Learning Short-Term Differences and Long-Term Dependencies for Entity Alignment

Jia Chen[1], Zhixu Li[1,2,3(✉)], Pengpeng Zhao[1], An Liu[1], Lei Zhao[1], Zhigang Chen[3], and Xiangliang Zhang[4]

[1] School of Computer Science and Technology, Soochow University, Suzhou, China
jchen0812@stu.suda.edu.cn, {zhixuli,ppzhao,anliu,zhaol}@suda.edu.cn
[2] IFLYTEK Research, Suzhou, China
[3] State Key Laboratory of Cognitive Intelligence, iFLYTEK, Hefei, People's Republic of China
zgchen@iflytek.com
[4] King Abdullah University of Science and Technology, Thuwal, Saudi Arabia
xiangliang.zhang@kaust.edu.sa

Abstract. We study the problem of structure-based entity alignment between knowledge graphs (KGs). The recent mainstream solutions for it apply KG embedding techniques to map entities into a vector space, where the similarity between entities could be measured accordingly. However, these methods which are mostly based on TransE and its variants treat relation triples in KGs independently. As a result, they fail to capture some advanced interactions between entities that are implicit in the surrounding and multi-hop entities: One is the differences between the one-hop and two-hop neighborhood of an entity, which we call as short-term differences, while the other is the dependencies between entities that are far apart, which we call as long-term dependencies. Based on the above observations, this paper proposes a novel approach learning to capture both the short-term differences and the long-term dependencies in KGs for entity alignment using graph neural networks and self-attention mechanisms respectively. Our empirical study conducted on four couples of real-world datasets shows the superiority of our model, compared with the state-of-the-art methods.

Keywords: Knowledge graph · Entity alignment · KG embedding

1 Introduction

Entity alignment (EA) aims at associating entities across different knowledge graphs (KGs) if they refer to the same real-world object. As numerous KGs have been constructed and applied in recent years, it is necessary to connect or merge them to comprehensively represent the knowledge in one domain or a specific cross-domain. To measure the similarity between entities across different

J. Z. Pan et al. (Eds.): ISWC 2020, LNCS 12506, pp. 92–109, 2020.
https://doi.org/10.1007/978-3-030-62419-4_6

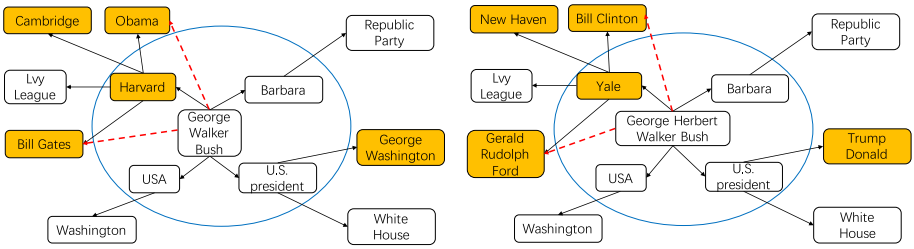

Fig. 1. Two implicit advanced interactions between entities in the EA scenario (Color figure online)

KGs, traditional methods mainly rely on designing proper features [12,14] which are labor-intensive and hard to transfer, while recent work [4,16] focus on applying various kinds of KG embedding models, which have shown their effectiveness on entity alignment. Although much other information such as the attributes, attribute values, and descriptions of entities could be utilized to enhance the embedding results for entity alignment, the mainstream studies always concentrate on using structure information, i.e., the relation triples in KGs.

The key idea of embedding-based methods is to encode entities and relations into vector spaces and then find alignment between entities according to their embedding similarities. Most embedding-based methods are developed from TransE [2], a seminal work in the KG embedding field, encoding entities and relations into vector spaces and making them satisfy some specific mathematical operations. MTransE [4] and SEA [16] are extensions of the TransE model for entity alignment task, they encode two KGs via TransE separately and learn transitions to map an entity's embedding in one KG to its counterpart in the other one. Different from adjusting entities using mapping functions, IPTransE [25] and BootEA [18] propose to swap aligned entities to calibrate the embeddings of entities in two KGs into a unified embedding space and generate more aligned entities during the process of iteration. JAPE [17], KDcoE [3], AttrE [19] and MultiKE [24] combine structure-based TransE with other external embeddings, such as attribute-based TransE, name-based TransE and entity description embeddings to get better representations for entities.

However, TransE and its variants learn embeddings from triples independently and thus fail to capture some advanced interactions between entities that are implicit in the surrounding and multi-hop entities. One implicit complex interaction between entities is the differences between the one-hop and two-hop neighborhood of an entity, which we call as **short-term differences**, while the other is the dependencies between entities that are far apart, which we call as **long-term dependencies**. As the example illustrated in Fig. 1, the traditional embedding results of "Geroge Walker Bush" and "Geroge Herbert Walker Bush" learned from their one-hop neighbors (i.e., those within the blue circle) may hardly distinguish them, but the short-term differences of the two entities could let them be better represented and distinguished as many of their two-hop

entities are different. Meanwhile, learning dependencies between an entity and its multi-hop entities, for example, the entities linked by the red dotted line in Fig. 1, could better represent entities than using TransE, which only learns one-hop dependencies in the form of triples thus propagates information inefficiently. Capturing the above two kinds of interactions prevents entities from suffering from low expressiveness.

The recent progress in graph neural networks (GNNs) [21] and sequence learning methods [5] have fueled a lot of research on applying these advanced models to entity alignment. GCN-Align [22] is the first to train graph convolutional networks (GCNs) on KGs, it quantifies relations in triples and embeds entities of each KG into a unified space. However, like the TransE-based methods, it also overlooks the differences in the surrounding entities and the dependencies between multi-hop entities. Instead of directly modeling triples or neighbors, RSNs [8] is proposed to learn from relational paths through an RNN-based sequence model where long-term dependencies between entities can be captured. Nevertheless, its sequence model precludes parallelization and the generated paths are biased to entities and relations of high degree. Besides, it is also not sensitive to short-term differences.

In this paper, we propose a novel model that can capture both the short-term differences and long-term dependencies. First, we leverage the self-attention mechanism [20] to model dependencies between entities without regard to the distance between items in the sequences generated by a well-designed degree-aware random walk. Second, unlike GCNs that update nodes' vectors by incorporating first-order neighbors, we capture short-term differences of entities by repeatedly mixing neighborhood information aggregated at various distances [1]. Then, a linear combination is used to concatenate embeddings learned from the above two modules as the final representation. Finally, we find alignments by calculating their embedding similarities.

In summary, our main contributions are listed as follows:

- This work is the first attempt to captures both the long-term dependencies and the short-term differences in KGs for entity alignment.
- We propose to utilize the self-attention mechanism instead of RNN to model long-term dependencies between entities and devise a degree-aware random walk to generate high-quality sequences in KGs.
- To obtain the representation of the short-term differences of entities, we introduce a new graph neural network, from which we can better interpret the short-term semantics of an entity from its surrounding neighbors.
- Our extensive empirical study conducted on four real-world datasets shows the superiority and efficacy of our method, compared with the state-of-the-art structure-based methods.

2 Related Work

KG embedding techniques have evolved rapidly in recent years and their usefulness has been demonstrated in entity alignment. The current methods for embedding-based entity alignment models fall into the following three categories:

2.1 TransE-Based Entity Alignment

TransE [2] is the most representative model in KG embedding approaches. It holds the assumption that when mapping a triple (h, r, t) in KGs into a vector space, the triple should satisfy $\mathbf{h} + \mathbf{r} \approx \mathbf{t}$, where $\mathbf{h}$, $\mathbf{r}$ and $\mathbf{t}$ are the corresponding vectors of h, r, t in the vector space. MTransE [4] encodes triples of each KG in a separated embedding space via TransE, and makes use of seed alignments to learn transitions for each embedding vector to its counterpart in the other space. IPTransE [25] and BootEA [18] learn to map embeddings of different KGs into the same space and perform iterative entity alignment to update the joint embeddings and provide more training samples. Instead of utilizing the limited aligned entities, SEA [16] is proposed to learn transitions from both labeled and unlabeled data. It also mitigates the effect of degree differences in the existing KG embedding methods by adversarial training. Further, several works, such as JAPE [17] and AttrE [19], considers to learn representations from the structure and attribute information in KGs while KDcoE [3] and MultiKE [24] leverage descriptions, names and attributes to enhance the structure-based embedding methods. All the methods above focus on the utilization of existing TransE-based methods that treat triples separately, including MultiKE, which learns comprehensive entity embeddings from different views. However, the hidden complex information involved in the surrounding and multi-hop entities is neglected and thus remained to be exploited.

2.2 GNN-Based Entity Alignment

As entities can be seen as nodes in graphs, GNN is then proposed for entity alignment. GCN-Align [22] is the first attempt to generate node-level embeddings by encoding information about the nodes' neighborhoods via GCN, which is a type of convolutional network that directly operates on graph data. It assumes that equivalent entities are usually neighbored by some other equivalent entities and they tend to have similar attributes, so when GCN produces neighborhood-aware embeddings of entities, alignment can be predicted by a pre-defined distance function. Inga [15] introduces an iterative training mechanism on GCN-Align and improves the initial attribute feature by considering local and global attribute information. Unlike the above node-level matching, [23] formulates the alignment task as a graph matching problem. It proposes a topic graph structure and matches all entities in two topic entity graphs by attentive node-level matching. However, none of them considers the importance of differences in the surrounding entities of an entity. Besides, although high-order dependency information, i.e., long-term dependencies, can be captured by increasing the number of GNN propagation layers, embeddings tend to be the same for different nodes and the noise introduced can not be ignored, as reported in [11].

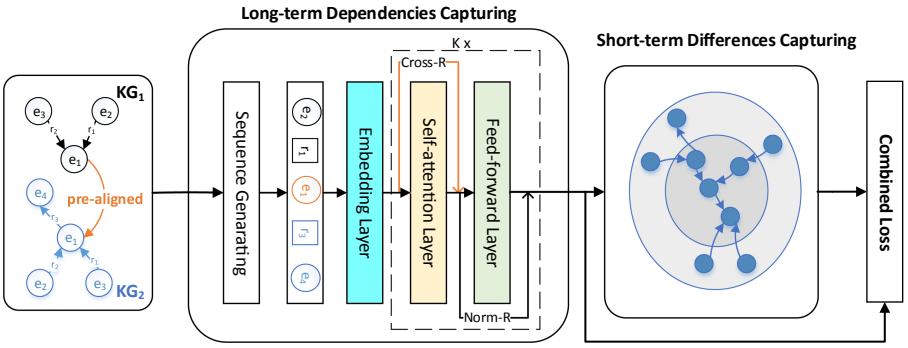

Fig. 2. The architecture of our proposed model

2.3 Sequence-Based Entity Alignment

Sequence is a ubiquitous data structure, employed extensively in natural language processing and related fields. RSNs [8] is the first to investigate long-term relational dependencies in KGs. It uses biased random walk to sample paths in KGs which are composed of entities and relations alternately and then models relational paths through recurrent skipping networks rather than recurrent neural networks due to the unique triple structure in KGs. In specific, RSNs makes the output hidden states of relations learn a residual from their direct former entities in the sequences. Though it achieves a significant performance improvement and outperforms TransE-based and GCN-based methods, it is also unaware of the short-term differences of entities and unable to capture long-term dependencies between entities well enough. As pointed out in a recent study of machine translation [20], without any recurrent or convolutional structures, long-term dependencies between words in sequences can be captured effectively by solely using self-attention mechanisms. Therefore, we seek to propose a model that can capture both the long-term dependencies and the short-term differences in KGs for entity alignment using self-attention mechanisms and graph neural networks, respectively.

3 Our Proposed Model

In this section, we introduce the proposed model for entity alignment. We first formulate the problem of entity alignment in KGs and then describe the proposed model in detail.

3.1 Problem Formulation

In this paper, we consider structure-based entity alignment between KGs, which aims to learn entity embeddings merely based on structure information in KGs (i.e., relation triples). A KG can be noted as a 3-tuple $G = (E, R, T)$, where E,

R, and T denote the set of entities, relations and triples, respectively. Triples have three components: head entity h, relation r and tail entity t. A triple is always presented as (h, r, t).

Our problem is formulated as follows: Given two KGs $G_1 = (E_1, R_1, T_1)$, $G_2 = (E_2, R_2, T_2)$ and some pre-aligned entity pairs $S = \{(e_i, e_j)|e_i \in E_1, e_j \in E_2\}$, entity alignment is aimed at discovering new equivalent entities $N = \{(e_{i'}, e_{j'})|e_{i'} \in E_1, e_{j'} \in E_2\}$ based on S.

3.2 Capturing Long-Term Dependencies

The overview of our model is presented in Fig. 2. The first part is designed to facilitate the learning of long-term dependencies between entities. It first connects two KGs as one joint KG by pre-aligned entities and adds reverse relations between entities to enhance connectivity. Then, after generating entity-relation sequences for each triple through a degree-aware random walk method, leverages a self-attention based sequence predicting model to capture the long-term dependencies.

Sequence Generating. Sequence generating takes two KGs and a set of pre-aligned entities as input to produce sequences $P = \{p_1, p_2, \ldots p_i, \ldots, p_l\}$, where $p_i = (e_{i_1}, r_{i_2}, \ldots, e_{i_m})$, made up of entities and relations alternately. To obtain desired sequences, inspired by random walk which is widely used in networks [6], we propose a degree-aware random walk for KG to maintain a balanced collection of information between long-tail entities and frequent entities. More specifically, we leverage the idea of node2vec [7] and introduce a depth bias and a degree bias to reduce the likelihood of revisiting an entity's one-hop neighbors and increase the likelihood of visiting entities of low-degree, respectively. To formalize, when the current entity is e_i, the transition probability to the next entity e_{i+1}, denoted as $P(e_{i+1}|e_i)$, is calculated as follows:

$$P(e_{i+1}|e_i) = \alpha_{dp}(e_i, e_{i+1}) \cdot \beta_{dg}(e_i, e_{i+1}) \tag{1}$$

where $\alpha_{dp}(e_i, e_{i+1})$ is the depth bias and $\beta_{dg}(e_i, e_{i+1})$ is the degree bias. They are defined as follows:

$$\alpha_{dp}(e_i, e_{i+1}) = \begin{cases} q & d(e_{i-1}, e_{i+1}) = 2 \\ 1 - q & d(e_{i-1}, e_{i+1}) \neq 2 \end{cases} \tag{2}$$

where q is a hyperparameter ranging from 0 to 1 controlling the depths, e_{i-1} is the previous entity of e_i and $d(e_{i-1}, e_{i+1})$ denotes the shortest distance from e_{i-1} to e_{i+1}.

$$\beta_{dg}(e_i, e_{i+1}) = \begin{cases} \frac{1}{d_{e_{i+1}} + f_r} & \exists r \in R, (e_i, r, e_{i+1}) \in T \\ 0 & otherwise \end{cases} \tag{3}$$

where $d_{e_{i+1}}$ gains the number of entities linked with e_{i+1}, and we name it the degree of entity e_{i+1}. f_r indicates the occurrence number of relation r in all triples.

It is noticeable that the bias we designed favor entities that haven't appeared in a specific sequence and have a low degree, which ensures the high quality of the generated sequences.

Self-attention-Based Sequence Predicting. The inputs of this module are sequences P and the goal of it is to effectively capture the long-term dependencies between items in P. The beginning of this module consists of three kinds of embeddings: entity embedding $\boldsymbol{E}$, relation embedding $\boldsymbol{R}$, and position embedding $\boldsymbol{P}$. After adding the embeddings of items in sequences to their corresponding positional embeddings, the output of the embedding layer $\hat{\boldsymbol{E}}$ is obtained and fed to the self-attention layer.

$$\begin{aligned} \boldsymbol{S} &= Concat(head_1, \ldots, head_h) \\ where\, head_h &= softmax(\frac{(\hat{\boldsymbol{E}}\boldsymbol{W}_h^Q)(\hat{\boldsymbol{E}}\boldsymbol{W}_h^K)^T}{\sqrt{d}})(\hat{\boldsymbol{E}}\boldsymbol{W}_h^V) \end{aligned} \tag{4}$$

where $\boldsymbol{W}_h^Q$, $\boldsymbol{W}_h^K$, and $\boldsymbol{W}_h^V$ are projection matrices of $head_h$, and the output of self-attention layer is the simple concatenation of multiple heads.

Hereafter, distinct from the normal residual connection (Norm-R) in [20], we use a specific crossed residual (Cross-R) connection $\boldsymbol{C}$ to better optimize the model due to the unique triple structure in KGs. It means that when capturing long-term dependencies, the importance of tripe structure should be reflected, so when the input is a relation, a residual from its previous entity is needed.

$$\boldsymbol{C}_i = \begin{cases} \boldsymbol{S}_i & \frac{i+1}{2} \in \mathbb{N}^+ \\ \boldsymbol{S}_i + \hat{\boldsymbol{E}}_{\frac{i}{2}} & \frac{i}{2} \in \mathbb{N}^+ \end{cases} \tag{5}$$

Then, we employ a point-wise feed-forward network and a normal residual connection (Norm-R) to allow interactions within different latent dimensions inspired by [20].

$$\boldsymbol{F} = \boldsymbol{C} + ReLU(\boldsymbol{C}\boldsymbol{W}_1 + b_1)\boldsymbol{W}_2 + b_2 \tag{6}$$

where $ReLU$ is activation function, $\boldsymbol{W}_1$ and $\boldsymbol{W}_2$ are trainable matrices and b_1 and b_2 are bias vectors.

We can further stack more layers to explore different types of dependencies. Formally, for the k-th layer, we recursively formulate the representation as:

$$\hat{\boldsymbol{E}}^{(k)} = SA\big(\hat{\boldsymbol{E}}^{(k-1)}\big) \tag{7}$$

where $k > 1$, $\hat{\boldsymbol{E}}^{(1)} = \hat{\boldsymbol{E}}$ and SA is made up of the above self-attention layer with crossed residual connection and feed-forward layer with normal residual connection.

After performing k layers, we obtain representations of long-term dependencies for entities and relations. We optimize the embeddings of items in sequences by adopting the sequence predicting method and leverage the idea used in [13] to accelerate training. The prediction loss of one sequence is defined as follows:

$$\mathcal{L}_{LD}^{(1)} = -\sum_{i=1}^{m-1}\Big(\log\sigma(\hat{\boldsymbol{E}}_i \cdot \boldsymbol{y}_i) + \sum_{j=1}^{n}\mathbb{E}_{y'_j\sim P(y')}\big[\log\sigma(-\hat{\boldsymbol{E}}_i \cdot \boldsymbol{y}'_j)\big]\Big) \quad (8)$$

where m is the number of items in the sequence, σ is an activation function, $\boldsymbol{y}_i$ is the object of the *i-th* item while y'_j is the *j-th* negative sample of it. n is the number of negative samples and $P(y')$ is the noise distribution where only items whose frequencies appear in the first three-quarters can be found, it changes according to the current item is entity or relation, which indicates that the object and its corresponding negative examples come from the same type. Adding the prediction loss of all l sequences is the whole loss of the long-term dependencies capturing model.

$$\mathcal{L}_{LD} = \sum_{i=1}^{l}\mathcal{L}_{LD}^{(i)} \quad (9)$$

3.3 Capturing Short-Term Differences

Next, we build upon the architecture of graph neural networks to capture short-term differences of entities. The major difference to [10] lies in that our graph convolutional network encodes neighborhood information at various distances rather than first-order neighborhood features, as such, short-term semantics of an entity can be augmented.

Formally, this module takes the vectors learned from the above self-attention based layers as input to produce embeddings of short-term differences for each entity as follows:

$$\boldsymbol{H}^{(i)} = \begin{cases} \hat{\boldsymbol{E}}^{(k)} & \text{if } i = 0 \\ \Big\Vert_{j\in A}\ \sigma(\hat{\boldsymbol{A}}^j \boldsymbol{H}^{(i-1)}\boldsymbol{W}_j^{(i)}) & \text{if } i \in [1..t] \end{cases} \quad (10)$$

where $\Vert$ denotes column-wise concatenation, A is the hyperparameter containing a set of integer adjacency powers, σ denotes an element-wise activation function, $\hat{\boldsymbol{A}}^j$ represents the self-defined adjacency matrix $\hat{\boldsymbol{A}}$ that is multiplied by itself j times and $\boldsymbol{W}_j^{(i)}$ is the weight matrix when gathering the j-order neighborhood information in the i-th layer. Note that, different from the commonly used adjacency matrix in GNNs: $\hat{\boldsymbol{A}} = \boldsymbol{D}^{-\frac{1}{2}}(\boldsymbol{A} + \boldsymbol{I}_n)\boldsymbol{D}^{-\frac{1}{2}}$, where $\boldsymbol{D}$ is the degree matrix, $\boldsymbol{A}$ is the connectivity matrix indicating the graph structure and $\boldsymbol{I}_n$ is the $n \times n$ identity matrix making $\hat{\boldsymbol{A}}$ symmetrically normalized with self-connections, we remove I_n from $\hat{\boldsymbol{A}}$ since our goal is to capture short-term differences surrounding an entity. In addition, we set $\boldsymbol{W}_j^{(1)}$ to be identity matrix because there is

Table 1. Statistics of datasets

Datasets	Source KGs	#Entity	#Norm-R	#Norm-T	#Dense-R	#Dense-T
DBP-WD	DBPdeia (English)	15,000	253	38,421	220	68,598
	Wikidata (English)	15,000	144	40,159	135	75,465
DBP-YG	DBPdeia (English)	15,000	219	33,571	206	71,257
	YAGO3 (English)	15,000	30	34,660	30	97,131
EN-FR	DBPdeia (English)	15,000	221	36,508	217	71,929
	DBPdeia (French)	15,000	177	33,532	174	66,760
EN-DE	DBPdeia (English)	15,000	225	38,281	207	56,983
	DBPdeia (German)	15,000	118	37,069	117	59,848

no need to transform the input features which have already captured long-term dependencies into higher-level features.

The short-term differences for equivalent entities are expected to be alike, hence we train the GNN model to encode equivalent entities as close as possible in the embedding space by using the pre-aligned entity pairs. We use the following loss function to measure its plausibility:

$$\mathcal{L}_{SD} = \sum_{(e_i,e_j)\in S} \sum_{(e'_i,e'_j)\in S'} \left[\|\boldsymbol{e}_i - \boldsymbol{e}_j\| + \gamma - \|\boldsymbol{e}'_i - \boldsymbol{e}'_j\|\right]_+ \tag{11}$$

where $[x]_+ = \max\{0, x\}$, $\|\cdot\|$ denotes either L_1 or L_2 vector norm, γ is a margin hyperparameter which is greater than 0, and S' denotes the set of negative-sampled entity alignments by replacing e_i or e_j in S.

3.4 Combined Loss

To preserve both the long-term and short-term complex information of entities, we jointly minimize the following loss function:

$$\mathcal{L}_{joint} = \mathcal{L}_{LD} + \mathcal{L}_{SD} \tag{12}$$

Considering the complementarity between the embeddings of long-term dependencies captured by the self-attention-based sequence model and that of short-term differences captured by GNNs, we concatenate them linearly and use the combined embeddings as entities' final representations.

$$\boldsymbol{e}_i^f = \left[\theta \boldsymbol{e}_i^{ld}; (1-\theta)\boldsymbol{e}_i^{sd}\right] \tag{13}$$

where θ is a parameter balancing the distribution between $\boldsymbol{e}_i^{ld}$ and $\boldsymbol{e}_i^{sd}$, which are the embedding of long-term dependencies and the embedding of short-term differences, respectively.

4 Experiments

This section covers four parts. We start by considering datasets and details of the model in the experimental setup and then introduce baselines briefly. Ultimately, experiments and analyses are shown to validate the proposed model.

4.1 Experimental Setup

Datasets. To evaluate the effectiveness of our proposed model comprehensively, we reuse four couples of real-world datasets, namely DBP-WD, DBP-YG, EN-FR and EN-DE, recently proposed in [8]. It is worth mentioning that each dataset is sampled from real-world KGs: DBPedia (DBP), YAGO3 (YG) and Wikidata (WD) by the PageRank algorithm to ensure that their entity distributions keep consistent with original KGs, and each has two kinds of entity distributions: a normal one and a dense one. DBP-WD and DBP-YG are mono-lingual KGs while EN-FR and EN-DE are cross-lingual KGs. The statistics of the datasets are shown in Table 1 where #Norm-R denotes the number of relations in the normal datasets while #Norm-T denotes that of triples in them, and similar notations for the dense datasets. For both the normal and dense datasets, each KG contains 15,000 entities.

Implementation Details. Our model is implemented with TensorFlow. We use an embedding dimension of 256 for all methods compared in 4.2 on all datasets. Adam [9] is adopted to optimize $\mathcal{L}_{LD}$ and $\mathcal{L}_{SD}$ in turn. For the sequence generating part, we set $q = 0.9$, $m = 15$ and for the sequence predicting phase, we use 8 head in the self-attention layer and set $k = 3$. When aggregate neighborhood information, i and A in Eq. (10) are set to 2 and $\{0, 1, 2\}$, respectively and the non-linearity function σ is tanh. Besides, dropout and layer normalization are adopted to stabilize the training. For fair comparison, we sample 30% of the aligned entity set as the training set and the rest for testing for all approaches. We choose Hits@1, Hits@10 and MRR to evaluate the alignment results, where Hits@k indicates the proportion of correctly aligned entities ranked in the top k and MRR is the abbreviation of mean reciprocal rank measuring the average reciprocal values of testing entities.

4.2 Baselines

We include the following methods for performance comparison, which have been thoroughly discussed in Sect. 2:

- **MTransE** [4]**:** This is the first to leverage KG embeddings to cross-lingual knowledge alignment. It explores different kinds of mapping functions between two KGs to find alignment in an efficient way.
- **IPTranE** [25]**:** It jointly encodes different KGs into the same space and improves alignment performance by iteration strategy.
- **JAPE** [17]**:** As the first attempt to learn embeddings of cross-lingual KGs while preserving their attribute information, this model achieves a certain improvement by updating its embeddings through attribute correlations which comes from attribute type similarity.
- **BootEA** [18]**:** It introduces a bootstrapping approach to iteratively generate more likely training data for alignment-oriented KG embedding and employs an editing method to weaken error accumulation caused by iteration.

Table 2. Results on the normal and mono-lingual datasets

Datasets	DBP-WD			DBP-YG		
Metrics	Hits@1	Hits@10	MRR	Hits@1	Hits@10	MRR
MTransE	22.3	50.1	0.32	24.6	54.0	0.34
IPTransE	23.1	51.7	0.33	22.7	50.0	0.32
JAPE	21.9	50.1	0.31	23.3	52.7	0.33
BootEA	32.3	63.1	0.42	31.3	62.5	0.42
GCN-Align	17.7	37.8	0.25	19.3	41.5	0.27
SEA	31.0	51.9	0.38	30.3	45.9	0.36
RSNs	38.8	65.7	0.49	40.0	67.5	0.50
Ours w/o LD	24.4	54.9	0.34	29.8	58.9	0.40
Ours w/o SD	44.1	70.7	0.53	44.5	71.0	0.53
Ours	**46.8**	**75.3**	**0.56**	**46.5**	**74.0**	**0.56**
%Improv	20.62%	14.61%	14.29%	16.25%	9.63%	12.00%

Table 3. Results on the normal and cross-lingual datasets

Datasets	EN-FR			EN-DE		
Metrics	Hits@1	Hits@10	MRR	Hits@1	Hits@10	MRR
MTransE	25.1	55.1	0.35	31.2	58.6	0.40
IPTransE	25.5	55.7	0.36	31.3	59.2	0.41
JAPE	25.6	56.2	0.36	32.0	59.9	0.41
BootEA	31.3	62.9	0.42	44.2	70.1	0.53
GCN-Align	15.5	34.5	0.22	25.3	46.4	0.33
SEA	25.8	40.0	0.31	42.5	59.6	0.49
RSNs	34.7	63.1	0.44	48.7	72.0	0.57
Ours w/o LD	22.0	51.1	0.32	37.1	62.7	0.46
Ours w/o SD	38.3	65.8	0.47	51.3	73.4	0.59
Ours	**41.1**	**70.6**	**0.51**	**53.9**	**77.8**	**0.62**
%Improv.	18.44%	11.89%	15.91%	10.68%	8.06%	8.77%

- **GCN-Align** [22]**:** It proposes a novel approach for cross-lingual KG alignment, whose core is graph convolutional networks. This GNN-based method takes relations in triples as edges with weights and produces neighborhood-aware embeddings of entities to discover entity alignments.
- **SEA** [16]**:** An extension of the MTransE-based alignment model, whose purpose is to design a semi-supervised entity alignment model to leverage abundant unlabeled data leaving unused before and improve the original KG embedding with awareness of degree difference by an adversarial framework.

Table 4. Results on the dense and mono-lingual datasets

Datasets	DBP-WD			DBP-YG		
Metrics	Hits@1	Hits@10	MRR	Hits@1	Hits@10	MRR
MTransE	38.9	68.7	0.49	22.8	51.3	0.32
IPTransE	43.5	74.5	0.54	23.6	51.3	0.33
JAPE	39.3	70.5	0.50	26.8	57.3	0.37
BootEA	67.8	91.2	0.76	68.2	89.8	0.76
GCN-Align	43.1	71.3	0.53	31.3	57.5	0.40
SEA	67.2	85.2	0.74	68.1	84.1	0.74
RSNs	76.3	92.4	0.83	82.6	95.8	0.87
Ours w/o LD	53.5	84.7	0.64	65.4	88.2	0.74
Ours w/o SD	79.9	93.7	0.85	85.6	96.5	0.89
Ours	**81.8**	**95.9**	**0.87**	**86.9**	**97.4**	**0.91**
%Improv.	7.21%	3.79%	4.82%	5.21%	1.67%	4.60%

Table 5. Results on the dense and cross-lingual datasets

Datasets	EN-FR			EN-DE		
Metrics	Hits@1	Hits@10	MRR	Hits@1	Hits@10	MRR
MTransE	37.7	70.0	0.49	34.7	62.0	0.44
IPTransE	42.9	78.3	0.55	34.0	63.2	0.44
JAPE	40.7	72.7	0.52	37.5	66.1	0.47
BootEA	64.8	91.9	0.74	66.5	87.1	0.73
GCN-Align	37.3	70.9	0.49	32.1	55.2	0.40
SEA	62.3	85.7	0.71	65.2	79.4	0.70
RSNs	75.6	92.5	0.82	73.9	89.0	0.79
Ours w/o LD	51.6	86.0	0.63	60.9	82.2	0.68
Ours w/o SD	80.6	94.2	0.85	77.6	91.2	0.82
Ours	**83.6**	**97.2**	**0.89**	**79.4**	**93.0**	**0.84**
%Improv.	10.58%	5.08%	8.54%	7.44%	4.49%	6.33%

– **RSNs** [8]**:** It is a state-of-the-art structure-based alignment model, which learns embeddings from relational paths instead of triples. It captures long-term dependencies between KGs by a sequence model integrating recurrent neural networks with residual learning.

4.3 Comparisons of Performance

The experimental results on the mono-lingual and cross-lingual datasets are presented in Table 2, 3, 4 and 5, respectively. Table 2 and 3 are results on normal

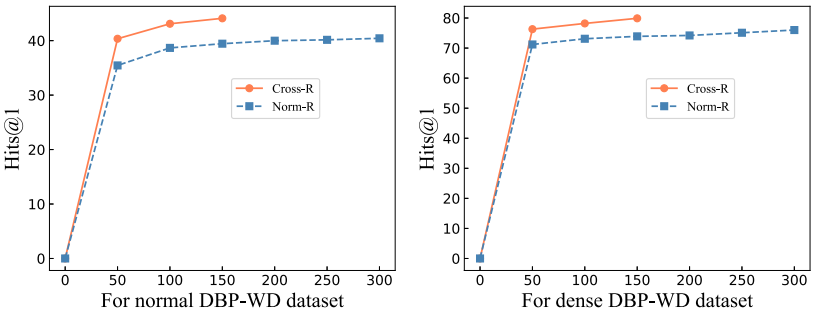

Fig. 3. Effect of the proposed Cross-R.

datasets while Table 4 and 5 are that on dense datasets. The best results are shown in bold, along with the percentage of improvement when comparing our method with the best baseline method. Ours w/o LD and Ours w/o SD denote our model without capturing long-term dependencies and short-term differences, respectively, and they are disabled by ignoring the loss. From the results, we have the following findings:

For the baseline methods, TransE-based models perform better than GNN-based GCN-Align in most cases. One possible reason is that GCN-Align weakens the role of relations and only expresses them numerically in the adjacency matrix. Among the TransE-based methods, BootEA always achieves better performance than SEA as it finds alignments in an efficient and iterative manner while SEA prevents entities of similar degree from being aggregated into the same region in the embedding space without discovering more alignments. It is noticeable that the sequence-based method RSNs outperforms all the other baselines, whether it is a TransE-based or GNN-based, on both the normal and dense datasets, which verifies the effectiveness and significance of capturing long-term relational dependencies between entities in KGs.

Compared to RSNs, the performance of Ours w/o SD demonstrates that introducing the self-attention mechanism is of importance to enhance the long-term dependency representations of entities. Ours w/o LD outperforms GCN-Align by a large margin and achieves comparable performance to some TransE-based approaches mainly because its multi-hop neighborhood concatenating layers allows GNN to capture the complex and hidden interactions existing in the surrounding entities. By considering both the long-term dependencies and short-term differences in KGs, our complete method consistently yields the best performance on all datasets in terms of all evaluation metrics, especially on Hits@1 and MRR, indicating the complementarity between the self-attention-based sequence model and the GNN-based model. In particular, our model improves over the best baseline RSNs w.r.t. Hits@1 by more than 10% and 5% on the normal and dense datasets, respectively. The superiority is more evident on the normal and monolingual datasets with more than 15% improvement. One possible reason is that cross-lingual datasets are extracted from one KG with different languages.

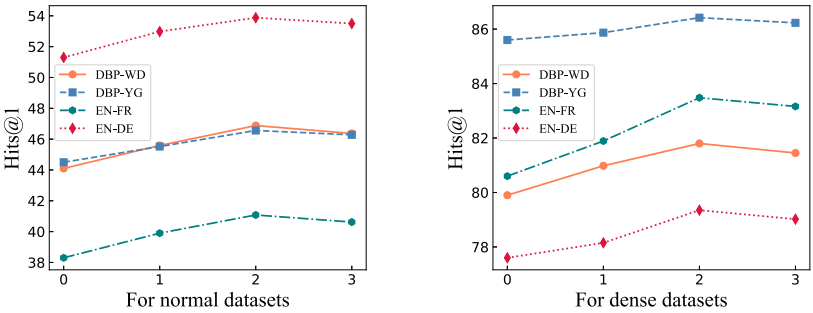

Fig. 4. Effect of the number of GNN layers t.

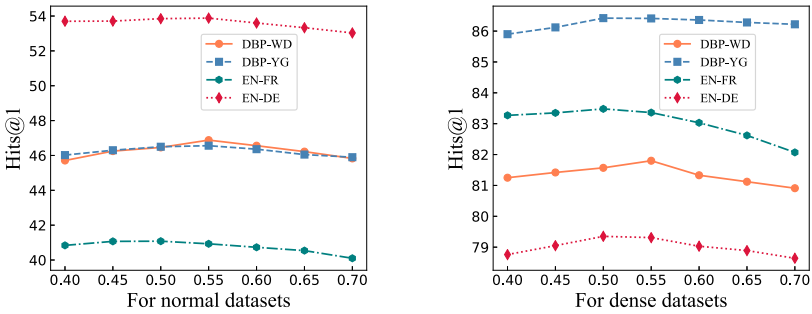

Fig. 5. Effect of parameter θ.

Hence they are less heterogeneous than monolingual datasets and thus easier to be represented. Moreover, compared with our model without long-term dependencies and short-term differences, the results justify the validity and efficacy of the complete model.

4.4 Model Analyses

In this subsection, we take a deep insight into the model to further understand the proposed architecture. As Hits@1 is the preferable metric, we only show the results on it. To evaluate the feasibility of our proposed crossed residual connection (Cross-R), we conduct experiments to compare it with the original norm residual connection (Norm-R). Both of them are under the same settings of long-term dependencies capturing model and disabled loss of short-term differences. Besides, we select two representative parameters for in-depth discussion.

Effect of the Crossed Residual Connection. Figure 3 shows the results of different residual connections on normal and dense DBP-WD dataset. Our proposed Cross-R achieves better performance with less epoch on both of them, which indicates that considering the unique triple structure is vital for learning long-term dependencies in KGs.

Table 6. Statistics of the person dataset

Datasets	Source KGs	#Entity	#Relation	#Triple
Person	English	15,170	2,228	203,502
	French	15,393	2,442	170,605

Table 7. Results on the person dataset

Model	MTransE	IPTransE	JAPE	GCN-Align	BootEA	SEA	RSNs	Ours
Hits@1	16.77	18.21	15.68	17.24	29.72	37.28	44.63	49.13
Hits@10	25.35	27.41	28.69	31.16	61.19	63.56	70.10	76.87
MRR	0.20	0.21	0.21	0.22	0.40	0.47	0.57	0.63

Effect of the Number of GNN Layers t**.** We vary the depth of GNN to investigate the impact of usage of GNN in our model. In particular, the layer number is searched in the range of {0,1,2,3} and the results are summarized in Fig. 4. We find it beneficial to increase the depth of GNN to boost performance on both normal and dense datasets. Two-layer GNN performs best across all the board, suggesting that updating the differences between adjacent two hops twice could be sufficient to capture short-term signals.

Effect of Parameter θ**.** Although we can infer the effectiveness of capturing long-term dependencies and short-term differences implicitly from Table 2, 3, 4 and 5, we would like to see how the two different modules contribute to our complete model. As depicted in Fig. 5, the parameter θ, which controls the contribution of the long-term dependency representations, is capable of balancing between the long-term dependencies and short-term differences. Jointly analyzing Table 2, 3, 4 and 5 with Fig. 5, we find that adaptively combining these two kinds of embeddings outperforms a single model and the best results are always achieved at $\theta = 0.55$. It indicates that the sequence model is of crucial significance and the GNN model is indispensable to enable the model with short-term differences capturing ability. Both of them play important roles in improving entity alignment performance.

4.5 Case Study

In this subsection, we perform an extra experimental evaluation on a specific domain provided in [16] to comprehensively evaluate the performance of our approach. The statistics of this person dataset is given in Table 6, which includes KGs in English and French, each of which contains more than 15,000 entities and 2,000 relations. There are 10,108 aligned entities between the two KGs.

The evaluation results are presented in Table 7. Our proposed model still outperforms other baselines on all metrics with about 10% improvement on Hits@1 and MRR compared to the second best method, which is consistent with the previous experiments and demonstrates the effectiveness of our model.

5 Conclusions

In this paper, short-term differences and long-term dependencies in KGs for entity alignment are illustrated for the first time. We propose a novel approach to manage to capture both short-term differences and long-term dependencies in KGs for our task. Specifically, we utilize and adapt the self-attention mechanism instead of RNN to model long-term dependencies between entities and devise a degree-aware random walk to generate high-quality sequences in KGs for learning. Next, to acquire the representation of the short-term differences of entities, we introduce a new graph neural network to treat triples as a graph, from which we can better interpret the short-term semantics of an entity by repeatedly mixing neighborhood information aggregated at various distances. Eventually, embeddings of entities can be obtained by combining short-term and long-term semantics in a linear way. Extensive experiments and analyses on four couples of real-world datasets demonstrate that our model consistently outperformed the state-of-the-art methods.

Acknowledgments. This research is partially supported by National Key R&D Program of China (No. 2018AAA0101900), the Priority Academic Program Development of Jiangsu Higher Education Institutions, Natural Science Foundation of Jiangsu Province (No. BK20191420), National Natural Science Foundation of China (Grant No. 62072323, 61632016), Natural Science Research Project of Jiangsu Higher Education Institution (No. 17KJA520003, 18KJA520010).

References

1. Abu-El-Haija, S., et al.: MixHop: higher-order graph convolution architectures via sparsified neighborhood mixing. In: Proceedings of the 36th International Conference on Machine Learning, vol. 97, pp. 21–29. PMLR, Long Beach (2019)
2. Bordes, A., Usunier, N., Garcia-Duran, A., Weston, J., Yakhnenko, O.: Translating embeddings for modeling multi-relational data. In: 27th Annual Conference on Neural Information Processing Systems 2013, Nevada, United States, pp. 2787–2795 (2013)
3. Chen, M., Tian, Y., Chang, K.W., Skiena, S., Zaniolo, C.: Co-training embeddings of knowledge graphs and entity descriptions for cross-lingual entity alignment. In: Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, Stockholm, Sweden, pp. 3998–4004 (2018)
4. Chen, M., Tian, Y., Yang, M., Zaniolo, C.: Multilingual knowledge graph embeddings for cross-lingual knowledge alignment. In: Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, Melbourne, Australia, pp. 1511–1517 (2017)
5. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 4171–4186 (2019)
6. Goyal, P., Ferrara, E.: Graph embedding techniques, applications, and performance: a survey. Knowl.-Based Syst. **151**, 78–94 (2018)

7. Grover, A., Leskovec, J.: node2vec: scalable feature learning for networks. In: Proceedings of the 22nd International Conference on Knowledge Discovery and Data Mining, pp. 855–864. ACM, San Francisco (2016)
8. Guo, L., Sun, Z., Hu, W.: Learning to exploit long-term relational dependencies in knowledge graphs. In: Proceedings of the 36th International Conference on Machine Learning, vol. 97, pp. 2505–2514. PMLR, Long Beach (2019)
9. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. In: 3rd International Conference on Learning Representations, San Diego, California (2015)
10. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. In: 5th International Conference on Learning Representations, Toulon, France (2017)
11. Li, Q., Han, Z., Wu, X.M.: Deeper insights into graph convolutional networks for semi-supervised learning. In: Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, New Orleans, Louisiana, pp. 3538–3545 (2018)
12. Mahdisoltani, F., Biega, J., Suchanek, F.M.: YAGO3: a knowledge base from multilingual Wikipedias. In: Seventh Biennial Conference on Innovative Data Systems Research, Asilomar, California (2015)
13. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: 27th Annual Conference on Neural Information Processing Systems, Lake Tahoe, Nevada, pp. 3111–3119 (2013)
14. Ngomo, A.C.N., Auer, S.: LIMES-a time-efficient approach for large-scale link discovery on the web of data. In: Twenty-Second International Joint Conference on Artificial Intelligence, Catalonia, Spain, pp. 2312–2317 (2011)
15. Pang, N., Zeng, W., Tang, J., Tan, Z., Zhao, X.: Iterative entity alignment with improved neural attribute embedding. In: Proceedings of the Workshop on Deep Learning for Knowledge Graphs, vol. 2377, pp. 41–46. CEUR-WS.org, Portoroz (2019)
16. Pei, S., Yu, L., Hoehndorf, R., Zhang, X.: Semi-supervised entity alignment via knowledge graph embedding with awareness of degree difference. In: The World Wide Web Conference, pp. 3130–3136. ACM, San Francisco (2019)
17. Sun, Z., Hu, W., Li, C.: Cross-lingual entity alignment via joint attribute-preserving embedding. In: d'Amato, C., et al. (eds.) ISWC 2017. LNCS, vol. 10587, pp. 628–644. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-68288-4_37
18. Sun, Z., Hu, W., Zhang, Q., Qu, Y.: Bootstrapping entity alignment with knowledge graph embedding. In: Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, Stockholm, Sweden, pp. 4396–4402 (2018)
19. Trisedya, B.D., Qi, J., Zhang, R.: Entity alignment between knowledge graphs using attribute embeddings. In: The Thirty-Third AAAI Conference on Artificial Intelligence, Honolulu, Hawaii, pp. 297–304 (2019)
20. Vaswani, A., et al.: Attention is all you need. In: Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems, Long Beach, California, pp. 5998–6008 (2017)
21. Velickovic, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., Bengio, Y.: Graph attention networks. In: 6th International Conference on Learning Representations, Vancouver, Canada (2018)
22. Wang, Z., Lv, Q., Lan, X., Zhang, Y.: Cross-lingual knowledge graph alignment via graph convolutional networks. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pp. 349–357. Association for Computational Linguistics, Brussels (2018)

23. Xu, K., et al.: Cross-lingual knowledge graph alignment via graph matching neural network. In: Proceedings of the 57th Conference of the Association for Computational Linguistics, pp. 3156–3161. Association for Computational Linguistics, Florence (2019)
24. Zhang, Q., Sun, Z., Hu, W., Chen, M., Guo, L., Qu, Y.: Multi-view knowledge graph embedding for entity alignment. In: Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, Macao, China, pp. 5429–5435 (2019)
25. Zhu, H., Xie, R., Liu, Z., Sun, M.: Iterative entity alignment via joint knowledge embeddings. In: Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, Melbourne, Australia, pp. 4258–4264 (2017)