Received December 10, 2019, accepted December 20, 2019, date of publication January 6, 2020, date of current version January 14, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2964028

Timespan-Aware Dynamic Knowledge Graph Embedding by Incorporating Temporal Evolution

XIAOLI TANG[1], RUI YUAN[2], QIANYU LI[1], TENGYUN WANG[1], HAIZHI YANG[1], YUNDONG CAI[3], AND HENGJIE SONG[1]

[1]School of Software Engineering, South China University of Technology, Guangzhou 510006, China
[2]School of Journalism and Communication, The Chinese University of Hong Kong, Hong Kong
[3]Joint NTU-UBC Research Centre of Excellence in Active Living for the Elderly, Nanyang Technological University, Singapore

Corresponding author: Hengjie Song (sehjsong@scut.edu.cn)

This work was supported in part by the Pre-Research Foundation of China under Grant 61400010205, in part by the National Natural Science Foundation of China under Grant 71671069, and in part by the National Key Research and Development Program of China under Grant 2018YFC0830900.

ABSTRACT Recently, Knowledge Graph Embedding (KGE) has attracted considerable research efforts, since it simplifies the manipulation while preserving the inherent structure of the KG. However to some extent, most existing KGE approaches ignore the historical changes of structural information involved in dynamic knowledge graphs (DKGs). To deal with this problem, this paper presents a Timespan-aware Dynamic knowledge Graph Embedding Evolution (TDG2E) method that considers temporal evolving process of DKGs. The major innovations of our paper are two-fold. Firstly, a Gated Recurrent Units (GRU) based model is utilized in TDG2E to deal with the dependency among sub-KGs that is inevitably involved in the learning process of the dynamic knowledge graph embedding. Furthermore, we incorporate an auxiliary loss to supervise the learning process of the next sub-KG by utilizing previous structural information (i.e., the hidden state of GRU). In contrast with existing approaches in the literature (e.g., HyTE and t-TransE), TDG2E preserves structural information of current sub-KG and the temporal evolving process of the DKG simultaneously. Secondly, to further deal with the time unbalance issue underlying the DKGs, a Timespan Gate is designed in GRU. It makes TDG2E possible to model the temporal evolving process of DKGs more effectively by incorporating the timespan between adjacent sub-KGs. Extensive experiments on two large temporal datasets (i.e., YAGO11k and Wikidata12k) extracted from real-world KGs validate that the proposed TDG2E significantly outperforms traditional KGE methods in terms of Mean Rank and Hit Rate.

INDEX TERMS Knowledge graph embedding, gated recurrent units, time unbalance.

I. INTRODUCTION

Recent years have witnessed the rapid growth of Knowledge Graph (KG). A great number of KGs, including Freebase [1], DBpedia [2], YAGO [3] and NELL [4] and so on, have been constructed and applied successfully to many real-world applications, ranging from semantic parsing [5], [6] and named entity disambiguation [7], [8], to information extraction [9], [10] and question answering [11], [12]. A KG is a large multi-relational graph in which nodes represent entities and typed edges correspond to relations among adjacent entities. Each edge in a KG encodes a factual belief (also called a fact) and is represented as a triple of the form (head entity, relation, tail entity), indicating that the *head entity* and the *tail entity* are connected by the *relation*, e.g., *(Bill Clinton, wasPresidentOf, USA)*. Though it is effective to represent inevitably structured data, the underlying symbolic nature of such triples usually makes KGs hard to manipulate [13]. To deal with this issue, a new research direction known as Knowledge Graph Embedding (KGE) has been proposed and quickly gained massive attention [14]–[20]. The main idea of KGE is to embed entities and relations of a KG into low dimensional spaces so as to simplify the manipulation while preserving the inherent structure of the KG [13].

Existing research on KGE has mainly focused on static knowledge graphs [21], [22]. However, in real-life scenarios, KGs, such as social knowledge graphs in Twitter, citation knowledge graphs in DBLP [23], tend to be dynamic in which facts are only valid in a specific time period and evolve

The associate editor coordinating the review of this manuscript and approving it for publication was Chang Choi.

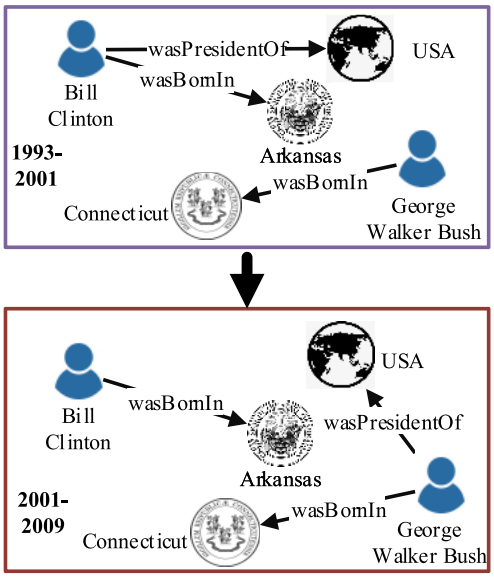


FIGURE 1. An illustrative example on the necessity of considering the temporal information and evolutionary patterns of KGs. Since Bill Clinton served as the 42^{th} president of the United States from 1993 to 2001 and left from the presidency in 2001, the KG shown on the top, turned into the one below in 2001. All facts in the top KG were valid from 1993 to 2001, while all facts of the KG below were valid from 2001 to 2009.

over time. For example, as shown in Figure 1, Bill Clinton served as the 42^{th} president of the United States from 1993 to 2001 and left from the presidency on January 20, 2001. According to this information, the fact *(Bill Clinton, wasPresidentOf, USA)* was true only from 1993 to 2001 and disappeared from the KG after January 20, 2001. Facts of dynamic KGs (DKGs) should be described with temporal information. Since static KGE methods completely ignore the temporal information, it makes static KGE methods fail to work on these real-life scenarios. Thus, it is necessary to design an embedding method for DKGs.

In order to incorporate temporal information in the learned embeddings while maintaining the inherent structure of DKGs, an obvious way is to fragment a DKG into multiple static sub-KGs with each sub-KG corresponding to a specific time bin [24]. Then the embeddings can be learned on these bins separately. Although this kind of models take into account the temporal information of the KG in the embedding process, they cannot explicitly model the evolving process of DKGs. This is because these models are designed to fit on different time bins independently, which makes it difficult to share statistical strength among different time bins. Furthermore, the model trained independently in a specific time bin cannot remain robust when the structure of the KG changes drastically at a specific point. Suffering from the aforementioned disadvantages, the existing research on DKG embedding is rather limited. Therefore, it is critical to design a KGE method that can preserve structural information of current sub-KG, while preserving evolutionary patterns of the DKG simultaneously.

In this paper, we propose a robust dynamic knowledge graph embedding method called Timespan-aware Dynamic knowledge Graph Embedding Evolution (TDG2E) to encode temporal information directly in the learned embeddings. TDG2E can fit the current KG structure well and consider the historical changes of structural information. Specifically, TDG2E fragments the temporally-scoped input KG into multiple static sub-KGs, each of which corresponds to one time bin. And then TDG2E projects the entities and relations of each sub-KG into temporally aware hyperplanes. Next, a Gated Recurrent Units (GRU) [25] based model is utilized to deal with the dependency among adjacent sub-KGs. Considering that cumulative structural information leads to the consecutive structure directly, we further incorporate an auxiliary loss that uses previous structural information to supervise the learning of the next sub-KG. Furthermore, TDG2E designs a Timespan Gate in GRU to solve the time unbalance issue faced by DKGs. We highlight our contributions as follows:

1) In contrast to the state-of-the-art dynamic KGE methods [24], [26], which learn different sub-KGs independently, the proposed TDG2E can preserve structural information of individual sub-KGs, while preserving evolutionary patterns of the DKGs. Specifically, TDG2E utilizes a GRU based model to capture dependency among adjacent sub-KGs that is inevitably involved in DKGs. Furthermore, TDG2E introduces an auxiliary loss, using previous structural information (i.e., the hidden state of GRU) to supervise the learning process of the following hyperplanes. Consequently, TDG2E obtains better performance in the applications.
2) As presented in [24], the distribution of timestamps in the KG is unbalanced. However, existing graph embedding methods [24], [26], [27] have not yet fully solved this problem. To deal with this issue well, TDG2E designs a Timespan Gate in GRU to model the evolutionary patterns of DKGs more effectively by incorporating the timespan between adjacent sub-KGs. It makes TDG2E possible to control the transition effect from past structural pattern to further structural pattern.

Based on two large temporal datasets extracted from real-world KGs, we have conducted extensive experiments to evaluate the effectiveness of TDG2E. The results reveal that the improvements are significant compared with other state-of-the-art methods in terms of common evaluation metrics (e.g., Mean Rank and Hit@K).

The rest of this paper is organized as follows. Section II reviews the related work. Section III provides some preliminary concepts and presents the problem. Section IV delves into our proposed method followed by learning algorithm in Section V. Then, the experimental results on benchmark datasets and discussions are provided in Section VI. Finally, we list concluding remarks and discuss the future work in Section VII.

II. RELATED WORK

Our work is related to *static knowledge graph embedding*, *dynamic graph embedding* and *dynamic knowledge graph embedding*.[1]

A. STATIC KNOWLEDGE GRAPH EMBEDDING

Extensive studies have been done on static knowledge graph embedding. These can be roughly categorized into three different paradigms. TransE [15], TransH [16], TransR [17], TransD [31], TranSparse [32] and TransF [26] are translation based approaches, whose goals are to minimize the distance between two entities where one of them is translated by the relation. RESCAL [14], DistMult [33], HolE [34], ComplEx [35], SimplE [36] are matrix factorization based models, which impose on matrices. References [37]–[39] are deep learning-based models. Deep learning approaches typically use feed-forward or convolutional neural networks for scoring edges in a KG.

B. DYNAMIC GRAPH EMBEDDING

In [40], Zhu *et al.* propose a dynamic graph embedding method based on matrix factorization. Reference [41] explores the evolution patterns of triads, which can preserve structural information and get the latent representation vectors for vertices at different timestamps. Reference [42] proposes a network embedding method based on Hawkes process, which is powerful in modeling sequences.

There are some studies on **recurrent graph neural models** for sequential or temporal graph-structured data [29], [43]–[46]. The principle of these models is to adopt a message-passing framework for aggregating nodes' neighborhood information (e.g., via graph convolutional operations). GN [43], [44] and RRN [45] learn node representations with message-passing between timestamps. EvolveGCN in [46] incorporates an RNN to learn node representations across different time stamps. RE-NET [29] adapts a RNN with message passing procedure between adjacent entities to encode temporal dependency between entity interactions.

Different from models mentioned above, our model focuses on the dynamic knowledge graph, which is a multi-relational, directed graph with time-stamped edges (relationships) between nodes (entities).

C. DYNAMIC KNOWLEDGE GRAPH EMBEDDING

There have been some studies on incorporating temporal information into modeling DKG. t-TransE [27] provides a link prediction method by using temporal order constraints to model transformation between time-sensitive relations. Know-Evolve [47] models the occurrence of a fact as a temporal point process and deals with concurrent events based on a problematic formulation. Reference [48] regards timestamps as a sequence of digits (from 0 to 9), then uses LSTMs to encode the relation vectors and the time digits. [49] models the interactions between relations and time, and studies various ways to combine the time embedding vector with relation embedding vector, such as concatenate, sum or dot product operations. Recently, HyTE [24] proposes a KGE method based on projected-time translation. It represents different times as different hyperplanes, and segregates the embedding space into different time zones by these hyperplanes. In contrast to previous time-sensitive KG embedding methods, HyTE can directly encode temporal information in the learned embeddings.

Our model focuses on Time-Conditioned tasks rather than Time-Predicting tasks [28]. HyTE is the most similar work to our study, however, it ignores the evolutionary patterns of DKGs. The advantages of our method over the previous work are two-fold. First, unlike the previous methods that only model different time bins to fit the corresponding structure well, our proposed model also considers the changing trend of the recent history, and introduces an auxiliary loss to supervise the learning of embeddings. By doing so, our model can make embeddings of KG elements preserve the structural information of current sub-KG and evolutionary patterns of dynamics. Second, we deal with the time unbalance issue of KGs, which further improves the adaptability of the proposed method and plays an essential role in obtaining superior performance over different DKGs.

III. PROBLEM FORMULATION

In this section, we will first clarify some terminologies used in this paper, and then explicitly formulate our problem. For clarity, some important symbols used throughout this paper are listed in Table 1.

Usually, KGs are treated as static ones, which consist of triples in the form of (h_i, r_i, t_i) $(i \in [1, N], N$ is the number of facts in KG $\mathcal{G})$. h_i and t_i are the head entity and the tail entity respectively, which are connected by relation r_i. When triples of a KG $\mathcal{G}$ come with separate time dimensions, $\mathcal{G}$ is dynamic. For the quadruple $(h_i, r_i, t_i, [t_s^i, t_e^i])$, $[t_s^i, t_e^i]$ denotes the existence time range of the fact represented by triplet (h_i, r_i, t_i) in the DKG $\mathcal{G}$. t_s^i denotes the start time of the fact, and t_e^i denotes the end time of the fact. Given the series of timestamps, $\tau \in \{1, 2, \ldots, T\}$, the DKG $\mathcal{G}$ can be split into T static sub-KGs $\mathcal{G}_1, \ldots, \mathcal{G}_T$, and each sub-KG consists of multiple triples that are valid in the corresponding timestamp. Thus, KG $\mathcal{G}$ can be expressed as

$$\mathcal{G} = \mathcal{G}_1 \cup \mathcal{G}_2 \cup \cdots \mathcal{G}_\tau \cdots \cup \mathcal{G}_T, \tag{1}$$

where $\tau \in [1, T]$. Similar to existing methods, we denote head entity h_i, tail entity t_i and relation r_i by $\boldsymbol{h}_i \in \mathbb{R}^{d\times 1}$, $\boldsymbol{t}_i \in \mathbb{R}^{d\times 1}$ and $\boldsymbol{r}_i \in \mathbb{R}^{d\times 1}$, respectively.

The goal of dynamic KGE is to learn $\boldsymbol{h}_i$, $\boldsymbol{t}_i$, $\boldsymbol{r}_i$, and the appropriate mapping functions $\Gamma_\tau, \tau \in [1, T]$ to fulfill the requirements of the following three tasks:

- Head entity prediction: for an incomplete fact $(?, r_i, t_i)$ at τ, Γ_τ is able to predict the head entity h_i.

[1] Taking into complex interactions between entities in knowledge graph, we separate the knowledge graph embedding from general graph as most studies [28]–[30] do.

TABLE 1. A summary of our symbols and descriptions.

Symbol	Description
τ	the timestamp
$\mathcal{G}$	the given DKG (dynamic knowledge grpah) ($\mathcal{G}_\tau$ is the static sub-KG at timestamp τ.)
$\boldsymbol{h}_i, \boldsymbol{r}_i, \boldsymbol{t}_i$	representations of head entity h_i, relation r_i and tail entity t_i respectively
d	the dimension of embeddings
$\boldsymbol{W}$	the time-specific hyperplanes, $\boldsymbol{W} = [\boldsymbol{w}_1, \boldsymbol{w}_2, \cdots, \boldsymbol{w}_T]$ ($\boldsymbol{w}_\tau$ is the hyperplane at timestamp τ.)
$P_\tau(*)$	the projection function related to hyperplane $\boldsymbol{w}_\tau$
$\mathcal{S}_\tau^+$	the positive triple set at timestamp τ (s_i^+ is a fact in $\mathcal{S}_\tau^+$.)
$\mathcal{S}_{\tau,e}^-$	the negative triple set about entities at timestamp τ ($s_{j,e}^-$ is a triple in $\mathcal{S}_{\tau,e}^-$.)
$\mathcal{S}_{\tau,r}^-$	the negative triple set about relations at timestamp τ ($s_{k,r}^-$ is a triple in $\mathcal{S}_{\tau,r}^-$.)
α	the trade-off parameter to balance the loss associated with relation negative triples
γ	the margin in margin-based ranking loss
$\boldsymbol{R}$	the parameters used in the GRU, $\boldsymbol{R} = [\boldsymbol{R}_z, \boldsymbol{R}_r, \boldsymbol{R}_T, \boldsymbol{R}_t, \boldsymbol{R}_p]$
$\boldsymbol{p}_\tau$	the τ-th hidden state of GRU
$\boldsymbol{z}_\tau, \boldsymbol{r}_\tau$	the τ-th update gate and reset gate of GRU respectively
$\boldsymbol{T}_\tau$	the τ-th Timespan gate of the adpated GRU
Δt_τ	the τ-th timespan
β	the trade-off parameter to balance the auxiliary loss
M	the number of negative sampling triples

- Relation prediction: for an incomplete fact $(h_i, ?, t_i)$ at τ, Γ_τ is able to predict the relation r_i.
- Tail entity prediction: for an incomplete fact $(h_i, r_i, ?)$ at τ, Γ_τ is able to predict the tail entity t_i.

IV. THE PROPOSED METHOD: TDG2E

In this section, we give an overview of the proposed TDG2E followed by detailed descriptions of main components.

A. ENCODING TEMPORAL INFORMATION

Here, we will describe how to encode temporal information into the learned embeddings of KG elements based on TransE. As mentioned in Section II, TransE is the most representative static KGE method which represents both entities and relations as vectors in the same space. Given a fact (h_i, r_i, t_i), the relation r_i is interpreted as a translation vector $\boldsymbol{r}_i$ so that the embedded head entity h_i and tail entity t_i can be connected by the translation vector $\boldsymbol{r}_i$ with low error, i.e., $\boldsymbol{h}_i + \boldsymbol{r}_i \approx \boldsymbol{t}_i$. Obviously, if the triplet (h_i, r_i, t_i) does not hold in the KG, TransE will enforce $\boldsymbol{h}_i + \boldsymbol{r}_i$ to be far away from $\boldsymbol{t}_i$. Based on this idea, TransE can obtain embeddings by minimizing a margin-based ranking loss over the whole training set [15].

For example, the fact *"Bill Clinton served as the president of the United States from 1993 to 2001."* can be abstracted with a triple as follows,

$$(h_i : BillClinton, r_i : wasPresidentOf, t_i : USA). \quad (2)$$

For another fact *"George Walker Bush served as the president of the United States from 2001 to 2009."*, we can denote it as follows,

$$(h_j : GeorgeWalkerBush, r_j : wasPresidentOf, t_j : USA). \quad (3)$$

Since the fact that *"George Walker Bush served as the 42th president of USA while George Walker Bush served as the 43th one."*, triple (2) and triple (3) share the same tail entity and relation while having different head entities. If the time information is ignored, TransE simply mandates $\boldsymbol{h}_i + \boldsymbol{r}_i = \boldsymbol{t}_i$ and $\boldsymbol{h}_j + \boldsymbol{r}_j = \boldsymbol{t}_j$. Since $\boldsymbol{t}_i = \boldsymbol{t}_j$ and $\boldsymbol{r}_i = \boldsymbol{r}_j$, a wrong conclusion, $\boldsymbol{h}_i = \boldsymbol{h}_j$, will be deduced by TransE.

In order to deal with the above issue, a traditional way is to utilize temporally aware hyperplanes to fragment the embedding space into different time bins. With the help of the time-specific hyperplane at time τ, the representation of the triple valid at time τ will be projected into hyperplane $\boldsymbol{w}_\tau \in \mathbb{R}^{d\times 1}$ as follows:

$$P_\tau(\boldsymbol{h}_i) = \boldsymbol{h}_i - (\boldsymbol{w}_\tau{}^T \boldsymbol{h}_i)\boldsymbol{w}_\tau, \quad (4)$$

$$P_\tau(\boldsymbol{t}_i) = \boldsymbol{t}_i - (\boldsymbol{w}_\tau{}^T \boldsymbol{t}_i)\boldsymbol{w}_\tau, \quad (5)$$

$$P_\tau(\boldsymbol{r}_i) = \boldsymbol{r}_i - (\boldsymbol{w}_\tau{}^T \boldsymbol{r}_i)\boldsymbol{w}_\tau, \quad (6)$$

where $\boldsymbol{w}_\tau$ is restricted as $\|\boldsymbol{w}_\tau\|_2 = 1$. $P_\tau(\boldsymbol{h}_i)$, $P_\tau(\boldsymbol{r}_i)$ and $P_\tau(\boldsymbol{t}_i)$ denote the projection of the head entity h_i, relation r_i and tail entity t_i on the hyperplane $\boldsymbol{w}_\tau$, respectively. By doing so, triples with the same tail entity and the same relation at different times will be projected into different subspaces, and the head entities of these triples will be also represented as different embeddings in different subspaces. Therefore, the many-to-one problem caused by time factor in DKGs mentioned above is easily handled.

Inspired by the previous method [50], we learn the embeddings in Eq. 4, Eq. 5 and Eq. 6 by minimizing the following margin-based ranking loss,

$$\underset{\boldsymbol{W},\boldsymbol{h},\boldsymbol{r},\boldsymbol{t}}{\operatorname{argmin}} \mathcal{L}_{emb}(\boldsymbol{W}, \boldsymbol{h}, \boldsymbol{r}, \boldsymbol{t}) = \sum_{\tau=1}^{T} \sum_{s_i^+}^{S_\tau^+} \sum_{s_{j,e}^-, s_{k,r}^-}^{S_{\tau,e}^-, S_{\tau,r}^-} max(0, 2\mathcal{L}_{tre}(s_i^+) + \gamma - \mathcal{L}_{tre}(s_{j,e}^-) - \alpha \mathcal{L}_{tre}(s_{k,r}^-)), \quad (7)$$

where $\boldsymbol{W} = [\boldsymbol{w}_1, \boldsymbol{w}_2, \cdots, \boldsymbol{w}_T]$. Note that different from the general margin-based ranking loss, which is widely used among other KGE methods, the margin-based ranking loss in Eq. 7 incorporates a task-oriented negative sampling strategy that considers the negative sampling of not only entities but also relations.

S_τ^+ in Eq. 7 is the positive triple set at timestamp τ and can be denoted as:

$$S_\tau^+ = \{(h_i, r_i, t_i) | (h_i, r_i, t_i) \in \mathcal{G}_\tau, i \in [1, |\mathcal{G}_\tau|]\}, \quad (8)$$

$|\mathcal{G}_\tau|$ is the size of static knowledge graph $\mathcal{G}_\tau$. $S_{\tau,e}^-$ is the sampled negative triple set which is generated by replacing the head entity or tail entity in S_τ^+, abstracted as:

$$S_{\tau,e}^- = \{(h_i', r_i, t_i), (h_i, r_i, t_i') | (h_i, r_i, t_i) \in \mathcal{G}_\tau, (h_i', r_i, t_i) \in \bar{\mathcal{G}}_\tau, (h_i, r_i, t_i') \in \bar{\mathcal{G}}_\tau\}, \quad (9)$$

$\bar{\mathcal{G}}_\tau$ is the complement of $\mathcal{G}_t$, i.e., $\mathcal{G}_\tau \cup \bar{\mathcal{G}}_\tau = \mathcal{G}$ and $\mathcal{G}_\tau \cap \bar{\mathcal{G}}_\tau = \emptyset$. $S_{\tau,r}^-$ is the sampled negative triple set which is generated by replacing the relation in S_τ^+, abstracted as:

$$S_{\tau,r}^- = \{(h_i, r_i', t_i) | (h_i, r_i, t_i) \in \mathcal{G}_\tau, (h_i, r_i', t_i) \in \bar{\mathcal{G}}_\tau\}. \quad (10)$$

$\mathcal{L}_{tre}$ in Eq. 7 is the loss corresponding to the projection of triples on $\boldsymbol{w}_\tau$, e.g.,

$$\mathcal{L}_{tre}(s_i^+) = \mathcal{L}_{tre}(h_i, r_i, t_i) = \|P_\tau(\boldsymbol{h}_i) + P_\tau(\boldsymbol{r}_i) - P_\tau(\boldsymbol{t}_i)\|. \tag{11}$$

Following the previous method [50], a parameter α is added for $\mathcal{L}_{tre}(s_{k,r}^-)$ in Eq. 7 to avoid excessive use of negative triples about relations, which may affect the ability of the model to differentiate entities.

B. TEMPORAL EVOLUTION EMBEDDING MODEL

According to previous methods [24], [27], the optimal $\boldsymbol{W}, \boldsymbol{h}, \boldsymbol{r}, \boldsymbol{t}$ can be solved by minimizing $\mathcal{L}_{emb}$ in Eq. 7. However, as it can be seen from the right part of Eq. 7, $\mathcal{L}_{emb}$ is only the total sum of the margin-based ranking criterion on each static sub-KG. In other words, for a fixed set of $\boldsymbol{h}, \boldsymbol{r}, \boldsymbol{t}$, the learning process of $\boldsymbol{w}_\tau$ corresponding to each timestamp is independent from that of other timestamps. As a result, the model learned by minimizing $\mathcal{L}_{emb}$ contains only sub-KG structure information at different timestamps, but not the temporal evolving pattern of the global DKG.

Actually, in dynamic situations, a DKG keeps evolving and its sub-KGs are dependent [42]. Thus, it is important to capture the dynamics of the global DKG and dependencies among sub-KGs. Considering that Gated Recurrent Units (GRU) [25] have achieved impressive results on a range of sequence modeling problems such as language modeling and speech recognition, we adopt GRU to process the hyperplanes at different timestamps while accumulating the margin-based ranking criterion of each sub-KG. Specifically, the hyperplane $\boldsymbol{w}_\tau$ is taken as the input of τ-th unit. The τ-th update gate $\boldsymbol{z}_\tau$, reset gate $\boldsymbol{r}_\tau$ and hidden state $\boldsymbol{p}_\tau$ of GRU are calculated as:

$$\boldsymbol{z}_\tau = \sigma(\boldsymbol{R}_z[\boldsymbol{p}_{\tau-1}, \boldsymbol{w}_\tau]), \tag{12}$$

$$\boldsymbol{r}_\tau = \sigma(\boldsymbol{R}_r[\boldsymbol{p}_{\tau-1}, \boldsymbol{w}_\tau]), \tag{13}$$

$$\widetilde{\boldsymbol{p}_\tau} = \tanh(\boldsymbol{R}_p[\boldsymbol{r}_\tau \circ \boldsymbol{p}_{\tau-1}, \boldsymbol{w}_\tau]), \tag{14}$$

$$\boldsymbol{p}_\tau = (\boldsymbol{1} - \boldsymbol{z}_\tau) \circ \boldsymbol{p}_{\tau-1} + \boldsymbol{z}_\tau \circ \widetilde{\boldsymbol{p}_\tau}, \tag{15}$$

where operator "$\circ$" denotes the Hadamard product and $\sigma(\boldsymbol{x}) = \frac{1}{1+exp^{-\boldsymbol{x}}}$ is the sigmoid activation function. tanh is the hyperbolic tangent activation function. $\boldsymbol{R}_z \in \mathbb{R}^{d\times d}$, $\boldsymbol{R}_r \in \mathbb{R}^{d\times d}$ and $\boldsymbol{R}_p \in \mathbb{R}^{d\times d}$ are all weight matrices.

$\boldsymbol{p}_\tau$ is an updated hidden representation containing information about the sequential sub-KGs until time τ. As we all know, to some extent, cumulative structural information at each timestamp leads to the consecutive structure directly, so we introduce an auxiliary loss to utilize hidden representation $\boldsymbol{p}_\tau$ for supervising the learning of hyperplane $\boldsymbol{w}_{\tau+1}$, which is formulated as:

$$\mathcal{L}_{aux}(\boldsymbol{W}, \boldsymbol{R}_z, \boldsymbol{R}_r, \boldsymbol{R}_p) = \sum_{\tau=1}^{T-1} \|\boldsymbol{p}_\tau - \boldsymbol{w}_{\tau+1}\|_2. \tag{16}$$

The benefits of the auxiliary loss is twofold. From the aspect of evolution learning, the introduction of the auxiliary loss helps each hyperplane preserves structural information of the current sub-KG, while preserving evolutionary patterns of the dynamic knowledge graph simultaneously. As for the optimization of GRU, the auxiliary loss reduces the difficulty of back propagation when GRU deals with tons of sub-KGs.

TABLE 2. Comparison of different time bins. (# means 'the number of'.).

	time bin 1596-1777	time bin 2013-2014
# facts changed	300	300
timespan	181 years	1 year
speed of change	low	high
impact on successors	small	strong

C. TIMESPAN-AWARE TEMPORAL EVOLUTION EMBEDDING MODEL

Another problem with Eq. 7 occurs during the application on real DKGs. As presented in [24], the distribution of timestamps in the KG is unbalanced. A usual practice to deal with this issue is to dispense the timestamps into a number of different bins, with each bin involving 300 triples. By doing so, less frequent year mentions are grouped into the same time bin but years with high frequency forms individual bins. For example, in KG Wikidata12k [49] which is extracted from a preprocessed dataset of Wikidata, there are bins like 1596-1777, 1791-1815 with a large span as the events occurring on those points of time are quite less in the KG. The years like 2013, 2014 being highly frequent are self-contained. Although these kind of methods can distribute the timestamps in the KG uniformly, they ignore the impact of different time bins on modeling the evolving process of the DKG. Compared to the one with a short span, a time bin with a long span has a stronger impact on its successor. For example, with regard to time bin 1596-1777, as 300 triples appeared or disappeared during the 181-year span, the KG was steady, which means that time bin 1596-1777 and its recent history have a strong impact on its successor; while as far as 2013-2014 is concerned, as 300 triples appeared or disappeared during the 1-year span, the KG changed violently, indicting that time bin 2013-2014 and its recent history have a small impact on its successor.

The above instance means that, to some extent, the timespan between sub-KGs affects the learning of hyperplanes. Specifically, the timespan controls the effect of the corresponding cumulative structural information on its successor. To better deal with the time unbalance issue among DKGs, we take timespans between sub-KGs into account.

Considering that the effect of timespans, determining the effect of cumulative structural information on successors, is similar to that of the update gate in GRU, which helps the model to determine how much of the past information (from previous time steps) needs to be passed along to the future [25], we propose to add a gate like the update gate to GRU for embodying the effect of timespans. The gate we added is called *Timespan Gate*, shown as $\boldsymbol{T}_\tau$ in Figure 2.

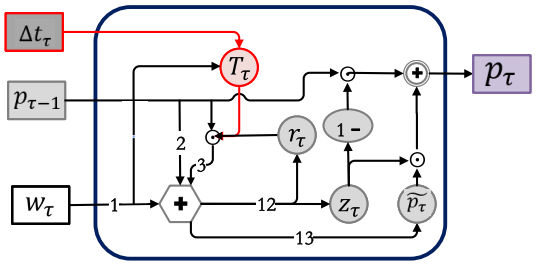


FIGURE 2. Illustration of the adapted GRU proposed by us. The part we added is marked in red. The *Timespan Gate* $\boldsymbol{T}_\tau$, shown as the red circle, is used to incorporate timespans, mitigating time unbalance of KGs. The operator "⊙" denotes the Hadamard product and "⊕" represents a matrix addition operation.

To calculate the *Timespan Gate*, we first denote the span between the $\{\tau+1\}$-th time bin and τ-th time bin as the τ-th timespan Δt_τ, abstracted as,

$$\Delta t_\tau = t_s^{\tau+1} - t_s^\tau, \tag{17}$$

where t_s^τ is the start time of the τ-th time bin. The Timespan Gate $\boldsymbol{T}_\tau$ is calculated as:

$$\boldsymbol{T}_\tau = \sigma(\boldsymbol{R}_T \boldsymbol{w}_\tau + \sigma(\Delta t_\tau \boldsymbol{R}_t) + \boldsymbol{b}_t), \tag{18}$$

where $\boldsymbol{R}_T \in \mathbb{R}^{d\times d}$ and $\boldsymbol{R}_t \in \mathbb{R}^{d\times d}$ are the weight matrices. $\boldsymbol{b}_t$ is the bias.

$\boldsymbol{T}_\tau$ in Eq. 18 captures the correlation between the current sub-KG and the timespan Δt_τ. Under the influence of $\boldsymbol{T}_\tau$, $\widetilde{\boldsymbol{p}_\tau}$ in Eq. 14 and hidden representation $\boldsymbol{p}_\tau$ in Eq. 15 are modified as :

$$\widetilde{\boldsymbol{p}_\tau} = \tanh(\boldsymbol{R}_p[\boldsymbol{r}_\tau \circ \boldsymbol{T}_\tau \circ \boldsymbol{p}_{\tau-1}, \boldsymbol{w}_\tau]), \tag{19}$$

$$\boldsymbol{p}_\tau = (\boldsymbol{1} - \boldsymbol{z}_\tau) \circ \boldsymbol{p}_{\tau-1} + \boldsymbol{z}_\tau \circ \widetilde{\boldsymbol{p}_\tau}. \tag{20}$$

$\boldsymbol{T}_\tau$ is helpful in two ways. As Eq. 19 and 20 show, on one hand, $\boldsymbol{p}_{\tau-1}$ is filtered by not only the reset gate $\boldsymbol{r}_\tau$, but also the time gate $\boldsymbol{T}_\tau$. So $\boldsymbol{T}_\tau$ can control the influence of the cumulative structural information. On the other hand, Δt_τ is firstly stored in $\boldsymbol{T}_\tau$, then transferred to $\boldsymbol{p}_\tau$, and would be transferred to $\boldsymbol{p}_{\tau+1}, \boldsymbol{p}_{\tau+2}, \cdots$. Thus $\boldsymbol{T}_\tau$ helps to store Δt_τ to model the evolution of the DKG.

V. MODEL LEARNING

In this section, we present some details of how we can learn the optimal parameters of TDG2E given a DKG $\mathcal{G}$.

By combining Eq. 7 and Eq. 16, we derive an overall optimization approach for the following unified optimization problem, which is constructed by using $\mathcal{L}(\boldsymbol{W}, \boldsymbol{h}, \boldsymbol{r}, \boldsymbol{t}, \boldsymbol{R})$ as the general loss function.

$$\begin{aligned}
&\underset{\boldsymbol{W},\boldsymbol{h},\boldsymbol{r},\boldsymbol{t},\boldsymbol{R}}{\operatorname{argmin}} \mathcal{L}(\boldsymbol{W}, \boldsymbol{h}, \boldsymbol{r}, \boldsymbol{t}, \boldsymbol{R}) = \beta \mathcal{L}_{aux}(\boldsymbol{W}, \boldsymbol{R}) \\
&\qquad + \mathcal{L}_{emb}(\boldsymbol{W}, \boldsymbol{h}, \boldsymbol{r}, \boldsymbol{t})| < S_\tau^+, S_{\tau,e}^-, S_{\tau,r}^- >, \\
&s.t.\ \|\boldsymbol{h}_i\|_2 = 1, \|\boldsymbol{r}_i\|_2 = 1, \|\boldsymbol{t}_i\|_2 = 1, i \in [1, N] \\
&\qquad \|\boldsymbol{w}_\tau\|_2 = 1, \tau \in [1, T]
\end{aligned} \tag{21}$$

where β is a tradeoff parameter and $\boldsymbol{R}$ includes $\boldsymbol{R}_z, \boldsymbol{R}_r, \boldsymbol{R}_T, \boldsymbol{R}_t$ and $\boldsymbol{R}_p$.

In this section, we roughly derive approaches to solve the optimization problem constructed in Eq. 21. Firstly, we convert the optimization problem to an unconstrained one,

$$\begin{aligned}
&\underset{\boldsymbol{W},\boldsymbol{h},\boldsymbol{r},\boldsymbol{t},\boldsymbol{R}}{\operatorname{argmin}} \mathcal{L}(\boldsymbol{W}, \boldsymbol{h}, \boldsymbol{r}, \boldsymbol{t}, \boldsymbol{R}) \\
&= \beta \mathcal{L}_{aux}(\boldsymbol{W}, \boldsymbol{R}) + \mathcal{L}_{emb}(\boldsymbol{W}, \boldsymbol{h}, \boldsymbol{r}, \boldsymbol{t}) \\
&\quad + \xi \sum_{i=1}^{N} [(\|\boldsymbol{h}_i\|_2 - 1)^2 + (\|\boldsymbol{r}_i\|_2 - 1)^2 + (\|\boldsymbol{t}_i\|_2 - 1)^2] \\
&\quad + \xi \sum_{\tau=1}^{T} (\|\boldsymbol{w}_\tau\|_2 - 1)^2 + \xi \|\boldsymbol{R}\|_2^2,
\end{aligned} \tag{22}$$

where ξ is a tradeoff parameter. Then, we learn $\boldsymbol{W}, \boldsymbol{h}, \boldsymbol{r}, \boldsymbol{t}$ and $\boldsymbol{R}$ alternatively and iteratively. To be specific, at the ρ-th iteration, we first fix the matrix $\boldsymbol{h}, \boldsymbol{r}, \boldsymbol{t}$ and $\boldsymbol{R}$ and update the value of each $\boldsymbol{w}_\tau$ in $\boldsymbol{W}$ using gradient descent based on the following rule,

$$(\boldsymbol{w}_\tau)_{\rho+1} = (\boldsymbol{w}_\tau)_\rho - \eta \frac{\partial \mathcal{L}(\boldsymbol{W}, \boldsymbol{h}, \boldsymbol{r}, \boldsymbol{t}, \boldsymbol{R})}{\partial \boldsymbol{w}_\tau}, \tag{23}$$

where η is the learning rate, and

$$\begin{aligned}
\frac{\partial \mathcal{L}(\boldsymbol{W}, \boldsymbol{h}, \boldsymbol{r}, \boldsymbol{t}, \boldsymbol{R})}{\partial \boldsymbol{w}_\tau} = &\sum_{\tau=1}^{T} \sum_{s_i^+}^{S_\tau^+} \sum_{s_{j,e}^-, s_{k,r}^-}^{S_{\tau,e}^-, S_{\tau,r}^-} [2\nabla_{\boldsymbol{w}_\tau} \mathcal{L}_{tre}(s_i^+) \\
&- \nabla_{\boldsymbol{w}_\tau} \mathcal{L}_{tre}(s_{j,e}^-) - \alpha \nabla_{\boldsymbol{w}_\tau} \mathcal{L}_{tre}(s_{k,r}^-)]_\dagger \\
&+ \beta \nabla_{\boldsymbol{w}_\tau} \mathcal{L}_{aux}(\boldsymbol{W}, \boldsymbol{R}) + 2\xi(\|\boldsymbol{w}_\tau\|_2 - 1)\boldsymbol{w}_\tau,
\end{aligned} \tag{24}$$

where $[x]_\dagger$ is an indication function, i.e., if $x > 0$ then $[x]_\dagger = x$, otherwise $[x]_\dagger = 0$. Due to the limited space, we don't detail $\nabla_{\boldsymbol{w}_\tau} \mathcal{L}_{aux}(\boldsymbol{W}, \boldsymbol{R})$ in this paper.

After updating the value of $\boldsymbol{W}$, we then alternatively fix $\boldsymbol{W}$ and $\boldsymbol{R}$, and update $\boldsymbol{h}$, $\boldsymbol{r}$ and $\boldsymbol{t}$ respectively. Take $\boldsymbol{h}_i$ for example, we update it based on the following rules,

$$(\boldsymbol{h}_i)_{\rho+1} = (\boldsymbol{h}_i)_\rho - \eta \frac{\partial \mathcal{L}(\boldsymbol{W}, \boldsymbol{h}, \boldsymbol{r}, \boldsymbol{t}, \boldsymbol{R})}{\partial \boldsymbol{h}_i}, \tag{25}$$

where

$$\begin{aligned}
&\frac{\partial \mathcal{L}(\boldsymbol{W}, \boldsymbol{h}, \boldsymbol{r}, \boldsymbol{t}, \boldsymbol{R})}{\partial \boldsymbol{h}_i} \\
&= \sum_{t=1}^{T} \sum_{s_i^+}^{S_\tau^+} \sum_{s_{j,e}^-, s_{k,r}^-}^{S_{\tau,e}^-, S_{\tau,r}^-} [2\nabla_{\boldsymbol{h}_i} \mathcal{L}_{tre}(s_i^+) \\
&- \nabla_{\boldsymbol{h}_i} \mathcal{L}_{tre}(s_{j,e}^-) - \alpha \nabla_{\boldsymbol{h}_i} \mathcal{L}_{tre}(s_{k,r}^-)]_\dagger + 2\xi(\|\boldsymbol{h}_i\|_2 - 1)\boldsymbol{h}_i,
\end{aligned} \tag{26}$$

Finally, we fix $\boldsymbol{W}, \boldsymbol{p}, \boldsymbol{r}$ and $\boldsymbol{t}$ and then update $\boldsymbol{R}_z, \boldsymbol{R}_r, \boldsymbol{R}_T, \boldsymbol{R}_p, \boldsymbol{R}_t$ respectively.

We update $\boldsymbol{W}, \boldsymbol{h}, \boldsymbol{r}, \boldsymbol{t}$ and $\boldsymbol{R}$ alternatively and iteratively until changes of values in the objective function $\mathcal{L}(\boldsymbol{W}, \boldsymbol{h}, \boldsymbol{r}, \boldsymbol{t}, \boldsymbol{R})$ are less than a threshold ε. Considering that there are numerous possible combinations of negative triples, we sample several negative triples for each training triplet. The negative triples are randomly sampled and separated into three groups (M negative head entity samples, M negative relation samples, and M negative tail entity samples).

TABLE 3. Detailed statistics of the Wikidata12K Dataset and YAGO11K Dataset. (# means 'the number of').

	YAGO11k	Wikidata12K
# Entities	10,623	12,554
# Relations	10	24
# Train	16.4k	32.5k
# Valid	2k	4k
# Test	2k	4k

VI. EXPERIMENTS

In this section, we evaluate the effectiveness of our proposed TDG2E in comparison with state-of-the-art KGE methods. Specifically, we first introduce the experiment setup, including datasets, compared baselines and evaluation schema. Then, we compare the performance of TDG2E with that of baselines and analyze the results in detail.

A. EXPERIMENT SETUP

1) DATASETS

KGs such as Wikidata [51] and YAGO [3] have time annotations on a subset of the facts. We use the temporally rich sub-KG extracted from them for training and testing our algorithm as well as the baselines.

- **YAGO11k**. This is a temporal graph generated from YAGO3 knowledge graph [52]. It is formed of 20.5k triples and includes 10,623 entities.
- **Wikidata12k**. It is extracted from Wikidata proposed by [24]. There are 40k triples in total with 12.6k entities in it.

In order to ensure a healthy connectivity within the graph, we filter out sparse triples via picking out the top 10 most frequently temporally rich relations in YAGO3 and top 24 in Wikidata12k. Following the strategy adopted in HyTE, we leave out links containing the entity with only a single mention in the sub-KG. Statistics of the Wikidata12k dataset and the YAGO11k dataset are summarized in Table 3.

2) BASELINES

To verify the effectiveness of TDG2E, we choose the following six state-of-the-art models as the baselines:

- **TransE** [15] learns embeddings of KG elements by translation. It is a static KGE method. It treats the relation as a translation vector between two corresponding embedded entities, where the embedded entities can be connected by the translation vector with low error.
- **TransH** [16] comes as an improvement of TransE by introducing relation-specific hyperplanes. By doing so, TransH enables different roles of an entity in different relations, which plays an import role in dealing with reflexive or many-to-one/one-to-many/many-to-many relations.
- **HolE** [34] is utilized as a static KGE method in this paper. It is a matrix factorization based model, attempts to learn compositional vector space representations of entire knowledge graphs.
- **t-TransE** [27] provides a link prediction method by using temporal order constraints to model transformation between time-sensitive relations. Specifically, in the embedding process, t-TransE enforces the embeddings to be temporally consistent.
- **HyTE** [24] proposes a graph embedding method based on projected-time translation. It represents different times as different hyperplanes, and segregates the embedding space into different time zones by these hyperplanes. In contrast to existing time-aware KGE methods, HyTE is able to encode temporal information directly in the learned embeddings.

Note that the hyper-parameters of baselines are set as the best ones reported in original papers.

3) EVALUATION SCHEMA

Similar to HyTE, we conduct three types of tasks on each dataset to verify the performance of TDG2E, i,e., head entity prediction, tail entity prediction, and relation prediction as mentioned in Section III. The settings for these tasks have been described in the problem statement. For example, the head entity prediction task is to predict the missing head entity with the given tail entity and relation. Specifically, with the learned embeddings and temporal hyperplanes, TDG2E uses Eq. 11 to calculate the loss corresponding to triples formed by each potential candidates head entity and the observed tail entity and relation. Then, TDG2E ranks the loss for the real head entity. TDG2E finally reports the following evaluation metrics on all test data.

- **Mean Rank**. Mean Rank is the averaged rank of the missing elements. In this paper, we report Mean Rank in all three tasks.
- **Hit@K**. Hit at rank K, Hit@K for short, is 1 if the miss elements are ranked within the top K candidates, otherwise 0. Considering the great difference in the magnitude of the entities and relations, we evaluate Hit@10 in the entity prediction task and Hit@1 in the relation prediction task.

For each evaluation metric, all models are repeatedly trained for 5 times and we report the averaged evaluation metrics of each model.

4) IMPLEMENTATION DETAILS

Both YAGO11k and Wikidata12k contain time annotations to the granularity of days. Following the strategy adopted by HyTE, for the temporal scoping task, we only deal with year level granularity by dropping the month and date information. Timestamps are then treated as 61 and 78 different spans for YAGO and Wikidata respectively. As mentioned in Section IV-C, we handle the unbalance issue that may occur in terms of triples number in a particular span by applying a minimum threshold of 300 triples per span during construction.

For fair comparison, we learn all models from scratch without any pretrained parameters. We optimize all

TABLE 4. Comparison of different models on Wikidata12K and YAGO11K datasets. ↑ means the higher the better while ↓ means the lower the better. The best results are in bold.

Datasets	Wikidata12K						YAGO11K					
Metirc	Mean Rank↓			Hits@10%↑		Hits@1%↑	Mean Rank↓			Hits@10%↑		Hits@1%↑
Tasks	head	tail	relation	head	tail	relation	head	tail	relation	head	tail	relation
TransE	742	524	1.48	5.63	11.34	85.42	2139	522	1.72	1.18	4.39	78.67
TransH	629	430	1.42	11.64	23.67	86.18	1967	341	1.51	1.45	5.56	76.51
HolE	810	741	2.26	12.41	24.69	74.29	1931	1716	2.53	1.34	6.33	70.35
t-TransE	418	289	1.84	14.58	23.76	82.68	1595	349	1.62	12.94	27.01	76.58
HyTE	242	181	1.13	24.69	41.38	92.54	1030	**118**	1.24	15.83	38.45	82.13
TDG2E-T	168	110	**1.10**	31.37	44.57	92.50	788	150	**1.15**	20.23	39.92	87.05
TDG2E	**163**	**107**	**1.10**	**32.11**	**48.23**	**92.55**	**782**	148	**1.15**	**21.07**	**41.03**	**87.82**

TABLE 5. Optimal hyper-parameter settings of TDG2E. d denotes the dimension of embeddings; γ is the margin; M is the number of negative sampling triples; ξ, β and α are tradeoff parameters; η is the learning rate.

	d	γ	M	ξ	α	β	η
YAGOO11k	128	1	5	1	0.01	100	0.0001
Wikidata12K	128	1	5	1	0.01	100	0.0001

models with Adaptive Moment Estimation (Adam) and apply a grid search to find out the best settings of hyper-parameters. Hyper-parameters involved in TDG2E, as shown in the loss function and the model learning process, are tradeoff parameters β, ξ and α, learning rate η, negative sampling triples M, embedding size of entity, relation d and margin γ. The tradeoff parameters β, ξ, the learning rate η and the margin γ are chosen individually from the set $\{10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 1.0, 100.0\}$. The best configuration is chosen by corresponding lowest Mean Rank on the validation set. Other hyper-parameters of our proposed model are empirically set as follows: the batch size is 256, embedding size of entity and relation d is 128, α is 0.01, ξ is 1, η, γ is 1 and M is 5 on both the Wikidata12K and YAGOO11k datasets, respectively. The optimal hyper-parameter settings are shown in Table 5.

B. PERFORMANCE COMPARISONS WITH BASELINES

In this section, we report the performance comparisons among TDG2E and baselines. The best results generated by different models on two datasets are presented in Table 4 and the iteration process of Hit Rate generated by different models on YAGO11K dataset is showed in Figure 3. It can be observed that:

1) TDG2E performs best among all methods in terms of Mean Rank and Hit@10 on the two datasets. Take Hit@10 as an example, as shown in Table 4, TDG2E outperforms the best baseline by 30.053% and 16.554% on the Wikidata12K dataset for head entity and tail entity prediction task, respectively. TDG2E also achieves significant improvements of Mean Rank on entity prediction tasks, including the head entity prediction and the tail entity prediction. TDG2E performs worse than the best baseline, i.e., HyTE, in terms of Mean Rank on the relation prediction task on YAGO11K dataset. However, note that the Mean Rank metric is less important since it can be easily reduced by an obstinate triple with a low rank [53].
2) Compared to performance on YAGO11K dataset, performance on Wikidata12K dataset of all methods is better. This is because YAGO11K is sparser than Wikidata12K. With lesser training data on YAGO11K dataset, all methods can not be trained enough, which leads to suboptimal performance.
3) In general, the performance of TransE, TransH and HolE on both entity prediction tasks and the relation prediction task is worse than that of t-TransE, HyTE and TDG2E. This is because both TransE, TransH and HolE only learn one embedding for each entity or relationship, without taking into account the temporal information. With the temporal information encoded in the dynamic knowledge graphs, embeddings learned in t-TransE, HyTE and TDG2E can easily reflect the variations of the entity or the relation over time, therefore they worked better. The results of performance comparisons among static knowledge graph embedding methods and dynamic ones prove the importance of temporal information.
4) Although t-TransE and HyTE have incorporated temporal information into the learned embeddings of DKG elements, their performance is still worse than that of TDG2E. That is because t-TransE and HyTE are designed to fit on different time bins independently, which makes it difficult to share statistical strength among different time bins. And a model trained independently in a specific time bin cannot remain robust when the structure of the graph changes drastically at a specific point. In contrast, TDG2E attempts to capture dependencies among different time bins. This significant performance gain empirically validates our claim that the evolutionary patterns of graph structures among sequential timestamps help to learn richer embeddings of the dynamic knowledge graph elements, in addition to the graph structure information at different timestamps.
5) Comparing with others, TDG2E exhibits a little bit unstable performance in relation prediction. This is partly caused by negative samples of relations. Different from other KGE methods, which only considers the negative sampling of entities in the learning

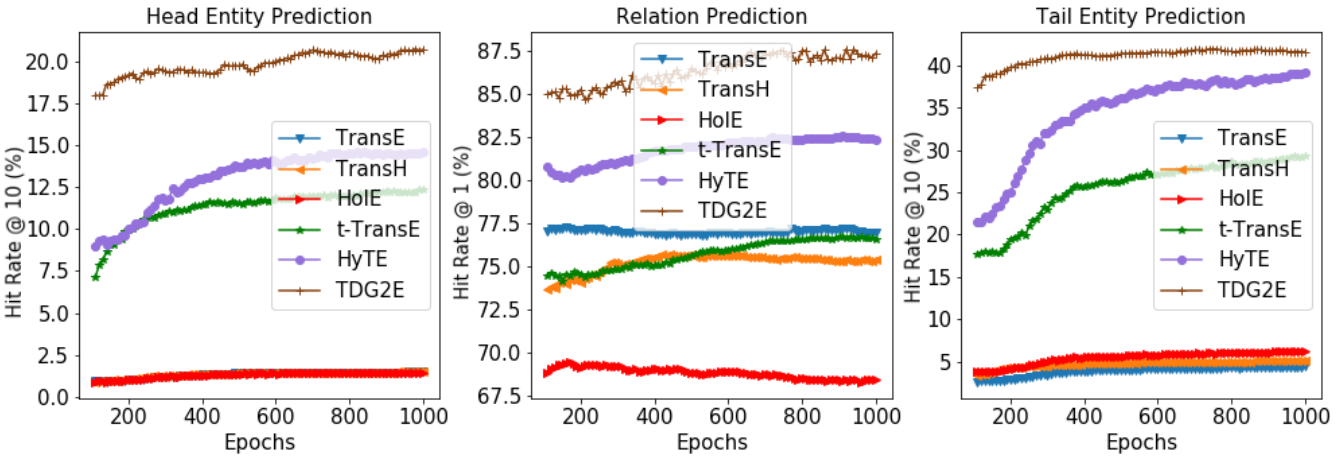


FIGURE 3. The iteration process of Hit Rate generated by different models on YAGO11K dataset.

TABLE 6. Instances of Qualitative Results of different methods on relation prediction task.

Test quadruples	TransE	HyTE	TDG2E
Gordon Carroll, ?, Baltimore,[1928, 1928]	diedIn	wasBornIn*	wasBornIn*
Sanders Anne Laubenthal, ?, Mobile Alabama,[1943, 1943]	wasBornIn*	wasBornIn*	wasBornIn*
Sanders Anne Laubenthal, ?, Washington.,[2002, 2002]	wasBornIn	diedIn*	diedIn*
Eugene Sander, ?, Cornell Univ.,[1959, 1965]	worksAt	graduatedFrom*	graduatedFrom*
Lauren Miller,?,Lakeland Florida,[1982,1982]	wasBornIn*	isMarriedTo	wasBornIn*
Lauren Miller,?,Seth Rogen,[2011,2011]	isMarriedTo*	isMarriedTo*	isMarriedTo*
Ernesto Maceda, ?, Nacionalist Party, [1971,1987]	isMarriedTo	isAffiliatedTo*	isAffiliatedTo*

* the correct relation.

process of embeddings, we adopt the task-oriented negative sampling strategy which considers not only negative sampling of entities but also that of relations as shown in Eq. 7. To illustrate the effect of negative relation sampling on TDG2E' performance intuitively, we adapt TDG2E to *TDG2E-nr* that only considers the negative sampling of entities in the learning process of embeddings. The iteration process of Hit@1 in relation prediction generated by TDG2E, TDG2E-nr and other models is showed in Figure 4. From the figure, we can see that TDG2E-nr's performance is a little bit stable, verifying that TDG2E's unstable performance is in part because of negative samples of relations.

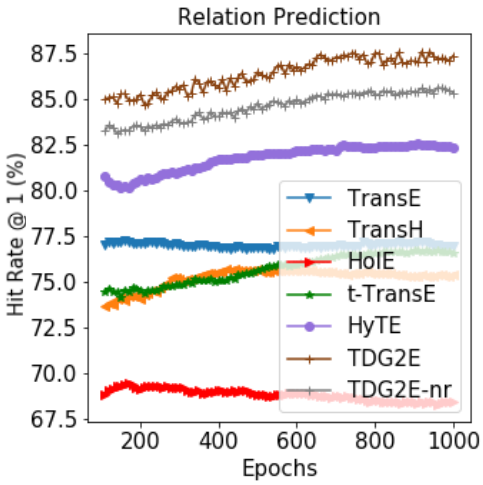


FIGURE 4. The iteration process of Hit@1 in relation prediction generated by different models on YAGO11K dataset.

C. PERFORMANCE COMPARISONS WITH THE VARIANT

To demonstrate the importance to consider the impact of timespans on modeling the evolving process of DKGs, we also compare our method TDG2E with its variant, named as **TDG2E-T**. Different from TDG2E, TDG2E-T adopts the general GRU to process each hyperplane $\boldsymbol{w}_\tau$, without the introduction of the Timespan Gate. In TDG2E-T, the hidden state and auxiliary loss are calculated as Eq. 15 and 16 respectively. The performance comparisons between TDG2E-T and TDG2E are reported in Table 4.

We notice that, in general, performance of TDG2E-T on both Wikidata12K and YAGO11K datasets does not exceed that of TDG2E. This is because that Timespan Gate introduced in TDG2E can balance the impacts of different time bins on their successors, which plays an important role to capture the evolving process of DKGs correctly.

D. PARAMETER ANALYSIS

As mentioned above, we optimize all models with Adaptive Moment Estimation (Adam) and apply a grid search to

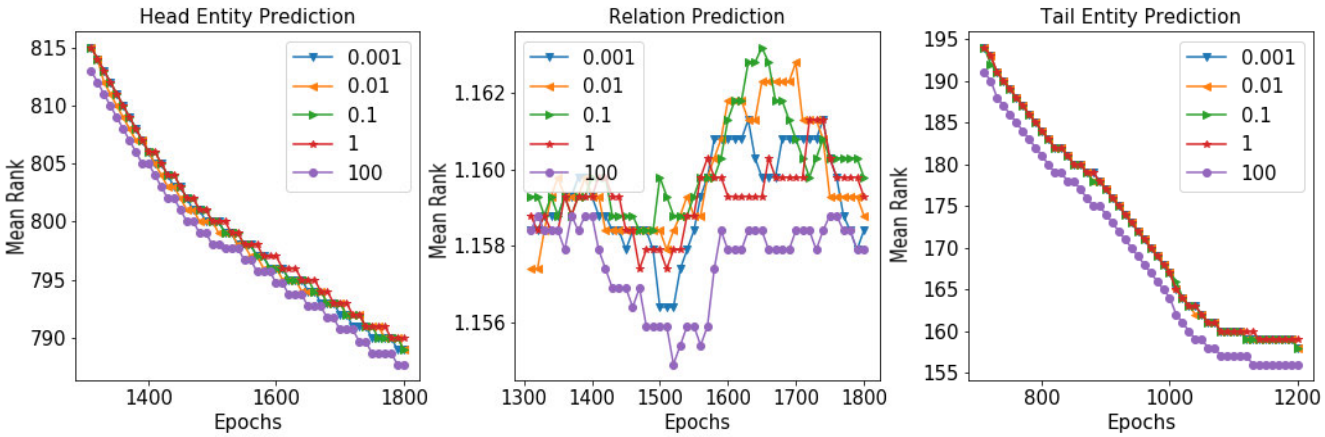


FIGURE 5. Sensitivity Study on Trade-off Parameters β. To make it intuitive and simple, we choose different epoch spans for three tasks. Note that the three sub-figures don't share the same order of magnitude.

find out the best settings of hyper-parameters for TDG2E. For the trade-off parameter β in Eq. 22, which balances the weight of auxiliary loss, we also adopt the grid search strategy to find its most suitable value by searching in $\{10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 1.0, 100.0\}$.

Fig. 5 reports the potential impacts imposed by the value of trade-off parameter β to the Mean Rank of TDG2E on the YAGO11K dataset. As shown in Fig.5, the Mean Rank of TDG2E on the three tasks, including the head entity prediction task, the relation prediction task and the tail entity prediction task, fluctuates greatly at first. This is because that the epoch is small and the model is not stable. The fluctuation of Mean Rank then tends to be mild as epoch increases. And the Mean Rank of TDG2E gets its smallest at the maximum epoch when β is around 100. These results indicate that an unsuitable β will increase the Mean Rank of TDG2E, which is not conducive to prediction. Therefore, we set the number of trade-off parameter, β to 100.

E. QUALITATIVE RESULTS

In order to demonstrate the advantages of our approach more intuitively, we carry out some qualitative analyses. Table 6 reports some instances for the relation prediction task on YAGO11K dataset. We can see that:

1) TransE is confused by the temporal relations, including *wasBornIn* and *diedIn*. TransE predicts wrongly that Gordon Carroll "died in Baltimore in 1928", while the fact is that Gordon Carroll "was born in Baltimore in 1928". However, HyTE and TDG2E can figure out the truth correctly. This instance can show the importance of considering the temporal information in DKGs.
2) As shown by the second instance, both TransE and dynamic knowledge graph embedding methods, including HyTE and TDG2E, can correctly predict *wasBornIn* for the incomplete fact *(Sanders Anne Laubenthal, ?, Mobile Alabam)* in 1943. However, when the query year is 2002, TransE predicts wrongly that Sanders Anne Laubentha "was born in Washingto", while HyTE and TDG2E can predict *diedIn* correctly. This is because TransE is fit for static knowledge graph embedding, without taking the temporal information into consideration. In contrast, HyTE and TDG2E encode a prior knowledge that Sanders Anne Laubenthal "was born in 1943", "created Excalibur in 1973" from the training data, and come to the correct relation through the relative temporal ordering.
3) HyTE makes some type inconsistency in relation prediction. For the missing fact (*Lauren Miller, ?, Lakeland Florida*) in 1982, HyTE predicts *isMarriedTo* wrongly. This can be attributed to the fact that HyTE does not impose any type related constraints in its embedding process.
4) TDG2E does well in dealing with all these seven instances as shown in Table 6. In our qualitative experiments, we can observe abundant instances of this kind. TDG2E is naturally learning some relation ordering in parallel with the temporal direction.

VII. CONCLUSION AND FUTURE WORK

In this paper, we propose a novel model termed as TDG2E, which aims to directly encode temporal information in the learned embeddings of dynamic knowledge graphs. In contrast with other state-of-the-art methods of static/dynamic embedding, the main characteristics of our approach are two fold. Firstly, TDG2E incorporates temporal evolving process underlying the given DKG by utilizing a GRU based model. Furthermore, considering the impacts of cumulative structural information to the consecutive structure, TDG2E further introduces an auxiliary loss to use hiddden state of GRU for supervising the learning of the next sub-KG. By taking into account the evolutionary patterns and the cumulative structural information, the proposed TDG2E can preserve not only structural information of current sub-KG but also evolutionary patterns of the DKG. Secondly, to further deal with

the time unbalance issue underlying the DKGs, a Timespan Gate is designed in GRU. It enables TDG2E to model the evolutionary patterns of DKGs more effectively by incorporating the timespan. Experiments on Wikidata12K dataset and YAGO11K dataset verify the superiority of TDG2E over other state-of-the-art baseline methods in terms of Mean Rank and Hit Rate.

In the future, we plan to further evaluate the effectiveness of our model on more real-world datasets. Moreover, considering that the hyperplane captures only the time information while ignores the relation-specific information, we would like to incorporate comprehensive hyperplanes to further improve our model.

ACKNOWLEDGMENT

(Xiaoli Tang and Rui Yuan contributed equally to this work.)

REFERENCES

[1] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor, "Freebase: A collaboratively created graph database for structuring human knowledge," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, 2008, pp. 1247–1250.

[2] J. Lehmann, R. Isele, M. Jakob, A. Jentzsch, D. Kontokostas, P. N. Mendes, S. Hellmann, M. Morsey, P. Van Kleef, S. Auer, and C. Bizer, "DBpedia—A large-scale, multilingual knowledge base extracted from Wikipedia," *Semantic Web*, vol. 6, no. 2, pp. 167–195, 2015.

[3] F. M. Suchanek, G. Kasneci, and G. Weikum, "YAGO: A core of semantic knowledge," in *Proc. ACM 16th Int. Conf. World Wide Web*, 2007, pp. 697–706.

[4] A. Carlson, J. Betteridge, B. Kisiel, B. Settles, E. R. Hruschka, and T. M. Mitchell, "Toward an architecture for never-ending language learning," in *Proc. 24th AAAI Conf. Artif. Intell.*, 2010.

[5] J. Berant, A. Chou, R. Frostig, and P. Liang, "Semantic parsing on freebase from question-answer pairs," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2013, pp. 1533–1544.

[6] L. Heck, D. Hakkani-Tür, and G. Tur, "Leveraging knowledge graphs for Web-scale unsupervised semantic parsing," in *Proc. Interspeech*. Graz, Austria: International Speech Communication Association, Aug. 2013. [Online]. Available: https://www.microsoft.com/en-us/research/publication/leveraging-knowledge-graphs-for-web-scale-unsupervised-semantic-parsing/

[7] D. Damljanovic and K. Bontcheva, "Named entity disambiguation using linked data," in *Proc. 9th Extended Semantic Web Conf.*, 2012, pp. 231–240.

[8] Z. Zheng, X. Si, F. Li, E. Y. Chang, and X. Zhu, "Entity disambiguation with freebase," in *Proc. IEEE/WIC/ACM Int. Joint Conf. Web Intell. Intell. Agent Technol.*, vol. 1, Dec. 2012, pp. 82–89.

[9] R. Hoffmann, C. Zhang, X. Ling, L. Zettlemoyer, and D. S. Weld, "Knowledge-based weak supervision for information extraction of overlapping relations," in *Proc. 49th Annu. Meeting Assoc. for Comput. Linguistics, Hum. Lang. Technol.*, vol. 1, 2011, pp. 541–550.

[10] J. Daiber, M. Jakob, C. Hokamp, and P. N. Mendes, "Improving efficiency and accuracy in multilingual entity extraction," in *Proc. ACM 9th Int. Conf. Semantic Syst.*, 2013, pp. 121–124.

[11] A. Bordes, J. Weston, and N. Usunier, "Open question answering with weakly supervised embedding models," in *Proc. Joint Eur. Conf. Mach. Learn. Knowl. Discovery Databases*. Berlin, Germany: Springer, 2014, pp. 165–180.

[12] A. Bordes, S. Chopra, and J. Weston, "Question answering with subgraph embeddings," 2014, *arXiv:1406.3676*. [Online]. Available: https://arxiv.org/abs/1406.3676

[13] Q. Wang, Z. Mao, B. Wang, and L. Guo, "Knowledge graph embedding: A survey of approaches and applications," *IEEE Trans. Knowl. Data Eng.*, vol. 29, no. 12, pp. 2724–2743, Dec. 2017.

[14] M. Nickel, V. Tresp, and H.-P. Kriegel, "A three-way model for collective learning on multi-relational data," in *Proc. ICML*, vol. 11, 2011, pp. 809–816.

[15] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, and O. Yakhnenko, "Translating embeddings for modeling multi-relational data," in *Proc. Adv. Neural Inf. Process. Syst.*, 2013, pp. 2787–2795.

[16] Z. Wang, J. Zhang, J. Feng, and Z. Chen, "Knowledge graph embedding by translating on hyperplanes," in *Proc. 28th AAAI Conf. Artif. Intell.*, 2014.

[17] Y. Lin, Z. Liu, M. Sun, Y. Liu, and X. Zhu, "Learning entity and relation embeddings for knowledge graph completion," in *Proc. 29th AAAI Conf. Artif. Intell.*, 2015.

[18] R. Jenatton, N. L. Roux, A. Bordes, and G. R. Obozinski, "A latent factor model for highly multi-relational data," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 3167–3175.

[19] A. Bordes, X. Glorot, J. Weston, and Y. Bengio, "A semantic matching energy function for learning with multi-relational data," *Mach Learn*, vol. 94, no. 2, pp. 233–259, Feb. 2014.

[20] R. Socher, D. Chen, C. D. Manning, and A. Ng, "Reasoning with neural tensor networks for knowledge base completion," in *Proc. Adv. Neural Inf. Process. Syst.*, 2013, pp. 926–934.

[21] P. Goyal and E. Ferrara, "Graph embedding techniques, applications, and performance: A survey," *Knowl.-Based Syst.*, vol. 151, pp. 78–94, Jul. 2018.

[22] H. Cai, V. W. Zheng, and K. C.-C. Chang, "A comprehensive survey of graph embedding: Problems, techniques, and applications," *IEEE Trans. Knowl. Data Eng.*, vol. 30, no. 9, pp. 1616–1637, Sep. 2018.

[23] J. Yang and J. Leskovec, "Defining and evaluating network communities based on ground-truth," *Knowl. Inf. Syst.*, vol. 42, no. 1, pp. 181–213, Jan. 2015.

[24] S. S. Dasgupta, S. N. Ray, and P. Talukdar, "Hyte: Hyperplane-based temporally aware knowledge graph embedding," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2018, pp. 2001–2011.

[25] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," 2014, *arXiv:1412.3555*. [Online]. Available: https://arxiv.org/abs/1412.3555

[26] J. Feng, M. Huang, M. Wang, M. Zhou, Y. Hao, and X. Zhu, "Knowledge graph embedding by flexible translation," in *Proc. 15th Int. Conf. Principles Knowl. Represent. Reasoning*, 2016.

[27] T. Jiang, T. Liu, T. Ge, L. Sha, S. Li, B. Chang, and Z. Sui, "Encoding temporal information for time-aware link prediction," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2016, pp. 2350–2354.

[28] S. M. Kazemi, R. Goel, K. Jain, I. Kobyzev, A. Sethi, P. Forsyth, and P. Poupart, "Relational representation learning for dynamic (knowledge) graphs: A survey," 2019, *arXiv:1905.11485*. [Online]. Available: https://arxiv.org/abs/1905.11485

[29] W. Jin, H. Jiang, M. Qu, T. Chen, C. Zhang, P. Szekely, and X. Ren, "Recurrent event network: Global structure inference over temporal knowledge graph," 2019, *arXiv:1904.05530*. [Online]. Available: https://arxiv.org/abs/1904.05530

[30] Y. Liu, W. Hua, K. Xin, and X. Zhou, "Context-aware temporal knowledge graph embedding," in *Proc. Int. Conf. Web Inf. Syst. Eng.* Cham, Switzerland: Springer, 2019, pp. 583–598.

[31] G. Ji, S. He, L. Xu, K. Liu, and J. Zhao, "Knowledge graph embedding via dynamic mapping matrix," in *Proc. 53rd Annu. Meeting Assoc. Comput. Linguistics, 7th Int. Joint Conf. Natural Lang. Process.*, vol. 1, 2015, pp. 687–696.

[32] G. Ji, K. Liu, S. He, and J. Zhao, "Knowledge graph completion with adaptive sparse transfer matrix," in *Proc. 30th AAAI Conf. Artif. Intell.*, 2016.

[33] B. Yang, W.-T. Yih, X. He, J. Gao, and L. Deng, "Embedding entities and relations for learning and inference in knowledge bases," 2014, *arXiv:1412.6575*. [Online]. Available: https://arxiv.org/abs/1412.6575

[34] M. Nickel, L. Rosasco, and T. Poggio, "Holographic embeddings of knowledge graphs," in *Proc. 30th Aaai Conf. Artif. Intell.*, 2016.

[35] T. Trouillon, J. Welbl, S. Riedel, E. Gaussier, and G. Bouchard, "Complex embeddings for simple link prediction," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 2071–2080.

[36] S. M. Kazemi and D. Poole, "Simple embedding for link prediction in knowledge graphs," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 4284–4295.

[37] I. Balažević, C. Allen, and T. M. Hospedales, "Hypernetwork knowledge graph embeddings," in *Proc. Int. Conf. Artif. Neural Netw.* Springer, 2019, pp. 553–565.

[38] T. Dettmers, P. Minervini, P. Stenetorp, and S. Riedel, "Convolutional 2D knowledge graph embeddings," in *Proc. 32nd AAAI Conf. Artif. Intell.*, 2018.

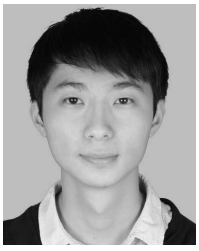
[39] X. Dong, E. Gabrilovich, G. Heitz, W. Horn, N. Lao, K. Murphy, T. Strohmann, S. Sun, and W. Zhang, "Knowledge vault: A Web-scale approach to probabilistic knowledge fusion," in *Proc. 20th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2014, pp. 601–610.
[40] L. Zhu, D. Guo, J. Yin, G. Ver Steeg, and A. Galstyan, "Scalable temporal latent space inference for link prediction in dynamic social networks," *IEEE Trans. Knowl. Data Eng.*, vol. 28, no. 10, pp. 2765–2777, Jul. 2016.
[41] L. Zhou, Y. Yang, X. Ren, F. Wu, and Y. Zhuang, "Dynamic network embedding by modeling triadic closure process," in *Proc. 32nd AAAI Conf. Artif. Intell.*, 2018.
[42] Y. Zuo, G. Liu, H. Lin, J. Guo, X. Hu, and J. Wu, "Embedding temporal network via neighborhood formation," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2018, pp. 2857–2866.
[43] A. Sanchez-Gonzalez, N. Heess, J. T. Springenberg, J. Merel, M. Riedmiller, R. Hadsell, and P. Battaglia, "Graph networks as learnable physics engines for inference and control," 2018, *arXiv:1806.01242*. [Online]. Available: https://arxiv.org/abs/1806.01242
[44] P. W. Battaglia *et al.*, "Relational inductive biases, deep learning, and graph networks," 2018, *arXiv:1806.01261*. [Online]. Available: https://arxiv.org/abs/1806.01261
[45] R. Palm, U. Paquet, and O. Winther, "Recurrent relational networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 3368–3378.
[46] A. Pareja, G. Domeniconi, J. Chen, T. Ma, T. Suzumura, H. Kanezashi, T. Kaler, and C. E. Leiserson, "EvolveGCN: Evolving graph convolutional networks for dynamic graphs," 2019, *arXiv:1902.10191*. [Online]. Available: https://arxiv.org/abs/1902.10191
[47] R. Trivedi, H. Dai, Y. Wang, and L. Song, "Know-evolve: Deep temporal reasoning for dynamic knowledge graphs," in *Proc. 34th Int. Conf. Mach. Learn. (JMLR)*, vol. 70, 2017, pp. 3462–3471.
[48] A. García-Durán, S. Dumančić, and M. Niepert, "Learning sequence encoders for temporal knowledge graph completion," 2018, *arXiv:1809.03202*. [Online]. Available: https://arxiv.org/abs/1809.03202
[49] J. Leblay and M. W. Chekol, "Deriving validity time in knowledge graph," in *Proc. Companion Web Conf. Int. World Wide Web Conf. Steering Committee*, 2018, pp. 1771–1776.
[50] Q. Li, X. Tang, T. Wang, H. Yang, and H. Song, "Unifying task-oriented knowledge graph learning and recommendation," *IEEE Access*, vol. 7, pp. 115816–115828, 2019.
[51] F. Erxleben, M. Günther, M. Krötzsch, J. Mendez, and D. Vrandečić, "Introducing Wikidata to the linked data Web," in *Proc. Int. Semantic Web Conf.* Cham, Switzerland: Springer, 2014, pp. 50–65.
[52] F. Mahdisoltani, J. Biega, and F. M. Suchanek, "YAGO3: A knowledge base from multilingual Wikipedias," Tech. Rep., 2013.
[53] Z. Wang and J.-Z. Li, "Text-enhanced representation learning for knowledge graph," in *Proc. IJCAI*, 2016, pp. 1293–1299.

XIAOLI TANG is currently pursuing the degree with the School of Software Engineering, South China University of Technology. Her research interests include recommender systems, information retrieval, and knowledge graph.

RUI YUAN received the B.S. degree from the School of Communication and Design, Sun Yat-sen University, and the M.S. degree from the School of Journalism and Communication, The Chinese University of Hong Kong. His research interests include big data analysis and data mining.

QIANYU LI is currently pursuing the Ph.D. degree with the School of Software Engineering, South China University of Technology. Her research interests include machine learning and data mining, in particular, on knowledge graph embedding.

TENGYUN WANG is currently pursuing the Ph.D. degree with the School of Software Engineering, South China University of Technology. His research interests include data mining, machine learning, and applied data science, including computational advertising.



HAIZHI YANG is currently pursuing the Ph.D. degree with the School of Software Engineering, South China University of Technology. His research interests include machine learning, data mining, and applied data science, including computational advertising.

YUNDONG CAI is currently a Researcher with the Joint NTU-UBC Research Centre of Excellence in Active Living for the Elderly, Nanyang Technological University, Singapore. His research interests include multiagent systems, interactive storytelling, and blockchain.

HENGJIE SONG is currently a Professor with the School of Software Engineering, South China University of Technology. He has published several high quality articles on the best journals and conference proceedings, including the *ACM Transactions on the Web*, the IEEE CIMs, the IEEE TRANSACTIONS ON FUZZY SYSTEMS, *Neural Networks* (Elsevier), AAAI, and ICDM. His research interests include artificial intelligence and the applications of AI in commercial search engines.