Entity Alignment with Reliable Path Reasoning and Relation-Aware Heterogeneous Graph Transformer

Weishan Cai$^{1,2}$, Wenjun Ma$^1$, Jieyu Zhan$^1$, Yuncheng Jiang$^{1,3}$*

1School of Computer Science, South China Normal University, China
2School of Computer and Information Engineering, Hanshan Normal University, China
3School of Artificial Intelligence, South China Normal University, China
caiws@m.scnu.edu.cn, phoenixsam@sina.com, zhanjiedu,ycjiang@scnu.edu.cn

Abstract

Entity Alignment (EA) has attracted widespread attention in both academia and industry, which aims to seek entities with same meanings from different Knowledge Graphs (KGs). There are substantial multi-step relation paths between entities in KGs, indicating the semantic relations of entities. However, existing methods rarely consider path information because not all natural paths facilitate for EA judgment. In this paper, we propose a more effective entity alignment framework, RPR-RHGT, which integrates relation and path structure information, as well as the heterogeneous information in KGs. Impressively, an initial reliable path reasoning algorithm is developed to generate the paths favorable for EA task from the relation structures of KGs. This is the first algorithm in the literature to successfully use unrestricted path information. In addition, to efficiently capture heterogeneous features in entity neighborhoods, a relation-aware heterogeneous graph transformer is designed to model the relation and path structures of KGs. Extensive experiments on three well-known datasets show RPR-RHGT significantly outperforms 10 state-of-the-art methods, exceeding the best performing baseline up to 8.62% on Hits@1. We also show its better performance than the baselines on different ratios of training set, and harder datasets.

1 Introduction

Most Knowledge Graphs (KGs) are often disconnected from each other because they are constructed with different technologies and languages, which poses challenges for merging and integrating different KGs. Entity Alignment (EA) is a task to connect entities with the same meaning in different KGs, which plays a fundamental role in the knowledge fusion of KGs. Recently, EA methods based on the Graph Neural Networks (GNNs) are more favored by researchers than the translation-based methods. GNNs not only exhibit excellent performance in aggregating the neighborhood features of nodes, but also can design corresponding feature acquisition methods for EA tasks, while translation-based methods are designed for link prediction.

Although current GNNs-based methods have achieved promising results, they still suffer from the following three limitations. First, many methods [Wu et al., 2019; Sun et al., 2020] treat KGs as homogeneous graphs without considering the heterogeneous features of sides between entities. Actually, the heterogeneous information helps to improve the accuracy and robustness of alignment judgments. Second, some semantic information other than relation structures is considered by many works, such as entity attributes [Liu et al., 2020; Cai et al., 2022], text descriptions [Yang et al., 2019], and multi-modal information [Liu et al., 2021]. However, the more semantic information a method integrates, the more data its application requires, which cannot be satisfied in some scenarios. Third, some other works [Wu et al., 2020; Zhu et al., 2020] only rely on the relation structures, and obtain inter-graph information based on Graph Matching Networks (GMN) [Li et al., 2019] to mine more similar features between aligned entities. Nonetheless, the matching modules they introduced for learning inter-graph information runs through the entire training process with high temporal and space complexity.

Therefore, for the first limitation above, we design a Relation-Aware Heterogeneous Graph Transformer (RHGT) to effectively extract the similarity features of aligned entities in their heterogeneous structures. For the latter two limitations above, we develop a Reliable Path Reasoning algorithm (RPR) that can directly extract the path structures favorable for EA tasks from the original relation structures. Existing methods rarely consider the path information of KGs (i.e., the indirect neighborhood of aligned entities), despite their success in modeling of direct relationship facts. It is known that substantial multi-step relational paths exist between entities, indicating their semantic relationships. But not all natural paths facilitate EA judgment, and some even backfire. Although IPTransE [Zhu et al., 2017] considers the reliability of paths, it assumes all relations between KGs are pre-aligned. Essentially, our idea is to make full leverage of the richness of KGs by simultaneously comparing the similarities of relation and path structures of aligned entities. We believe the paths that frequently appear near pre-aligned entities can be regarded as reliable and used to align other entities. The fusion of relation and path structure information complements
each other, alleviating the inconsistency between each type of information of aligned entities.

After all, we combine above two methods into a entity alignment framework called RPR-RHGT, which not only considers the heterogeneous information of sides in KGs, but also mines the path information within the relation structures of KGs. Extensive experiments on three well-known benchmark datasets show RPR-RHGT not only outperforms 10 state-of-the-art models significantly, but also has impressive scalability and robustness.

2 Related Work

Translation-based Entity Alignment. Such methods are mainly based on TransE [Bordes et al., 2013] and its variants. MTransE [Chen et al., 2017] is the pioneering work, which uses TransE to model the entities and relations and evaluates the transforms between two vector spaces based on pre-aligned entities. Other works utilize additional information or external knowledge of KGs, such as attribute structures [Sun et al., 2017; Zhang et al., 2019b; Trisedya et al., 2019], entity descriptions [Chen et al., 2018], entity names [Zhang et al., 2019b], ontology schemata [Xiang et al., 2021], to find more similar features of aligned entities. There are also some works [Sun et al., 2018; Zhu et al., 2017] that try to discover more new aligned entities by iterative strategies.

GNNs-based Entity Alignment. GNNs-based methods mainly utilize Graph Convolutional Networks (GCNs) and Graph Attention Networks (GATs) to aggregate the neighborhood feature of each entity, thereby obtaining the neighborhood similarity between aligned entities. Most of them directly compare the neighborhood similarity between aligned entities in relation structures [Wu et al., 2019]. There are several attempts to simultaneously consider the similarities in the attribute and relation structures [Liu et al., 2020]. Some other works smartly model both intra-graph and cross-graph information, and learn similarities by building cross-graph attention mechanism using GMN [Wu et al., 2020; Zhu et al., 2020]. Besides, some researchers believe that the heterogeneity of edges in KGs should be considered when aggregating the neighborhood features, because KGs are heterogeneous graphs. They propose or apply some heterogeneous graph embedding methods to learn better representations for entities [Cai et al., 2022]. All aforementioned works only consider the similarity of direct neighborhoods between aligned entities. However, aligned entities have some similarity in their indirect neighborhoods. Hence, we attempt to obtain the similarity between aligned entities in the relation structures and multi-hop path structures of KGs simultaneously in the paper.

Heterogeneous GNNs. Recently, many works have tried to extend GNNs to the modeling of heterogeneous graphs. RGCNs [Schlichtkrull et al., 2018] and RGATs [Busbridge et al., 2019] model heterogeneous graphs by using a weight matrix for each relation. HAN [Wang et al., 2019] proposes a hierarchical attention mechanism to learn the weights of nodes and meta-paths from node-level and semantic-level attention, respectively. HetGNN [Zhang et al., 2019a] adopts different Recurrent Neural Networks (RNNs) for different types of nodes to integrate multi-modal features. However, due to the large number of relations in KGs, the training complexity is high when applying them to model KGs. More recently, HGT [Hu et al., 2020] and RHGT [Mei et al., 2022] try to model the heterogeneity by heterogeneous graph transformers. But they are not designed to capture neighborhood similarity, so it is difficult to directly apply to EA tasks. Therefore, an improved heterogeneous graph transformer is designed to consider the heterogeneity of KGs, thereby obtaining high-quality entity embeddings for EA tasks.

3 Proposed Framework

3.1 Problem Definition

To increase the neighborhood semantics of entities, we introduce a meta-path-based similarity framework for EA. The classic meta-path paradigm is defined as a sequence of relations between objects, so we define the new compound relation between two entities as a relation path in this paper. For example, suppose (e₁, e₂) is an aligned entity, where superscripts denote different KGs. There is a path relation (r₁, r₂) near e₁, because the following relation exists:  e₁ → r₁ e₁ → r₂ e₁, but there may not be a similar path near e₂. Therefore, not all paths in the neighborhood of entities are reliable for EA learning. In other words, we should only keep partially reliable paths to learn the neighborhood similarity of aligned entities.

Definition 1 (Reliable Path Set). In this paper, we use P = {p₁, ..., pₜ, ..., pₚ} represents reliable path set, where each pᵢ = (rᵢ, rᵢ₊₁) is effective for EA learning. “Reliable path” here refer to path that facilitate EA learning, rather than the meaningful path. We believe the paths that frequently occur in the neighborhoods of pre-aligned entities can be considered reliable. In this paper, we only consider paths based on two-hop relations, and the study of a wider range of path structures will be left to future work.

Definition 2 (KG with Reliable Path Set). We define KG as G = (E, R, T_r, P, T_p), where E is entity set, R is relation set, T_r ⊆ E × R × E is relation triple set, P is reliable path set, and T_p = {p₁, ..., pₚ} is path triple set.

Definition 3 (Entity Alignment). G¹ = (E¹, R¹, T_r¹, P¹, T_p¹) and G² = (E², R², T_r², P², T_p²) are two KGs to be aligned. Let L = {l₁, l₂} be the pre-aligned entity set, where lᵢ refers to the same real-world object. Entity Alignment tasks aim to find the remaining aligned entities between two KGs.

Formally, we use bold letter for embedding vector. For example, E¹ represents the embedding matrix of entities in G¹, and eᵢ represents the the i-th row of E¹. In addition, the entity name is the most common text used to identify an entity, which can be used to effectively capture the semantic similarity of aligned entities. Therefore, we apply pre-trained word embeddings to generate initial representations of entities, E₀, and use them as the input of our framework.
3.2 Overview Framework of RPR-RHGT

In this section, we introduce our proposed framework RPR-RHGT, a novel robust EA framework based on a reliable path reasoning algorithm and a relation-aware heterogeneous graph transformer. Specifically, RPR-RHGT is mainly composed of three modules, as shown in Figure 1: (1) Reliable Path Reasoning (RPR). A reliable path reasoning algorithm is developed to infer the reliable relation paths and form path structures of two KGs. (2) Relation-Aware Heterogeneous Graph Transformer (RHGT). We design the RHGT to capture the features of specific patterns of relations and paths with fewer parameters, which contain the heterogeneous neighborhood features of aligned entities in relation and path structures. (3) Alignment Learning. This module computes the loss function and similarity matrices of path-based and relation-based entity embeddings, and evaluates the probabilities of EA.

3.3 Reliable Path Reasoning (RPR)

As discussed in Section 1, not all relation paths are reliable for EA learning. It is known that each KG is constructed according to relatively stable data sources and construction rules. Our key insight is the path with a high number of matches between the neighborhoods of pre-aligned entities (small range) can be regarded as reliable, which can be used to match judgments of other entities (large range). We first establish the path neighborhood matching between each pre-aligned pair (see Figure 2(a)), derive the matching paths (see Figure 2(b)), finally select those paths with high numbers of matches to form a reliable path set $P$.

Specifically, for a given pair $(e^1_a, e^2_a) \in \mathbb{L}$, the similarity matrix $S$ denotes the similarities between path neighborhoods $PN(e^1_a)$ and $PN(e^2_a)$, where $PN(\cdot)$ indicates path neighborhood of an entity. Firstly, the entities with maximum similarities in each row of $S$ are selected as the matching neighbors. As shown in Figure 2(a), the matching result of $e^1_1$ (one neighbor of $e^1_a$) is $e^2_{n-1}$, because their similarity is the largest in first row. However, there may be multiple neighbors of $e^1_a$ that match the same neighbor of $e^2_a$, such as $e^1_2$ and $e^2_3$ match with $e^2_{n-1}$ simultaneously. Therefore, the neighbor matching requires some one-to-one constraints: 1) the similarity of matching neighbors must reach a certain threshold: $MN(S) = \{e^2_i | |S_{ij}| \leq \tau_{sim} \}$; 2) sort the similarity values that satisfy the threshold from high to low, and then perform one-to-one matching: $Match_{1:1}(MN(S)) = [(e^1_1, e^2_1), (e^1_2, e^2_2), \ldots]$. So as a result, $e^1_1$ is chosen to match $e^2_{n-1}$, because $0.9 > 0.7$. Obviously, only some neighbors of $e^2_a$ may end up finding matching neighbors.

Secondly, for each $(e^1_1, e^2_j) \in Match_{1:1}(MN(S))$, we can deduce the path matching pair $(p^1_k, p^2_j)$ according to the following reasoning relationship, as shown in Figure 2(b):

$$
e^1_1 \leftrightarrow e^2_j \Rightarrow (e^1_a, p^1_k, e^1_1) \leftrightarrow (e^2_j, p^2_j, e^2_j) \Rightarrow p^1_k \leftrightarrow p^2_j,
$$

where $\leftrightarrow$ indicates the matching relationship; $(e^1_a, p^1_k, e^1_1)$ and $(e^2_j, p^2_j, e^2_j)$ are the path triples.

The last step is to count the matching number of each matching path, $counter(p^1_k \leftrightarrow p^2_j)$, and select those paths with high numbers of matches to form reliable path set $P$:

$$P = \{ (p^1_k, p^2_j) | counter(p^1_k \leftrightarrow p^2_j) > \tau_{path} \},$$

where $\tau_{path}$ is set according to the specific dataset. Algorithm 1 gives the procedure of our RPR algorithm.
Algorithm 1 Procedure of RPR Algorithm.

Input: (1) $G = (E, R, T_{rel})$; (2) pre-aligned entities $L$; (3) entity name embeddings $E^h$.
Output: reliable path set $P$, path triple set $T_{path}$.
1: Set $P_{all} \leftarrow \emptyset$;
2: for $(e^h_1, e^h_2) \in L$ do
3: Compute matching neighbors of path structures $Match_{1:3}(MN(S))$ between $PN(e^h_1)$ and $PN(e^h_2)$;
4: for $(e^h_1, e^h_2) \in Match_{1:3}(MN(S))$ do
5: Deduce the path matching pair $(p^l_k, p^l_t)$ using Eq.(1);
6: $P_{all} \leftarrow P_{all} \cup (p^l_k, p^l_t)$;
7: end for
8: end for
9: Generate the reliable path set $P$ using Eq. (2);
10: Generate the path triple set $T_{path}$ using Definition 2;
11: Return $P$ and $T_{path}$.

3.4 Relation-Aware Heterogeneous Graph Transformer (RHGT)

The process of Graph Transformer [Yun et al., 2019] aggregating all neighborhood features of node $h$ can be briefly expressed as:

$$e^{(l)}_h \leftarrow \text{Aggregate} \left( \text{Attention}(h, t) \cdot \text{Message}(h, t) \right),$$  (3)

where Attention is to estimate the importance of each neighborhood node; Message is to extract the feature of each neighborhood node; and Aggregate aggregates the neighborhood message through attention weights.

As shown in Eq.(3), Graph Transformer does not consider the edge features. Inspired by [Hu et al., 2020], we design a relation-aware heterogeneous graph transformer (RHGT), which enables our model to distinguish the heterogeneity features of relations and paths, to better obtain the neighborhood similarity of aligned entities. Let $E^{(l)}$ denote the output of $(l)$-th layer of RHGT, which is also the input of the $(l+1)$-th layer. Initially, $E^{(0)} = E^h$.

When the input of RHGT is the relation triples, the output is relation-based embeddings, and when the input is the path triples, the output is path-based embeddings. As shown in Figure 3, RHGT is mainly composed of four layers.

(a) Relation Embedding. Considering that the head entities and tail entities associated with aligned relations or aligned paths have certain similarities, we generate relation features by aggregating the features of associated entities. Specifically, the embedding of $r$ is approximated by averaging the embeddings of its associated head entities $H_r$ and associated tail entities $T_r$ as:

$$R^i(r) = \frac{\sum_{e^h_i \in H_r} b_r e^{(l-1)}_i}{|H_r|} + \frac{\sum_{e^t_i \in T_r} b_t e^{(l-1)}_i}{|T_r|},$$  (4)

where $|\cdot|$ indicates the size of collection; $b_r, b_t$ are the attention vectors; $\|\cdot\|$ denotes concatenation and $\sigma$ is ReLU function.

(b) Heterogeneous Attention. Inspired by [Hu et al., 2020], we map the entity $h$ into a key vector $K^i(h)$ and its neighborhood entity $t$ into a query vector $Q^i(t)$. The key difference from other methods is that instead of directly using the dot product of key and query vector as attention, we use the dot product between their concatenated result and $R^i(r)$. $R^i(r)$ comes from the feature aggregation of the associated head and tail entities (see Eq.(4)), so it will not deviate too far from the embeddings of its associated entities. Moreover, $R^i(r)$ denotes heterogeneous features of edges, so neighbors associated with different edges contribute differently to the entity $h$. Specifically, we compute the multi-head attention for each neighborhood relation $(h, r, t)$, as follows:

$$H_{Attention}(h, r, t) = \sum_{e^h_i \in H_h} \text{Softmax}(HATT_{head}(h, r, t)), $$

$$\text{HATT}_{head}(h, r, t) = \alpha^T \left( K^i(h) \| Q^i(t) \| R^i(r) \right) / \sqrt{d/|h|},$$  (5)

where $K^i(h) = K_{Linear}^i(e^{(l-1)}_h)$; $Q^i(t) = Q_{Linear}^i(e^{(l-1)}_t)$; $R_N(h)$ denotes the neighborhood of entity $h$; $a \in \mathbb{R}^{d/|h| \times 1}$ is the attention parameter; $|h|$ is the number of attention heads and $d/|h|$ is the vector dimension per head. Note that the Softmax process is to make the sum of attention vectors of all neighborhood entities equal to 1.

(c) Heterogeneous Message. Similarly, we hope to incorporate relations into the message passing process to distinguish the differences of different types of edges. For any $(h, r, t) \in T$, its multi-head message is computed as follows:

$$H_{Message}(h, r, t) = \sum_{i \in H_h} HMSG_{head}(h, r, t), $$

$$HMSG_{head}(h, r, t) = \sum_{i \in H_h} \left( V_{Linear}^i(e^{(l-1)}_h) \| R^i(r) \right).$$  (6)

To get the $i$-th head message $HMSG_{head}(h, r, t)$, we first apply a linear projection $V_{Linear}^i$ to project the features of tail entity $t$, and then concatenate the features of $t$ and relation $r$. The final heterogeneous message can be obtained by concatenating all $h_i$ message headers.

(d) Heterogeneous Aggregate. The final step is to aggregate heterogeneous multi-head attentions and messages of entities (see Figure 3 (c)), thereby aggregating the information from neighbors with different feature to entity $h$. The update vector $e^{(l)}_h$ of $h$ can be obtained simply by averaging the
corresponding messages from neighborhood entities with the attention coefficients as weights:

$$
\hat{e}_h^{(l)} = \sum_{(r,t) \in R(h,n)} \text{HAttention}(h,r,t) \cdot \text{HMessage}(h,r,t),
$$

(7)

where $\oplus$ denotes the overlay operation. To incorporate the name features and the features obtained by the multilayer neural network, the residual connection is used to generate the final updated embeddings as following:

$$
e_h^{(l)} = w_\beta A \cdot \text{Linear}(\hat{e}_h^{(l)}) + (1 - w_\beta) N \cdot \text{Linear}(e_h^{(l-1)}),
$$

(8)

where $w_\beta$ is trainable weights, $A \cdot \text{Linear}(\cdot)$, $N \cdot \text{Linear}(\cdot)$ are linear projections. Finally, we can generate relation-based embeddings $E_{rel}$ and path-based embeddings $E_{path}$ based on entire relation structure $T_{rel}$ and path structure $T_{path}$, respectively, and use them for end-to-end EA tasks.

3.5 Alignment Learning

After obtaining the final entity representations, we use Manhattan distance to measure the similarity of candidate entity pair. A smaller distance means a higher probability of entity alignment. The following function is used to compute the similarity of candidate entity pair based on $E_{rel}$ and $E_{path}$:

$$
d_f(e_i^1, e_j^1) = \|e_i^1 - e_j^1\|_{L_1},
$$

(9)

where $f = \{rel, path\}$; $L_1$ indicates the Manhattan distance.

To capture various aspects of the entities, previous methods usually concatenate the multi-source embeddings of entities and directly use them for the loss function. However, we argue that the contribution of relation-based and path-based embeddings to EA should be different, since these two structures of a entity may be quite diverse. Therefore, instead of directly using concatenated embeddings, we assign different weights to the loss functions of the two embeddings, thereby distinguishing their different contributions during training. In view of this, the following margin-based ranking loss function is used in model training, the goal of which is to keep the embedding distance of positive pair as small as possible and the embedding distance of negative pair as large as possible:

$$
\mathcal{L} = \sum_{(p,q) \in \mathbb{L}_r \setminus \mathbb{L}_{r,rel}} [d_{rel}(p,q) - d_{rel}(p',q') + \gamma_1]_+ + \theta(\sum_{(p,q) \in \mathbb{L}_p \setminus \mathbb{L}_{p,\text{path}}} [d_{path}(p,q) - d_{path}(p',q') + \gamma_2]_+),
$$

(10)

where $[\cdot]_+ = \max\{0, \cdot\}$; $\mathbb{L}_{r,rel}$ and $\mathbb{L}_{p,\text{path}}$ represent the negative pair of relation-based and path-based embeddings, respectively; $\gamma_1, \gamma_2 > 0$ are the margin hyper-parameters for separating positive and negative pairs, respectively.

4 Experiments

In this section, we evaluate the performance of \textit{RPR-RHGT} on three widely used benchmark datasets. The code is now available at https://github.com/cwswork/RPR-RHGT.

4.1 Experiment Settings

\textbf{Datasets.} Three experimental datasets contain cross-lingual datasets and mono-lingual dataset: WK31-15K [Sun et al., 2020b] is from multi-lingual DBpedia and used to evaluate model performance on sparse and dense datasets, where each subset contains two versions: V1 is sparse set obtained by using IDS algorithm, and V2 is twice as dense as V1. \textit{DBP-15K} [Sun et al., 2017] is the most used dataset in the literature, and is also from DBpedia. \textit{Dwy-100K} [Sun et al., 2018] contains two mono-lingual KGs, which serve as large-scale datasets to better evaluate the scalability of experimental models. Table 1 outlines the statistics of above datasets which also contains the numbers of paths and path triples generated by Algorithm 1, to demonstrate the effect of the \textit{RPR} module. Due to five-fold cross-validation used on \textit{WK31-15K} and \textit{DBP-15K}, the “Path.Paths” and “Path.Triplets” of these two datasets are the average statistic for the five training sets.

\textbf{Metrics.} Hits@$k$ is the proportion of correctly aligned ranked at the top-$k$ candidates; MRR (Mean Meciprocal Rank) is the average of the reciprocal ranks. Higher Hits@$k$ and MRR scores indicate better performance of EA.

\textbf{Baselines.} For \textit{WK31-15K} and \textit{DBP-15K}, we compare \textit{RPR-RHGT} with eight previous state-of-the-art alignment models (mentioned in Section 2): MTransE [Chen et al., 2017], IPTransE [Zhu et al., 2017], JAPE [Sun et al., 2017], BootEA [Sun et al., 2018], AttrE [Trivedya et al., 2019], RDGCN [Wu et al., 2019], NMN [Wu et al., 2020], RAGA [Zhu et al., 2021]. Since only a few models are evaluated on \textit{Dwy-100K}, we compare with the following models: MultiKE [Zhang et al., 2019b], RDGCN [Wu et al., 2019], NMN [Wu et al., 2020], COTSAE [Yang et al., 2020].

\textbf{Implementation Settings.} For \textit{WK31-15K} and \textit{DBP-15K}, the proportion of train, validation and test is 2:1:7, the same as [Sun et al., 2020b]. For \textit{Dwy-100K}, we adopt the same train (30%) / test (70%) split as baselines. We use fastText $^1$ to generate entity name embeddings that are uniformly applied to baseline recurrence, including RDGCN, NMN, RAGA, MultiKE and COTSAE. The embedding dimensions of 15K and 100K datasets are 300 and 200, respectively.

\begin{table}[h]
\centering
\begin{tabular}{|c|c|c|c|c|}
\hline
Datasets & KGs & Entities & Relation & Path \\
& & & Rel & Triples & Triples \\
\hline
\textit{JA-EN(DBP)} & JA & 65,744 & 2,043 & 164,373 & 139 & 283,311 \\
\textit{EN} & 95,680 & 2,096 & 233,319 & 172 & 559,984 \\
\textit{FR-EN(DBP)} & FR & 66,858 & 1,379 & 192,191 & 38 & 58,517 \\
\textit{EN} & 105,899 & 2,209 & 278,990 & 46 & 505,443 \\
\textit{ZH-EN(DBP)} & ZH & 66,469 & 2,830 & 153,929 & 140 & 166,991 \\
\textit{EN} & 98,125 & 2,317 & 237,674 & 46 & 436,418 \\
\hline
\textit{EN-DE(V1)} & EN & 15,000 & 215 & 47,676 & 13 & 12,393 \\
\textit{DE} & 15,000 & 131 & 50,419 & 17 & 18,153 \\
\hline
\textit{EN-DE(V2)} & EN & 15,000 & 169 & 84,867 & 38 & 58,517 \\
\textit{DE} & 15,000 & 96 & 92,632 & 38 & 77,243 \\
\hline
\textit{EN-FR(V1)} & EN & 15,000 & 267 & 47,334 & 46 & 51,349 \\
\textit{FR} & 15,000 & 210 & 40,864 & 46 & 50,504 \\
\hline
\textit{EN-FR(V2)} & EN & 15,000 & 193 & 96,318 & 80 & 379,112 \\
\textit{FR} & 15,000 & 166 & 80,112 & 80 & 294,751 \\
\hline
\textit{DBP-WD} & DBP & 100,000 & 330 & 463,294 & 460 & 1,834,831 \\
\textit{WD} & 100,000 & 220 & 448,774 & 460 & 2,709,929 \\
\hline
\textit{DBP-YG} & YG & 100,000 & 302 & 428,952 & 115 & 1,148,939 \\
\textit{BP} & 100,000 & 21 & 502,563 & 115 & 2,893,006 \\
\hline
\end{tabular}
\caption{Statistics of datasets.}
\end{table}

\footnotesize{$^1$https://fasttext.cc/docs/en/crawl-vectors.html}$
Table 2: Overall performances of all models on WK31-15K and DBP-15K. "*" marks the results obtained from OpenEA [Sun et al., 2020b]. Other results are produced using their source code.

| Datasets          | EN-DE(V1) | EN-DE(V2) | EN-FR(V1) | EN-FR(V2) |
|------------------|-----------|-----------|-----------|-----------|
|                  | Hits@1    | Hits@5    | Hits@1    | Hits@5    | Hits@1    | Hits@5    | Hits@1    | Hits@5    |
|                  | MRR        | MRR        | MRR        | MRR        | MRR        | MRR        |
| MTransE          | 30.7      | 51.8      | 19.3      | 35.2      | 27.4      | 24.7      | 46.7      | 35.1      |
| w/o.RPR          | 24.0      | 43.6      | .336      | .351      |
| *IPTransE        | 35.0      | 51.5      | 47.6      | 67.8      | 57.1      | 16.9      | 32.0      | 243      |
| *JAPE            | 28.8      | 51.2      | 39.4      | 61.7      | 25.0      | 26.2      | 49.7      | 37.2      |
| *BootEA          | 67.5      | 82.0      | 74.0      | 83.3      | 86.9      | 50.7      | 71.8      | 603      |
| *AttrE           | 51.7      | 68.7      | 59.7      | 65.0      | 76.2      | 48.1      | 67.1      | 569      |
| RDGCN            | 81.98     | 87.65     | 846      | 81.61     | 86.98     | 841      | 80.53     | 8766      |
| NMM              | 85.57     | 90.45     | 877      | 85.18     | 89.57     | 871      | 85.12     | 9074      |
| RAGA             | 87.90     | 94.28     | 908      | 81.34     | 89.15     | 849      | 82.71     | 9155      |
| AttrE            | 35.96     | 60.31     | .475      | 40.21     | 66.09     | .522      | 16.02     | 33.29     |
| w/o.RPR          | 20.4      | 40.52     | .303      | 19.74     | 40.37     | .297      | 20.89     | 42.09     |
| *IPTransE        | 27.92     | 52.70     | .396      | 31.22     | 57.42     | .434      | 17.34     | 37.05     |
| *JAPE            | 23.86     | 44.50     | .340      | 22.98     | 45.22     | .336      | 26.46     | 50.30     |
| *BootEA          | 52.71     | 71.89     | .616      | 57.61     | 77.27     | .666      | 55.45     | 73.72     |
| *AttrE           | 35.96     | 60.31     | .475      | 40.21     | 66.09     | .522      | 16.02     | 33.29     |
| RDGCN            | 81.22     | 87.98     | .844      | 80.88     | 88.08     | .842      | 62.11     | 73.88     |
| NMM              | 84.29     | 90.47     | .870      | 83.46     | 90.10     | .864      | 65.16     | 76.64     |
| RAGA             | 79.29     | 89.12     | .838      | 85.27     | 93.17     | .889      | 68.72     | 82.55     |

Table 3: Overall performance of all models on DWY-100K. All baseline performances are taken from their papers.

| Datasets          | DBP-WD | DBP-YG |
|------------------|--------|--------|
|                  | Hits@1 | Hits@10 | MRR | Hits@1 | Hits@10 | MRR |
| MultiKE          | 91.86  | 96.26   | .935 | 88.03  | 95.32   | .906 |
| RDGCN            | 97.90  | 99.10   | .945 | 94.39  | 98.74   | .961 |
| NMM              | 98.10  | 99.20   | .945 | 94.39  | 98.74   | .961 |
| COTSAE           | 92.68  | 97.86   | .945 | 94.39  | 98.74   | .961 |
| w/o.RPR          | 99.11  | 99.84   | .994 | 96.30  | 98.78   | .972 |
| *RPR-RHGT        | 99.26  | 99.36   | .995 | 96.58  | 98.86   | .974 |

Table 2: Overall performances of all models on WK31-15K and DBP-15K. "*" marks the results obtained from OpenEA [Sun et al., 2020b]. Other results are produced using their source code.

For all datasets, we use the same weight hyper-parameters: $\alpha_{sml} = 0.5, \tau_{path} = 20, h_n = 4, \gamma_1 = \gamma_2 = 10, \theta = 0.3$.

4.2 Main Results

Tables 2 and 3 report all performances on three datasets. The Hits@k is in percentage (%), while number in bold denotes the best result of all models and number in underline denotes the best result of baselines.

Results on WK31-15K. As shown in Table 2, RPR-RHGT achieves the best performance on WK31-15K, exceeding by 4.28%~8.62% on Hits@1. By reducing the numbers of relations and triples, WK31-15K challenges the ability of EA models to model sparse KGs. RPR-RHGT achieves significant improvements over the baselines on both sparse KGs and dense KGs. Besides, it is noteworthy that the improvements of RPR-RHGT on Hits@1 are much higher than that on Hits@5, indicating that RPR-RHGT can more accurately identify true entity among the top-5 indistinguishable alignment candidates. This experiment shows RPR-RHGT can compensate the neighborhood sparsity problem of some entities to a certain extent.

Results on DBP-15K. From observing Table 2, the Hits@1 of RPR-RHGT on DBP-15K is higher than the best baselines by 4.35%~0.58%, which indicates that our model performs best on all DBP-15K. It is noteworthy that the performance of RAGA on ZH-EN(DBP) is comparable to that of RPR-RHGT. We believe that ZH-EN(DBP) has more mismatched paths as one of the reasons. As shown in Table 1, ZH-EN(DBP) has more relations than other datasets, but no more reliable paths obtained by RPR algorithm. Besides, NMM is one of the best performing baselines and effectively captures the cross-graph information and relation information of KG, while RPR-RHGT still achieves good performance. Although the gap between RPR-RHGT and RAGA is smaller, RPR-RHGT has an advantage on DBP-15K.

Results on DWY-100K. As the largest dataset, DWY-100K raises challenges to the time and space complexity of EA models. As show in Table 3, although RPR-RHGT does not rely on attribute structures, it still outperforms all baselines on DWY-100K. Since DWY-100K is several times larger than other datasets, this experiment demonstrates that RPR-RHGT has good scalability and superiority in larger real-world and monolingual KGs.

4.3 Ablation Experiments

w/o.RPR is the RPR-RHGT without RPR module, the results of which are shown at the bottom of Tables 2 and 3. It
can be observed that w/o.RPR performs better than all baseline models on all datasets, except for ZH-EN(DBP), which confirms the effectiveness of RHGT design. Besides, RPR-RHGT achieves better performance than w/o.RPR across all metrics and datasets. This experiment confirms the assumption that the relational and path structure information of KGs can mutually reinforce each other.

4.4 Further Analysis

Sensitivity to Ratios of Pre-Aligned Entities. To explore the impact of pre-aligned entities on EA model training, we implement a further evaluation based on different ratios of training set. We take EN-DE(V1) and EN-DE(V2) as examples, and vary the ratio from 5% to 30%, while the validation dataset remains at 10%. RDGCN, NMN and RAGA are chosen as comparison models, all of which use name embeddings and perform best among baselines. As shown in Figure 4, our two models maintain consistent performance, significantly outperforming the baselines on training sets for all scales. This indicates that RPR-RHGT can achieve satisfactory results based on fewer pre-aligned entities.

Analysis on Harder Datasets. For a more objective evaluation of EA models, we take EN-DE(V1) and EN-DE(V2) as examples (called regular datasets), to construct two experimental datasets with relatively low similarities of entity names (called harder datasets). Specifically, we first compute the name embedding similarities of aligned entity pairs and rank them (low to high), then pick the highest-ranked 50% as the harder datasets, which are divided in the same way as above. To compare the effects of name embeddings on the performances of regular and harder datasets, we also compute the alignment accuracy of entity embeddings based only on their name embeddings without training, i.e., Regular(name) and Harder(name). As shown in Figure 5, the performances of all models based on name embeddings drop on harder datasets. However, comparing the performance on regular datasets, the performance of all models on harder dataset shows a more significant improvement over the performance of name embeddings. In particular, RPR-RHGT achieves up to 32.48% and 41.1% improvement over the name embeddings in Hits@1 on two harder datasets. This result demonstrates the robustness of RPR-RHGT, which can still promote effective EA on the datasets with less similar entity names.

Analysis on Training Time and Alignment Time. To evaluate the training and alignment efficiency of RPR-RHGT, we compare the training time and alignment time of the following four models on EN-DE(V1). The results running on a workstation with CPU (EPYC 3975WX +256G RAM) and GPU (RTX A4000 with 16G) are shown in Table 4, which shows large differences between different methods. Although the training time of RPR-RHGT is not optimal, its time complexity is competitive. Overall, our model balances well between effectiveness and efficiency.

5 Conclusions

Traditional GNNs either donot consider the heterogeneous information of KGs, or cannot effectively extract heterogeneous information that is effective for EA tasks. This paper proposes a new EA framework, RPR-RHGT, which focuses on mining reliable path information and heterogeneous information, thereby making full use of KGs’ own relation structures to improve alignment accuracy. First, we develop a RPR algorithm, which infers reliable paths from relation structures and only needs to be executed once. This algorithm is the first in the literature to successfully use unrestricted path information. Second, we improve a RHGT model for modeling the heterogeneity of KGs, to better capture the heterogeneous neighborhood similarity of aligned entities. Experimental results show RPR-RHGT not only outperforms state-of-the-art models, but also achieves better performance in multiple ablation studies and analysis experiments. In the future, we will continue to explore better ways to mine the heterogeneous information and path information of KGs for EA tasks.

Acknowledgments

The works described in this paper are supported by The National Natural Science Foundation of China under Grant Nos. 61772210 and U1911201; Guangdong Province Universities Pearl River Scholar Funded Scheme (2018); The Project of Science and Technology in Guangzhou in China under Grant No. 202007040006.
References

[Bordes et al., 2013] Antoine Bordes, Nicolas Usunier, Alberto García-Durán, Jason Weston, and Oksana Yakhnenko. Translating Embeddings for Modeling Multi-relational Data. In Proceedings of NeurIPS, 2013.

[Busbridge et al., 2019] Dan Busbridge, Dane Sherburn, Pietro Cavallo, and Nils Y. Hammerla. Relational Graph Attention Networks. In Proceedings of ICLR, 2019.

[Cai et al., 2022] Weishan Cai, Yizhao Wang, Shun Mao, Jieyu Zhan, and Yuncheng Jiang. Multi-heterogeneous Neighborhood-aware for Knowledge Graphs Alignment. Information Processing & Management, (1), 2022.

[Chen et al., 2017] Muhao Chen, Yingtao Tian, Mohan Yang, and Carlo Zaniolo. Multilingual Knowledge Graph Embeddings for Cross-lingual Knowledge Alignment. In Proceedings of IJCAI, 2017.

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

[Hu et al., 2020] Ziniu Hu, Yuxiao Dong, Kuansan Wang, and Yizhou Sun. Heterogeneous Graph Transformer. In Proceedings of WWW, 2020.

[Li et al., 2019] Yujia Li, Chenjie Gu, Thomas Dullien, Oriol Vinyals, and Pushmeet Kohli. Graph Matching Networks for Learning the Similarity of Graph Structured Objects. In Proceedings of ICML, 2019.

[Liu et al., 2020] Zhiyuan Liu, Xinyao Cao, Liangming Pan, Juanzi Li, Zhiyuan Liu, and Tat-Seng Chua. Exploring and Evaluating Attributes, Values, and Structures for Entity Alignment. In Proceedings of EMNLP, 2020.

[Liu et al., 2021] Fangyu Liu, Muhao Chen, Dan Roth, and Nigel Collier. Visual Pivoting for (Unsupervised) Entity Alignment. In Proceedings of AAAI, 2021.

[Mei et al., 2022] Xin Mei, Xiaoyan Cai, Libin Yang, and Nanxin Wang. Relation-aware Heterogeneous Graph Transformer based drug repurposing. Expert Systems with Applications, 2022.

[Schlichtkrull et al., 2018] Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne Van Der Berg, Ivan Titov, and Max Welling. Modeling relational data with graph convolutional networks. In Proceedings of ESWC, 2018.

[Sun et al., 2017] Zequn Sun, Wei Hu, and Chengkai Li. Cross-Lingual Entity Alignment via Joint Attribute-Preserving Embedding. In Proceedings of ISWC, 2017.

[Sun et al., 2018] Zequn Sun, Wei Hu, Qingheng Zhang, and Yuzhong Qu. Bootstrapping Entity Alignment with Knowledge Graph Embedding. In Proceedings of IJCAI, 2018.

[Sun et al., 2020a] Zequn Sun, Chengming Wang, Wei Hu, Muhao Chen, Jian Dai, Wei Zhang, and Yuzhong Qu. Knowledge Graph Alignment Network with Gated Multi-Hop Neighborhood Aggregation. In Proceedings of AAAI, 2020.

[Sun et al., 2020b] Zequn Sun, Qingheng Zhang, Wei Hu, Chengming Wang, Muhao Chen, Farahnaz Akrami, and Chengkai Li. A Benchmarking Study of Embedding-based Entity Alignment for Knowledge Graphs. In Proceedings of PVLDB, 2020.

[Trisedya et al., 2019] Bayu Distiawan Trisedya, Jianzhong Qi, and Rui Zhang. Entity Alignment between Knowledge Graphs Using Attribute Embeddings. In Proceedings of AAAI, 2019.

[Wang et al., 2019] Xiao Wang, Houye Ji, Chuan Shi, Bai Wang, Yanfeng Ye, Peng Cui, and Philip S Yu. Heterogeneous Graph Attention Network. In Proceedings of WWW, 2019.

[Wu et al., 2019] Yuting Wu, Xiao Liu, Yansong Feng, Zheng Wang, Rui Yan, and Dongyan Zhao. Relation-Aware Entity Alignment for Heterogeneous Knowledge Graphs. In Proceedings of IJCAI, 2019.

[Wu et al., 2020] Yuting Wu, Xiao Liu, Yansong Feng, Zheng Wang, and Dongyan Zhao. Neighborhood Matching Network for Entity Alignment. In Proceedings of ACL, 2020.

[Xiang et al., 2021] Yuejia Xiang, Ziheng Zhang, Jiaoyan Chen, Xi Chen, Zhenxi Lin, and Yefeng Zheng. OntoEA: Ontology-guided Entity Alignment via Joint Knowledge Graph Embedding. In Proceedings of ACL, 2021.

[Yang et al., 2019] Hsiu-Wei Yang, Yanyan Zou, Peng Shi, Wei Lu, Jimmy Lin, and Xu Sun. Aligning Cross-Lingual Entities with Multi-Aspect Information. In Proceedings of EMNLP-IJCNLP, 2019.

[Yang et al., 2020] Kai Yang, Shaoqin Liu, Junfeng Zhao, Yasha Wang, and Bing Xie. COTSAE: CO-Training of Structure and Attribute Embeddings for Entity Alignment. In Proceedings of AAAI, 2020.

[Yun et al., 2019] Seongjun Yun, Minbyul Jeong, Raehyun Kim, Jaewoo Kang, and Hyunwoo J Kim. Graph Transformer Networks. In Proceedings of NeurIPS, 2019.

[Zhang et al., 2019a] Chuxu Zhang, Dongjin Song, Chao Huang, Ananthram Swami, and Nitesh V. Chawla. Heterogeneous Graph Neural Network. In Proceedings of SIGKDD, 2019.

[Zhang et al., 2019b] Qingheng Zhang, Zequn Sun, Wei Hu, Muhao Chen, Lingbing Guo, and Yuzhong Qu. Multi-view Knowledge Graph Embedding for Entity Alignment. In Proceedings of IJCAI, 2019.

[Zhu et al., 2017a] Hao Zhu, Ruobing Xie, Zhiyuan Liu, and Maosong Sun. Iterative Entity Alignment via Joint Knowledge Embeddings. In Proceedings of IJCAI, 2017.

[Zhu et al., 2020] Qi Zhu, Hao Wei, Bunyamin Sisman, Da Zheng, Christos Faloutsos, Xin Luna Dong, and Jiawei Han. Collective Multi-type Entity Alignment Between Knowledge Graphs. In Proceedings of WWW, 2020.

[Zhu et al., 2021] Renbo Zhu, Meng Ma, and Ping Wang. RAGA: Relation-Aware Graph Attention Networks for Global Entity Alignment. In Proceedings of PAKDD, 2021.