Time-aware Entity Alignment using Temporal Relational Attention

Chengjin Xu
University of Bonn
Bonn, Germany
xuc@iai.uni-bonn.de

Fenglong Su
National University of Defense Technology
Changsha, China

Bo Xiong
University of Stuttgart
Stuttgart, Germany

Jens Lehmann
University of Bonn
Fraunhofer IAIS
Bonn, Germany

ABSTRACT

Knowledge graph (KG) alignment is to match entities in different KGs, which is important to knowledge fusion and integration. Temporal KGs (TKGs) extend traditional Knowledge Graphs (KGs) by associating static triples with specific timestamps (e.g., temporal scopes or time points). Moreover, open-world KGs (OKGs) are dynamic with new emerging entities and timestamps. While entity alignment (EA) between KGs has drawn increasing attention from the research community, EA between TKGs and OKGs still remains unexplored. In this work, we propose a novel Temporal Relational Entity Alignment method (TREA) which is able to learn alignment-oriented TKG embeddings and represent new emerging entities. We first map entities, relations and timestamps into an embedding space, and the initial feature of each entity is represented by fusing the embeddings of its connected relations and timestamps as well as its neighboring entities. A graph neural network (GNN) is employed to capture intra-graph information and a temporal relational attention mechanism is utilized to integrate relation and time features of links between nodes. Finally, a margin-based full multi-class log-loss is used for efficient training and a sequential time regularizer is used to model unobserved timestamps. We use three well-established TKG datasets, as references for evaluating temporal and non-temporal EA methods. Experimental results show that our method outperforms the state-of-the-art EA methods.

CCS CONCEPTS

• **Computing methodologies** → **Knowledge representation and reasoning**; **Temporal reasoning**.

KEYWORDS

Graph Attention Networks, Temporal Knowledge Graph, Entity Alignment

WWW '22, April 25–29, 2022, Virtual Event, Lyon, France.

ACM ISBN 978-1-4503-9096-5/22/04…$15.00
https://doi.org/10.1145/3485447.3511922

ACM Reference Format:
Chengjin Xu, Fenglong Su, Bo Xiong, and Jens Lehmann. 2022. Time-aware Entity Alignment using Temporal Relational Attention. In *Proceedings of the ACM Web Conference 2022 (WWW '22), April 25–29, 2022, Virtual Event, Lyon, France.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3485447.3511922

1 INTRODUCTION

Knowledge Graph (KG) is a knowledge base that uses a graph-structured data model or topology to integrate data. The recent growth of KG, such as the Google knowledge graph, YAGO [27], and DBpedia [16], has triggered success in various applications to deal with knowledge-intensive downstream like search engines and recommendation systems [9]. A typical KG is composed of entities as nodes and relations as edges and can be represented as a collection of triples (e_s, r, e_o), indicating a specific relationship between the subject entity and the object entity, e.g., (*Boris Johnson*, *PrimeMinisterOf*, *United Kingdom*).

However, most KGs are independently extracted from separate data sources, focusing on different domains or languages. Therefore, using a single KG is oftentimes insufficient to the need of downstream applications and it is essential to fuse heterogeneous knowledge among different KGs where same entities exist in different surface forms. Entity alignment (EA) in different KGs play an important role for KG fusion. To address this problem, a lot of embedding-based EA methods [5, 21, 22, 30, 36] have been proposed to embed entities into low-dimensional vector spaces, and obtain the pairs of equivalent entities by computing the pair-wise distance between their vector representations.

Recently, studies of temporal dynamics in knowledge graphs have intrigued a lot of interest for researchers. Temporal KGs like Wikidata [7], YAGO3 [19] and ICEWS [15] incorporate time information into triples, which means some links between entities have two properties, i.e., relation and time, as shown in Figure 1. Triples attached with time information are represented as quadruples, shaped like (e_s, r, e_o, t), where t denotes the timestamp and can be represented in various forms, e.g., time points, start/end time, or time intervals. Noteworthily, timestamps in most TKGs are presented using Arabic numerals and similar formats. Thus, timestamps expressing the same temporal information across different TKGs can be easily aligned by manually unifying their formats.

However, most of the existing embedding-based EA methods do not consider time information in KGs, which might lead to the

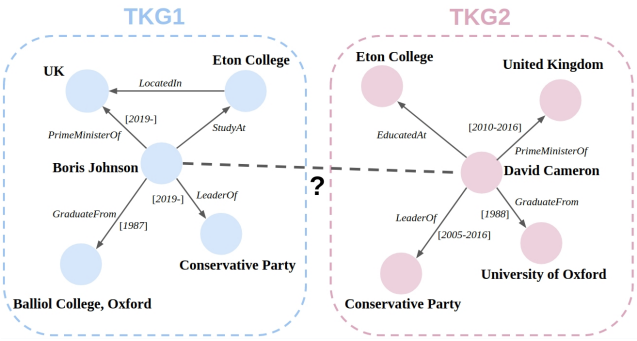

Figure 1: An illustrative example of limitations of the existing time-unaware entity align methods.

misalignment between similar entities. As in the case of Figure 1 in which the left and right subgraphs are extracted from two separate TKG respectively, time-unaware EA methods are likely to recognize *Boris Johnson* and *David Cameron* as an entity pair to be aligned due to the similarity of their neighboring nodes and connected relations. On the other hand, these two entities can be easily distinguished by considering the timestamps of links within their neighborhood. Moreover, the existing EA methods can not model new emerging entities and timestamps, which is important for learning and inferencing on open-world KGs (OKGs).

To address the above challenges, we introduce a novel Temporal Relational Entity Alignment method, TREA, for EA between TKGs and OKGs. TREA maps entities, relations and timestamps in TKGs into an embedding space, and the the initial feature of each entity is represented by fusing the embeddings of its connected relations and timestamps as well as its neighboring entities so that a new emerging entity can also be represented. We employs a graph neural network (GNN) to learn the semantics of entities and capture structural information, and a temporal relational attention mechanism is used to incorporate time and relation information of links into the GNN by assigning respective weights to different nodes within a neighborhood according to embeddings of the corresponding timestamps and relation. Then, a margin-based full multi-class log-loss is used for efficient training and a sequential time regularizer is used to model unobserved timestamps. At last, entities are aligned by computing the distances between their multi-aspect representations.

In summary, the main contributions of this paper are as follows:

- We propose a GNN-based embedding method to entity alignment between TKGs, which can model temporal relational graphs with an efficient time-aware attention mechanism.
- The existing EA methods hold a closed-world assumption, i.e., where KGs are fixed, and entities or timestamps cannot be easily added. In this work, we relax this assumption and propose a new open-world EA task where new entities and timestamps are emerging in testing phase.
- A neighborhood aggregation representation and a sequential time regularizer are proposed, which can model representations of new emerging entities and timestamps, respectively. Additionally, a margin-based full multi-class log-loss function is used for fast training.
- The experimental results on three well-built TKG benchmarks show that our method achieves the state-of-the-art on temporal EA and open-world EA.

2 RELATED WORK

2.1 TKG and OKG Embedding

With the development of TKGs, recent researches have shown interest in extending KGE models to TKGs. Typically, such works improve performances of some existing KGE methods on time-aware link prediction by mapping timestamps into the embedding space and defining a time-aware score function which involves time embeddings [8, 13, 41–43]. Another line of TKGE works including RENET [11] and TeMP [37] combine R-GCN and temporal recurrent networks (e.g., RNN, GRU or temporal transformers) as encoders, and use shallow KGE models (e.g., ComplEx [33]) as decoders. Some TKGE models [10, 11] can predict future link. However, the closed-world assumption is partly kept since the entity set of a KG is consistent. On the other hand, some existing OKGE [26, 35] models consider new emerging entities but ignore time information. Noteworthily, the above TKGE and OKGE methods are not directly compatible with the EA setting.

2.2 Temporal Graph Neural Network

Graph Neural Network (GNN) is famous for its strong modeling capability on the non-Euclidean structure and thus has become one of the hottest topics in graph learning these years. GNN models, e.g., GCN [12] and GAT [34] have seen a series of recent successes in problems related to graph-structured data. Since many real-world graphs like social networks and recommendation systems are dynamic or temporal, it is essential to carefully consider time information for learning good representations on such graphs. Most of the existing temporal GNN models [2, 17, 20, 25, 45] generally discretize a temporal graph into multiple static snapshot in a timeline and use combinations of GNN models and recurrent models whereby the former handle graph information and the latter capture dynamism. Such combination architectures often need long training time.

2.3 Entity Alignment

Entity alignment (EA) is to find equivalent entities between multiple KGs. A lot of embedding-based EA methods have been proposed to embed KGs into a vector space and align entities by measure the similarities between their embeddings. Some embedding-based EA methods [5, 29, 30, 49] exploit TransE to encode entities and relations into dense embeddings and learn structure information from separate KGs, and use an alignment function to map the embeddings of entities in different KGs into a unified vector space via pre-aligned entities.

More recent research works introduce GNNs into EA 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 methods are proposed to incorporate relation information into GCNs. A part of GCN-based EA methods [1, 22, 23, 28, 46] assign different weight coefficients to entities

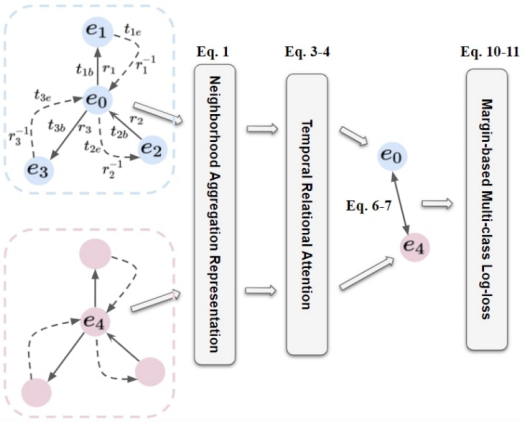

Figure 2: Framework of our method. Dashed arrows represent the created inverse links.

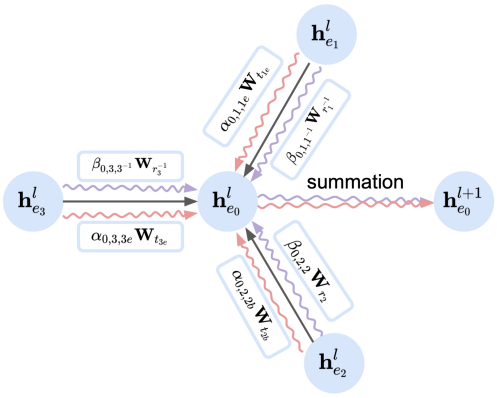

Figure 3: An illustration of temporal relation attention by entity e_0 on its neighborhood. Red and purple arrows denote computations of time and relation attention, respectively.

according to relation types between them, which empowers the models to distinguish the importance between different entities. Another part of GNN-based EA methods incorporate auxiliary information in addition to structure information in KGs. Some of them [38–40, 47] use pre-trained entity name vectors, and meanwhile others [18, 24] utilize an attributed value encoder to learn attribute information. A very recent work [44] show that the incorporation of time information can improve the performances of GNN-based EA methods. So far, there is no existing EA model which can be directly compatible with both TKGs and OKGs.

3 PROBLEM FORMULATION

A TKG is a directed relational temporal graph $\mathcal{G} = (\mathcal{E}, \mathcal{R}, \mathcal{T}, \mathcal{Q})$ comprising four sets: entities $\mathcal{E}$, relations $\mathcal{R}$, timestamps $\mathcal{T}$, and quadruples $\mathcal{Q} \subseteq \mathcal{E} \times \mathcal{R} \times \mathcal{E} \times \mathcal{T}$. A TKG stores the real-world information in the form of quadruples (e_s, r, e_o, t), where $e_s, e_o \in \mathcal{E}$. Given two TKGs $\mathcal{G}_1 = (\mathcal{E}_1, \mathcal{R}_1, \mathcal{T}_1, \mathcal{Q}_1)$, $\mathcal{G}_2 = (\mathcal{E}_2, \mathcal{R}_2, \mathcal{T}_2, \mathcal{Q}_2)$, and a pre-aligned entity pair set $\mathcal{S} = \{(e_i, e_j)|e_i \in \mathcal{E}_1, e_j \in \mathcal{E}_2, e_i \equiv e_j\}$ as alignment seeds where $\equiv$ denotes equivalence. The task of EA between TKGs aims to obtain more potential equivalent entity pairs.

The task of EA between OKGs aims to predict entity pairs involving unseen entities or those entities associated with future time stamps. In another word, a TKG $\mathcal{G}$ is updated as $(\mathcal{E} \bigcup \mathcal{E}', \mathcal{R}, \mathcal{T} \bigcup \mathcal{T}', \mathcal{Q} \bigcup \mathcal{Q}')$ where $\mathcal{E}'$, $\mathcal{T}'$ and $\mathcal{Q}'$ are sets of unseen entities, timestamps and quadruples emerging at the inference phase.

4 THE PROPOSED METHOD

As mentioned in the introduction, a lot of recent knowledge bases involves temporal facts and real-world knowledge bases are often dynamic with new emerging entities and timestamps. However, most of the existing embedding-based EA methods disregard time information and is incapable to model new emerging entities and timestamps. To address these defects, we propose a novel *Temporal Relational Entity Alignment* method, **TREA**. Figure 2 depicts that TREA consists of three major parts: (i) *Neighborhood Aggregation Representation* (NAR); (ii) *Temporal Relational Attention* (TRA); (iii) *Margin-based Multi-class Log-loss* (MML). In this section, we elaborate on the above three parts of TREA.

4.1 Neighborhood Aggregation Representation

Timestamps in TKGs can be represented in different forms, i.e., time points, start/end time, and time intervals and some TKGs involve non-temporal facts. Following previous work [44], we use a time range ($[t_b, t_e]$) where t_b denotes the begin time and t_e denotes the end time to represent each timestamp. Specifically, we have $t_b = t_e$ for a time point and $t_e = t_0$ or $t_b = t_0$ for a start time or end time where $t_0 \in \mathcal{T}$ denotes the first time step and indicates that the time data is unobtainable. We create an inverse link for each link to accumulate the direction information. Each inverse link involves a reciprocal relation r^{-1} corresponding to the relation r of the original link. Thus, the relation set $\mathcal{R}$ is extended to have its inverse. And the begin time t_b and end time t_e of each timestamp are separately attached to the original link and the inverse link. Thus, each fact $(e_s, r, e_o, [t_b, t_e])$ can be decomposed into two quadruples (e_s, r, e_o, t_b) and (e_s, r^{-1}, e_o, t_e).

We map all of entities, relations (including reverse relations) and time steps in TKGs into a vector space $\mathbb{R}^k$ where k denotes the dimension of the vector space. Embeddings of the entity e_i, relation r_m, time step t_n are denoted as $h_{e_i}, h_{r_m}, h_{t_n} \in \mathbb{R}^k$, respectively. To enforce neighborhood information into the entity representation, we average the embeddings of each embedding and its neighboring entities and then concatenate the average entity embedding with the features of the inward links in the entity's neighborhood. The complete neighborhood aggregation representation $h^0_{e_i}$ of an entity e_i can be formulated as,

$$\mathbf{h}^0_{e_i} = \left[\frac{1}{|\mathcal{N}^e_i|+1} \sum_{e_j \in \mathcal{N}^e_i \cup e_i} \mathbf{h}_{e_j} \,||\, \frac{1}{|\mathcal{N}^r_i|} \sum_{r_j \in \mathcal{N}^r_i} \mathbf{h}_{r_j} \,||\, \frac{1}{|\mathcal{N}^\tau_i|} \sum_{\tau_j \in \mathcal{N}^\tau_i} \mathbf{h}_{\tau_j} \right] \quad (1)$$

where $\mathcal{N}^e_i$ is the set of neighboring entities of e_i, $\mathcal{N}^r_i$ and $\mathcal{N}^\tau_i$ are sets of relations and time steps which connect inwardly to e_i. $||$ denotes the concatenation operator. For a new emerging entity e_i without an entity embedding h_{e_i}, we only consider its neighboring entities $e_j \in \mathcal{N}^e_i$ which are observed in the original KG. By using

neighborhood aggregation representation, we can represent both observed entities and new emerging entity.

4.2 Temporal Relational Attention

Graph Attention Network (GAT) extends the vanilla Graph Convolutional Network (GCN) by employing a self-attention mechanism to calculate the hidden representations of each node by attending over its neighbors. A GAT layer can be defined as follow,

$$\mathbf{h}_{e_i}^{l+1} = \sigma\Big(\sum_{e_j \in \mathcal{N}_{e_i}^e \cup e_i} \alpha_{i,j} \mathbf{W} \mathbf{h}_{e_j}^{l} \Big), \tag{2}$$

where $\sigma(\cdot)$ denotes the nonlinear activation function, $\mathbf{W}$ denotes the shared transformation matrix, and $\alpha_{i,j}$ denotes the attention coefficient of e_j to e_i. To develop a temporal relational attention mechanism, we define new time-specific attention coefficient $\alpha_{i,j,n}$, relation-specific attention coefficient $\beta_{i,j,m}$, diagonal temporal transformation matrix $\mathbf{W}_t$ and diagonal relational transformation matrix $\mathbf{W}_r$. The attention coefficients are computed as,

$$\begin{aligned} \alpha_{i,j,n} &= \frac{\exp(\boldsymbol{v}_t^T \mathbf{h}_{t_n})}{\sum_{e_j \in \mathcal{N}_i^e} \sum_{[t_{n'}, r_{m'}] \in \mathcal{L}_{ij}} \exp(\boldsymbol{v}_t^T \mathbf{h}_{t_{n'}})}, \\ \beta_{i,j,m} &= \frac{\exp(\boldsymbol{v}_r^T \mathbf{h}_{r_m})}{\sum_{e_j \in \mathcal{N}_i^e} \sum_{[t_{n'}, r_{m'}] \in \mathcal{L}_{ij}} \exp(\boldsymbol{v}_r^T \mathbf{h}_{r_{m'}})}, \end{aligned} \tag{3}$$

where $\boldsymbol{v}_t, \boldsymbol{v}_r \in \mathbb{R}^k$ denote shared temporal and relational attention weight vectors, $\mathbf{h}_{t_n} \in \mathbb{R}^k$ denotes the embedding of the time step t_n, $\mathbf{h}_{r_m} \in \mathbb{R}^k$ denotes the embedding of the relation r_m, $\mathcal{L}_{ij}$ denotes the set of inward links from e_j to e_i and $[t_{n'}, r_{m'}] \in \mathcal{L}_{ij}$ indicates the presence of an observed quadruple $(e_j, r_{m'}, e_i, t_{n'})$.

As shown in Figure 3, the entity feature update equation can be renewed by substituting Eq 3 into Eq 2,

$$\mathbf{h}_{e_i}^{l+1} = tanh\Big(\sum_{e_j \in \mathcal{N}_i^e} \sum_{[t_n, r_m] \in \mathcal{L}_{ij}} (\alpha_{i,j,n} \mathbf{W}_t + \beta_{i,j,m} \mathbf{W}_r) \mathbf{h}_{e_j}^{l} \Big). \tag{4}$$

A cross-layer representation is employed to capture multi-hoop neighboring information by stacking entity features from different layers. We define a global output features $\mathbf{h}_{e_i}^{out}$ of e_i as,

$$\mathbf{h}_{e_i}^{out} = [\mathbf{h}_{e_i}^0 || \mathbf{h}_{e_i}^1 || \cdots || \mathbf{h}_{e_i}^L], \tag{5}$$

where L denotes the number of attention layers.

To obtain multi-aspect representations for entities, we concatenate entity output features generated from GNN with averages of embeddings of their neighboring relations and timestamps. The multi-view entity representation of e_i is defined as follows,

$$\mathbf{h}_{e_i}^{mul} = \Big[\mathbf{h}_{e_i}^{out} || \frac{1}{|\mathcal{N}_i^r|} \sum_{r_m \in \mathcal{N}_i^r} \mathbf{h}_{r_m} || \frac{1}{|\mathcal{N}_i^t|} \sum_{t_n \in \mathcal{N}_i^t} \mathbf{h}_{t_n} \Big]. \tag{6}$$

4.3 Margin-based Multi-class Log-loss

The optimization objective of an embedding-based EA model is to enforce that entities of each alignment seed have close representations. During training, we use L2 distance as the metric to define the difference of representations of two entities e_i and e_j as follows,

$$d(e_i, e_j) = ||\mathbf{h}_{e_i}^{mul} - \mathbf{h}_{e_j}^{mul}||_2^2. \tag{7}$$

Most of existing embedding-based EA methods employ a pairwise margin ranking loss (MRL) function to minimize the distances between training entity pairs as follows,

$$\mathcal{L} = \sum_{(e_i, e_j) \in \mathcal{S}} [\gamma + d(e_i, e_j) - d(e_i', e_j')]_+, \tag{8}$$

where $\mathcal{S}$ denote the set of alignment seeds, γ is a fixed margin, $[x]_+$ represents the operation $Max(x, 0)$, e_i' and e_j' denote the negative samples corresponding to e_i and e_j. Since the negative samples are randomly selected and the margin ranking loss function treats each negative sample equally, the whole training process might be influenced by easy negative samples which are low-quality and uninformative, and thus suffer from slow convergence.

Lacroix et al. [14] employs a full negative sampling strategy instead of random negative sampling for training link prediction model to achieve fast convergence. And a multi-class logistic loss function is used to ensure that the optimization objective mainly focus on hard negative samples,

$$\mathcal{L} = \sum_{i \in \mathcal{S}^P} \Big[-s(i) + \log \sum_{j \in \mathcal{S}_i^N \cup i} \exp(s(j)) \Big], \tag{9}$$

where S^P denote the sets of positive samples, $\mathcal{S}_i^N$ denotes negative samples corresponding to the positive sample i, $s(i)$ represents the score of the sample i.

In this work, we also adopt full negative sampling for fast convergence and a LogSumExp operation is used to find hard negative samples [21]. The margin-based logistic loss function for entity pairs can be defined as follows,

$$\begin{aligned} \mathcal{L}_p = &\sum_{(e_i, e_j) \in \mathcal{S}} \log\Big[1 + \sum_{e_j' \in \mathcal{E}_2} \exp(\gamma + d(e_i, e_j) - d(e_i, e_j')) \Big] \\ &+ \sum_{(e_i, e_j) \in \mathcal{S}} \log\Big[1 + \sum_{e_i' \in \mathcal{E}_1} \exp(\gamma + d(e_i, e_j) - d(e_i', e_j)) \Big], \end{aligned} \tag{10}$$

To retain the distance and sequence information between different timestamps, we use a sequential time regularizer for time embeddings in addition to the loss function for entity pairs, based on the assumption that embeddings of two distant time steps are relatively more different than those of two adjacent time steps. The complete loss function used for our model is defined as,

$$\mathcal{L} = \mathcal{L}_p + \lambda_t \sum_{i=1}^{|\mathcal{T}|-1} ||\mathbf{h}_{t_{i+1}} - \mathbf{h}_{t_i} - \mathbf{h}_{t_b}||_2^2, \tag{11}$$

where $\mathcal{T} = \mathcal{T}_1 \cup \mathcal{T}_2$, λ_t denotes the time regularization weight, $\mathbf{h}_{t_b}$ is the embedding of the temporal slope.

Following the previous EA work [22, 23, 44], we adopt Cross-domain Similarity Local Scaling (CSLS) [6] as the distance metric during testing to measure similarities between entity embeddings.

4.4 Trainable Parameters

We compare the theoretical value of the amount of trainable parameters of TREA with several existing EA models. As shown in Table 1, compared to parameter-efficient translational entity align models like MTransE, TREA uses additional parameters only for reciprocal relation embeddings, time embeddings and attention weight vectors, which are much fewer than parameters of entity

embeddings in most cases. We further show the actual values of the amount of trainable parameters of these EA models on different datasets in Section 5.7.

EA Methods	Number of Trainable Parameters
MTransE	$d(\lvert\mathcal{E}_1\rvert + \lvert\mathcal{E}_2\rvert + \lvert\mathcal{R}_1\rvert + \lvert\mathcal{R}_2\rvert)$
JAPE	$d(\lvert\mathcal{E}_1\rvert + \lvert\mathcal{E}_2\rvert + \lvert\mathcal{R}_1\rvert + \lvert\mathcal{R}_2\rvert)$
BootEA	$d(\lvert\mathcal{E}_1\rvert + \lvert\mathcal{E}_2\rvert + \lvert\mathcal{R}_1\rvert + \lvert\mathcal{R}_2\rvert)$
GCN-Align	$d(\lvert\mathcal{E}_1\rvert + \lvert\mathcal{E}_2\rvert) + 2d^2$
MuGNN	$d(\lvert\mathcal{E}_1\rvert + \lvert\mathcal{E}_2\rvert + \lvert\mathcal{R}_1\rvert + \lvert\mathcal{R}_2\rvert) + 2d + d^2$
MRAEA	$d(\lvert\mathcal{E}_1\rvert + \lvert\mathcal{E}_2\rvert + 2\lvert\mathcal{R}_1\rvert + 2\lvert\mathcal{R}_2\rvert) + 3dML$
HyperKA	$d(\lvert\mathcal{E}_1\rvert + \lvert\mathcal{E}_2\rvert + \lvert\mathcal{R}_1\rvert + \lvert\mathcal{R}_2\rvert) + d^2L$
RREA	$d(\lvert\mathcal{E}_1\rvert + \lvert\mathcal{E}_2\rvert + 2\lvert\mathcal{R}_1\rvert + 2\lvert\mathcal{R}_2\rvert) + 3dL$
KE-GCN	$d(\lvert\mathcal{E}_1\rvert + \lvert\mathcal{E}_2\rvert + \lvert\mathcal{R}_1\rvert + \lvert\mathcal{R}_2\rvert) + d^2L(\lvert\mathcal{R}_1\rvert + \lvert\mathcal{R}_2\rvert + 2)$
TEA-GNN	$d(\lvert\mathcal{E}_1\rvert + \lvert\mathcal{E}_2\rvert + 2\lvert\mathcal{R}_1\rvert + 2\lvert\mathcal{R}_2\rvert + \lvert\mathcal{T}\rvert) + 6dL$
TREA	$d(\lvert\mathcal{E}_1\rvert + \lvert\mathcal{E}_2\rvert + 2\lvert\mathcal{R}_1\rvert + 2\lvert\mathcal{R}_2\rvert + \lvert\mathcal{T}\rvert) + 8dL$

Table 1: Comparison of numbers of trainable parameters between our EA method and several popular EA methods. Note that M denotes the number of attention heads of MRAEA.

5 EXPERIMENT

5.1 Temporal Datasets for Entity Alignment

In this paper, we use three TKG datasets extracted from ICEWS [15], YAGO [27], Wikidata [7] as references for evaluating temporal and non-temporal EA methods, i.e., **DICEWS**, **YAGO-WIKI50K** and **YAGO-WIKI20K** [44]. The statics of the three temporal EA datasets are listed in Table 2.

ICEW05-15 [8] is a subset of ICEWS facts occurring during 2005 to 2015, and is commonly used as a TKG benchmark dataset in the community. It is noteworthy that time annotations in ICEWS are all time points, e.g., (*Barack Obama*, *Visit*, *Ukraine*, *2014-07-08*). **DICEWS** is built from ICEWS05-15 in the similar way to the construction of DFB datasets [49]. Between $\mathcal{Q}_1$ and $\mathcal{Q}_2$ of DICEWS, the overlap ratio of their shared quadruples are 50%.

YAGO-WIKI50K and **YAGO-WIKI20K** are extracted from Wikidata and YAGO, between which there are a large number of identical entities represented in different surface forms. The first dataset contains about 50,000 entity pairs, and each fact in the dataset is temporal. Meanwhile, YAGO-WIKI20K is a hybrid dataset with 20,000 entity pairs. It is noteworthy that timestamps in YAGO-WIKI datasets are represented in various forms, e.g., time points, start or end time, time intervals. Statics of time-aware EA datasets are listed in Table 2. Please see the appendix for detailed dataset descriptions.

5.2 Baseline

As mentioned in Section 2.3, most of embedding- based EA methods can be roughly classified into two categories, i.e., translational methods and GNN-based methods. In this paper, we compare our proposed method with three popular tranlational EA methods and seven state-of-the-art GNN-based EA methods as follows,

- Translational Models: MTransE [5], JAPE [29], AlignE [30];
- GNN-based Models: GCN-Align [36], MuGNN [1], MRAEA [22], HyperKA [28], RREA [23], KE-GCN [46], TEA-GNN [44].

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

Table 2: Statistics of our proposed temporal EA datasets. $\lvert\mathcal{P}\rvert$ denotes the total number of reference entity pairs.

Due to the lack of attribute information in TKG datasets, we use the SE (Structural Embedding) variants of JAPE and GCN-Align, and the basic versions of MRAEA and RREA. And attribute-aware EA methods [3, 18, 24, 32, 48] are not selected as baseline models for the same reason. We also do not select EA models [4, 38–40, 47, 50] which use literal information of entities as auxiliary information since our model solely rely on structural information in TKGs for EA. And we use AlignE instead of BootEA since we do not adopt bootstrapping to generate semi-supervised structure data for other baseline models and ours. The current time-aware link prediction methods [8, 11, 13, 37, 42, 43] are not suitable for EA setting since the the EA task does not necessarily score facts.

5.3 Experimental Setup

Implementation Enviroments: We implement our method using Tensorflow and Keras. Except that the experiments of MTransE is implemented based on OpenEA framework [31], all experiments of baseline models are implemented based on their publicly available resource codes. All of experiments are conducted on a single GeForce GTX TITAN X GPU with 12GB RAM.

Evaluation Metrics: CSLS are used as the distance metric for all baseline models as well as our models. We use Mean Reciprocal Rank (MRR) and Hits@K as our evaluation metrics. The Hits@K score is calculated by measuring the proportion of correctly aligned pairs ranked in the top-K. MRR denotes the mean of the reciprocals of these ranks.

Dataset splitting: As we mentioned, timestamps can be easily aligned by uniforming time formats. By contrast, it is difficult to obtain pre-aligned entity pairs in practical applications. Thus, we prefer to using a small proportion (2-12%) of entity pairs as alignment seeds $\mathcal{S}$. For DICEWS, we consider using two of its variants, DICEWS-1K and DICEWS-200, which contain 1,000 and 200 alignment seeds, respectively. For YAGO-WIKI50K, we use its variants, YAGO-WIKI50K-5K and YAGO-WIKI50K-1K, with $\lvert\mathcal{S}\rvert = 5,000$ and $\lvert\mathcal{S}\rvert = 1,000$ respectively, for evaluation. For YAGO-WIKI20K, we take 400 entity pairs as alignment seeds. We compare our method against baselines on DICEWS and YAGO-WIKI50K datasets and conduct a sensitivity study on YAGO-WIKI20K.

Open-world setting: We resplit quadruples of DICEWS dataset according to timestamps of facts. Specifically, we use two quadruple sets $\mathcal{Q}_1'$, $\mathcal{Q}_2'$ which contain quadruples in $\mathcal{Q}_1$, $\mathcal{Q}_2$ occuring before

Models	DICEWS-1K			DICEWS-200			YAGO-WIKI50K-5K			YAGO-WIKI50K-1K		
	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
MuGNN	.617	.525	.794	.412	.367	.583	.808	.762	.890	.632	.589	.733
MRAEA	.745	.675	.870	.564	.476	.733	.848	.806	.913	.685	.623	.801
HyperKA	.669	.588	.842	.474	.383	.653	.829	.784	.900	.665	.610	.775
RREA	.780	.722	.883	.719	.659	.824	.868	.828	.938	.753	.696	.859
KE-GCN	.650	.549	.827	.451	.373	.625	.831	.780	.910	.654	.600	.761
TEA-GNN	.911	.887	.947	.902	.876	.941	.909	.879	.961	.775	.723	.871
TREA	**.933**	**.914**	**.966**	**.927**	**.910**	**.960**	**.958**	**.940**	**.989**	**.885**	**.840**	**.937**

Table 3: Entity alignment results on DICEWS and YAGO-WIKI50K datasets. The best results are written bold. Baseline models are ordered by their publication dates.

2014-01-01 for training. By doing this, 1,064 and 1,079 entities in $\mathcal{E}_1$ and $\mathcal{E}_2$ as well as 730 timestamps in $\mathcal{T}$ are not observed during the training process. We select 1,000 pre-aligned pairs as the training set, none of which involves unobserved entities. The unseen entities, timestamps and quadruples only appear in testing phase. We do not use YAGO-WIKI datasets since most of facts in these datasets occur in the last few timestamps.

Hyper-parameter Configurations: For all baseline models, we mainly focus on the grid research of embedding dimensions d and margins γ and follow their default optimal configurations regarding other hyper-parameters, e.g., learning rates lr, batch sizes b, negative sampling rates η, dropout rates dr, numbers of attentional layers L, numbers of maximum epochs ep, and so forth. For all baselines and our method TREA, we tune d in the range of $(25, 50, 75, 100)$, λ_t in the range of $(0, 0.001, 0.005, 0.01, 0.05, \ldots, 1)$ and γ in the range of $(0, 0.5, 1, 2, 3, 5, 7, 10, 15, 20)$. For a fair comparison, we use the same setup for TREA as TEA-GNN and RREA to fix $L = 2$ and $dr = 0.3$. Specially, we fix $b = 1024$, $ep = 100$ and adopt a RMSprop optimizer with $lr = 0.005$ for TREA. The optimal configuration of TREA regarding d, λ_t and γ are listed as below: $d = 100$, $\lambda_t = 0.001$, $\gamma = 0$ for DICEWS-1K and DICEWS-200; $d = 50$, $\lambda_t = 0.01$, $\gamma = 0$ for YAGO-WIKI50K-5K; $d = 50$, $\lambda_t = 0.005$, $\gamma = 1$ for YAGO-WIKI50K-1K; $d = 100$, $\lambda_t = 0$, $\gamma = 0$ for YAGO-WIKI20K.

5.4 Main Results

In table 3, we report the performances of our method and all baseline methods on DICEWS and YAGO-WIKI50K datasets. Among all baseline models, TEA-GNN obtains the best performance since TEA-GNN can also capture time information by utilizing a temporal relational graph neural network. Compared to TEA-GNN, TREA improves Hits@1 by 3.0%, 3.9%, 6.9% and 16.2% on four datasets respectively. Noteworthily, the performance difference between TEA-GNN and TREA on DICEWS datasets is smaller. One possible reason is that facts in two TKGs of each DICEW dataset are all from ICEWS05-15 and have a high overlap ratio of 50%, and thus both TEA-GNN and TREA are able to align entities well between these two homogeneous TKGs with their similar abilities of modeling time information. By contrast, two TKGs of each YAGO-WIKI50K dataset are extracted from different knowledge bases, and YAGO-WIKI50K datasets are much sparse than DICEWS datasets at the entity level. Thus, it is more important to force input feature of each entity to reflect its neighborhood semantics, i.e., the embeddings of neighboring entities, relations and timestamps. Moreover, using Margin-based Multi-class Log-loss (MML) can be more helpful to find hard negative samples among a larger amount of negative heterogeneous entity pairs. Compared to DICEWS datasets where two TKGs are homogeneous, YAGO-WIKI50K datasets are extracted from different resources and are sparser at entity level which are closer to application scenarios in the real world. Therefore, TREA achieves more significant improvements on YAGO-WIKI50K datasets with Neighborhood Aggregation Represetation (NAR) and MML. We conduct an ablation study to empirically verify the above arguments in the later section.

5.5 Ablation Study

Although TREA and TEA-GNN use similar temporal relational attention (TRA) mechanisms, TREA achieves better performances with NAR, MML and sequential time regularizer (STR). To demonstrate the effectiveness of each design of TREA, we conduct an ablation experiment on DICEWS-200 and YAGO-WIKI50K-1K. We implement two variants of TREA including a time-unaware variant TREA (-Ti) which is implemented by taking $t \in \mathcal{T}$ as unknown time information, i.e., $t \equiv t_0$, a variant TREA (-Re) which is set to be relation-unaware in a similar way. And we also evaluate two variants of TREA trained without NAR and STR, respectively, i.e., TREA (-NAR) and TREA (-STR). Additionally, TREA trained with Margin Rank Loss (MRL), i.e., TREA (W. MRL) is considered.

As shown in Table 4, there are obvious declines of the performances of TREA on both datasets after the removal of time information, which supports our intuition that the incorporation of time information have a remarkable effect on EA results between TKGs. We also provide an example study in the appendix with regard to this argument. The removal of relation information slightly changes

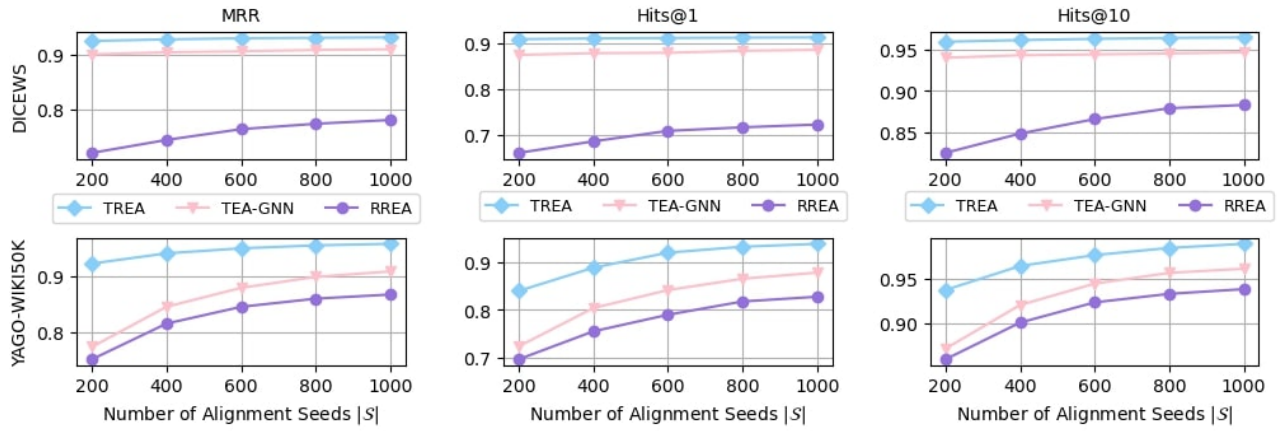

Figure 4: Entity alignment results w.r.t. different sizes of pre-aligned seed entity pairs on temporal datasets.

Models	DICEWS-200			YAGO-WIKI50K-1K		
	MRR	Hits@1	Hits@10	MRR	Hits@1	Hits@10
TREA	.927	.910	.960	.885	.840	.937
(-Ti)	.803	.741	.923	.785	.734	.885
(-Re)	.784	.727	.895	.869	.823	.924
(-NAR)	.923	.907	.955	.874	.827	.932
(-STR)	.908	.884	.940	.878	.830	.931
(w. MRL)	.929	.907	.965	.798	.745	.873

Table 4: Results of Ablation experiment.

the performances on YAGO-WIKI50K-5K dataset where two TKGs, $\mathcal{G}_1$ and $\mathcal{G}_2$ are extracted from different sources.

On YAGO-WIKI50K-1K, NAR improves TREA's performance and TREA trained with MML outperforms TREA trained with MRL. However, these two technologies fails to have an positive effect on DICEWS-200. As mentioned in Section 5.4, NAR is important to making the input information of entities as rich as possible when TKGs are sparse at the entity level. Thus, NAR is less necessary for dense TKGs like DICEWS. Moreover, training process on DICEWS-200 might be mainly influenced by positive samples due to the lack of hard negative samples while hard negative samples play an important role in the training process on YAGO-WIKI50K-1K. Thus, MML scarcely change the performance of TREA on DICEWS-200, but have significant effects on YAGO-WIKI50K-1K. Specifically, e_1 and e_2 can be regarded as hard negative samples of each other if there exist multiple observed similar-looking quadruple pairs, shaped like (e_s, r, e_1, t) and (e_s, r, e_2, t), which have the same subjects, relations and timestamps but involve these two entities respectively. For each observed quadruple (e_s, r, e_o, t) in YAGO-WIKI50K-1K, there are on average 5.22 similar-looking quadruples shaped like (e_s, r, e'_o, t). By contrast, this number drops to 0.15 in DICEWS-200. Thus, the average number of hard negative samples of each entity in training data of DICEWS-200 is much lower than YAGO-WIKI50K-1K. Different from NAR and MML, STR is helpful for improving the performance of TREA on DICEWS200 because the distribution of time data in DICEWS200 is dense and uniform. Meanwhile, some timestamps are missed in YAGO-WIKI50K and most of timestamps are concentrated in the last few years with a long tail of other timestamps, which can not be well modelled by a sequential time model.

5.6 Robustness Study

It is costly to annotate pre-aligned entity pairs manually, especially for the large-scale KGs. Thus, it is essential for an EA method to maintain an effective performance with a small proportion of pre-aligned entities. To verify the robustness of TREA, we test TREA and the two best performing baseline models, i.e., TEA-GNN and RREA, with $|\mathcal{S}|$ varying from 200 to 1,000 with step size of 200 and $|\mathcal{S}|$ varying from 1,000 to 5,000 with step size of 1,000 on DICEWS and YAGO-WIKI50K, respectively. As shown in Figure 4, TREA is not only superior to TEA-GNN and RREA in all seed sizes, but also has a more gradual slope curve. This demonstrates that our model is less dependent on additional training data due to its robust model structure and learning effectiveness, and it is promising to have good capability of generalization.

5.7 Efficiency Study

Table 5 lists numbers of trainable parameters of our method and all baseline models, and their overall time costs on DICEWS and YAGO-WIKI50K dataset, including data loading, pre-processing, training, and evaluating. As shown in Table 5, the training efficiency of TREA exceeds most of baseline models. Only GCN-Align has less training time than TREA on both datasets, since it uses a small negative sampling rate and it is evaluated only once during the training process. While most of GNN-based EA models trained with MRL need thousands of training epochs, TREA can converge within 100 epochs by adopting MML. With the inclusion of time embeddings, TREA does not excessively increase the number of trainable parameters on DICEWS datasets, compared to baseline models. On YAGO-WIKI50K datasets, TREA has even fewer free parameters than most baseline models since lower-dimensional embeddings ($d = 50$) are used. In general, the high efficiency of TREA makes the time-aware EA on large-scale KGs possible.

Models	DICEWS		YAGO-WIKI50K	
	Time Cost	Parameter Number	Time Cost	Parameter Number
MTransE	284	1.95M	1,624	9.89M
JAPE	953	1.95M	4,083	9.89M
AlignE	9,797	1.95M	5,384	9.89M
GCN-Align	39	1.92M	613	9.91M
MuGNN	2,173	1.96M	22,457	9.90M
MRAEA	2,647	2.01M	14,338	7.42M
HyperKA	6,389	1.97M	40,951	9.91M
RREA	1,538	2.00M	6,487	4.95M
KE-GCN	1,704	2.18M	9,510	11.61M
TEA-GNN	4,410	2.40M	9,350	4.95M
TREA	128	2.41M	2,655	4.96M
(w. MRL)	1,948	2.41M	6,327	4.96M

Table 5: Time costs (seconds) and numbers of trainable parameters of EA methods.

5.8 Sensitivity Study

YAGO-WIKI20K is a temporally-hybrid dataset where some facts are non-temporal. Thus, a part of entities are insensitive to temporal change. Testing entity pairs can be categorized into **highly time-sensitive** entity pairs and **lowly time-sensitive** entity pairs according to the ratio of the number of entities' time-aware connected links over the amount of all links within their neighborhood [44]. The EA results of TREA and its time-unaware variant on the highly time-sensitive test set and the lowly time-sensitive test set are reported in Table 6. One can find that TREA significantly outperforms its time-unaware variant on the highly time-sensitive test set while they have close performances on lowly time-sensitive test set, which means that the incorporation of time information have more significant effect on higher time-sensitive entity pairs.

Models	Highly Time-Sensitive			Lowly Time-Sensitive		
	MRR	Hits@1	Hits@10	MRR	Hits@1	Hits@10
TREA	.909	.875	.961	.370	.319	.470
(- Ti)	.834	.781	.930	.367	.317	.466

Table 6: Entity alignment results on YAGO-WIKI20K.

5.9 Open-World Learning

In the real world, most of KGs are dynamic with new emerging entities and timestamps. However, the existing EA methods hold a closed-world assumption that KGs are fixed and is unable to model representations of new emerging entities and timestamps. Taking RREA and TEA-GNN as examples, neither of them can model input features of unseen entities since their embeddings are unobtainable or untrained. Moreover, RREA does not consider time information and TEA-GNN can not model unseen timestamps. By contrast, TREA can model the input features with its known neighborhood information by using Eq. 1, i.e., NAR. In Eq. 11, we force the change of timestamp embeddings to satisfy a linear equation over time by using STR. Thus, we can roughly estimate embeddings of future timestamps with embeddings of observed timestamps and the temporal slope.

We compare TREA with TEA-GNN and RREA on DICEWS under open-world setting. To enable TEA-GNN and RREA to model unseen entities, we drop links flowing from unobserved entities towards observed entities and remove unseen entities' input features from their final representations in testing phase, which causes the information loss inevitably. We use the last seen timestamps to represent future timestamps for TEA-GNN since the embedding of unknown time information t_0 is not updated during the training process. As mentioned in Section 5.3, 1,064 and 1,079 entities as well as 730 timestamps are unseen before 2014-01-01 in DICEWS. We select 1,000 entity pairs only involving observed entities as training set and the rest are testing set. Among testing entity pairs, 1,027 entity pairs involve unseen entities, called **unobserved entity pairs**, and others are called **observed entity pairs**. Table 7 shows that TREA significantly outperforms TEA-GNN and TREA under open-world setting, especially on unobserved entities. These experimental results support our argument that TREA can effectively perform EA tasks between OKGs. We also provide an example study in the appendix as argumentation.

Models	Unobserved Entity Pairs			Observed Entity Pairs		
	MRR	Hits@1	Hits@10	MRR	Hits@1	Hits@10
RREA	.253	.075	.483	.407	.361	.580
TEA-GNN	.324	.155	.590	.513	.392	.748
TREA	.479	.342	.748	.643	.549	.825

Table 7: Open-world Entity alignment results on DICEWS.

6 CONCLUSION

Embedding models have been successful for entity alignment between KGs, but lack consideration of time information and the dynamism of open-world KGs. To address this challenge, we present a GNN-based method which uses an efficient attention mechanism to learn both time and relation information in TKGs. In addition, a neighborhood aggregation representation is used to incorporate neighborhood information into entitie's input features and is able to represent observed entities and new emerging entities. A margin-based multi-class log-loss is used for fast parameter optimization. A sequential time regularizer helps to model future time representations. Experimental results on three TKG benchmarks show that our method achieves higher performances than the state-of-the-art EA methods under both close-world setting and open-world setting.

ACKNOWLEDGMENTS

We acknowledge the support of the following projects: SPEAKER (BMWi FKZ 01MK20011A), JOSEPH (Fraunhofer Zukunftsstiftung), the EU projects Cleopatra (GA 812997), PLATOON(GA 872592), TAILOR(GA 952215), KnowGraphs(GA 860801), the BMBF projects MLwin(01IS18050) and the BMBF excellence clusters ML2R (BmBF FKZ 01 15 18038 A/B/C) and ScaDS.AI (IS18026A-F). We also thank the China Scholarship Council (CSC) and the International Max Planck Research School for Intelligent Systems (IMPRS-IS) for supporting Chengjin Xu and Bo Xiong, respectively.

REFERENCES

[1] 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* (2019).

[2] 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* (2018).

[3] Muhao Chen, Weijia Shi, Ben Zhou, and Dan Roth. 2020. Cross-lingual entity alignment with incidental supervision. *arXiv preprint arXiv:2005.00171* (2020).

[4] Muhao Chen, Yingtao Tian, Kai-Wei Chang, Steven Skiena, and Carlo Zaniolo. 2018. Co-training Embeddings of Knowledge Graphs and Entity Descriptions for Cross-lingual Entity Alignment. In *IJCAI*.

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

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

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

[8] Alberto García-Durán, Sebastijan Dumančić, and Mathias Niepert. 2018. Learning Sequence Encoders for Temporal Knowledge Graph Completion. *arXiv preprint arXiv:1809.03202* (2018).

[9] Qingyu Guo, Fuzhen Zhuang, Chuan Qin, Hengshu Zhu, Xing Xie, Hui Xiong, and Qing He. 2020. A Survey on Knowledge Graph-Based Recommender Systems. arXiv:2003.00911 [cs.IR]

[10] Zhen Han, Peng Chen, Yunpu Ma, and Volker Tresp. 2020. Explainable subgraph reasoning for forecasting on temporal knowledge graphs. In *International Conference on Learning Representations*.

[11] Woojeong Jin, Changlin Zhang, Pedro Szekely, and Xiang Ren. 2019. Recurrent Event Network for Reasoning over Temporal Knowledge Graphs. *arXiv preprint arXiv:1904.05530* (2019).

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

[13] Timothée Lacroix, Guillaume Obozinski, and Nicolas Usunier. 2020. Tensor Decompositions for temporal knowledge base completion. *arXiv preprint arXiv:2004.04926* (2020).

[14] Timothée Lacroix, Nicolas Usunier, and Guillaume Obozinski. 2018. Canonical tensor decomposition for knowledge base completion. International Conference on Machine Learning (ICML).

[15] Jennifer Lautenschlager, Steve Shellman, and Michael Ward. 2015. Icews event aggregations. *Harvard Dataverse* 3 (2015).

[16] 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 (2015), 167–195.

[17] Yaguang Li, Rose Yu, Cyrus Shahabi, and Yan Liu. 2017. Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. *arXiv preprint arXiv:1707.01926* (2017).

[18] Zhiyuan Liu, Yixin Cao, Liangming Pan, Juanzi Li, and Tat-Seng Chua. 2020. Exploring and evaluating attributes, values, and structures for entity alignment. *arXiv preprint arXiv:2010.03249* (2020).

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

[20] Franco Manessi, Alessandro Rozza, and Mario Manzo. 2020. Dynamic graph convolutional networks. *Pattern Recognition* 97 (2020), 107000.

[21] Xin Mao, Wenting Wang, Yuanbin Wu, and Man Lan. 2021. Boosting the Speed of Entity Alignment 10×: Dual Attention Matching Network with Normalized Hard Sample Mining. In *Proceedings of the Web Conference 2021*. 821–832.

[22] Xin Mao, Wenting Wang, Huimin Xu, Man Lan, and Yuanbin Wu. 2020. 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*. 420–428.

[23] Xin Mao, Wenting Wang, Huimin Xu, Yuanbin Wu, and Man Lan. 2020. Relational Reflection Entity Alignment. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. 1095–1104.

[24] Tam Thanh Nguyen, Thanh Trung Huynh, Hongzhi Yin, Vinh Van Tong, Darnbi Sakong, Bolong Zheng, and Quoc Viet Hung Nguyen. 2020. Entity alignment for knowledge graphs with multi-order convolutional networks. *IEEE Transactions on Knowledge and Data Engineering* (2020).

[25] 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*, Vol. 34. 5363–5370.

[26] Baoxu Shi and Tim Weninger. 2018. Open-world knowledge graph completion. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 32.

[27] 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*. ACM, 697–706.

[28] Zequn Sun, Muhao Chen, Wei Hu, Chengming Wang, Jian Dai, and Wei Zhang. 2020. Knowledge Association with Hyperbolic Knowledge Graph Embeddings. In *EMNLP*.

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

[30] Zequn Sun, Wei Hu, Qingheng Zhang, and Yuzhong Qu. 2018. Bootstrapping Entity Alignment with Knowledge Graph Embedding.. In *IJCAI*, Vol. 18. 4396–4402.

[31] 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 (2020), 2326–2340. http://www.vldb.org/pvldb/vol13/p2326-sun.pdf

[32] Bayu Distiawan Trisedya, Jianzhong Qi, and Rui Zhang. 2019. Entity alignment between knowledge graphs using attribute embeddings. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33. 297–304.

[33] Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex Embeddings for Simple Link Prediction. *arXiv preprint arXiv:1606.06357* (2016).

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

[35] Peifeng Wang, Jialong Han, Chenliang Li, and Rong Pan. 2019. Logic attention based neighborhood aggregation for inductive knowledge graph embedding. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33. 7152–7159.

[36] 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*. 349–357.

[37] 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* (2020).

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

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

[40] Yuting Wu, Xiao Liu, Yansong Feng, Zheng Wang, and Dongyan Zhao. 2020. Neighborhood matching network for entity alignment. In *Proceedings of the 58th Annual Meeting of the Association for Computational L6477is6487*. 6477–6487.

[41] Chengjin Xu, Yung-Yu Chen, Mojtaba Nayyeri, and Jens Lehmann. 2021. Temporal Knowledge Graph Completion using a Linear Temporal Regularizer and Multivector Embeddings. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 2569–2578.

[42] 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* (2019).

[43] Chengjin Xu, Mojtaba Nayyeri, Fouad Alkhoury, Hamed Shariat Yazdi, and Jens Lehmann. 2020. TeRo: A Time-aware Knowledge Graph Embedding via Temporal Rotation. *arXiv preprint arXiv:2010.01029* (2020).

[44] Chengjin Xu, Fenglong Su, and Jens Lehmann. 2021. Time-aware Graph Neural Network for Entity Alignment between Temporal Knowledge Graphs. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. 8999–9010.

[45] Sijie Yan, Yuanjun Xiong, and Dahua Lin. 2018. Spatial temporal graph convolutional networks for skeleton-based action recognition. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 32.

[46] Donghan Yu, Yiming Yang, Ruohong Zhang, and Yuexin Wu. 2021. Knowledge Embedding Based Graph Convolutional Network. In *Proceedings of the Web Conference 2021*. 1619–1628.

[47] Weixin Zeng, Xiang Zhao, Jiuyang Tang, and Xuemin Lin. 2020. Collective entity alignment via adaptive features. In *2020 IEEE 36th International Conference on Data Engineering (ICDE)*. IEEE, 1870–1873.

[48] Qingheng Zhang, Zequn Sun, Wei Hu, Muhao Chen, Lingbing Guo, and Yuzhong Qu. 2019. Multi-view Knowledge Graph Embedding for Entity Alignment. In *IJCAI*. 5429–5435.

[49] Hao Zhu, Ruobing Xie, Zhiyuan Liu, and Maosong Sun. 2017. Iterative Entity Alignment via Joint Knowledge Embeddings.. In *IJCAI*, Vol. 17. 4258–4264.

[50] Renbo Zhu, Meng Ma, and Ping Wang. 2021. RAGA: Relation-Aware Graph Attention Networks for Global Entity Alignment.. In *PAKDD (1)*. Springer, 501–513.

Entities to Be Aligned	**Ashley_Wooliscroft** (in $\mathcal{E}_1$ of YAGO-WIKI50K-1K)
Predictions	TREA: **Ashley Wooliscroft**; TREA (-Ti): **Matt Haddrell** (in $\mathcal{E}_2$ of YAGO-WIKI50K-1K)
Similar Facts (Links) Involving Aligned Entities Between $\mathcal{Q}_1$ and $\mathcal{Q}_2$	(Ashley_Wooliscroft, playsFor, Newcastle_Town_F.C., [2003, 2004]), (Ashley_Wooliscroft, playsFor, Leek_Town_F.C., [2004, 2006]), (Ashley_Wooliscroft, playsFor, Kidsgrove_Athletic_F.C., [2006, 2007]), ... (in $\mathcal{Q}_1$ of YAGO-WIKI50K-1K) (Matt Haddrell, member of sports team, Newcastle Town F.C., [2006, 2007]), (Matt Haddrell, member of sports team, Leek Town F.C.., [2004, 2005]), (Matt Haddrell, member of sports team, Kidsgrove Athletic F.C., [2009, 2010]), ... (in $\mathcal{Q}_2$ of YAGO-WIKI50K-1K)

Table 8: Examples of different alignment predictions between TREA and TREA (-Ti).

Unobserved Entity Alignment	TREA	TEA-GNN
Yoon Sang-jick $\in \mathcal{E}_1$	**Yoon Sang-jick** $\in \mathcal{E}_2$	Andrew Robb $\in \mathcal{E}_2$
Edwin Lacierda $\in \mathcal{E}_1$	**Edwin Lacierda** $\in \mathcal{E}_2$	Wen Jiabao $\in \mathcal{E}_2$
Dunya Maumoon $\in \mathcal{E}_1$	**Dunya Maumoon** $\in \mathcal{E}_2$	Maumoon Abdul Gayoom $\in \mathcal{E}_2$

Table 9: Examples of Different Alignment Predictions between TREA and TEA-GNN under open-world setting.

A DETAILS OF TEMPORAL ENTITY ALIGNMENT DATASETS

In this paper, we use three TKG datasets extracted from ICEWS [15], YAGO [27], Wikidata [7] as references for evaluating temporal and non-temporal EA methods, i.e., **DICEWS**, **YAGO-WIKI50K** and **YAGO-WIKI20K** [44]. The statics of the three temporal EA datasets are listed in Table 2.

Integrated Crisis Early Warning System (ICEWS) is a publicly available large-scale event-based database that contains political events with specific time annotations extracted from millions of real-world news stories. It is noteworthy that time annotations in ICEWS are all time points, e.g., (*Barack Obama*, *Visit*, *Ukraine*, *2014-07-08*). ICEW05-15 [8] is a subset of ICEWS which contains 10,094 entities, 251 relations, 4,017 time steps and 461,329 temporal facts occurring during 2005 to 2015, and is commonly used as a TKG benchmark dataset in the community. **DICEWS** is built based on ICEWS05-15 in the similar way to the establishment of DFB datasets [49]. First, ICEWS05-15 quadruples is randomly divided into two subsets $\mathcal{Q}_1$ and $\mathcal{Q}_2$ of similar size, and making the overlap ratio of the amount of shared quadruples between $\mathcal{Q}_1$ and $\mathcal{Q}_2$ to all quadruples equal to 50%. These two TKGs have the same set of time steps $\mathcal{T} = \mathcal{T}_1 \cup \mathcal{T}_2$, i.e., the sequence of dates in the year 2005.

Wikidata is a free and open knowledge base that store the structured data from Wikipedia. YAGO is also an open source knowledge base and is extracted from Wikipedia and other sources. In these two knowledge base, there are a large number of identical entities represented in different surface forms and a part of facts are attached with timestamps of various forms, e.g., time points, start/end time and time intervals. Lacroix et al. [13] built a large-scale TKG dataset from Wikidata, which contains 43,2715 entities, 407 relations and 1,724 time steps (only year information was kept) by filtering out high-frequency entities and relations. The whole dataset has over 7 millions of triples in total and about 10% of them are attached to specific timestamps. To build **YAGO-WIKI50K** from YAGO and Wikidata, top 50,000 entities are first selected according to their frequencies in this dataset and link them to their equivalent YAGO entities according to their QIDs and the mappings of YAGO entities to Wikidata QIDs. Two TKGs are generated by filtering out facts only involving the selected entities from the above-mentioned subset of Wikidata and all YAGO facts and then attach complementary time metadata to the corresponding YAGO facts. In this step, a small part of entities are removed. At last, non-temporal facts are removed from these two filtered TKGs to make sure that YAGO-WIKI50K is **fully temporal** and only keep the year information of timestamps in YAGO facts and uniform the time formats of both TKGs to generate the shared time set $\mathcal{T}$. **YAGO-WIKI20K** is constructed in the same way except that the amount of selected Wikidata entities is reduced to 20,000 and keep the non-temporal facts in two TKGs of this **temporally hybrid** dataset.

B EXAMPLE STUDY

We provide an example in Table 9 that TREA gives different predictions from TREA (-Ti) with consideration of additional time information. In YAGO-WIKI50K-5K dataset, the entity *Ashley Wooliscroft* only have 5 occurrences in $\mathcal{Q}_1$ and the top retrieved entity in $\mathcal{E}_2$ by TREA (-Ti) is *Matt Haddrell*. Since these two players played for three same football clubs, the time-unaware model wrongly aligns these two different entities from $\mathcal{E}_1$ and $\mathcal{E}_2$ regardless of the fact that they played for different periods of time. By contrast, TREA gives correct prediction with the consideration of time information. In Table , we show some examples that TEA-GNN gives wrong predictions and TREA predicts correctly for unobserved entity pairs under open-world setting. Without NAR and STR, TEA-GNN more frequently gives wrong predictions for unobserved entities since it can only recognize straightforward neighborhood information, but can hardly learn the semantics of new emerging entities.