Time-aware Graph Neural Networks for Entity Alignment between Temporal Knowledge Graphs

Anonymous EMNLP submission

Abstract

Entity alignment aims to identify equivalent entity pairs between different knowledge graphs (KGs). Recently, the availability of temporal KGs (TKGs) that contain time information created the need for reasoning over time in such TKGs. Existing embedding-based entity alignment approaches disregard time information that commonly exists in many large-scale KGs, leaving much room for improvement. In this paper, we focus on the task of aligning entity pairs between TKGs and propose a novel Time-aware Entity Alignment approach based on Graph Neural Networks (TEA-GNN). We embed entities, relations and timestamps of different KGs into a vector space and use GNNs to learn entity representations. To incorporate both relation and time information into the GNN structure of our model, we use a self-attention mechanism which assigns different weights to different nodes with orthogonal transformation matrices computed from embeddings of the relevant relations and timestamps in a neighborhood. Experimental results on multiple real-world TKG datasets show that our method significantly outperforms the state-of-the-art methods due to the inclusion of time information.

1 Introduction

Knowledge Graphs (KGs) provide a means for structured knowledge representation through connected nodes via edges. The nodes represent entities and the edges connecting these nodes denote relations. A KG stores facts as triples of the form (e_s, r, e_o), where e_s is the subject entity, e_o is the object entity, and r is the relation between entities. Many large-scale KGs including YAGO (Suchanek et al., 2007) and DBpedia (Lehmann et al., 2015) have been established and are widely used in NLP applications, e.g., question answering and language modeling (Wang et al., 2017).

Since most KGs are developed independently and many of them are supplementary in contents, one of core challenges of KGs is to align equivalent entity pairs between different KGs. To address this issue, embedding-based approaches are leveraged to model entities and relations across multiple KGs and measure the similarities between entities (Sun et al., 2020). It has been proven that the utility of multi-relation information is helpful for an effective entity alignment approach (Mao et al., 2020a).

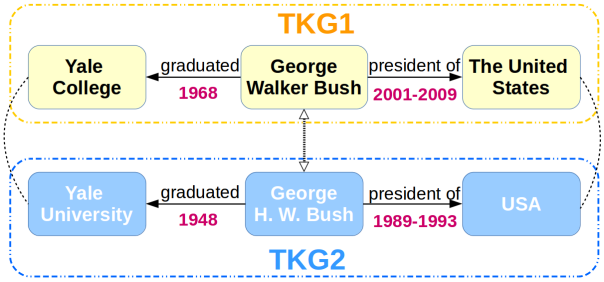

Figure 1: Illustration of the limitation of the existing time-agnostic entity align approaches.

In addition to relation information, many KGs including YAGO3 (Mahdisoltani et al., 2013), Wikidata (Erxleben et al., 2014) and ICEWS (Lautenschlager et al., 2015) also contain time information between entities, i.e., some edges between entities have two properties, relation and time as shown in Figure 1. Facts in such temporal KGs (TKGs) can be represented as quadruples shaped like (e_s, r, e_o, τ) where τ denotes the timestamp. Noteworthily, timestamps in most TKGs are presented in Arabic numerals and have similar formats. Thus, timestamps representing the same dates across multiple TKGs can be easily aligned by manually uniforming their formats.

However, the existing embedding-based entity alignment approaches disregard time information in TKGs, leaving much room for improvement. Taking the case in Figure 1 as an example, given two entities, George H. W. Bush and George Walker Bush, existing in two TKGs respectively, time-agnostic embedding-based approaches are likely to ignore time information and wrongly

recognize these two entities as the same person in the real world due to the homogeneity of their neighborhood information.

To address this issue, an intuitive solution is to incorporate time information into entity alignment models. Inspired by the recent successful applications of GNN models in entity alignment, in this paper, we propose a novel Time-aware Entity Alignment approach based on Graph Neural Networks (TEA-GNN) for entity alignment between TKGs. Different from some temporal GNN models which discretize temporal graphs into multiple snapshots, we treat timestamps as properties of links between entities. We first map all entities, relations and timestamps in TKGs into an embedding space. To incorporate relation and time information into the GNN structure, we utilize a self-attention mechanism which assigns different importance weights to different nodes within a neighborhood according to orthogonal transformation matrices computed with the embedddings of the corresponding relations and timestamps. To further integrate time information into the final entity representations, we concatenate output features of entities with the summation of their neighboring time embeddings to get multi-view entity representations.

Specifically, we create a reverse relation r^{-1} for each relation r to integrate direction information. And a time-aware fact involving a time interval $(e_s, r, e_o, [\tau_b, \tau_e])$, where τ_b and τ_e denote the begin and end time, is separated into two quadruples (e_s, r, e_o, τ_b) and $(e_o, r^{-1}, e_s, \tau_e)$, which represent the begin and the end of the relation, respectively. In this way, TEA-GNN can adapt well to datasets where timestamps are represented in various forms: time points, begin or end time, time intervals.

To verify our proposed approach, we evaluate TEA-GNN and its time-agnostic variant as well as several state-of-the-art entity alignment approaches on real-world datasets extracted from ICEWS, YAGO3 and Wikidata. Experimental results show that TEA-GNN significantly outperforms all baseline models with the inclusion of time information. To the best of our knowledge, this work is the first attempt to perform entity alignment between TKGs using a time-aware embedding-based approach.

2 Related Work

2.1 Knowledge Graph Embedding

KG embedding (KGE) aims to embed entities and relations into a low-dimensional vector space and measure the plausibility of each triples (e_s, r, e_o) by defining a score function. A typical KGE model is TransE (Bordes et al., 2013) which is based on the assumption of $e_s + r \approx e_o$. In addition to translational KGE models including TransE and its variants (Wang et al., 2014; Nayyeri et al., 2019, 2020), other KGE models can be classified into semantic matching models (Yang et al., 2014; Xu et al., 2020b) or neural network-based models (Dettmers et al., 2018; Schlichtkrull et al., 2018).

With the development of TKGs, TKG embedding (TKGE) draws increasing attention (Leblay and Chekol, 2018; Xu et al., 2019, 2020a; Lacroix et al., 2020). An example of typical TKGE models is TTransE (Leblay and Chekol, 2018) which represents timestamps as latent vectors with entities and relations and incorporates time embeddings into its score function $||e_s + r + \tau - e_o||$. The success of TKGE models shows that the inclusion of time information is helpful for reasoning over TKGs.

2.2 Graph Neural Network

Benefitting from the ability to model non-Euclidean space, GNN has become increasingly popular in many areas, including social networks and KGs (Schlichtkrull et al., 2018). Graph Convolutional Network (GCN) (Kipf and Welling, 2016) is an extension of GNN, which generates node-level embeddings by aggregating information from the nodes' neighborhoods. Furthermore, Graph Attention Network (GAT) (Veličković et al., 2017) employs a self-attention mechanism to calculate the hidden representations of each entity by attending over its neighbors.

With the success of these GNN models in the static setting, we approach further practical scenarios where the graph temporally evolves. Existing approaches (Chen et al., 2018; Manessi et al., 2020; Pareja et al., 2020; Wu et al., 2020) generally discretize a temporal graph into multiple static snapshots in a timeline and utilize a combination of GNNs and recurrent architectures (e.g., LSTM), whereby the former digest graph information and the latter handle dynamism.

2.3 Knowledge Graph Alignment

Many KG alignment approaches are proposed to find equivalent entities across multiple KGs by measuring the similarities between entity embeddings. Most embedding-based entity alignment approaches can be classified into two categories, i.e., translational models and GNN-based models.

Typical translational entity align models are based on embeddings learned from TransE and its variants. MTransE (Chen et al., 2016) learns a mapping between two separate KGE spaces. JAPE (Sun et al., 2017) proposes to jointly learn structure embeddings and attribute embeddings in a uniform optimization objective. IPTransE (Zhu et al., 2017) and BootEA (Sun et al., 2018) employ a semi-supervised learning strategy which iteratively label new entity alignment as supervision. The main limitation of translational models is their inability of modeling 1-n, n-1 and n-n relations. Besides, they may lack to exploit the global view of entities since TransE is trained on individual triples.

Many recent studies introduce GNNs into entity alignment task, which is originated with the ability to model global information of graphs. GCN-Align utilizes GCNs to embed entities of each KG into a unified vector space without the prior knowledge of relations. After that, a bunch of GCN-based approaches are proposed to incorporate relation information into GCNs. HGCN (Wu et al., 2019a) jointly learn both entity and relation representations via a GCN-based framework and RDGCN (Wu et al., 2019b) construct a dual relation graph for embedding learning. MuGNN (Cao et al., 2019), MRAEA and RREA (Mao et al., 2020a,b) assign different weight coefficients to entities according to relation types between them, which empowers the models to distinguish the importance between different entities. Our framework TEA-GNN adopts a similar idea with additional time information.

3 Problem Formulation

Formally, a TKG is represented as $\mathcal{G} = (\mathcal{E}, \mathcal{R}, \mathcal{T}, \mathcal{Q})$ where $\mathcal{E}$, $\mathcal{R}$ and $\mathcal{T}$ are the sets of entities, relations and timestamps, respectively. $\mathcal{Q} \subset \mathcal{E} \times \mathcal{R} \times \mathcal{E} \times \mathcal{T}$ is the set of factual quadruples. Let $\mathcal{G}_1 = (\mathcal{E}_1, \mathcal{R}_1, \mathcal{T}_1, \mathcal{Q}_1)$ and $\mathcal{G}_2 = (\mathcal{E}_2, \mathcal{R}_2, \mathcal{T}_2, \mathcal{Q}_2)$ be two TKGs, and $\mathcal{S} = \{(e_{i1}, e_{i2}) | e_{i1} \in \mathcal{E}_1, e_{i2} \in \mathcal{E}_2\}$ be the set of pre-aligned entity pairs between $\mathcal{G}_1$ and $\mathcal{G}_2$. As mentioned in Section 1, timestamps in different TKGs can be easily aligned by manually uniforming their formats. A uniform time set $\mathcal{T}^* = \mathcal{T}_1 \cup \mathcal{T}_2$ can be constructed for both TKGs. Therefore, two TKGs can be renewed as $\mathcal{G}_1 = (\mathcal{E}_1, \mathcal{R}_1, \mathcal{T}^*, \mathcal{Q}_1)$ and $\mathcal{G}_2 = (\mathcal{E}_2, \mathcal{R}_2, \mathcal{T}^*, \mathcal{Q}_2)$ sharing the same set of timestamps. The task of time-aware entity alignment aims to find new aligned entity pairs between $\mathcal{G}_1$ and $\mathcal{G}_2$ based on the prior knowledge of $\mathcal{S}$ and $\mathcal{T}^*$.

4 The Proposed Approach

To exploit both relation and time information for entity alignment, we first create a reverse link for each link so that each pair of reverse links between entities can represent relation directions and handle the begin and end of the relation. An orthogonal transformation-based self-attention mechanism is employed in each GCN layer to assign different weights to entities according to relation and time information between them. Finally, entity alignments are predicted by applying a distance function to multi-view representations of entities.

4.1 Reverse Link Generation

Time information τ in a temporal fact (e_s, r, e_o, τ) can be represented in various forms, e.g., time points, begin or end time and time intervals. A time interval is shaped like $[\tau_b, \tau_e]$ where τ_b and τ_e denote the actual begin time and end time of the fact, respectively. A time point can be represented as $[\tau_b, \tau_e]$ where $\tau_b = \tau_e$. Noteworthily, we represent a begin or end time as $[\tau_b, \tau_0]$ or $[\tau_0, \tau_e]$ where $\tau_0 \in \mathcal{T}^*$ is the first time step in the time set denoting the unknown time information. A fact without known time information can be denoted as $(e_s, r, e_o, [\tau_0, \tau_0])$ to deal with heterogeneous temporal knowledge bases where a significant amount of relations might be non-temporal.

In order to integrate relation direction, we create a reverse relation r^{-1} for each relation r and extend the relation set $\mathcal{R} = \{r_0, r_1, \cdots, r_{|\mathcal{R}|-1}\} \rightarrow \{r_0, r_0^{-1}, \cdots, r_{|\mathcal{R}|-1}, r_{|\mathcal{R}|-1}^{-1}\}$. And each fact $(e_s, r, e_o, [\tau_b, \tau_e])$ is decomposed into two quadruples (e_s, r, e_o, τ_b) and $(e_o, r^{-1}, e_s, \tau_e)$ to handle the begin and the end of the relation, respectively.

4.2 Time-Aware Self-Attention

We map all of entities, relations (including reverse relations) and time steps in both TKGs into a same vector space $\mathbb{R}^k$ where k denotes the embedding dimension. Embeddings of the entity e_i, relation r_j, time step τ_m are denoted as $h_{e_i}, h_{r_j}, h_{\tau_m} \in \mathbb{R}^k$.

Some recent studies (Smith et al., 2017; Pei et al., 2019) show that orthogonal transformation matrix is desirable and robust when transforming one isomorphic embedding to another. Thus, for each relation embedding h_{r_j} and time embedding h_{τ_m}, we define the corresponding orthogonal transformation matrices $M_{r_j}, M_{\tau_m} \in \mathbb{R}^{k \times k}$ as follows,

$$M_{r_j} = I - 2h_{r_j}h_{r_j}^T, \; M_{\tau_m} = I - 2h_{\tau_m}h_{\tau_m}^T, \quad (1)$$

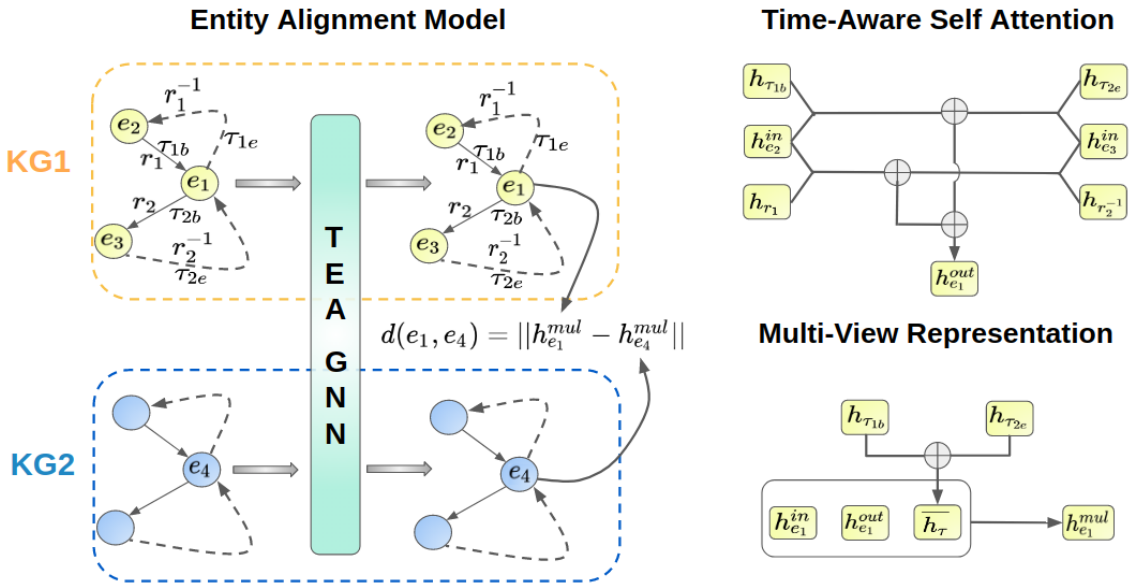

Figure 2: Framework of our approach. Dashed arrows represent the created reverse links.

where embeddings h_{r_j} and h_{τ_m} are normalized to ensure $h_{r_j}^T h_{r_j} = h_{\tau_m}^T h_{\tau_m} = 1$. By doing this, we can easily prove that transformation matrices M_{r_j} and M_{τ_m} are orthogonal. Taking M_{τ_m} as an example, we can obtain

$$\begin{aligned} M_{\tau_m}^T M_{\tau_m} &= (I - 2h_{\tau_m} h_{\tau_m}^T)^T (I - 2h_{\tau_m} h_{\tau_m}^T) \\ &= I - 4h_{\tau_m} h_{\tau_m}^T + 4h_{\tau_m} h_{\tau_m}^T h_{\tau_m} h_{\tau_m}^T = I. \end{aligned} \quad (2)$$

By using such orthogonal transformation matrices, the norms and the relative distances of entities can remain unchanged after transformation, i.e.,

$$\begin{aligned} ||h_{e_i} M_{\tau_m}|| &= ||h_{e_i}||, \\ h_{e_i}^T h_{e_j} &= (h_{e_i} M_{\tau_m})^T (h_{e_j} M_{\tau_m}). \end{aligned} \quad (3)$$

A self-attention mechanism is used to integrate both time and relation information into entity representations by assigning different weights to different neighboring nodes according to the orthogonal transformation matrices of relations and timestamps of the corresponding inward links. In the case of Figure 2, the inward links in the neighborhood of the entity e_1 include $(e_2, r_1, e_1, \tau_{1b})$ and $(e_3, r_2^{-1}, e_1, \tau_{2e})$ in which e_1 performs as the object entity. We define the time-specific weighted importance $\alpha_{i,j,m}$ and the relation-specific weighted importance $\beta_{i,j,m}$ of the mth inward link from the neighboring entity e_j to e_i as follows,

$$\begin{aligned} \alpha_{i,j,m} &= \nu_\tau^T [h_{e_i}^{in} || M_{\tau_m} h_{e_j}^{in} || h_{\tau_m}], \\ \beta_{i,j,m} &= \nu_r^T [h_{e_i}^{in} || M_{r_m} h_{e_j}^{in} || h_{r_m}], \end{aligned} \quad (4)$$

where $\nu_\tau^T, \nu_r^T \in \mathbb{R}^{3k}$ are shared temporal and relational attention weight vectors. $h_{e_i}^{in}, h_{e_j}^{in} \in \mathbb{R}^k$ are the input features of entities e_i and e_j. The entities' input features in the first network layer are their original embeddings. h_{τ_m} and h_{r_m} are embeddings of the timestamp and relation in the mth inward link. Following GAT (Veličković et al., 2017), we define the normalized element $\omega_{i,j,m}$ and $\upsilon_{i,j,m}$ representing the temporal and relational connectivity from entity e_i to e_j using softmax functions,

$$\begin{aligned} \omega_{i,j,m} &= \frac{\exp(\alpha_{i,j,m})}{\sum_{e_j \in \mathcal{N}_i^e} \sum_{\tau_m \in \mathcal{L}_{ij}^\tau} \exp(\alpha_{i,j,m})}, \\ \upsilon_{i,j,m} &= \frac{\exp(\beta_{i,j,m})}{\sum_{e_j \in \mathcal{N}_i^e} \sum_{r_m \in \mathcal{L}_{ij}^r} \exp(\beta_{i,j,m})}, \end{aligned} \quad (5)$$

where $\mathcal{N}_i^e$ is the set of neighboring entities of e_i and $\mathcal{L}_{ij}^r$ and $\mathcal{L}_{ij}^\tau$ denote the sets of relations and time steps in the links from e_j to e_i.

The output features $h_{e_i}^{out}$ are obtained with an aggregate which linearly combines the temporal and relational orthogonal transformations of the input features of neighboring entities and a nonlinear ReLU activation function $\sigma(\cdot)$, i.e.,

$$\begin{aligned} h_{e_i}^{out} = \sigma\Big(\sum_{e_j \in \mathcal{N}_i^e} \sum_{[\tau_m, r_m] \in \mathcal{L}_{ij}} & \omega_{i,j,m} M_{\tau_m} h_{e_j}^{in} \\ & + \upsilon_{i,j,m} M_{r_m} h_{e_j}^{in} \Big). \end{aligned} \quad (6)$$

where $\mathcal{L}_{ij}$ denotes the set of links from e_j to e_i.

4.3 Entity Alignment Model

Entity align model aims to embed two KGs into a unified vector space by pushing the seed alignments of entities together. In this work, the entity align

Dataset	$\lvert\mathcal{E}_1\rvert$	$\lvert\mathcal{E}_2\rvert$	$\lvert\mathcal{R}_1\rvert$	$\lvert\mathcal{R}_2\rvert$	$\lvert\mathcal{T}^*\rvert$	$\lvert\mathcal{Q}_1\rvert$	$\lvert\mathcal{Q}_2\rvert$	$\lvert\mathcal{P}\rvert$	$\lvert\mathcal{S}\rvert$
ICEWS-12/2	9,517	9,537	247	246	4,017	307,552	307,553	8,566	1,000/200
YAGO-WIKI50K-10/2	49,629	49,222	11	30	245	221,050	317,814	49,172	5,000/1,000
YAGO-WIKI20K	19,493	19,929	32	130	405	83,583	142,568	19,462	400

Table 1: Statistics of original datasets (not including reverse relations and reverse links).

model consists of multiple TEA-GNN layers and a distance function which measures the similarities between final representations of entities.

Let the the l-th layer's output features of entity e_i as $h_{e_i}^{out(l)}$. A cross-layer representation is employed to capture multi-hoop neighboring information in previous work (Mao et al., 2020a) by concatenating output features of different layers. In the same way, we define the global output features $\hat{h}_{e_i}^{out}$ of e_i as

$$\hat{h}_{e_i}^{out} = [h_{e_i}^{out(0)} || h_{e_i}^{out(1)} || \cdots || h_{e_i}^{out(L)}], \quad (7)$$

where L is the number of layers and $h_{e_i}^{out(0)} = h_{e_i}^{in}$ are the input features.

We further concatenate the average embeddings of connected timestamps with output features of entities to get multi-view embeddings as final entity representations, i.e.,

$$h_{e_i}^{mul} = [\hat{h}_{e_i}^{out} || \frac{1}{|\mathcal{N}_{e_i}^{\tau}|} \sum_{\tau_m \in \mathcal{N}_{e_i}^{\tau}} h_{\tau_m}], \quad (8)$$

where $\mathcal{N}_{e_i}^{\tau}$ represents the set of timestamps around entity e_i.

Entity alignments are predicted based on the distances between the final output features of entities from two KGs. For two entities $e_i \in \mathcal{E}_1$ and $e_j \in \mathcal{E}_2$ from different sources, we use L1 distance to measure the distance between them as follows,

$$d(e_i, e_j) = ||\hat{h}_{e_i}^{out} - \hat{h}_{e_j}^{out}||_1, \quad (9)$$

A margin rank loss is used as the optimization objective of the entity align model, i.e.,

$$\mathcal{L} = \sum_{(e_i,e_j)\in\mathcal{S}} \sum_{(e_i,e_j')\in\mathcal{S}'} \sigma(d(e_i,e_j) + \lambda - d(e_i,e_j')) + \sum_{(e_i,e_j)\in\mathcal{S}} \sum_{(e_i',e_j)\in\mathcal{S}'} \sigma(d(e_i,e_j) + \lambda - d(e_i',e_j)). \quad (10)$$

where λ denotes the margin, $\mathcal{S}'$ is the set of generated negative entity pairs, $e_i' \in \mathcal{E}_1$ and $e_j' \in \mathcal{E}_2$ are the negative entities of e_i and e_j. Negative entities are sampled randomly and an RMSprop optimizer is used to minimize the loss function.

During testing, we adopt CSLS (Conneau et al., 2017) as the distance metric to measure similarities between entity embeddings.

5 Experiments

5.1 Datasets

In this work, we build five datasets from ICEWS05-15 (García-Durán et al., 2018), YAGO3 (Mahdisoltani et al., 2013) and Wikidata (Erxleben et al., 2014) to evaluate our approach.

ICEWS05-15[1] is originally extracted from ICEWS (Lautenschlager et al., 2015) which is a repository that contains political events with specific time annotations, e.g. (Barack Obama, Make a visit, Ukraine, 2014-07-08). It is noteworthy that time annotations in ICEWS are all time points. ICEWS05-15 contains events during 2005 to 2015. We build two datasets **ICEWS-12** and **ICEWS-2** in the similar way to the construction of DFB datasets (Zhu et al., 2017). We first randomly divide ICEWS05-15 quadruples into two subsets $\mathcal{Q}_1$ and $\mathcal{Q}_2$ of similar size, and make the overlap ratio of the amount of shared quadruples between $\mathcal{Q}_1$ and $\mathcal{Q}_2$ to all quadruples equal to 50%. The only difference between ICEWS-12 and ICEWS-2 is the proportion of alignment seed $\mathcal{S}$. In ICEWS-12 and ICEWS-2, about 12% and 2% of entity pairs between TKGs are pre-known. Moreover, We set the time unit of ICEWS datasets as 1 day, which means that each day is an individual time step.

YAGO3 and Wikidata are two common large-scale knowledge bases containing time information of various forms including time points, beginning or end time, and time intervals. Lacroix et al. (2020) extract a subset[2] from Wikidata in which 90% of facts are non-temporal while others have time annotations attached. We select top 50,000 entities according to their frequencies in Wikidata and link them to their equivalent YAGO entities[3] according to their QIDs and the mappings of YAGO entities to Wikidata QIDs. We generate two TKGs only involving the selected entities from the original Wikidata dataset and all YAGO facts, and then attach complementary time information to meta YAGO facts. We build two time-

[1] https://github.com/nle-ml/mmkb
[2] https://github.com/facebookresearch/tkbc
[3] http://resources.mpi-inf.mpg.de/yago-naga/yago3.1

Models	ICEWS-12			ICEWS-2			YAGO-WIKI50K-10			YAGO-WIKI50K-2		
	MRR	Hits@1	Hits@10	MRR	Hits@1	Hits@10	MRR	Hits@1	Hits@10	MRR	Hits@1	Hits@10
MTransE	.150	.101	.241	.104	.067	.175	.322	.242	.477	.033	.012	.067
JAPE	.198	.144	.298	.138	.098	.210	.345	.271	.488	.157	.101	.262
AlignE	.593	.508	.751	.303	.222	.457	.800	.756	.883	.618	.565	.714
GCN-Align	.291	.204	.466	.231	.165	.363	.581	.512	.711	.279	.217	.398
MRAEA	.745	.675	.870	.564	.476	.733	.848	.806	.913	.685	.623	.801
RREA	.780	.722	.883	.719	.659	.824	.868	.828	.938	.753	.696	.859
TU-GNN	.693	.610	.848	.610	.518	.788	.839	.795	.916	.712	.647	.834
TEA-GNN	**.911**	**.887**	**.947**	**.902**	**.876**	**.941**	**.909**	**.879**	**.961**	**.775**	**.723**	**.871**

Table 2: Entity alignment results on ICEWS and YAGO-WIKI50K datasets. The best results are written bold.

aware datasets **YAGO-WIKI50K-10** and **YAGO-WIKI50K-2** by removing non-temporal facts in the generated TKGs and using different ratios of alignment seeds $\mathcal{S}$. In addition, we build a hybrid dataset **YAGO-WIKI20K** containing both temporal and non-temporal facts with 400 pairs of alignment seeds by reducing sizes of entity sets of two TKGs to around 20,000. To generate the shared time set $\mathcal{T}^*$ for a YAGO-WIKI dataset, we drop month and date information and use the first time step τ_0 to represent unobtainable time information.

Statics of all datasets are listed in Table 1. $\mathcal{P}$ denotes the set of reference entity pairs. The set of reference entity pairs other than pre-aligned entity pairs, i.e., $\mathcal{P} - \mathcal{S}$ are used for testing.

5.2 Experimental Setup

Following the previous work, we perform entity alignment as a ranking task based on similarities between entity embeddings, and use Mean Reciprocal Rank (MRR) and Hits@N (N=1, 10) as evaluation metrics. The default configuration of our model is as follows: embedding dimension $k = 100$, learning rate $lr = 0.005$, number of TEA-GNN layers $L = 2$, margin $\gamma = 1$ and dropout rate is 0.3. Below we only list the non-default hyperparamters: $\gamma = 3$ for ICEWS-2 and YAGO-WIKI20K; $k = 25$ for YAGO-WIKI50K-10 and YAGO-WIKI50K-2. To verify the effectiveness of integration of time information, we implement a time-unaware variant of TEA-GNN which takes all time steps $\tau_i \in \mathcal{T}^*$ as unknown time information τ_0, denoted as TU-GNN. The non-default hyperparameters of TU-GNN are as follows: $\gamma = 3$ for ICEWS-12 and YAGO-WIKI20K; $\gamma = 5$ for ICEWS-2; $k = 25$ for YAGO-WIKI50K-10 and YAGO-WIKI50K-2.

In this work, we compare our proposed models with three strong translational baseline models and three state-of-the-art GNN-based models including MTransE (Chen et al., 2016), JAPE (Sun et al., 2017), AlignE (Sun et al., 2018), GCN-Align (Wang et al., 2018), MRAEA (Mao et al., 2020a) and RREA (Mao et al., 2020b). We choose AlignE instead of BootEA since we do not use iterative learning for other models including our proposed models. Due to the lack of attribute information, we use the SE (Structural Embedding) variants of JAPE and GCN-Align as baseline models. Except that the experiments of MTransE is implemented based on OpenEA framework (Sun et al., 2020), all experiments of baseline models are implemented based on their resource codes. All target models including our proposed models are trained on a GeForce GTX 1080Ti GPU. For a fair comparison, we set the maximum embedding dimension as 100 for all target models. Details of implementation and grid research for hyperparameters can be found in Appendix A.

5.3 Results and Analysis

Main Results Table 2 shows the entity alignment results of our proposed models and all baselines on ICEWS and YAGO-WIKI50K datasets. It can be shown that TEA-GNN remarkably outperforms all baseline models on four TKG datasets across all metrics. Compared to RREA which achieves the best results among than all baseline models, TEA-GNN obtains the improvement of 22.9%, 32.9%, 6.2% and 3.9% regarding Hits@1 on four TKG datasets, respectively.

Qualitative Study To study the effect of the integration of time information on the entity alignment performances of TEA-GNN, we conduct a qualitative study of TEA-GNN and its time-unaware variant TU-GNN. Table 3 lists several examples that TEA-GNN gives different predictions from TU-GNN with consideration of additional time information. In the first case, TU-GNN wrongly aligns two entities from $\mathcal{G}_1$ and $\mathcal{G}_2$ of ICEWS-2,

Entities to Be Aligned	Predictions	Similar Facts (Links) Involving Aligned Entities Betwen
Daniel Scioli (in $\mathcal{E}_1$ of ICEWS-2)	TEA-GNN: Daniel Scioli TU-GNN: Agustín Rossi	(Daniel Scioli, Accuse, Senate (Argentina), 2015-06-28), (Presidential Candidate (Argentina), Make Statement, Daniel Scioli, 2015-03-07), ... (in $\mathcal{Q}_1$ of ICEWS-2) (Agustín Rossi, Accuse, Senate (Argentina), 2009-08-26), (Presidential Candidate (Argentina), Make Statement, Agustín Rossi, 2015-04-18), ... (in $\mathcal{Q}_2$ of ICEWS-2)
Leon Benko (Q1389599) (in $\mathcal{E}_2$ of YAGO-WIKI50K-2)	TEA-GNN: <Leon_Benko> TU-GNN: <Olivier_Fontenette>	(<Olivier_Fontenette>, <playsFor>, <K.V._Kortrijk>, [2008, -]), ... (in $\mathcal{Q}_1$ of YAGO-WIKI50K-2) (Leon Benko (Q1389599), member of sports team, K.V. Kortrijk (Q618620), [2009, 2010]), ... (in $\mathcal{Q}_2$ of YAGO-WIKI50K-2)

Table 3: Examples of Different Alignment Predictions between TEA-GNN and TU-GNN.

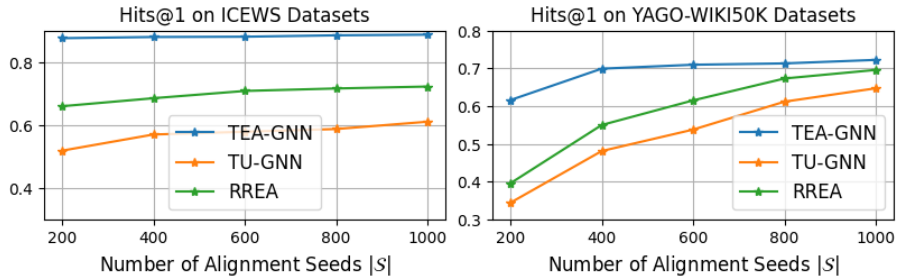

Figure 3: Hits@1 of TEA-GNN, TU-GNN and RREA on entity alignment, w.r.t. number of alignment seeds $|\mathcal{S}|$.

i.e., Daniel Scioli and Agustín Rossi, because these two entities have very similar connected links in $\mathcal{G}_1$ and $\mathcal{G}_2$ regardless of time information. As shown in Table 3, some links respective to these two entities in $\mathcal{G}_1$ and $\mathcal{G}_2$ have the same linked entities and relation types, leading to the result that TU-GNN identifies them as an equivalent entity pair. On the other hand, TEA-GNN can correctly distinguish these two entities since the relevant links have different timestamps. Similarly, TU-GNN recognizes a Wikidata entity Leon Benko (Q1389599) and a YAGO entity <Olivier_Fontenette> as the same person since these two person played for the same football club, while TEA-GNN can learn that they played for different periods and thus are not the same person in real world. These cases demonstrate the effect of time information on the performances of our proposed entity alignment models.

Sensitivity Study Numerically, compared to TU-GNN, TEA-GNN improves Hits@1 by 45.4% and 69.1% on ICEWS-12 and ICEWS-2, 10.6% and 11.6% on YAGO-WIKI50K-10 and YAGO-WIKI50K-2. It can be shown that the improvements on datasets with less alignment seeds are more significant. To futher verify this observation, we evaluate the performances of these two models and RREA on ICEWS and YAGO-WIKI50K datasets with different numbers of alignment seeds. As shown in Figure 3, the performance difference between TEA-GNN and two time-unaware models becomes greater with the decreasing of the numbers $|\mathcal{S}|$ of alignment seeds from 1000 to 200.

We also conduct a study on the prediction accuracy of aligned entities which have different time sensitivity. As mentioned in Section 5.1, we generate a hybrid dataset YAGO-WIKI20K where 17.5% of YAGO facts and 36.6% of Wikidata facts are non-temporal. We divide all testing entity pairs in this dataset into two categories based on their sensitivity to time information, i.e., **highly time-sensitive** entity pairs and **lowly time-sensitive** entity pairs. Time sensitivity s_i of a single entity e_i is defined as the ratio of the number of its time-aware connected links in which $\tau \neq \tau_0$ over the total number of all links $\mathcal{L}_i$ within its neighborhood, i.e.,

$$s_i = 1 - |\mathcal{L}_i^{\tau_0}|/|\mathcal{L}_i|, \tag{11}$$

where $\mathcal{L}_i^{\tau_0}$ denotes the set of time-unaware links connecting e_i. Given an entity pair (e_{i1}, e_{i2}) between $\mathcal{G}_1$ and $\mathcal{G}_2$, we call them as a higly time-sensitive entity pair if $s_{i1} \geqslant 0.5$ and $s_{i2} \geqslant 0.5$. Otherwise, they are lowly time-sensitive.

Among 19,062 testing entity pairs of YAGO-WIKI20K, 6,898 of them are highly time-sensitive and others are lowly time-sensitive according to the above definitions. The entity alignment results of TEA-GNN and TU-GNN on the highly time-sensitive test set and the lowly time-sensitive test set are reported in Table 4. It can be shown that TEA-GNN and TU-GNN have close performance on entity alignment for lowly time-sensitive entity

	Highly Time-Sensitive			Lowly Time-Sensitive			In Total		
	MRR	Hits@1	Hits@10	MRR	Hits@1	Hits@10	MRR	Hits@1	Hits@10
TEA-GNN	.888	853	.950	.364	.319	.449	.553	.512	.630
TU-GNN	.790	.737	.888	.366	.315	.463	.519	.468	.617

Table 4: Entity alignment results on different test sets of YAGO-WIKI20K.

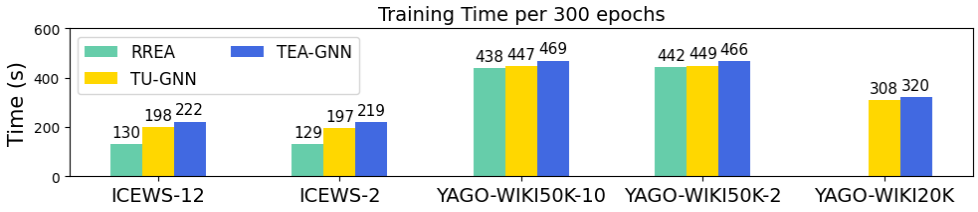

Figure 4: Training time per 300 epochs on different datasets.

pairs while TEA-GNN remarkably outperforms TU-GNN on the highly time-sensitive test set. In other words, the effect of incorporation of time information are more significant when testing entity pairs are more time-sensitive.

Complexity Study Given two TKGs to be aligned, i.e., $\mathcal{G}_1 = (\mathcal{E}_1, \mathcal{R}_1, \mathcal{T}^*, \mathcal{Q}_1)$ and $\mathcal{G}_2 = (\mathcal{E}_2, \mathcal{R}_2, \mathcal{T}^*, \mathcal{Q}_2)$, the total number of trainable parameters $|p|$ of TEA-GNN is equal to,

$$|p| = k \times (|\mathcal{E}_1| + |\mathcal{E}_2| + 2|\mathcal{R}_1| + 2|\mathcal{R}_2| + |\mathcal{T}^*|) + 3k \times L + 3k \times L, \quad (12)$$

where L denotes the number of TEA-GNN layers, the last two terms represent numbers of parameters of shared temporal and relational attention weight vectors involved in Equation 8. Compared to parameter-efficient translational entity align models like MTransE, JAPE, in which the numbers of parameters are $k \times (|\mathcal{E}_1| + |\mathcal{E}_2| + |\mathcal{R}_1| + |\mathcal{R}_2|)$, TEA-GNN uses additional parameters only for reverse relation embeddings, time embeddings and attention weight vectors, which are much fewer than parameters of entity embeddings in most cases.

As shown in Figure 4, the processing of the additional time information does not excessively increase the training time for TEA-GNN, compared to RREA and TU-GNN. Since we set the maximum number of epochs as 6000, the training processes of our proposed models on different datasets can be completed within a couple of hours on a single GeForce GTX 1080Ti GPU.

6 Conclusion

The main contributions of this paper are threefold:

- We propose a novel GNN-based approach TEA-GNN which can model temporal relational graphs with an orthogonal transformation based time-aware self-attention mechanism and perform entity alignment tasks between TKGs. To the best of our knowledge, this work is the first attempt to integrate time information into an embedding-based entity alignment approach.
- Existing temporal GNN models typically discretize a temporal graphs into multiple static snapshots and utilize a combination of GNNs and recurrent architectures. Differently, we treat timestamps as attentive properties of links between nodes. This method has been proven to be time-efficient in our case and could potentially be used for non-relational temporal graph representation learning.
- Multiple new datasets are created in this work for evaluating the performance of entity alignment models on TKGs. Experiments show that TEA-GNN remarkably outperforms the state-of-the-art entity alignment models on various well-built TKG datasets.

For future work, we will try to integrate other types of information, e.g,. attribute information, into our model and extend our model for other learning tasks of temporal graphs.

References

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Advances in Neural Information Processing Systems*, pages 2787–2795.

Yixin Cao, Zhiyuan Liu, Chengjiang Li, Juanzi Li, and Tat-Seng Chua. 2019. Multi-channel graph neural network for entity alignment. *arXiv preprint arXiv:1908.09898*.

Jinyin Chen, Xuanheng Xu, Yangyang Wu, and Haibin Zheng. 2018. Gc-lstm: Graph convolution embedded lstm for dynamic link prediction. *arXiv preprint arXiv:1812.04206*.

Muhao Chen, Yingtao Tian, Mohan Yang, and Carlo Zaniolo. 2016. Multilingual knowledge graph embeddings for cross-lingual knowledge alignment. *arXiv preprint arXiv:1611.03954*.

Alexis Conneau, Guillaume Lample, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. 2017. Word translation without parallel data. *arXiv preprint arXiv:1710.04087*.

Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. 2018. Convolutional 2d knowledge graph embeddings. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Fredo Erxleben, Michael Günther, Markus Krötzsch, Julian Mendez, and Denny Vrandečić. 2014. Introducing wikidata to the linked data web. In *International Semantic Web Conference*, pages 50–65. Springer.

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

Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

Timothée Lacroix, Guillaume Obozinski, and Nicolas Usunier. 2020. Tensor decompositions for temporal knowledge base completion. *arXiv preprint arXiv:2004.04926*.

Jennifer Lautenschlager, Steve Shellman, and Michael Ward. 2015. Icews event aggregations.

Julien Leblay and Melisachew Wudage Chekol. 2018. Deriving validity time in knowledge graph. In *Companion of the The Web Conference 2018 on The Web Conference 2018*, pages 1771–1776. International World Wide Web Conferences Steering Committee.

Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick Van Kleef, Sören Auer, et al. 2015. Dbpedia–a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic web*, 6(2):167–195.

Farzaneh Mahdisoltani, Joanna Biega, and Fabian M Suchanek. 2013. Yago3: A knowledge base from multilingual wikipedias. In *CIDR*.

Franco Manessi, Alessandro Rozza, and Mario Manzo. 2020. Dynamic graph convolutional networks. *Pattern Recognition*, 97:107000.

Xin Mao, Wenting Wang, Huimin Xu, Man Lan, and Yuanbin Wu. 2020a. Mraea: an efficient and robust entity alignment approach for cross-lingual knowledge graph. In *Proceedings of the 13th International Conference on Web Search and Data Mining*, pages 420–428.

Xin Mao, Wenting Wang, Huimin Xu, Yuanbin Wu, and Man Lan. 2020b. Relational reflection entity alignment. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, pages 1095–1104.

Mojtaba Nayyeri, Chengjin Xu, Sahar Vahdati, Nadezhda Vassilyeva, Emanuel Sallinger, Hamed Shariat Yazdi, and Jens Lehmann. 2020. Fantastic knowledge graph embeddings and how to find the right space for them. In *International Semantic Web Conference*, pages 438–455. Springer.

Mojtaba Nayyeri, Chengjin Xu, Yadollah Yaghoobzadeh, Hamed Shariat Yazdi, and Jens Lehmann. 2019. Toward understanding the effect of loss function on the performance of knowledge graph embedding.

Aldo Pareja, Giacomo Domeniconi, Jie Chen, Tengfei Ma, Toyotaro Suzumura, Hiroki Kanezashi, Tim Kaler, Tao Schardl, and Charles Leiserson. 2020. Evolvegcn: Evolving graph convolutional networks for dynamic graphs. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 5363–5370.

Shichao Pei, Lu Yu, and Xiangliang Zhang. 2019. Improving cross-lingual entity alignment via optimal transport. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pages 3231–3237. International Joint Conferences on Artificial Intelligence Organization.

Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *European semantic web conference*, pages 593–607. Springer.

Samuel L Smith, David HP Turban, Steven Hamblin, and Nils Y Hammerla. 2017. Offline bilingual word vectors, orthogonal transformations and the inverted softmax. *arXiv preprint arXiv:1702.03859*.

Fabian M Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. Yago: a core of semantic knowledge. In *Proceedings of the 16th international conference on World Wide Web*, pages 697–706. ACM.

Zequn Sun, Wei Hu, and Chengkai Li. 2017. Cross-lingual entity alignment via joint attribute-preserving embedding. In *International Semantic Web Conference*, pages 628–644. Springer.

Zequn Sun, Wei Hu, Qingheng Zhang, and Yuzhong Qu. 2018. Bootstrapping entity alignment with knowledge graph embedding. In *IJCAI*, volume 18, pages 4396–4402.

Zequn Sun, Qingheng Zhang, Wei Hu, Chengming Wang, Muhao Chen, Farahnaz Akrami, and Chengkai Li. 2020. A benchmarking study of embedding-based entity alignment for knowledge graphs. *Proceedings of the VLDB Endowment*, 13(11):2326–2340.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2017. Graph attention networks. *arXiv preprint arXiv:1710.10903*.

Quan Wang, Zhendong Mao, Bin Wang, and Li Guo. 2017. Knowledge graph embedding: A survey of approaches and applications. *IEEE Transactions on Knowledge and Data Engineering*, 29(12):2724–2743.

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

Zhichun Wang, Qingsong Lv, Xiaohan Lan, and Yu Zhang. 2018. Cross-lingual knowledge graph alignment via graph convolutional networks. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 349–357.

Jiapeng Wu, Meng Cao, Jackie Chi Kit Cheung, and William L Hamilton. 2020. Temp: Temporal message passing for temporal knowledge graph completion. *arXiv preprint arXiv:2010.03526*.

Yuting Wu, Xiao Liu, Yansong Feng, Zheng Wang, and Dongyan Zhao. 2019a. Jointly learning entity and relation representations for entity alignment. *arXiv preprint arXiv:1909.09317*.

Yuting Wu, Xiao Liu, Yansong Feng, Zheng Wang, and Dongyan Zhao. 2019b. Jointly learning entity and relation representations for entity alignment. *arXiv preprint arXiv:1909.09317*.

Chengjin Xu, Mojtaba Nayyeri, Fouad Alkhoury, Jens Lehmann, and Hamed Shariat Yazdi. 2019. Temporal knowledge graph embedding model based on additive time series decomposition. *arXiv preprint arXiv:1911.07893*.

Chengjin Xu, Mojtaba Nayyeri, Fouad Alkhoury, Hamed Shariat Yazdi, and Jens Lehmann. 2020a. Tero: A time-aware knowledge graph embedding via temporal rotation. *arXiv preprint arXiv:2010.01029*.

Chengjin Xu, Mojtaba Nayyeri, Yung-Yu Chen, and Jens Lehmann. 2020b. Knowledge graph embeddings in geometric algebras. *arXiv preprint arXiv:2010.00989*.

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

Hao Zhu, Ruobing Xie, Zhiyuan Liu, and Maosong Sun. 2017. Iterative entity alignment via joint knowledge embeddings. In *IJCAI*, volume 17, pages 4258–4264.

A Implementation Details

As mentioned in Section 5.2, we use the source codes [4][5][6][7][8] respective to baseline models for evaluation, except that we evaluate MTransE based on the implementation of OpenEA framework [9].

For all baseline models, we mostly follow their default optimal configurations regarding learning rates lr, batch sizes b, negative sampling rates η, dropout rates dr, numbers of GNN layers L and mainly focus on the grid research of embedding dimensions k and margins γ (negative weights α for JAPE). We also follow the respective original papers to fix the numbers of multi-head attention mechanisms as 2 for MRAEA and set the balance weight $\beta = 0.9$ for GCN-Align. For all baseline models and our proposed models, we tune k in the range of $(25, 50, 75, 100)$, and γ or α in the range of $(0, 0.5, 1, 2, 3, 5, 7, 10, 15, 20)$. Specially, we use the same margin hyperparameters as the original paper for AlignE. To make a fair comparison, we use the same setup for our proposed model as MRAEA and RREA to fix $L = 2$, $dr = 0.3$, $b = |\mathcal{E}_1| + |\mathcal{E}_2|$ and $\eta = b//|\mathcal{S}| + 1$ where $//$ denotes the round-down after division, and also conduct the same grid research of embedding dimensions k and margins γ for TEA-GNN and TU-GNN as what we do for baseline models.

All hyperparameters used for getting the results reported in Table 2 and 4 are listed in Table 5, 6, 7, 8 and 9. In our experiments, for MRAEA, RREA and our proposed models, the usage of the higher-dimensional embeddings would lead to out-of-memory problems since we only use a single mid-range GPU device for training and the learning of large-scale graph neural networks with numerous nodes and high-dimensional embeddings needs excessive memory footprint during the training process. It is predictable that we can possibly further boost the performances of TU-GNN and TEA-GNN on YAGOWIKI50K datasets by increasing their embedding dimensions with more high-performance GPU devices.

The source codes and datasets used in this work are submitted as the supplementary materials for reproducibility and will be released on Github after the anonymity period.

[4]https://github.com/nju-websoft/JAPE
[5]https://github.com/nju-websoft/BootEA
[6]https://github.com/1049451037/GCN-Align/
[7]https://github.com/MaoXinn/MRAEA
[8]https://github.com/MaoXinn/RREA
[9]https://github.com/nju-websoft/OpenEA/

Models	k	lr	b	η	γ (or α)	dr
MTransE	100	0.01	20,000	10	10	-
JAPE	100	0.01	10,000	1	2	-
AlignE	100	0.01	20,000	10	0.01,2,0.7	-
GCN-Align	100	20	-	5	3	0
MRAEA	100	0.001	19,054	20	1	0.3
RREA	100	0.005	19,054	20	3	0.3
TU-GNN	100	0.005	19,054	20	3	0.3
TEA-GNN	100	0.005	19,054	20	1	0.3

Table 5: Hyperparameters of target models for ICEWS-12.

Models	k	lr	b	η	γ (or α)	dr
MTransE	100	0.01	20,000	10	7	-
JAPE	100	0.01	10,000	1	3	-
AlignE	100	0.01	20,000	10	0.01,2,0.7	-
GCN-Align	100	20	-	5	7	0
MRAEA	100	0.001	19,054	96	2	0.3
RREA	100	0.005	19,054	96	2	0.3
TU-GNN	100	0.005	19,054	96	5	0.3
TEA-GNN	100	0.005	19,054	96	3	0.3

Table 6: Hyperparameters of target models for ICEWS-2.

Models	k	lr	b	η	γ (or α)	dr
MTransE	100	0.01	20,000	10	10	-
JAPE	100	0.01	10,000	1	3	-
AlignE	100	0.01	20,000	10	0.01,2,0.7	-
GCN-Align	100	20	-	5	3	0
MRAEA	75	0.001	98,851	20	1	0.3
RREA	50	0.005	98,851	20	1	0.3
TU-GNN	25	0.005	98,851	20	1	0.3
TEA-GNN	25	0.005	98,851	20	1	0.3

Table 7: Hyperparameters of target models for YAGO-WIKI50K-12.

Models	k	lr	b	η	γ (or α)	dr
MTransE	100	0.01	20,000	10	1	-
JAPE	100	0.01	10,000	1	2	-
AlignE	100	0.01	20,000	10	0.01,2,0.7	-
GCN-Align	100	20	-	5	10	0
MRAEA	75	0.001	98,851	99	2	0.3
RREA	50	0.005	98,851	99	1	0.3
TU-GNN	25	0.005	98,851	99	1	0.3
TEA-GNN	25	0.005	98,851	99	1	0.3

Table 8: Hyperparameters of target models for YAGO-WIKI50K-2.

Models	k	lr	b	η	γ (or α)	dr
TU-GNN	100	0.005	39,422	99	3	0.3
TEA-GNN	100	0.005	39,422	99	3	0.3

Table 9: Hyperparameters of target models for YAGO-WIKI20K.