we employ gradient descent techniques to train the proposed model.

5. Experiments

5.1. Datasets

We evaluate the proposed model on four datasets: ICEWS2014, ICEWS2018, WIKI, and YAGO. Among them, ICEWS2014 and ICEWS2018 are subsets of the ICEWS dataset [35] which contains information about political events with time annotations, and they contain event facts in 2014 and 2018, respectively. The WIKI dataset [36] consists of the temporal events from the Wikipedia data-

set. The YAGO [37] dataset is a TKGs base that incorporates information from Wikipedias, WordNet, and GeoNames. Here, we also use a subset of YAGO to evaluate our model. The detailed statistics of the four datasets are presented in Table 1.

5.2. Evaluation metrics

For each quadruple (s_q, r_q, o_q, t_q) in test set, we would predict two kinds of queries: $(s_q, r_q, ?, t_q)$ and $(?, r_q, o_q, t_q)$. However, the query $(?, r_q, o_q, t_q)$ can be transformed into $(o_q, r_q^{-1}, ?, t_q)$ to be predicted, so we apply data augmentation technique to generate training data (o, r^{-1}, s, t) for each quadruple (s, r, o, t) in training sets. For the answer o_q^* predicted by our model, we employ the Hits@1/3/10 scores and MRR [38] (mean reciprocal rank) to measure our model performance. MRR is the average of the reciprocal of the mean rank (MR) assigned to the true quadruple overall candidate quadruples, which is defined as follows,

$$\frac{1}{2 * |test|} \sum_{f=(s,r,o,t) \in test} \left(\frac{1}{R_{f,o}} + \frac{1}{R_{f,s}} \right). \quad (12)$$

Here, we denote $R_{f,s}$ and $R_{f,o}$ as the ranking for subject s and object o for the two queries, respectively. Hits@ n measures the percentage of test set rankings where a true quadruple is ranked within the top n candidate quadruples, that is,

$$\frac{1}{2 * |test|} \sum_{f=(s,r,o,t) \in test} (\mathbb{1}_{R_{f,o} \leq n} + \mathbb{1}_{R_{f,s} \leq n}) \quad (13)$$

Where $n = 1, 3, 10$, $\mathbb{1}_x$ is 1 if x holds and 0 otherwise.

In this work, we perform the time-aware filtering [39] scheme where all correct entities at the query timestamp except for the true query object are filtered out from the answers. Furthermore, the raw setting does not filter out the objects that have an impact on the reasoning results of the query, and the static filtering setting filters out all other objects that appear together with the query subject and relation at any timestamp. Compared with them, time-aware filtering obtains more reasonable results.

5.3. Implementation and hyperparameters

The proposed model is implemented in the Pytorch [40] framework and is optimized by Adam algorithm [41] with batch size 128. We set entity, relation, and time embedding dimensionality to 300. The temperature hyperparameter τ_t and τ_r are 0.1 and 3, respectively. Furthermore, we use batch normalization, input dropout, and embedding dropout to speed up training. Detailed

hyperparameter settings are reported in Table 2, including learning rate (lr), input dropout, embedding dropout, and importance factor γ . For fairness, we apply the above hyper-parameters to our baseline model TuckERTNT [7]. In addition, we reproduce the results marked (*) of some baseline models, and the results of other baseline models are taken from reported results of [42,39,43].

5.4. Baseline methods

Static Reasoning Methods: To evaluate the performance of the proposed model comprehensively, we compare the proposed model against a large number of static KGs reasoning models. Note that, we ignore the timestamps of the facts when conducting static reasoning methods on the TKGs dataset. The static KGs reasoning models to be compared include TransE [12], DistMult [16], ComplEx [17], TuckER [18].

Temporal Reasoning Methods: We also compare several state-of-the-art TKGs reasoning methods based on embedding, including interpolation and extrapolation TKGs reasoning methods. Interpolation TKGs reasoning methods include TTransE [31], TA-DistMult [34], DE-Simple [5], TNTComplEx [6], TuckERTNT [7]. Extrapolation TKGs reasoning methods include CyGNet [10], TANGO-Tucker [44], TANGO-DistMult [44], ReNET [11].

Variants of HGAT: To evaluate the importance of different components of HGAT, we propose four variants version of HGAT. (1) HGAT-w/o-RTA, which removes relation-aware and time-aware attention mechanism, and sets $\alpha_{t_i} = 1$ and $\beta_{t_i,i} = 1$ simultaneously. (2) HGAT-w-TA, it only equips with time-aware attention mechanism, and sets $\beta_{t_i,i} = 1$. (3) HGAT-w-RA, which equips with relation-aware attention mechanism, and sets $\alpha_{t_i} = 1$. (4) HGAT-dot, which takes the information aggregation method of Eq. 7 to learn the representation of entities.

5.5. Experimental results and analysis

5.5.1. Comparative study

Table 3 and Table 4 present the link prediction results of the proposed model and baseline models on four TKG datasets, and we have the following observations and analyses.

First, the results in Table 3 and Table 4 indicate that our model HGAT achieves considerable performance on these four datasets. To be specific, on MRR, HGAT is superior to ReNET 0.7% on ICEWS14, 6.5% on WIKI, 5.6% on YAGO. In addition, HGAT achieves similar performance to ReNET on ICEWS18. TKGs reasoning methods including T-TransE, TA-DistMult, De-simple, TNTComplEx, and TuckERTNT, only learn the dynamic representation of entity or

Table 1
Dataset statistics.

| Datasets | \mathcal{N}_{ent} | \mathcal{N}_{rel} | \mathcal{N}_{time} | Time Span | Time granularity | \mathcal{N}_{tr} | \mathcal{N}_{val} | \mathcal{N}_{ts} |
|-----------|---------------------|---------------------|----------------------|-----------|------------------|--------------------|---------------------|--------------------|
| ICEWS2014 | 7128 | 230 | 365 | 2014 | 24 h | 63685 | 13823 | 13222 |
| ICEWS2018 | 23033 | 256 | 304 | 2018 | 24 h | 373018 | 45995 | 49545 |
| WIKI | 12554 | 24 | 100 | – | 1 year | 539286 | 67538 | 63110 |
| YAGO | 10623 | 10 | 189 | – | 1 year | 161540 | 19523 | 20026 |

Table 2
Hyperparameter settings.

| Datasets | lr | input dropout | embedding dropout 1 | embedding dropout 2 | γ |
|-----------|-------|---------------|---------------------|---------------------|----------|
| ICEWS2014 | 0.01 | 0.3 | 0.4 | 0.5 | 0.5 |
| ICEWS2018 | 0.001 | 0.3 | 0.3 | 0.3 | 0.5 |
| WIKI | 0.01 | 0.3 | 0.4 | 0.5 | 0.7 |
| YAGO | 0.01 | 0.3 | 0.4 | 0.5 | 0.6 |

Table 3

The results of link prediction on ICEWS14 and ICEWS18 datasets. Compared metrics are time-aware filtered MRR and Hits@1/3/10. The best results are in bold.

| Method | ICEWS14 | | | | ICEWS18 | | | |
|----------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|
| | MRR | Hit@1 | Hit@3 | Hit@10 | MRR | Hit@1 | Hit@3 | Hit@10 |
| TransE | 0.224 | 0.133 | 0.256 | 0.412 | 0.122 | 0.058 | 0.128 | 0.251 |
| DistMult | 0.276 | 0.181 | 0.311 | 0.469 | 0.101 | 0.045 | 0.103 | 0.212 |
| CompLex | 0.308 | 0.215 | 0.344 | 0.495 | 0.210 | 0.118 | 0.234 | 0.398 |
| TuckER | 0.358* | 0.267* | 0.395* | 0.533* | 0.206 | 0.125 | 0.226 | 0.372 |
| T-TransE | 0.131 | 0.029 | 0.167 | 0.342 | 0.084 | 0.018 | 0.089 | 0.224 |
| TA-DistMult | 0.264 | 0.170 | 0.302 | 0.454 | 0.167 | 0.086 | 0.184 | 0.335 |
| De-simple | 0.326 | 0.244 | 0.356 | 0.491 | 0.193 | 0.115 | 0.218 | 0.348 |
| TNTCompLex | 0.304 | 0.212 | 0.340 | 0.472 | 0.275 | 0.195 | 0.308 | 0.428 |
| TuckERTNT | 0.363* | 0.272* | 0.401* | 0.542* | 0.261* | 0.164* | 0.288* | 0.447* |
| CyGNet | 0.327 | 0.236 | 0.363 | 0.506 | 0.249 | 0.159 | 0.282 | 0.426 |
| TANGO-Tucker | – | – | – | – | 0.286 | 0.193 | 0.321 | 0.470 |
| TANGO-DistMult | – | – | – | – | 0.267 | 0.192 | 0.300 | 0.440 |
| ReNET | 0.382 | 0.286 | 0.413 | 0.545 | 0.288 | 0.190 | 0.324 | 0.475 |
| HGAT | 0.389 | 0.297 | 0.424 | 0.564 | 0.285 | 0.196 | 0.327 | 0.466 |

Table 4

The results of link prediction on WIKI and YAGO datasets. Compared metrics are time-aware filtered MRR and Hits@1/3/10. The best results are in bold.

| Method | WIKI | | | | YAGO | | | |
|----------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|
| | MRR | Hit@1 | Hit@3 | Hit@10 | MRR | Hit@1 | Hit@3 | Hit@10 |
| TransE | 0.492* | 0.588* | 0.523* | 0.544* | 0.457* | 0.422* | 0.602* | 0.688* |
| DistMult | 0.496 | 0.461 | 0.528 | 0.541 | 0.548 | 0.473 | 0.598 | 0.685 |
| CompLex | 0.487* | 0.457* | 0.522* | 0.532* | 0.550* | 0.471* | 0.594* | 0.682* |
| TuckER | 0.500 | 0.461 | 0.536 | 0.548 | 0.548 | 0.474 | 0.596 | 0.689 |
| T-TransE | 0.292 | 0.216 | 0.344 | 0.423 | 0.311 | 0.181 | 0.409 | 0.512 |
| TA-DistMult | 0.445 | 0.399 | 0.487 | 0.517 | 0.549 | 0.481 | 0.596 | 0.667 |
| De-simple | 0.454 | 0.426 | 0.477 | 0.495 | 0.549 | 0.516 | 0.573 | 0.601 |
| TNTCompLex | 0.450 | 0.400 | 0.493 | 0.520 | 0.579 | 0.529 | 0.613 | 0.666 |
| TuckERTNT | 0.503* | 0.480* | 0.518* | 0.538* | 0.607* | 0.562* | 0.636* | 0.682* |
| CyGNet | 0.338 | 0.290 | 0.361 | 0.418 | 0.520 | 0.453 | 0.561 | 0.637 |
| TANGO-Tucker | 0.504 | 0.485 | 0.514 | 0.535 | 0.578 | 0.530 | 0.607 | 0.658 |
| TANGO-DistMult | 0.515 | 0.496 | 0.521 | 0.533 | 0.627 | 0.591 | 0.603 | 0.679 |
| ReNET | 0.496 | 0.468 | 0.511 | 0.534 | 0.580 | 0.530 | 0.610 | 0.662 |
| HGAT | 0.561 | 0.529 | 0.581 | 0.618 | 0.636 | 0.598 | 0.660 | 0.715 |

Table 5

The results of different variants of our model on ICEWS14 and ICEWS18 datasets. The best results are in bold.

| Method | ICEWS14 | | | | ICEWS18 | | | |
|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|
| | MRR | Hit@1 | Hit@3 | Hit@10 | MRR | Hit@1 | Hit@3 | Hit@10 |
| HGAT-w/o-RTA | 0.351 | 0.261 | 0.384 | 0.535 | 0.268 | 0.177 | 0.301 | 0.447 |
| HGAT-w-RA | 0.368 | 0.278 | 0.405 | 0.552 | 0.276 | 0.184 | 0.310 | 0.455 |
| HGAT-w-TA | 0.374 | 0.281 | 0.413 | 0.558 | 0.283 | 0.190 | 0.319 | 0.461 |
| HGAT-dot | 0.356 | 0.265 | 0.392 | 0.537 | 0.275 | 0.184 | 0.309 | 0.452 |
| HGAT | 0.389 | 0.297 | 0.424 | 0.564 | 0.285 | 0.196 | 0.327 | 0.466 |

relation in the decoder part and do not make full use of relevant historical information, and HGAT equipping with the proposed encoder has more merits than them on learning adaptive entity representation. Second, our decoder model TuckERTNT is superior to most of the baseline methods due to its powerfully expressive ability, especially on WIKI and YAGO datasets. This also provides support for the outstanding performance of the proposed model. Third, compared with the decoder-only model TuckERTNT, HGAT achieves substantial improvements against it on all the metrics. For instance, on metric MRR, HGAT outperforms TuckERTNT 2.6% on ICEWS14, 2.4% on ICEWS18, 5.8% on WIKI, and 2.9% on YAGO. This result justifies the effectiveness of the proposed encoder and shows that the learned information from historical facts is valuable to our model.

5.5.2. Model variants & ablation study

To further verify the effectiveness of the proposed model, we present several variants of HGAT by adjusting the use of its model components, and Table 5 and Table 6 report the link prediction results of these variants. The result shows that both time-aware and relation-aware attention exert an important effect in the proposed model on all datasets. For instance, on metric MRR, the relation-aware mechanism helps the proposed model improve 1.7% on ICEWS14, and the time-aware mechanism helps the proposed model improve 2.3% on ICEWS14. These results further demonstrate that the proposed model is capable of learning from historical facts in a targeted manner. Moreover, we also observe that HGAT outperforms HGAT-dot on all datasets and metrics. This justifies that designing an appropriate information aggregator is

Table 6

The results of different variants of our model on WIKI and YAGO datasets. The best results are in bold.

| Method | WIKI | | | | YAGO | | | |
|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|
| | MRR | Hit@1 | Hit@3 | Hit@10 | MRR | Hit@1 | Hit@3 | Hit@10 |
| HGAT-w/o-RTA | 0.499 | 0.458 | 0.523 | 0.571 | 0.587 | 0.538 | 0.617 | 0.680 |
| HGAT-w-RA | 0.522 | 0.485 | 0.548 | 0.583 | 0.611 | 0.565 | 0.638 | 0.692 |
| HGAT-w-TA | 0.536 | 0.502 | 0.558 | 0.594 | 0.628 | 0.583 | 0.652 | 0.705 |
| HGAT-dot | 0.500 | 0.475 | 0.515 | 0.542 | 0.588 | 0.539 | 0.618 | 0.673 |
| HGAT | 0.561 | 0.529 | 0.581 | 0.618 | 0.636 | 0.598 | 0.660 | 0.715 |

Table 7

The results of HGAT with different decoders.

| Method | ICEWS14 | | | ICEWS18 | | | WIKI | | | YAGO | | |
|-----------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|
| | MRR | Hit@1 | Hit@10 | MRR | Hit@1 | Hit@10 | MRR | Hit@1 | Hit@10 | MRR | Hit@1 | Hit@10 |
| HGAT-TuckER | 0.381 | 0.292 | 0.558 | 0.272 | 0.180 | 0.451 | 0.553 | 0.520 | 0.611 | 0.616 | 0.571 | 0.704 |
| HGAT-TNTComplEx | 0.383 | 0.287 | 0.553 | 0.277 | 0.191 | 0.457 | 0.545 | 0.503 | 0.602 | 0.608 | 0.563 | 0.612 |
| HGAT | 0.389 | 0.297 | 0.564 | 0.285 | 0.196 | 0.466 | 0.561 | 0.529 | 0.618 | 0.636 | 0.590 | 0.715 |

also important for our model. In addition, we also employ different models as the decoder of HGAT, including static model TuckER, temporal model TNTComplEx, and TuckERTNT. The results in Table 7 suggest that HGAT with decoder TuckERTNT achieves considerable performance. Furthermore, HGAT-TuckER and HGAT-TNTComplEx are also superior to the decoder TuckER and TNTComplEx, respectively. These results also demonstrate the effectiveness and superiority of the proposed encoder.

5.5.3. Case study

We also provide some case studies for HGAT. Table 8 presents an example of a query and its relevant historical facts, and corresponding weights computed by time-aware and relation-aware attention. From the results in Table 8, we have the following observations and analyses.

First, the time-aware attention mechanism which is inspired by our intuition results in large variations in α over time, which is beneficial to our model because it alleviates the problem of over-smooth in the process of information aggregation and focuses on the more valuable neighboring facts. The results in Table 5 and Table 6 validate our intuition and the attention score α can improve our model performance explicitly. Second, we find that the relation *Accuse* and *Criticizeordenounce* are more relevant to the task relation *Threaten* than other relations. Among the tail entities of the two relations, *Pakistan*, and *Iran* are the most weighted.

Table 8

Case study for the query (*Trump, Threaten*, ?, 2018-7-26).

| Query: (<i>Trump, Threaten</i> , ?, 2018-7-26) | | Answer: <i>Iran</i> | |
|---|--|--|--|
| Query-Relevant Historical Facts | | Attention Scores | |
| (<i>Trump, Make statement, North Korea</i> , 2018-1-2) | | $\alpha_1 = 1.180\text{e-}10, \beta_{1,1} = 4.931\text{e-}11, \mu_{1,1} = 1.673\text{e-}10$ | |
| (<i>Trump, Accuse, Pakistan</i> , 2018-1-2) | | $\alpha_1 = 1.180\text{e-}10, \beta_{1,2} = 0.044, \mu_{1,2} = 0.044$ | |
| (<i>Trump, Reduce material aid, Pakistan</i> , 2018-1-4) | | $\alpha_2 = 1.180\text{e-}10, \beta_{2,1} = 3.153\text{e-}4, \mu_{2,1} = 3.15\text{e-}4$ | |
| (<i>Trump, Consult, Sebastián Piñera</i> , 2018-1-4) | | $\alpha_2 = 1.108\text{e-}10, \beta_{2,2} = 4.934\text{e-}11, \mu_{2,1} = 1.673\text{e-}10$ | |
| (<i>Trump, Criticize or denounce</i> ⁻¹ , <i>Iran</i> , 2018-1-5) | | $\alpha_3 = 1.180\text{e-}10, \beta_{3,1} = 7.4\text{e-}3, \mu_{3,1} = 7.4\text{e-}3$ | |
| ... | | ... | |
| (<i>Trump, Criticize or denounce</i> , <i>Iran</i> , 2018-7-3) | | $\alpha_{8206} = 3.777\text{e-}9, \beta_{8206,1} = 0.025, \mu_{8206,1} = 0.025$ | |
| (<i>Trump, Host a visit, Russia</i> , 2018-7-3) | | $\alpha_{8206} = 3.777\text{e-}9, \beta_{8206,1} = 3.405\text{e-}10, \mu_{8206,1} = 4.182\text{e-}9$ | |
| ... | | ... | |
| (<i>Trump, Accuse, Iran</i> , 2018-7-25) | | $\alpha_{8897} = 0.038, \beta_{8897,1} = 0.044, \mu_{8897,1} = 0.082$ | |
| (<i>Trump, Make statement, Ecuador</i> , 2018-7-25) | | $\alpha_{8897} = 0.038, \beta_{8897,1} = 4.931\text{e-}11, \mu_{8897,1} = 0.038$ | |

These results are in line with our expectations and also justify that the relation-aware mechanism can differentiate the importance of structural neighbors. Third, it is worth noting that we also tried to learn from top-20 query-relevant historical facts, and found utilizing all query-relevant historical facts achieved the best performance. Last, we observe that the weights of most historical facts follow this rule $(\gamma * \alpha_{t_i} + (1 - \gamma) * \beta_{t_i,i}) \approx \max\{\gamma * \alpha_{t_i}, (1 - \gamma) * \beta_{t_i,i}\}$. For example, the importance factor is 0.5 on ICEWS18, and $(\gamma * \alpha_{t_i} + (1 - \gamma) * \beta_{t_i,i}) \sim (\alpha_{t_i} + \beta_{t_i,i}) \approx \max\{\alpha_{t_i}, \beta_{t_i,i}\}$. Therefore, the important information in the distance to be underestimated by time-aware attention can be pulled back, such as (*Trump, Criticizeordenounce*⁻¹, *Iran*, 2018-1-5). The summation operation also increases the contribution of nearby important information, such as (*Trump, Accuse, Iran*, 2018-7-25). From another perspective, the dot operation makes $\sum_{t_i,i} \alpha_{t_i} * \beta_{t_i,i} = 1$, which indicates the sum of weights of all relevant facts is 1. This kind of operation destroys the hierarchical weight structure designed in this paper, making these related facts get very small weights respectively, and it is hard to distinguish the importance of the facts. This is also the reason why the performance of HGAT-dot is similar to that of TuckERTNT. In short, the summation operation can better alleviate the influence of the two attention mechanisms on each other, so that the importance of related facts can be more reasonably distinguished. All these results justify the effectiveness of the proposed information aggregator.

6. Conclusion and future work

This work proposes a novel hierarchical graph attention network for TKGs reasoning tasks. The proposed two-layer graph attention encoder can model the time-oriented and task-oriented roles of the entities, respectively. The designed time-aware and relation-aware attention mechanism can focus on the historical facts that are valuable to the query. Therefore, these two attention mechanisms not only provide interpretability for our model but also enable the proposed model more stable and more consistent with our intuition. Experimental results on four benchmarks explicitly demonstrate the significant merits and superiority of the proposed model on the TKGs reasoning task. In the future, we plan to explore a more reasonable and more adaptable attention mechanism to assign a reasonable weight to each historical fact related to the query, thereby learning a more powerful encoder for TKGs reasoning.

Declaration of Competing Interest

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

Acknowledgment

We gratefully acknowledge the help and assistance from Bin Liu and Xinke Wang. Moreover, this work is supported by the National Natural Science Foundation of China (NSFC) (Nos. 61831022, 62276259, 62201572, 62206278, U21B2010), Beijing Municipal Science & Technology Commission, Administrative Commission of Zhongguancun Science Park (No. Z211100004821013), CCF-Baidu Open Fund (No. OF2022025).

References

- [1] Y. Koren, R. Bell, C. Volinsky, Matrix factorization techniques for recommender systems, *Proc. Comput.* (2009) 30–37.
- [2] L. Dong, F. Wei, M. Zhou, K. Xu, Question answering over Freebase with multi-column convolutional neural networks, in: *Proceedings of the Annual Meeting of the Association for Computational Linguistics and the International Joint Conference on Natural Language Processing*, 2015, pp. 260–269.
- [3] A. Bordes, N. Usunier, S. Chopra, J. Weston, Large-scale simple question answering with memory networks, in: *CoRR*, 2015.
- [4] C. Xiong, J.P. Callan, Query expansion with freebase, in: *Proceedings of the International Conference on The Theory of Information Retrieval*, 2015, p. 111–120.
- [5] R. Goel, S.M. Kazemi, M. Brubaker, P. Poupert, Diachronic embedding for temporal knowledge graph completion, in: *Proceedings of AAAI Conference on Artificial Intelligence*, 2020.
- [6] T. Lacroix, G. Obozinski, N. Usunier, Tensor decompositions for temporal knowledge base completion, in: *Proceedings of International Conference on Learning Representations*, 2020, p. 253–256.
- [7] P. Shao, G. Yang, D. Zhang, J. Tao, F. Che, T. Liu, Tucker decomposition-based temporal knowledge graph completion, *Knowl.-Based Syst.* (2020) 107841.
- [8] R. Trivedi, H. Dai, Y. Wang, L. Song, Know-evolve: Deep temporal reasoning for dynamic knowledge graphs, in: *Proceedings of the International Conference on Machine Learning*, 2017, p. 3462–3471.
- [9] R.S. Trivedi, M. Farajtabar, P. Biswal, H. Zha, Dyrep: Learning representations over dynamic graphs, in: *Proceedings of International Conference on Learning Representations*, 2019.
- [10] C. Zhu, M. Chen, C. Fan, G. Cheng, Y. Zhang, Learning from history: Modeling temporal knowledge graphs with sequential copy-generation networks, in: *Proceedings of AAAI Conference on Artificial Intelligence*, 2021, pp. 4732–4740.
- [11] W. Jin, M. Qu, X. Jin, X. Ren, Recurrent event network: Autoregressive structure inference over temporal knowledge graphs, in: *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2020.
- [12] A. Bordes, N. Usunier, A. Garcia-Durán, J. Weston, O. Yakhnenko, Translating embeddings for modeling multi-relational data, in: *Proceedings of the*

- International Conference on Neural Information Processing Systems*, 2013, pp. 2787–2795.
- [13] G. Ji, S. He, L. Xu, K. Liu, J. Zhao, Knowledge graph embedding via dynamic mapping matrix, in: *Proceedings of Annual Meeting of the Association for Computational Linguistics*, 2015, p. 687–696.
- [14] Z. Wang, J. Zhang, J. Feng, Z. Chen, Knowledge graph embedding by translating on hyperplanes, in: *Proceedings of AAAI Conference on Artificial Intelligence*, 2014, p. 1112–1119.
- [15] M. Nickel, V. Tresp, H.-P. Kriegel, A three-way model for collective learning on multi-relational data, in: *Proceedings of the International Conference on Machine Learning*, 2011, p. 809–816.
- [16] B. Yang, S.W.-T. Yih, X. He, J. Gao, L. Deng, Embedding entities and relations for learning and inference in knowledge bases, in: *Proceedings of the International Conference on Learning Representations*, 2015.
- [17] T. Trouillon, J. Welbl, S. Riedel, E. Gaussier, G. Bouchard, Complex embeddings for simple link prediction, in: *Proceedings of the International Conference on Machine Learning*, 2016, p. 2071–2080.
- [18] I. Balazević, C. Allen, T.M. Hospedales, Tucker: Tensor factorization for knowledge graph completion, in: *Proceedings of the Conference on Empirical Methods in Natural Language Processing and the International Joint Conference on Natural Language Processing*, 2019, pp. 5185–5194.
- [19] T. Dettmers, M. Pasquale, S. Pontus, S. Riedel, Convolutional 2d knowledge graph embeddings, in: *Proceedings of AAAI Conference on Artificial Intelligence*, 2018, pp. 1811–1818.
- [20] M. Schlichtkrull, T.N. Kipf, P. Bloem, R.V.D. Berg, I. Titov, M. Welling, Modeling relational data with graph convolutional networks, in: *Proceedings of The Semantic Web*, 2017, pp. 593–607.
- [21] N. Lao, T. Mitchell, W.W. Cohen, Random walk inference and learning in a large scale knowledge base, in: *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2011, p. 529–539.
- [22] M. Gardner, P. Talukdar, J. Krishnamurthy, T. Mitchell, Incorporating vector space similarity in random walk inference over knowledge bases, in: *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2014, pp. 397–406.
- [23] W. Xiong, T. Hoang, W.Y. Wang, DeepPath: A reinforcement learning method for knowledge graph reasoning, in: *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2017, pp. 564–573.
- [24] Y. Shen, J. Chen, P.-S. Huang, Y. Guo, J. Gao, M-walk: Learning to walk over graphs using monte carlo tree search, in: *Proceedings of the International Conference on Neural Information Processing Systems*, 2018, pp. 6787–6798.
- [25] R. Das, S. Dhuliawala, M. Zaheer, L. Vilnis, I. Durugkar, A. Krishnamurthy, A. Smola, A. McCallum, Go for a walk and arrive at the answer: Reasoning over paths in knowledge bases using reinforcement learning, in: *Proceedings of International Conference on Learning Representations*, 2018.
- [26] W. Chen, W. Xiong, X. Yan, W.Y. Wang, Variational knowledge graph reasoning, in: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2018, pp. 1823–1832.
- [27] M. Richardson, P. Domingos, Markov logic networks, in: *Proceedings of Mach. Learn.*, 2006, pp. 107–136.
- [28] M. Qu, J. Tang, Probabilistic logic neural networks for reasoning, in: *Proceedings of the International Conference on Neural Information Processing Systems*, 2019, pp. 7710–7720.
- [29] Y. Zhang, X. Chen, Y. Yang, A. Ramamurthy, B. Li, Y. Qi, L. Song, Efficient probabilistic logic reasoning with graph neural networks, in: *Proceedings of International Conference on Learning Representations*, 2020.
- [30] M. Qu, J. Chen, L.-P. Xhonneux, Y. Bengio, J. Tang, Rnnlogic: Learning logic rules for reasoning on knowledge graphs, in: *Proceedings of International Conference on Learning Representations*, 2021.
- [31] T. Jiang, T. Liu, T. Ge, L. Sha, S. Li, B. Chang, Z. Sui, Encoding temporal information for time-aware link prediction, in: *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2016, pp. 2350–2354.
- [32] P. Shao, T. Liu, F. Che, D. Zhang, J. Tao, Adaptive pseudo-siamese policy network for temporal knowledge prediction, *Neural Networks* (2023) 192–201.
- [33] S.S. Dasgupta, S.N. Ray, P. Talukdar, Hyte: Hyperplane-based temporally aware knowledge graph embedding, in: *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2018, pp. 2001–2011.
- [34] A. García-Durán, S. Dumancic, M. Niepert, Learning sequence encoders for temporal knowledge graph completion, in: *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2018, pp. 4816–4821.
- [35] E. Boschee, J. Lautenschlager, S. O'Brien, S. Shellman, J. Starz, M. Ward, Icews coded event data, *Harvard Dataverse*, 2015.
- [36] J. Leblay, M.W. Chekol, Deriving validity time in knowledge graph, in: *Proceedings of The Web Conference 2018*, 2018, pp. 1771–1776.
- [37] F. Mahdisoltani, J. Biega, F.M. Suchanek, Yago3: A knowledge base from multilingual wikipeas, in: *Proceedings of CIDR*, 2015.
- [38] M. Nickel, K. Murphy, V. Tresp, E. Gabrilovich, A review of relational machine learning for knowledge graphs, in: *Proceedings of the IEEE*, 2016.
- [39] Z. Han, P. Chen, Y. Ma, V. Tresp, Explainable subgraph reasoning for forecasting on temporal knowledge graphs, in: *Proceedings of International Conference on Learning Representations*, 2021.

- [40] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. Devito, Z. Lin, A. Desmaison, L. Antiga, A. Lerer, Automatic differentiation in pytorch, in: Proceedings of the International Conference on Neural Information Processing Systems, 2017.
- [41] J.C. Duchi, E. Hazan, Y. Singer, Adaptive subgradient methods for online learning and stochastic optimization, in: Proceedings of Machine Learning Research, 2011, p. 2121–2159.
- [42] Z. Han, Z. Ding, Y. Ma, Y. Gu, V. Tresp, Learning neural ordinary equations for forecasting future links on temporal knowledge graphs, in: Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, 2021, pp. 8352–8364.
- [43] H. Sun, J. Zhong, Y. Ma, Z. Han, K. He, Timetraveler: Reinforcement learning for temporal knowledge graph forecasting, in: Proceedings of the Conference on Empirical Methods in Natural Language Processing, 2021, pp. 8306–8319.
- [44] Z. Ding, Z. Han, Y. Ma, V. Tresp, Temporal knowledge graph forecasting with neural ODE, in: CoRR, 2021.

679
680
681
682
683
684
685
686