Knowledge Graph Alignment Network with Gated Multi-hop Neighborhood Aggregation

Zequn Sun[1*], **Chengming Wang**[1*], **Wei Hu**[1†], **Muhao Chen**[2], **Jian Dai**[3], **Wei Zhang**[3], **Yuzhong Qu**[1]

[1]State Key Laboratory for Novel Software Technology, Nanjing University, China
[2]Department of Computer Science, University of California, Los Angeles, USA
[3]Alibaba Group, China
{zqsun,cmwang}.nju@gmail.com, {whu, yzqu}@nju.edu.cn, muhaochen@ucla.edu, {yiding.dj, lantu.zw}@alibaba-inc.com

Abstract

Graph neural networks (GNNs) have emerged as a powerful paradigm for embedding-based entity alignment due to their capability of identifying isomorphic subgraphs. However, in real knowledge graphs (KGs), the counterpart entities usually have non-isomorphic neighborhood structures, which easily causes GNNs to yield different representations for them. To tackle this problem, we propose a new KG alignment network, namely AliNet, aiming at mitigating the non-isomorphism of neighborhood structures in an end-to-end manner. As the direct neighbors of counterpart entities are usually dissimilar due to the schema heterogeneity, AliNet introduces distant neighbors to expand the overlap between their neighborhood structures. It employs an attention mechanism to highlight helpful distant neighbors and reduce noises. Then, it controls the aggregation of both direct and distant neighborhood information using a gating mechanism. We further propose a relation loss to refine entity representations. We perform thorough experiments with detailed ablation studies and analyses on five entity alignment datasets, demonstrating the effectiveness of AliNet.

1 Introduction

Entity alignment is the task of finding entities from different knowledge graphs (KGs) that refer to the same real-world identity. Recently, increasing attention has been paid to the utilization of KG representation learning rather than symbolic formalism for tackling this task. Representation learning models encode KGs into vector spaces, where relation semantics of entities can be assessed by the learned embedding operations, such as the relation-specific translation (Bordes et al. 2013) or rotation (Sun et al. 2019). For embedding-based entity alignment, the similarity of entities is measured by the distance of entity embeddings. It has shown great potentials in dealing with the symbolic heterogeneity problem and benefits the entity alignment task in both monolingual and cross-lingual scenarios (Zhu et al. 2017; Chen et al. 2017).

Most recently, graph neural networks (GNNs) (Kipf and Welling 2017; Velickovic et al. 2018; Abu-El-Haija et al.
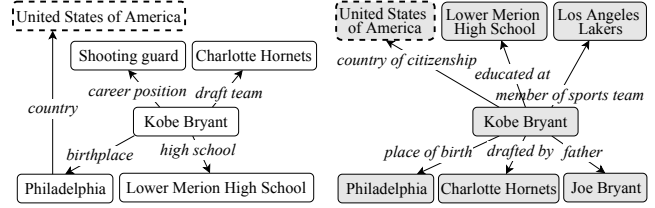

Figure 1: Non-isomorphic relational neighborhood of *Kobe Bryant* in DBpedia (left) and Wikidata (right), respectively.

2019) have emerged as a powerful model to learn vector representations for graph-structured data. In GNNs, the representation of a node is learned by recursively aggregating the representations of its neighboring nodes. A recent work (Morris et al. 2019) has proved that GNNs have the same expressiveness as the Weisfeiler-Leman (WL) test (Weisfeiler and Lehman 1968) in terms of identifying isomorphic subgraphs. It provides the theory basis of using GNNs for entity alignment between different KGs as similar entities usually have similar neighborhood. Recently, several studies (Wang et al. 2018; Xu et al. 2019b; Cao et al. 2019; Wu et al. 2019; Ye et al. 2019) have exploited GNNs for embedding-based entity alignment, and have achieved promising results.

However, existing GNN-based entity alignment models still face a critical problem. As different KGs usually have heterogeneous schemas and data incompleteness (Pujara et al. 2013), the counterpart entities usually have dissimilar neighborhood structures. Figure 1 gives an example. The neighborhood of the two entities referring to *Kobe Bryant* is inconsistent to each other, especially containing different sets of neighboring entities. The statistics on DBpedia-based benchmark datasets for entity alignment (Sun, Hu, and Li 2017) also show that the majority of aligned entity pairs have different neighboring entities. Particularly, the percentages of such entity pairs reach 89.97% between Chinese-English, 86.19% between Japanese-English and 90.71% between French-English, respectively. Different neighborhood structures would easily cause a GNN to yield different representations for counterpart entities.

The challenge of resolving this issue lies in the difficulty

of fully mitigating the non-isomorphism in the neighborhood structures of counterpart entities from different KGs. Even though we assume that the two KGs are complete (the goal of MuGNN (Cao et al. 2019)), due to the schema heterogeneity, the counterpart entities still inevitably have dissimilar neighborhood structures. For example, in Figure 1, *United States of America* is among the one-hop (direct) neighbors of *Kobe Bryant* in Wikidata. However in DBpedia, it is a two-hop neighbor. Motivated by the fact that the semantically-related information can appear in both direct and distant neighbors of counterpart entities, we propose the KG alignment network AliNet which aggregates both direct and distant neighborhood information. Specifically, each AliNet layer has multiple functions to aggregate the neighborhood information within multiple hops. To reduce noise information, we further employ an attention mechanism for the distant neighborhood aggregation to find out important neighbors in an end-to-end manner. Finally, we use the gating mechanism to combine the output representations of the multiple aggregation functions, obtaining the hidden representations in the current layer. We also design a relation loss to refine entity representations and enable AliNet to capture some special structures such as the triangular relational structure. We perform thorough experiments with detailed ablation studies and analyses on five entity alignment datasets, demonstrating the effectiveness of AliNet and each of its technical contributions.

2 Preliminaries

2.1 GNNs

In GNNs, the representation of a node is learned by recursively aggregating the feature vectors of its neighbors. Different aggregation strategies lead to different variants of GNNs.

GCN A very popular variant of GNNs is the vanilla GCN (Kipf and Welling 2017). The hidden representation of node i at the l-th layer ($l \geq 1$), denoted as $\mathbf{h}_i^{(l)}$, is computed by:

$$\mathbf{h}_i^{(l)} = \sigma\Big(\sum_{j \in \mathcal{N}_1(i) \cup \{i\}} \frac{1}{c_i} \mathbf{W}^{(l)} \mathbf{h}_j^{(l-1)} \Big), \qquad (1)$$

where $\mathcal{N}_1(\cdot)$ represents the set of one-hop neighbors of the given entity, $\mathbf{W}^{(l)}$ is the weight matrix of the l-th layer and c_i is the normalization constant. $\sigma(\cdot)$ is an activation function. The vanilla GCN encodes a node as the *mean pooling* of the representations of its neighbors and itself from the last layer. The input vector fed to the first layer is denoted as $\mathbf{h}_i^{(0)}$.

R-GCN Conventional GNNs only consider the node-wise connectivity in a graph and ignore edge labels such as the relations in KGs. R-GCN (Schlichtkrull et al. 2018) addresses this issue by distinguishing different neighbors with relation-specific weight matrices. It computes $\mathbf{h}_i^{(l)}$ as follows:

$$\mathbf{h}_i^{(l)} = \sigma\Big(\mathbf{W}_0^{(l)} \mathbf{h}_i^{(l-1)} + \sum_{r \in \mathcal{R}} \sum_{j \in \mathcal{N}_r(i)} \frac{1}{c_{i,r}} \mathbf{W}_r^{(l)} \mathbf{h}_j^{(l-1)} \Big), \quad (2)$$

where $\mathbf{W}_0^{(l)}$ is the weight matrix for the node itself and $\mathbf{W}_r^{(l)}$ is used specifically for the neighbors having relation r, i.e., $\mathcal{N}_r(i)$. $\mathcal{R}$ is the relation set and $c_{i,r}$ is for normalization.

2.2 Entity Alignment of KGs

We formally represent a KG as $\mathcal{G} = (\mathcal{E}, \mathcal{R}, \mathcal{T})$, where $\mathcal{E}$ is the set of entities, $\mathcal{R}$ is the set of relations, and $\mathcal{T} = \mathcal{E} \times \mathcal{R} \times \mathcal{E}$ is the set of triples. Without loss of generality, we consider the entity alignment task between two KGs, i.e., $\mathcal{G}_1 = (\mathcal{E}_1, \mathcal{R}_1, \mathcal{T}_1)$ and $\mathcal{G}_2 = (\mathcal{E}_2, \mathcal{R}_2, \mathcal{T}_2)$. Given partial pre-aligned entity pairs $\mathcal{A}^+ = \{(i, j) \in \mathcal{E}_1 \times \mathcal{E}_2 | i \equiv j\}$ where $\equiv$ means the alignment relationship, the goal of the task is to find alignment of remaining entities via entity embeddings.

2.3 GNNs for Entity Alignment

Recent GNN-based entity alignment models include GCN-Align (Wang et al. 2018), GMNN (Xu et al. 2019b), MuGNN (Cao et al. 2019), RDGCN (Wu et al. 2019) and AVR-GCN (Ye et al. 2019). GCN-Align and GMNN are built based on the vanilla GCN. RDGCN introduces dual relation graphs to enhance the vanilla GCN. AVR-GCN extends R-GCN using a TransE-like relation-specific translation operation (Bordes et al. 2013). Before aggregation, each entity representation is translated from its tail entity representations using relation vectors. We argue that such relation-specific translation and R-GCN introduce a high complexity with the overhead of trainable parameters. More importantly, the aforementioned models do not take the non-isomorphism in KG structures into consideration. While MuGNN (Cao et al. 2019) notices the structure incompleteness of KGs and proposes a two-step method of rule-based KG completion and multi-channel GNNs for entity alignment. However, the learned rules rely on relation alignment to resolve schema heterogeneity.

Isomorphic structures are beneficial GNNs would learn the same representation for the entities that have isomorphic neighborhood structures with identical feature vectors representing corresponding neighbors (Xu et al. 2019a). We show that, in some cases, if two entities have isomorphic neighborhood structures and only partially pre-aligned neighbor representations, GNNs can also capture the similarity of other neighbors to be aligned. Figure 2 (i) gives an example. For simplicity, here we consider a single-layer GCN. We can let pre-aligned entities have the same representation by minimizing their Euclidean distance, i.e., $\mathbf{h}_a^{(0)} = \mathbf{h}_{a'}^{(0)}$, $\mathbf{h}_b^{(0)} = \mathbf{h}_{b'}^{(0)}$ and $\mathbf{h}_d^{(0)} = \mathbf{h}_{d'}^{(0)}$ as well as $\mathbf{h}_a^{(1)} = \mathbf{h}_{a'}^{(1)}$, $\mathbf{h}_b^{(1)} = \mathbf{h}_{b'}^{(1)}$ and $\mathbf{h}_d^{(1)} = \mathbf{h}_{d'}^{(1)}$ in the ideal condition. By the mean-pooling based aggregation, we have $\mathbf{h}_b^{(1)} = \sigma(\mathbf{W}^{(1)}(\mathbf{h}_b^{(0)} + \mathbf{h}_a^{(0)} + \mathbf{h}_c^{(0)})/3)$ and $\mathbf{h}_{b'}^{(1)} = \sigma(\mathbf{W}^{(1)}(\mathbf{h}_{b'}^{(0)} + \mathbf{h}_{a'}^{(0)} + \mathbf{h}_{c'}^{(0)})/3)$, yielding $\mathbf{h}_c^{(0)} = \mathbf{h}_{c'}^{(0)}$. Finally, the counterpart entities would have the same representation. This indicates that the alignment information between entities can be propagated across the different GNN layers and different isomorphic graphs given partially pre-aligned neighborhood. However, for entity alignment between different KGs, it is impossible to require the two KGs to have isomorphic structures due to the schema heterogeneity. Figure 2 (ii) gives an example of non-isomorphic graph structures, where c and c' would have different representations due to their different neighborhood structures.

Only structures are not enough Conventional GNNs fall short of characterizing some special subgraph structures such
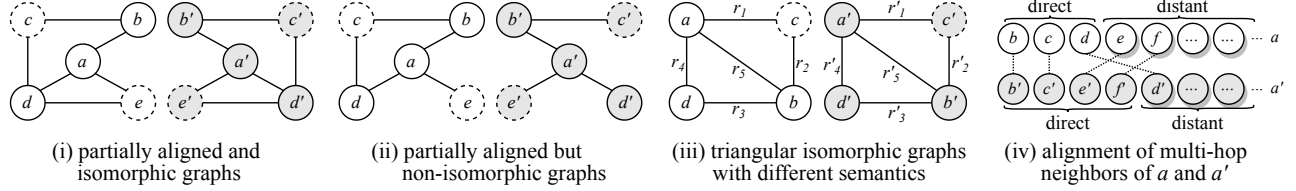
Figure 2: Illustration of GNNs for entity alignment. In (i), (ii) and (iii), (a, a'), (b, b') and (d, d') denote three pairs of pre-aligned entities while others are entities to be aligned. The dotted lines in (iv) means the alignment relationship.

as triangular graphs. Figure 2 (iii) shows a simple example. In this case, if we use mean-pooling aggregation, we would get that $\mathbf{h}_a^{(1)} = \mathbf{h}_{a'}^{(1)} = \mathbf{h}_b^{(1)} = \mathbf{h}_{b'}^{(1)}$ because the four entities have isomorphic neighborhood structures. While in fact, a and b are different entities with a specific relation. We should take relations into consideration. Although R-GCN (Schlichtkrull et al. 2018) considers relations in the aggregation function, it relies on relation alignment for identifying similar entities. Let us review Eq. (2). R-GCN needs to learn a weight matrix $\mathbf{W}_r$ for each relation r. If the relations of two KGs are not pre-aligned (e.g., $r_1 \equiv r'_1$ and $r_2 \equiv r'_2$), the relation-specific aggregation functions in R-GCN would fall short of propagating the alignment information of entities.

Compensation with distant neighborhood and relations
The schema heterogeneity of different KGs usually brings about the mixture of direct and distant neighbors of counterpart entities. To reduce the effects of the non-isomorphism in neighborhood structures, we propose introducing distant neighborhood information. We show a toy example in Figure 2 (iv). The one-hop neighbors of two counterpart entities a and a' are different and only contain two pairs of counterpart entities (b, b') and (c, c'). The one-hop neighbor d of a is in fact the distant neighbor d' of a'. The distant neighbors e and f of a are aligned with the one-hop neighbors e' and f' of a', respectively. It is intuitive that if we can include the distant neighbors e and f in the neighborhood aggregation for a, and also take d' into consideration for a', the GNN would learn more similar representations for a and a'. However, as can be seen, not all the distant neighbors are helpful. Therefore, the aggregation of distant neighbors should be attentive and selective. This is the key motivation of AliNet. To further enhance the expressiveness of AliNet, we also take relation semantics into consideration without introducing relation vectors.

3 Knowledge Graph Alignment Network

In AliNet, the entity representations are learned by a controlled aggregation of their neighborhood information within k hops by the gating mechanism. Without loss of generality, in the following, we show the case of aggregating both the one-hop and two-hop neighborhood information ($k = 2$). The network architecture is illustrated in Figure 3. Note that AliNet can also be extended to more hops.

3.1 Gated Multi-hop Neighborhood Aggregation

The one-hop neighbors of an entity are the most important neighborhood for GNNs to characterize the entity. We aggre-

gate these neighbor representations using the vanilla GCN layers. Specifically, at the i-th layer, the hidden representation of entity i by aggregating its one-hop neighbors, denoted as $\mathbf{h}_{i,1}^{(l)}$, can be computed using Eq. (1).

As discussed before, it is not enough to only aggregate one-hop neighbors. Although a GCN with L layers can capture the structural information within the entitys L-hop neighbors, such layer-by-layer propagation is not efficient. For two-hop neighborhood aggregation, we introduce the attention mechanism because directly employing the original aggregation of GCN would cause noise information to propagate through layers. Specifically, let $\mathcal{N}_2(\cdot)$ be the set of two-hop neighbors of the given entity. The hidden representation of entity i by aggregating its two-hop neighborhood information at the l-th layer, denoted as $\mathbf{h}_{i,2}^{(l)}$, is computed as follows:

$$\mathbf{h}_{i,2}^{(l)} = \sigma\Big(\sum_{j \in \mathcal{N}_2(i) \cup \{i\}} \alpha_{ij}^{(l)} \mathbf{W}_2^{(l)} \mathbf{h}_j^{(l-1)} \Big), \quad (3)$$

where $\alpha_{ij}^{(l)}$ is a learnable attention weight for entity i and its neighbor j. $\mathbf{W}_2^{(l)}$ is the weight matrix. The computation of attention weights is introduced in the next subsection.

Inspired by the skipping connections in neural networks (Srivastava, Greff, and Schmidhuber 2015; He et al. 2016; Guo, Sun, and Hu 2019). We propose to use the gating mechanism to combine the information from one-hop and two-hop neighbors directly. Specifically, the hidden representation $\mathbf{h}_i^{(l)}$ of entity i at the l-th layer is computed as follows:

$$\mathbf{h}_i^{(l)} = g(\mathbf{h}_{i,2}^{(l)}) \cdot \mathbf{h}_{i,1}^{(l)} + (1 - g(\mathbf{h}_{i,2}^{(l)})) \cdot \mathbf{h}_{i,2}^{(l)}, \quad (4)$$

where $g(\mathbf{h}_{i,2}^{(l)}) = \sigma(\mathbf{M}\mathbf{h}_{i,2}^{(l)} + \mathbf{b})$ serves as the gate to control the combination of both one-hop and two-hop neighborhood. $\mathbf{M}$ and $\mathbf{b}$ are the weight matrix and bias vector, respectively.

3.2 Attention for Distant Neighborhood

The number of the more distant neighbors of an entity can grow exponentially than the number of its one-hop neighbors. It is intuitive that not all the distant neighbors contribute to the characterization of the central entity. Hence, for two-hop neighborhood aggregation, we compute the attention weights between entities for highlighting useful neighbors. The graph attention network GAT (Velickovic et al. 2018) applies a shared linear transformation to entities in each attention function. However, as the central entity and its neighbors in KGs can be quite different, such shared transformation would
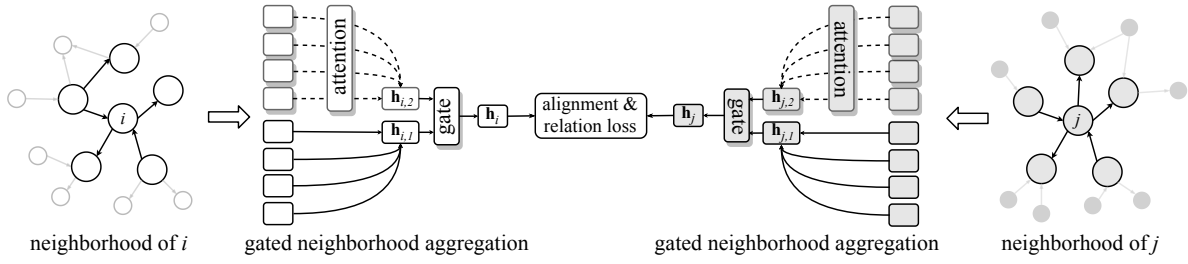
Figure 3: Overview of the KG alignment network (AliNet) with gated two-hop neighborhood aggregation.

cause a deleterious effect to correctly distinguishing between them. Instead, we use two matrices $\mathbf{M}_1^{(l)}$ and $\mathbf{M}_2^{(l)}$ for the linear transformations of the central entity and its neighbors, respectively. Formally, the attention weight $c_{ij}^{(l)} \in \mathbb{R}$ between i and j at the l-th layer is computed as follows:

$$c_{ij}^{(l)} = \text{LeaklyReLU}[(\mathbf{M}_1^{(l)}\mathbf{h}_i^{(l)})^\top (\mathbf{M}_2^{(l)}\mathbf{h}_j^{(l)})], \quad (5)$$

Finally, we normalize attention weights using the softmax function to make them comparable across different entities:

$$\alpha_{ij}^{(l)} = \text{softmax}_j(c_{ij}^{(l)}) = \frac{\exp(c_{ij}^{(l)})}{\sum_{n \in \mathcal{N}_2(i) \cup \{i\}} \exp(c_{in}^{(l)})}. \quad (6)$$

3.3 Contrastive Alignment Loss

We minimize the contrastive alignment loss to let the representations of aligned entities have a very small distance while those of unaligned entities have a large distance:

$$\mathcal{L}_1 = \sum_{(i,j) \in \mathcal{A}^+} ||\mathbf{h}_i - \mathbf{h}_j|| + \sum_{(i',j') \in \mathcal{A}^-} \alpha_1 [\lambda - ||\mathbf{h}_{i'} - \mathbf{h}_{j'}||]_+, \quad (7)$$

where $\mathcal{A}^-$ is the set of negative samples generated by randomly substituting one of the two pre-aligned entities. $|| \cdot ||$ denotes the L_2 vector norm. $[\cdot]_+ = \max(0, \cdot)$ The distance of negative samples is expected to be larger than a margin λ, i.e., $||\mathbf{h}_{i'} - \mathbf{h}_{j'}|| > \lambda$. α_1 is a hyper-parameter for balance.

Previous work usually uses the hidden outputs at the last layer as the final representations of entities, i.e. $\mathbf{h}_i = \mathbf{h}_i^{(L)}$ where L denotes the number of layers. However, as discussed in Section 2, the representations of each layer all contribute to propagating alignment information. Therefore, we use the hidden representations of all layers. Formally, we have

$$\mathbf{h}_i = \bigoplus_{l=1}^{L} \text{norm}(\mathbf{h}_i^{(l)}), \quad (8)$$

where $\oplus$ represents concatenation and $\text{norm}(\cdot)$ is the L_2 normalization for reducing the trivial optimization procedure of artificially increasing vector norm (Bordes et al. 2013).

3.4 Relation Semantics Modeling

As KGs provide semantic relations between entities, it is natural to incorporate the semantics of the relational facts into entity modeling. As discussed in Section 2, R-GCN needs the structures of two KGs to be highly similar or relation

alignment for entity alignment. Here, we borrow the translational assumption from TransE (Bordes et al. 2013). To avoid overhead of parameters, we do not introduce additional relation-specific embeddings. The representation for r, denoted as $\mathbf{r}$, can be retrieved via its related entity embeddings:

$$\mathbf{r} = \frac{1}{|\mathcal{T}_r|} \sum_{(s,o) \in \mathcal{T}_r} (\mathbf{h}_s - \mathbf{h}_o), \quad (9)$$

where $\mathcal{T}_r$ is the subject-object entity pairs of relation r. Then we minimize the following relation loss for refinement:

$$\mathcal{L}_2 = \sum_{r \in \mathcal{R}} \frac{1}{|\mathcal{T}_r|} \sum_{(s,o) \in \mathcal{T}_r} ||\mathbf{h}_s - \mathbf{h}_o - \mathbf{r}||, \quad (10)$$

where $\mathcal{R}$ is the set of the total relations in the two KGs.

3.5 Implementation

Next, we introduce implementation details of AliNet.

Objective The final objective of AliNet is the combination of the contrastive alignment loss and relation loss, aiming at injecting relation semantics to the preserved graph structures:

$$\mathcal{L} = \mathcal{L}_1 + \alpha_2 \mathcal{L}_2, \quad (11)$$

where α_2 is a hyper-parameter to weight the two losses. The objective is optimized using the Adam optimizer. All the learnable parameters including the input feature vectors of entities are initialized by the Xavier initialization (Glorot and Bengio 2010). The adjacency information is a sparse matrix obtained from the relational triples $\mathcal{T}_1$ and $\mathcal{T}_2$. The neighborhood aggregation can be done by the sparse matrix multiplication between the adjacency matrix and the entity representation matrix, making the storage complexity linear to the number of entities and triples.

Generalization to k-hop neighborhood Here we consider aggregating the neighborhood information within k hops. Let $\rho_1(\mathbf{h}_{i,1}^{(l)}, \mathbf{h}_{i,2}^{(l)})$ be the gating combination for the one-hop and two-hop neighborhood aggregation in Eq. (4). We use $k-1$ gating functions to combine the information recursively:

$$\mathbf{h}_i^{(l)} = \rho_{k-1}(\cdots \rho_2(\rho_1(\mathbf{h}_{i,1}^{(l)}, \mathbf{h}_{i,2}^{(l)}), \mathbf{h}_{i,3}^{(l)}) \cdots). \quad (12)$$

Neighborhood augmentation The proposed gated multi-hop neighborhood aggregation expands the direct neighbors of an entity in an end-to-end manner. To further implement this idea, we propose a heuristic method to add edges among

pre-aligned entities. Specifically, if two entities i and j of KG_1 have an edge while their counterparts i' and j' in KG_2 do not, we add an edge linking i' and j'. The goal is to mitigate the non-isomorphism by adding such balanced edges.

Alignment prediction   Once trained AliNet, we can predict entity alignment based on the nearest neighbor search among entity representations in the cross-KG scope. Given a source entity i to be aligned in KG_1, its counterpart in KG_2 is: $i' = \arg\min_{j \in \mathcal{E}_2} \pi(\mathbf{h}_i, \mathbf{h}_j)$, where $\pi()$ is a distance measure such as Euclidean distance. Here we still use the combined representations to measure the distance of entity embeddings.

4   Experiments

In this section, we evaluate AliNet on the entity alignment task. The source code of AliNet is accessible online[1].

4.1   Datasets

Following the latest progress (Sun et al. 2018; Cao et al. 2019), we use the following datasets and training-test splits.

- DBP15K (Sun, Hu, and Li 2017) has three datasets built from multi-lingual DBpedia, namely DBP$_{\text{ZH-EN}}$ (Chinese-English), DBP$_{\text{JA-EN}}$ (Japanese-English) and DBP$_{\text{FR-EN}}$ (French-English). Each dataset has $15,000$ reference entity alignment and about four hundred thousand triples.

- DWY100K (Sun et al. 2018) are extracted from DBpedia, Wikidata and YAGO3. It has two datasets, namely DBP-WD (DBpedia-Wikidata) and DBP-YG (DBpedia-YAGO3). Each dataset has $100,000$ reference entity alignment and more than nine hundred thousand triples.

4.2   Comparative Models

We compare with recent embedding-based entity alignment models: MTransE (Chen et al. 2017), IPTransE (Zhu et al. 2017), JAPE (Sun, Hu, and Li 2017), AlignE (Sun et al. 2018), GCN-Align (Wang et al. 2018), SEA (Pei et al. 2019), RSN (Guo, Sun, and Hu 2019) and MuGNN (Cao et al. 2019). Note that some recent GNN-based models like GMNN (Xu et al. 2019b) and RDGCN (Wu et al. 2019) incorporate the surface information of entities into their representations. As our model solely relies on structure information, we do not take these models into comparison. For ablation study, we develop three variants of AliNet, i.e., AliNet (w/o rel. loss) that does not optimize the relation loss, AliNet (w/o rel. loss & augment.) that does not employ the relation loss and neighborhood augmentation, and the full model AliNet.

For comprehensive comparison, we also choose some KG embedding models and GNN variants as baselines. Conventional KG embedding models are usually evaluated on the task of link prediction. However, as studied in (Zhang et al. 2019), some of them can also be used for entity alignment. We select TransH (Wang et al. 2014), ConvE (Dettmers et al. 2018) and RotatE (Sun et al. 2019). TransE (Bordes et al. 2013) has already been exploited for entity alignment by MTransE and IPTransE. For GNNs, we choose GCN (Kipf and Welling 2017), GAT (Velickovic et al. 2018) and R-CGN

[1]https://github.com/nju-websoft/AliNet

(Schlichtkrull et al. 2018) as baselines. The re-tuned versions of GCN, GAT and R-GCN are implemented by ourselves following the same pipeline as AliNet for fair comparison.

4.3   Implementation Details

We search among the following values for hyper-parameters, i.e., the learning rate in $\{0.0001, 0.0005, 0.001, 0.005, 0.01\}$, α_1 in $\{0.1, 0.2, \ldots, 0.5\}$, α_2 in $\{0.01, 0.05, 0.1, 0.2\}$, λ in $\{1.0, 1.1, \ldots, 2.0\}$, the hidden representation dimension of each layer in $\{100, 200, 300, 400, 500\}$, the number of layers L in $\{1, 2, 3, 4\}$, and the number of negative alignment pairs in $\{5, 10, 15, 20\}$. The selected setting is that $\lambda = 1.5$, $\alpha_1 = 0.1$, $\alpha_2 = 0.01$. The learning rate is 0.001. The batch size for DBP15K is $4,500$, and for DWY100K is $10,000$. We stack two AliNet layers ($L = 2$) and each layer combines the one-hop and two-hop information ($k = 2$). The dimensions of three layers (including the input layer) are 500, 400 and 300, respectively. The activation function for neighborhood aggregation is tanh(), and the one for the gating mechanism is ReLU(). We sample 10 negative samples for each pre-aligned entity pair. We use early stopping to terminate training based on the Hits@1 performance with a patience of 5 epochs. We use CSLS (Conneau et al. 2018) for nearest neighbor search.

Following convention, we report the Hits@1, Hits@10 and MRR (mean reciprocal rank) results to assess entity alignment performance. Higher Hits@1, Hits@10 and MRR scores indicate better performance. Note that, the Hits@1 is equivalent to precision. As the nearest neighbor search can always find a counterpart for each entity to be aligned, the recall and F1-measure also have the same value as Hits@1.

4.4   Main Results

We present the entity alignment results in Table 1. We can see that AliNet outperforms the state-of-the-art structure-based embedding models for entity alignment by Hits@1 and MRR. For example, on DBP$_{\text{FR-EN}}$, AliNet achieves a gain of 0.036 by Hits@1 compared with RSN, and 0.057 against MuGNN. We think that these results have demonstrated the superiority of AliNet. As the DBP15K datasets are extracted from the multi-lingual DBpedia, the schema heterogeneity of them is much weaker than that of DWY100K which are extracted from different KGs. AliNet also achieves the best Hits@10 results on DWY100K, demonstrating its practicability. We find that the neighborhood augmentation method leads AliNet a gain of $0.012 - 0.037$ by Hits@1. This is because it can reduce the non-isomorphism in the neighborhood structures of pre-aligned entities. The results further support our motivation of mitigating the non-isomorphism in KG structures for entity alignment. AliNet shows better performance than AliNet (w/o rel. loss), showing the effect of the relation loss.

In comparison to the re-tuned GNN variants GCN, GAT and R-GCN, AliNet also achieves better performance. As the GCN baseline has the same training process as AliNet and the difference only lies in the choice of GNN layers (i.e., GCN layers for one-hop neighborhood aggregation or AliNet layers for both one-hop and two-hop neighborhood aggregation), these results can demonstrate the effectiveness of integrating multi-hop information. Both GAT and R-GCN fail to outperform GCN. We attribute such observation to that

Methods	DBP$_{\text{ZH-EN}}$			DBP$_{\text{JA-EN}}$			DBP$_{\text{FR-EN}}$			DBP-WD			DBP-YG		
	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10	MRR
MTransE (Chen et al. 2017)	0.308	0.614	0.364	0.279	0.575	0.349	0.244	0.556	0.335	0.281	0.520	0.363	0.252	0.493	0.334
IPTransE (Zhu et al. 2017)	0.406	0.735	0.516	0.367	0.693	0.474	0.333	0.685	0.451	0.349	0.638	0.447	0.297	0.558	0.386
JAPE (Sun, Hu, and Li 2017)	0.412	0.745	0.490	0.363	0.685	0.476	0.324	0.667	0.430	0.318	0.589	0.411	0.236	0.484	0.320
AlignE (Sun et al. 2018)	0.472	0.792	0.581	0.448	0.789	0.563	0.481	0.824	0.599	0.566	0.827	0.655	0.633	0.848	0.707
GCN-Align (Wang et al. 2018)	0.413	0.744	0.549	0.399	0.745	0.546	0.373	0.745	0.532	0.506	0.772	0.600	0.597	0.838	0.682
SEA (Pei et al. 2019)	0.424	0.796	0.548	0.385	0.783	0.518	0.400	0.797	0.533	0.518	0.802	0.616	0.516	0.736	0.592
RSN (Guo, Sun, and Hu 2019)	0.508	0.745	0.591	0.507	0.737	0.590	0.516	0.768	0.605	0.607	0.793	0.673	0.689	0.878	0.756
MuGCN (Cao et al. 2019)	0.494	**0.844**	0.611	0.501	**0.857**	0.621	0.495	**0.870**	0.621	0.616	0.897	0.714	0.741	0.937	0.810
TransH (Wang et al. 2014)	0.377	0.711	0.490	0.339	0.681	0.462	0.313	0.668	0.433	0.351	0.641	0.450	0.314	0.574	0.402
ConvE (Dettmers et al. 2018)	0.169	0.329	0.224	0.192	0.343	0.246	0.240	0.459	0.316	0.403	0.628	0.483	0.503	0.736	0.582
RotatE (Sun et al. 2019)	0.485	0.788	0.589	0.442	0.761	0.550	0.345	0.738	0.476	0.479	0.776	0.579	0.599	0.835	0.680
GCN (Kipf and Welling 2017)	0.487	0.790	0.559	0.507	0.805	0.618	0.508	0.808	0.628	0.613	0.850	0.698	0.733	0.909	0.796
GAT (Velickovic et al. 2018)	0.418	0.667	0.508	0.446	0.695	0.537	0.442	0.731	0.546	0.540	0.781	0.625	0.563	0.806	0.648
R-GCN (Schlichtkrull et al. 2018)	0.463	0.734	0.564	0.471	0.754	0.571	0.469	0.758	0.570	0.574	0.791	0.651	0.617	0.829	0.692
AliNet (w/o rel. loss & augment)	0.511	0.798	0.611	0.527	0.794	0.622	0.520	0.848	0.635	0.642	0.877	0.726	0.745	0.918	0.806
AliNet (w/o rel. loss)	0.525	0.790	0.619	0.539	0.796	0.638	0.535	0.839	0.645	0.679	0.887	0.750	0.773	0.935	0.832
AliNet	**0.539**	0.826	**0.628**	**0.549**	0.831	**0.645**	**0.552**	0.852	**0.657**	**0.690**	**0.908**	**0.766**	**0.786**	**0.943**	**0.841**

Table 1: Result comparison on entity alignment

the direct neighbors of an entity are less dissimilar than distant neighbors, and hence may not require an attention-based neighborhood aggregation to select relevant neighboring entities. This is also the reason for choosing GCN layers rather than GAT layers in the one-hop neighborhood aggregation of AliNet. For R-GCN, as discussed in Section 2, it cannot well capture the similarity of neighborhood structures of counterpart entities. We also observe that the GCN baseline outperforms many other embedding-based entity alignment models including another GCN variant GCN-Align. This again validates the effectiveness of our framework.

4.5 Analyses

Aggregation strategies of multi-hop neighborhood The underlying idea of AliNet is to extend the neighborhood of entities by attentively aggregating multi-hop neighborhood with the gating mechanism. To gain a deep insight into this point, we further design three variants of AliNet using different strategies to aggregate multi-hop neighborhood. The first one, denoted as AliNet (mix), borrows the idea from MixHop (Abu-El-Haija et al. 2019), which has a similar motivation for node classification in general graphs. It takes the two-hop neighbors as one-hop and uses GCN layers to directly aggregate such mixed neighborhood information. The second one, denoted as AliNet (add), replaces the gating mechanism with addition operator. In the last variant AliNet (gat), we replace the proposed attention mechanism with GAT (Velickovic et al. 2018). Due to space limitation, we only show the results on DBP15K in Table 2. We find that AliNet (mix) fails to achieve promising performance, which indicates that using GCN layers for two-hop neighborhood aggregation is not effective because it would introduce much noise information. AliNet (add) does not show very satisfactory results because addition cannot selectively combine important representations across dimensions like the gating mechanism. AliNet (gat) also achieves slightly lower performance than AliNet, showing the effect of our attention mechanism. Based on these results and the performance of the GCN baseline shown in Table 1, we can come to the conclusion that the

Methods	DBP$_{\text{ZH-EN}}$			DBP$_{\text{JA-EN}}$			DBP$_{\text{FR-EN}}$		
	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10	MRR
AliNet (mix)	0.227	0.611	0.350	0.294	0.696	0.426	0.258	0.674	0.391
AliNet (add)	0.498	0.801	0.602	0.515	0.813	0.618	0.501	0.839	0.585
AliNet (gat)	0.517	0.803	0.618	0.531	0.810	0.632	0.523	0.845	0.636
AliNet	**0.539**	**0.826**	**0.628**	**0.549**	**0.831**	**0.645**	**0.552**	**0.852**	**0.657**

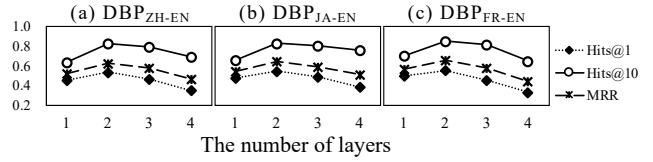Table 2: Results on DBP15K w.r.t. aggregation strategies

Figure 4: Results on DBP15K w.r.t. the number of layers.

multi-hop neighborhood information indeed contributes to entity alignment while the gating and attention mechanisms are crucial to capture important information in distant neighbors.

Impact of the number of layers and choice of k We first report the results of AliNet with 1 to 4 layers on DBP15K in Figure 4. AliNet with 2 layers achieves the best performance over all the three metrics. We observe that when AliNet has more layers, its performance declines as well. Although more layers allow AliNet to indirectly capture more distant neighborhood information by layer-to-layer propagation, such distant neighbors would introduce much noise and lead to more non-isomorphic neighborhood structures. Besides, we further show the results of the two-layer AliNet that considers different hops of neighborhood information in each layer in Table 3. We can see that considering two-hop neighborhood leads to the best results. This is similarly attributed to the aforementioned reasons regarding aggregation of multi-hop neighbors. This is further verified by an analysis about DBP15K. For example, in DBP$_{\text{ZH-EN}}$, each Chinese entity has 6.6 one-hop neighbors on average and this number for each English entity is 8.6. However, between their one-hop neighbors, there are only 4.5 pairs of counterpart entities, leaving 2.1 Chinese

Methods	DBP$_{\text{ZH-EN}}$			DBP$_{\text{JA-EN}}$			DBP$_{\text{FR-EN}}$		
	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10	MRR
GCN	0.487	0.790	0.559	0.507	0.805	0.618	0.508	0.808	0.628
AliNet	**0.539**	**0.826**	**0.628**	**0.549**	**0.831**	**0.645**	**0.552**	**0.852**	**0.657**
AliNet ($k=3$)	0.461	0.786	0.571	0.484	0.802	0.590	0.450	0.813	0.575
AliNet ($k=4$)	0.386	0.721	0.501	0.407	0.706	0.516	0.373	0.745	0.499

Table 3: Results on DBP15K w.r.t. k values

Figure 5: Average OC of one-hop neighbor sets of correct alignment during the first 200 training epochs on DBP15K.

one-hop neighbors and 4.1 English ones unaligned. If considering two-hop neighbors, the numbers of unaligned one-hop neighbors are reduced to 0.5 for Chinese and 0.9 for English, respectively. The numbers have less room to be reduced by introducing more distant neighbors. This suggests us that aggregating two-hop neighborhood information is enough.

Analysis of neighborhood overlap Furthermore, we make an empirical statistics on the overlap coefficient (OC) of the one-hop neighbors for each pair of counterpart entities in the correctly-found alignment. A high OC value for two entities means that they have a large overlap between their one-hop neighbors. We predict entity alignment and compute the average OC values of the correctly-predicted alignment every epoch. Figure 5 shows the value changes during the first 200 training epochs of AliNet and GCN. We find that the values display a decreasing trend. This indicates that it is relatively easy for GNN-based models to find the counterpart entities having a large proportion of common one-hop neighbors. The OC values of AliNet are smaller than those of GCN. This indicates that AliNet can effectively align the entities with smaller overlap in their one-hop neighbors.

Performance based on different layers In AliNet, we propose to use the combined representations of all layers as the final entity representations for predicting entity alignment. Here, we further examine the performance based on layer-specific entity representations. We report the entity alignment results on DBP15K in Figure 6 due to space limitation. The input layer is the randomly initialized feature vectors for entities to be tuned in AliNet. On top of this, we stack two AliNet layers (i.e., Layer 1 and Layer 2). "Combination" means the combined representations computed by Eq. (8). We can see that the representations of different layers show different performance of entity alignment. Layer 1 shows the best results among the three layers. As expected, the combined representations finally outperform the layer-specific representations.

5 Related Work

We hereby discuss related work to this paper. Particularly, as we have covered the majority of GNN-based entity alignment models in Section 2, we focus on discussing other families
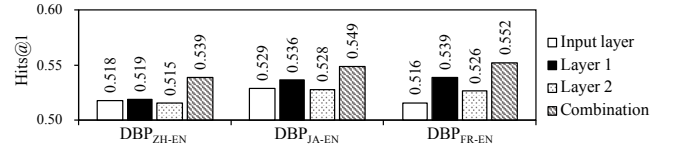

Figure 6: Hits@1 results w.r.t. different layers on DBP15K.

of models here. Most other models, such as MTransE (Chen et al. 2017), IPTransE (Zhu et al. 2017), JAPE (Sun, Hu, and Li 2017), AlignE and BootEA (Sun et al. 2018), NAEA (Zhu et al. 2019) as well as OTEA (Pei, Yu, and Zhang 2019), use TransE (Bordes et al. 2013) to learn entity embeddings. Meanwhile they learn a linear mapping or minimize the distance between the embeddings of pre-aligned entities. On top of KG structures, some work like KDCoE (Chen et al. 2018), AttrE (Trisedya, Qi, and Zhang 2019) and MultiKE (Zhang et al. 2019), incorporates additional profile information of entities such as textual descriptions and literal names for KG embedding. Differently, AliNet exploits the basic graph structures without using additional information. To further improve entity alignment performance, IPTransE, BootEA, KDCoE and NAEA use semi-supervised learning.

We also notice the recent work (Li et al. 2019) that uses GNNs for comparing the similarity of two graphs. Differently, we focus on the node-level rather than graph-level similarity comparison. (Kampffmeyer et al. 2019) captures the hierarchical structures of entities by introducing hypernym-hyponym links between nodes and their ancestors/descendants. Our work is also related to KG embedding that aims at learning vector representations for KG completion. There are translational models such as TransE (Bordes et al. 2013), TransH (Wang et al. 2014) and TransR (Lin et al. 2015), bilinear models such as ComplEx (Trouillon et al. 2016), SimplE (Kazemi and Poole 2018) and RotatE (Sun et al. 2019), and deep models such as ConvE (Dettmers et al. 2018), R-GCN (Schlichtkrull et al. 2018) and RSN (Guo, Sun, and Hu 2019). We refer interested readers to the recent survey (Lin et al. 2018) for more details on KG embedding.

6 Conclusion and Future Work

In this paper, we propose AliNet for entity alignment, aiming at mitigating the non-isomorphism among the neighborhood structures of counterpart entities in an end-to-end manner. AliNet captures the neighborhood information within multiple hops by a gating mechanism in each layer. It employs an attention mechanism for multi-hop neighborhood aggregation to reduce noises. We further propose a relation loss to enhance the expressiveness of AliNet. Our experiments on five datasets demonstrate the effectiveness of AliNet. For future work, we plan to incorporate side information of entities in other modalities into the preserved graph structures.

References

Abu-El-Haija, S.; Perozzi, B.; Kapoor, A.; Alipourfard, N.; Lerman, K.; Harutyunyan, H.; Steeg, G. V.; and Galstyan, A. 2019. Mixhop: Higher-order graph convolutional architectures via sparsified neighborhood mixing. In *ICML*, 21–29.

Bordes, A.; Usunier, N.; García-Durán, A.; Weston, J.; and Yakhnenko, O. 2013. Translating embeddings for modeling multi-relational data. In *NIPS*, 2787–2795.

Cao, Y.; Liu, Z.; Li, C.; Liu, Z.; Li, J.; and Chua, T.-S. 2019. Multi-channel graph neural network for entity alignment. In *ACL*, 1452–1461.

Chen, M.; Tian, Y.; Yang, M.; and Zaniolo, C. 2017. Multilingual knowledge graph embeddings for cross-lingual knowledge alignment. In *IJCAI*, 1511–1517.

Chen, M.; Tian, Y.; Chang, K.; Skiena, S.; and Zaniolo, C. 2018. Co-training embeddings of knowledge graphs and entity descriptions for cross-lingual entity alignment. In *IJCAI*, 3998–4004.

Conneau, A.; Lample, G.; Ranzato, M.; Denoyer, L.; and Jégou, H. 2018. Word translation without parallel data. In *ICLR*.

Dettmers, T.; Minervini, P.; Stenetorp, P.; and Riedel, S. 2018. Convolutional 2D knowledge graph embeddings. In *AAAI*, 1811–1818.

Glorot, X., and Bengio, Y. 2010. Understanding the difficulty of training deep feedforward neural networks. In *AISTATS*, volume 9, 249–256.

Guo, L.; Sun, Z.; and Hu, W. 2019. Learning to exploit long-term relational dependencies in knowledge graphs. In *ICML*, 2505–2514.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *CVPR*, 770–778.

Kampffmeyer, M.; Chen, Y.; Liang, X.; Wang, H.; Zhang, Y.; and Xing, E. P. 2019. Rethinking knowledge graph propagation for zero-shot learning. In *CVPR*, 11487–11496.

Kazemi, S. M., and Poole, D. 2018. Simple embedding for link prediction in knowledge graphs. In *NeurIPS*, 4289–4300.

Kipf, T. N., and Welling, M. 2017. Semi-supervised classification with graph convolutional networks. In *ICLR*.

Li, Y.; Gu, C.; Dullien, T.; Vinyals, O.; and Kohli, P. 2019. Graph matching networks for learning the similarity of graph structured objects. In *ICML*, 3835–3845.

Lin, Y.; Liu, Z.; Sun, M.; Liu, Y.; and Zhu, X. 2015. Learning entity and relation embeddings for knowledge graph completion. In *AAAI*, 2181–2187.

Lin, Y.; Han, X.; Xie, R.; Liu, Z.; and Sun, M. 2018. Knowledge representation learning: A quantitative review. *CoRR* abs/1812.10901.

Morris, C.; Ritzert, M.; Fey, M.; Hamilton, W. L.; Lenssen, J. E.; Rattan, G.; and Grohe, M. 2019. Weisfeiler and leman go neural: Higher-order graph neural networks. In *AAAI*, 4602–4609.

Pei, S.; Yu, L.; Hoehndorf, R.; and Zhang, X. 2019. Semi-supervised entity alignment via knowledge graph embedding with awareness of degree difference. In *WWW*, 3130–3136.

Pei, S.; Yu, L.; and Zhang, X. 2019. Improving cross-lingual entity alignment via optimal transport. In *IJCAI*, 3231–3237.

Pujara, J.; Miao, H.; Getoor, L.; and Cohen, W. W. 2013. Knowledge graph identification. In *ISWC*, 542–557.

Schlichtkrull, M. S.; Kipf, T. N.; Bloem, P.; van den Berg, R.; Titov, I.; and Welling, M. 2018. Modeling relational data with graph convolutional networks. In *ESWC*, 593–607.

Srivastava, R. K.; Greff, K.; and Schmidhuber, J. 2015. Highway networks. *CoRR* abs/1505.00387.

Sun, Z.; Hu, W.; Zhang, Q.; and Qu, Y. 2018. Bootstrapping entity alignment with knowledge graph embedding. In *IJCAI*, 4396–4402.

Sun, Z.; Deng, Z.-H.; Nie, J.-Y.; and Tang, J. 2019. Rotate: Knowledge graph embedding by relational rotation in complex space. In *ICLR*.

Sun, Z.; Hu, W.; and Li, C. 2017. Cross-lingual entity alignment via joint attribute-preserving embedding. In *ISWC*, 628–644.

Trisedya, B. D.; Qi, J.; and Zhang, R. 2019. Entity alignment between knowledge graphs using attribute embeddings. In *AAAI*, 297–304.

Trouillon, T.; Welbl, J.; Riedel, S.; Gaussier, É.; and Bouchard, G. 2016. Complex embeddings for simple link prediction. In *ICML*, 2071–2080.

Velickovic, P.; Cucurull, G.; Casanova, A.; Romero, A.; Liò, P.; and Bengio, Y. 2018. Graph attention networks. In *ICLR*.

Wang, Z.; Zhang, J.; Feng, J.; and Chen, Z. 2014. Knowledge graph embedding by translating on hyperplanes. In *AAAI*, 1112–1119.

Wang, Z.; Lv, Q.; Lan, X.; and Zhang, Y. 2018. Cross-lingual knowledge graph alignment via graph convolutional networks. In *EMNLP*, 349–357.

Weisfeiler, B., and Lehman, A. 1968. A reduction of a graph to a canonical form and an algebra arising during this reduction. *Nauchno-Technicheskaya Informatsia* 9:12–16.

Wu, Y.; Liu, X.; Feng, Y.; Wang, Z.; Yan, R.; and Zhao, D. 2019. Relation-aware entity alignment for heterogeneous knowledge graphs. In *IJCAI*, 5278–5284.

Xu, K.; Hu, W.; Leskovec, J.; and Jegelka, S. 2019a. How powerful are graph neural networks? In *ICLR*.

Xu, K.; Wang, L.; Yu, M.; Feng, Y.; Song, Y.; Wang, Z.; and Yu, D. 2019b. Cross-lingual knowledge graph alignment via graph matching neural network. In *ACL*, 3156–3161.

Ye, R.; Li, X.; Fang, Y.; Zang, H.; and Wang, M. 2019. A vectorized relational graph convolutional network for multi-relational network alignment. In *IJCAI*, 4135–4141.

Zhang, Q.; Sun, Z.; Hu, W.; Chen, M.; Guo, L.; and Qu, Y. 2019. Multi-view knowledge graph embedding for entity alignment. In *IJCAI*, 5429–5435.

Zhu, H.; Xie, R.; Liu, Z.; and Sun, M. 2017. Iterative entity alignment via joint knowledge embeddings. In *IJCAI*, 4258–4264.

Zhu, Q.; Zhou, X.; Wu, J.; Tan, J.; and Guo, L. 2019. Neighborhood-aware attentional representation for multilingual knowledge graphs. In *IJCAI*, 1943–1949.