Multi-view Knowledge Graph Embedding for Entity Alignment

Qingheng Zhang[1,*], Zequn Sun[1,*], Wei Hu[1,†], Muhao Chen[2], Lingbing Guo[1], 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
{qhzhang, zqsun, lbguo}.nju@gmail.com, {whu, yzqu}@nju.edu.cn, muhaochen@ucla.edu

Abstract

We study the problem of embedding-based entity alignment between knowledge graphs (KGs). Previous works mainly focus on the relational structure of entities. Some further incorporate another type of features, such as attributes, for refinement. However, a vast of entity features are still unexplored or not equally treated together, which impairs the accuracy and robustness of embedding-based entity alignment. In this paper, we propose a novel framework that unifies multiple views of entities to learn embeddings for entity alignment. Specifically, we embed entities based on the views of entity names, relations and attributes, with several combination strategies. Furthermore, we design some cross-KG inference methods to enhance the alignment between two KGs. Our experiments on real-world datasets show that the proposed framework significantly outperforms the state-of-the-art embedding-based entity alignment methods. The selected views, cross-KG inference and combination strategies all contribute to the performance improvement.

1 Introduction

Entity alignment, a.k.a. entity matching or resolution, aims to find entities in different knowledge graphs (KGs) referring to the same real-world identity. It plays a fundamental role in KG construction and fusion, and also supports many downstream applications, e.g., semantic search, question answering and recommender systems. Conventional methods for entity alignment identify similar entities based on the symbolic features, such as names, textual descriptions and attribute values. However, the computation of feature similarity often suffers from the semantic heterogeneity between different KGs [Isele and Bizer, 2013]. Recently, increasing attention has been paid to leveraging the KG embedding techniques for addressing this problem, where the key idea is to learn vector representations (called *embeddings*) of KGs and find entity alignment according to the similarity of the embeddings. The vector representations can benefit the task of learning similarities [Wu *et al.*, 2018]. Although existing embedding-based entity alignment methods have achieved promising results, they are still challenged by the following two limitations.

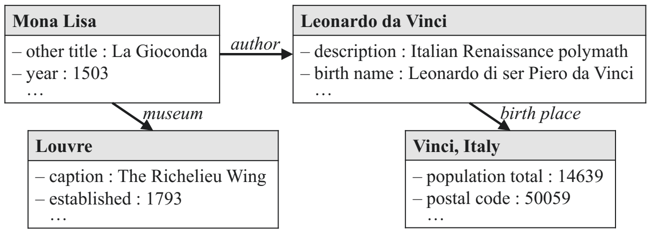
Mona Lisa
– other title : La Gioconda
– year : 1503
…
author
Leonardo da Vinci
– description : Italian Renaissance polymath
– birth name : Leonardo di ser Piero da Vinci
…
museum
Louvre
– caption : The Richelieu Wing
– established : 1793
…
birth place
Vinci, Italy
– population total : 14639
– postal code : 50059
…

Figure 1: An example of the multi-view features, e.g., names (denoted by **bold** font), relations (denoted by *italic* font) and attributes (denoted by regular font), of four entities in DBpedia.

First, entities in KGs have various features, but the current embedding-based entity alignment methods exploit just one or two types of them. For example, MTransE [Chen *et al.*, 2017], IPTransE [Zhu *et al.*, 2017] and BootEA [Sun *et al.*, 2018] only embed the relational structure of KGs for entity alignment, in addition to which JAPE [Sun *et al.*, 2017], KDCoE [Chen *et al.*, 2018] and AttrE [Trsedya *et al.*, 2019] complement attributes, textual descriptions or literals, respectively, to refine the embeddings. Actually, different types of features characterize different aspects of entity identities. Making use of them together would improve the accuracy and robustness.

Second, the existing embedding-based entity alignment methods rely on abundant *seed* entity alignment as labeled training data. However, in practice, such seed entity alignment is not always accessible and very costly to obtain [Isele and Bizer, 2013]. Furthermore, it is widely-acknowledged that entity alignment can benefit from relation and attribute alignment [Suchanek *et al.*, 2012]. Still, the existing methods [Zhu *et al.*, 2017; Trsedya *et al.*, 2019] assume that seed relation and attribute alignment can be easily found beforehand. In fact, learning embeddings from various features would enable the automatic population of more alignment information and relieve the reliance on the seed alignment.

To cope with the above limitations, we propose MultiKE, a new entity alignment framework based on multi-view KG embedding. The underlying idea is to divide the various features of KGs into multiple subsets (called *views*), which are complementary to each other (see Figure 1 for example). So entity embeddings can be learned from each particular view

*Equal contributors
†Corresponding author

and jointly optimized to improve the alignment performance. In summary, our main contributions are listed as follows:

- Based on the data model of KGs, we define three representative views based on the name, relation and attribute features. For each view, we employ an appropriate model to learn embeddings from it. (Section 3)
- For entity alignment, we design two cross-KG identity inference methods at the entity level as well as the relation and attribute level, respectively, to preserve and enhance the alignment between different KGs. (Section 4)
- We present three different strategies to combine multiple view-specific entity embeddings. Finally, we find entity alignment by the combined embeddings. (Section 5)
- Our experiments on two real-world datasets show that MultiKE largely outperforms existing embedding-based entity alignment methods. The selected views, cross-KG inference and combination strategies all contribute to the improvement. MultiKE also achieves promising results on unsupervised entity alignment and is comparable to conventional entity alignment methods. (Section 6)

2 Related Work

KG embedding. Learning KG embeddings has drawn much attention in recent years. Current KG embedding models can be divided into three kinds. *Translational* models, such as TransE [Bordes *et al.*, 2013], TransH [Wang *et al.*, 2014] TransR [Lin *et al.*, 2015] and TransD [Ji *et al.*, 2015], interpret a relation as a translation vector from a head entity to a tail. *Semantic matching* models use similarity-based functions to infer relation facts, e.g., the Hadamard product in DistMult [Yang *et al.*, 2015] and ComplEx [Trouillon *et al.*, 2016], and the circular correlation in HolE [Nickel *et al.*, 2016]. *Neural* models exploit deep learning techniques for KG embedding, e.g., multilayer perceptrons in ProjE [Shi and Weninger, 2017], convolutional neural networks in ConvE [Dettmers *et al.*, 2018], and graph convolutional networks in R-GCN [Schlichtkrull *et al.*, 2018]. All these models focus on relation facts and are mostly evaluated by the task of link prediction in a single KG.

Embedding-based entity alignment. The current methods represent different KGs as embeddings and find entity alignment by measuring the similarity between these embeddings. MTransE [Chen *et al.*, 2017] learns a mapping between two separate KG embedding spaces. IPTransE [Zhu *et al.*, 2017] and BootEA [Sun *et al.*, 2018] are two self-training methods, which embed two KGs in a unified space and iteratively label new entity alignment as supervision. GCN-Align [Wang *et al.*, 2018b] employs graph convolutional networks to model entities based on their neighborhood information. These methods align entities mostly based on the relation features of KGs. Additionally, several other methods leverage an additional type of features to enhance the relation-based embeddings. JAPE [Sun *et al.*, 2017] models the attribute correlations with Skip-Gram, KDCoE [Chen *et al.*, 2018] co-trains the embeddings of textual descriptions, and AttrE [Trsedya *et al.*, 2019] conducts character-level literal embeddings. However, all these features are not equally treated but used to refine the relation-based embeddings. More importantly, these methods are incapable of incorporating new features.

Multi-view representation learning. Learning representations from multi-view data can achieve strong generalization performance. Recently, multi-view representation learning has been widely used in the network embedding [Qu *et al.*, 2017; Wang *et al.*, 2018a] and natural language processing (NLP) [Clark *et al.*, 2018]. A typical process of multi-view representation learning is constituted by three major steps: (i) identify multiple views that can sufficiently represent the data; (ii) carry out representation learning on each view; and (iii) combine the multiple view-specific representations. We refer interested readers to [Li *et al.*, 2016] for more details.

3 Multi-view KG Embedding

3.1 Problem Statement

In this paper, we study multi-view KG embedding for entity alignment, which aims to learn comprehensive entity embeddings based on different views. Particularly, for entity alignment, we consider three views for an entity identity, namely the name view, the relation view and the attribute view. Names describe a basic and key characteristic of entities. In KGs, entity names are usually described, for example, by *rdfs:label*, or the local names of entity URIs[1]. The relations and attributes defined in KG schemata characterize entities with rich extrinsic and intrinsic information. Based on the view division above, we formalize a KG as a 7-tuple $\mathcal{G} = (\mathcal{E}, \mathcal{R}, \mathcal{A}, \mathcal{V}, \mathcal{N}, \mathcal{X}, \mathcal{Y})$, where $\mathcal{E}, \mathcal{R}, \mathcal{A}$ and $\mathcal{V}$ denote the sets of entities, relations, attributes and literals, respectively. $\mathcal{N} \subseteq \mathcal{E} \times \mathcal{V}$ denotes the name view of entities, $\mathcal{X} \subseteq \mathcal{E} \times \mathcal{R} \times \mathcal{E}$ denotes the relation view, and $\mathcal{Y} \subseteq \mathcal{E} \times \mathcal{A} \times \mathcal{V}$ denotes the attribute view.

Given a source KG $\mathcal{G}_a = (\mathcal{E}_a, \mathcal{R}_a, \mathcal{A}_a, \mathcal{V}_a, \mathcal{N}_a, \mathcal{X}_a, \mathcal{Y}_a)$ and a target KG $\mathcal{G}_b = (\mathcal{E}_b, \mathcal{R}_b, \mathcal{A}_b, \mathcal{V}_b, \mathcal{N}_b, \mathcal{X}_b, \mathcal{Y}_b)$, entity alignment aims to find a set of identical entities $\mathcal{M} = \{(e_i, e_j) \in \mathcal{E}_a \times \mathcal{E}_b \mid e_i \equiv e_j\}$, where "$\equiv$" denotes the equivalence relationship. We first learn d-dimensional entity embeddings and then find entity alignment based on the embeddings.

For notations, we use bold lowercase letters to represent embeddings and bold uppercase letters to matrices. The superscript with bracket of an entity embedding indicates from which view this embedding comes. The name, relation and attribute views are marked by $^{(1)}$,$^{(2)}$ and $^{(3)}$, respectively.

3.2 Literal Embedding

Literals are constituted by sequences of tokens. As the basis for multi-view embedding, literal embedding represents the discrete and symbolic literals by d-dimensional embeddings. Without loss of generality, let $l = (o_1, o_2, \dots, o_n)$ denote a literal of n tokens. LP$(\cdot)$ is defined as a lookup function that maps the input to an embedding. To resolve the different expressions of literals and capture their inherent semantics, we consider the following cases:

$$\text{LP}(o_i) = \begin{cases} \text{word\_embed}(o_i) & \text{if } o_i \text{ has a word embedding} \\ \text{char\_embed}(o_i) & \text{otherwise} \end{cases}, \quad (1)$$

[1]A local name is the string after the last hash or slash of a URI.

where word_embed$(\cdot)$ returns the corresponding word embedding of the input token. In fact, word embeddings have become a fundamental resource for many NLP applications, due to their capability of capturing semantic relatedness of words. Here, we use pre-trained word embeddings [Mikolov *et al.*, 2018] to collocate similar literals. char_embed$(\cdot)$ returns the average of the character embeddings that are pre-trained with the Skip-Gram model [Mikolov *et al.*, 2013] on the KG literal set $\mathcal{V}_a \cup \mathcal{V}_b$. Let $\phi(\cdot)$ be the literal embedding of the input. We employ an autoencoder to encode a list of token embeddings into one literal embedding in an unsupervised manner:

$$\phi(l) = \text{encode}([\text{LP}(o_1); \text{LP}(o_2); \dots; \text{LP}(o_n)]), \quad (2)$$

where encode$(\cdot)$ returns the compressed representation of the input embeddings and $[;]$ denotes the concatenation operation. We limit the maximum number of tokens to 5. Long literals are truncated while short ones are appended with placeholders.

3.3 Name View Embedding

We embed the name view using the above literal embeddings. Given an entity h, its name embedding is defined as follows:

$$\mathbf{h}^{(1)} = \phi(\text{name}(h)), \quad (3)$$

where name$(\cdot)$ extracts the name of the input object. We refer to the KG embeddings under the name view as $\Theta^{(1)}$.

3.4 Relation View Embedding

The relation view characterizes the structure of KGs, in which entities are linked by relations. To preserve such relational structures, we adopt TransE [Bordes *et al.*, 2013] to interpret a relation as a translation vector from its head entity to tail entity. Given a relation fact (h, r, t) in KGs, we use the following score function to measure the plausibility of the embeddings:

$$f_{\text{rel}}(\mathbf{h}^{(2)}, \mathbf{r}, \mathbf{t}^{(2)}) = -||\mathbf{h}^{(2)} + \mathbf{r} - \mathbf{t}^{(2)}||, \quad (4)$$

where $||\cdot||$ denotes either L_1 or L_2 vector norm. Then, we define the probability of (h, r, t) being a real relation fact (i.e., existing in the KGs) as follows:

$$\mathcal{P}_{\text{rel}}(\zeta_{(h,r,t)} = 1 \mid \Theta^{(2)}) = \text{sigmoid}(f_{\text{rel}}(\mathbf{h}^{(2)}, \mathbf{r}, \mathbf{t}^{(2)})), \quad (5)$$

where $\Theta^{(2)}$ denotes the KG embeddings under the relation view and $\zeta_{(h,r,t)}$ denotes the label ("1" or "-1") of (h, r, t). We parameterize $\Theta^{(2)}$ by minimizing the logistic loss below:

$$\mathcal{L}(\Theta^{(2)}) = \sum_{(h,r,t) \in \mathcal{X}^+ \cup \mathcal{X}^-} \log(1 + \exp(-\zeta_{(h,r,t)} f_{\text{rel}}(\mathbf{h}^{(2)}, \mathbf{r}, \mathbf{t}^{(2)}))), \quad (6)$$

where $\mathcal{X}^+ = \mathcal{X}_a \cup \mathcal{X}_b$ denotes the set of real relation facts in the source and target KGs, while $\mathcal{X}^-$ denotes the set of faked ones sampled by replacing the head or tail entities of real relation facts with random ones.

3.5 Attribute View Embedding

For the attribute view, we use a convolutional neural network (CNN) to extract features from the attributes and values of entities. We splice the embeddings of an attribute a and its value v into a matrix, denoted by $\langle \mathbf{a}; \mathbf{v} \rangle \in \mathbb{R}^{2 \times d}$, and feed it to a CNN to obtain the compressed representation:

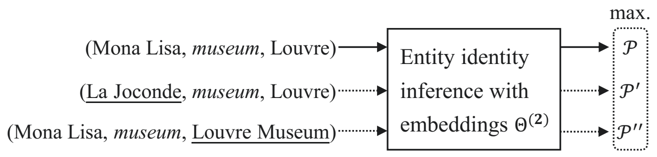
max.
(Mona Lisa, *museum*, Louvre)
(La Joconde, *museum*, Louvre)
(Mona Lisa, *museum*, Louvre Museum)
Entity identity inference with embeddings $\Theta^{(2)}$
$\mathcal{P}$
$\mathcal{P}'$
$\mathcal{P}''$

Figure 2: Illustration of cross-KG entity identity inference under the relation view, where entities without and with underlines come from DBpedia and Wikidata, respectively. Dotted arrows denote auxiliary inference probabilities.

$$\text{CNN}(\langle \mathbf{a}; \mathbf{v} \rangle) = \sigma(\text{vec}(\sigma(\langle \mathbf{a}; \mathbf{v} \rangle * \Omega))\mathbf{W}), \quad (7)$$

where CNN$(\cdot)$ denotes a convolution operation that slides a convolution kernel Ω of size $2 \times c$ $(c < d)$ over the input embedding matrix to extract high-level features. The feature map tensor is then reshaped to a vector by vec$(\cdot)$ and finally projected in the KG embedding space using a densely-connected layer parametrized by $\mathbf{W}$. $\sigma(\cdot)$ is an activation function.

Given an attribute fact (h, a, v) in KGs, we define the following score function to measure its plausibility:

$$f_{\text{attr}}(\mathbf{h}^{(3)}, \mathbf{a}, \mathbf{v}) = -||\mathbf{h}^{(3)} - \text{CNN}(\langle \mathbf{a}; \mathbf{v} \rangle)||. \quad (8)$$

Based on this, the head entities are expected to be close to its attributes and values. This objective can be achieved by minimizing the following logistic loss:

$$\mathcal{L}(\Theta^{(3)}) = \sum_{(h,a,v) \in \mathcal{Y}^+} \log(1 + \exp(-f_{\text{attr}}(\mathbf{h}^{(3)}, \mathbf{a}, \mathbf{v}))), \quad (9)$$

where $\mathcal{Y}^+ = \mathcal{Y}_a \cup \mathcal{Y}_b$ denotes the set of real attribute facts in the source and target KGs, and $\Theta^{(3)}$ denotes the KG embeddings under the attribute view. Negative sampling is not employed here, because we find that it does not lead to noticeable improvement on entity alignment.

4 Cross-KG Training for Entity Alignment

In this section, we propose the cross-KG entity identity inference to capture the alignment information between two KGs, based on seed entity alignment. Furthermore, we present the cross-KG relation and attribute identity inference to enhance entity alignment, based on the automatically found relation and attribute alignment.

4.1 Entity Identity Inference

As shown in Figure 2, we take as an example the cross-KG entity identity inference under the relation view. Our intuition is that swapping aligned entities in their involved relation facts should lead to the same probability of identity inference, because the entities refer to the same object in the real world [Zhu *et al.*, 2017; Sun *et al.*, 2018]. Given a relation fact (h, r, t), if $(h, \hat{h})$ appears in the seed entity alignment, we add the following auxiliary probability:

$$\mathcal{P}'_{\text{rel}}(\zeta_{(h,r,t)} = 1 \mid \Theta^{(2)}) = \text{sigmoid}(f_{\text{rel}}(\hat{\mathbf{h}}^{(2)}, \mathbf{r}, \mathbf{t}^{(2)})). \quad (10)$$

Also, if $(t, \hat{t})$ exists in the seed entity alignment, we add:

$$\mathcal{P}''_{\text{rel}}(\zeta_{(h,r,t)} = 1 \mid \Theta^{(2)}) = \text{sigmoid}(f_{\text{rel}}(\mathbf{h}^{(2)}, \mathbf{r}, \hat{\mathbf{t}}^{(2)})). \quad (11)$$

We maximize these auxiliary probabilities over the relation facts having those entities in the seed entity alignment. The loss is computed as follows:

$$\mathcal{L}_{\text{CE}}(\Theta^{(2)}) = \sum_{(h,r,t)\in\mathcal{X}'} \log(1+\exp(-f_{\text{rel}}(\hat{\mathbf{h}}^{(2)},\mathbf{r},\mathbf{t}^{(2)}))) + \sum_{(h,r,t)\in\mathcal{X}''} \log(1+\exp(-f_{\text{rel}}(\mathbf{h}^{(2)},\mathbf{r},\hat{\mathbf{t}}^{(2)}))), \quad (12)$$

where $\mathcal{X}'$ and $\mathcal{X}''$ refer to the sets of relation facts whose head and tail entities are in the seed entity alignment, respectively.

For the cross-KG entity inference under the attribute view, the loss, denoted by $\mathcal{L}_{\text{CE}}(\Theta^{(3)})$, is similarly defined:

$$\mathcal{L}_{\text{CE}}(\Theta^{(3)}) = \sum_{(h,a,v)\in\mathcal{Y}'} \log(1+\exp(-f_{\text{attr}}(\hat{\mathbf{h}}^{(3)},\mathbf{a},\mathbf{v}))), \quad (13)$$

where $\mathcal{Y}'$ denotes the set of attribute facts such that the head entities are in the seed entity alignment.

4.2 Relation and Attribute Identity Inference

Similar to the cross-KG entity identity inference, we add the auxiliary probabilities for the cross-KG relation and attribute identity inference. Let us consider the relations for example. Given a relation fact (h,r,t), if $(r,\hat{r})$ constitutes a relation alignment, we have:

$$\mathcal{P}'''_{\text{rel}}(\zeta_{(h,r,t)} = 1 \mid \Theta^{(2)}) = \text{sigmoid}(f_{\text{rel}}(\mathbf{h}^{(2)},\hat{\mathbf{r}},\mathbf{t}^{(2)})). \quad (14)$$

Unlike the previous works [Zhu *et al.*, 2017; Trsedya *et al.*, 2019] that assume the existence of seed relation and attribute alignment, we propose a *soft* alignment method to automatically find the relation and attribute alignment in training. Here, "soft" means that we do not require the relations or attributes in the alignment to be strictly equivalent. In fact, due to the heterogeneity between KG schemata, finding relation and attribute alignment can be harder than entity alignment [Cheatham and Hitzler, 2014], and sometimes the strict alignment even does not exist. We denote the soft relation alignment by $\mathcal{S}_{\text{rel}} = \{(r,\hat{r},\text{sim}(r,\hat{r})) \mid \text{sim}(r,\hat{r}) \geq \eta\}$, where $r,\hat{r}$ are relations from different KGs, $\text{sim}(r,\hat{r})$ denotes their similarity, and η denotes a threshold in $(0,1]$. We consider two kinds of similarity: name similarity based on literal embeddings and semantic similarity based on relation embeddings. We combine them as a weighted sum:

$$\text{sim}(r,\hat{r}) = \alpha_1 \cos(\phi(\text{name}(r)),\phi(\text{name}(\hat{r}))) + \alpha_2 \cos(\mathbf{r},\hat{\mathbf{r}}), \quad (15)$$

where $\alpha_1,\alpha_2 > 0$ are two weighting factors and $\alpha_1+\alpha_2 = 1$. $\cos(\cdot)$ calculates the cosine similarity of two embeddings.

We regard this similarity as a smooth coefficient to reduce the negative effect of inaccurate alignment and combine it into the loss of cross-KG relation identity inference:

$$\mathcal{L}_{\text{CRA}}(\Theta^{(2)}) = \sum_{(h,r,t)\in\mathcal{X}'''} \text{sim}(r,\hat{r}) \log(1+\exp(-f_{\text{rel}}(\mathbf{h}^{(2)},\hat{\mathbf{r}},\mathbf{t}^{(2)}))), \quad (16)$$

where $\mathcal{X}'''$ denotes the set of relation facts having the relations in $\mathcal{S}_{\text{rel}}$. Note that, the soft relation alignment is not fixed but updated iteratively during the training process of MultiKE.

The cross-KG attribute identify inference can be formulated in the same way. Due to the space limitation, we omit its loss function $\mathcal{L}_{\text{CRA}}(\Theta^{(3)})$ here.

5 View Combination

The view-specific embeddings characterize entity identities from different aspects. Intuitively, the general entity embeddings can benefit from multiple view-specific embeddings. In this section, we present three combination strategies.

5.1 Weighted View Averaging

A straightforward combination is to average the embeddings from different views. To emphasize on important views, we assign weights to view-specific entity embeddings. Let $\tilde{\mathbf{h}}$ denote the combined embedding for h. Without loss of generality, let D be the number of views, and we have $\tilde{\mathbf{h}} = \sum_{i=1}^{D} w_i \mathbf{h}^{(i)}$, where w_i is the weight of $\mathbf{h}^{(i)}$, and can be calculated by:

$$w_i = \frac{\cos(\mathbf{h}^{(i)},\bar{\mathbf{h}})}{\sum_{j=1}^{D}\cos(\mathbf{h}^{(j)},\bar{\mathbf{h}})}, \quad (17)$$

where $\bar{\mathbf{h}}$ is the average of multi-view embeddings of h, i.e., $\bar{\mathbf{h}} = \frac{1}{D}\sum_{j=1}^{D}\mathbf{h}^{(j)}$. If the embedding from one view is far away from its average embedding, it would have a lower weight. This is a kind of late combination, because it aggregates embeddings after they have been learned independently.

5.2 Shared Space Learning

This combination strategy seeks to induce an orthogonal mapping matrix from each view-specific embedding space to a shared space, based on the assumption that multiple views can be generated from the shared latent view. Let $\tilde{\mathbf{H}}$ be the combined embedding matrix for all entities, where each row corresponds to an entity, and $\mathbf{H}^{(i)}$ be the entity embedding matrix under the i^{th} view. We minimize the mapping loss:

$$\mathcal{L}_{\text{SSL}}(\tilde{\mathbf{H}},\mathbf{Z}) = \sum_{i=1}^{D}(||\tilde{\mathbf{H}} - \mathbf{H}^{(i)}\mathbf{Z}^{(i)}||_F^2 + ||\mathbf{I} - \mathbf{Z}^{(i)\top}\mathbf{Z}^{(i)}||_F^2), \quad (18)$$

where $\mathbf{Z} = \bigcup_{i=1}^{D}\mathbf{Z}^{(i)}$. $\mathbf{Z}^{(i)}$ serves as the mapping from the i^{th} view-specific embedding space to the shared space, and $||\cdot||_F$ denotes the Frobenius norm. The second term is used as a soft constraint to orthogonalize mapping matrices, where $\mathbf{I}$ is the identity matrix. The orthogonality helps keep the distances between embeddings in the view-specific space unchanged during the transformation, and thus the shared space can learn the alignment information from the multiple embedding spaces. Note that this is also a kind of late combination.

5.3 In-training Combination

Unlike the above strategies, this combination participates in the joint training of multi-view embeddings, which enables the multiple views to benefit from each other. The loss is:

$$\mathcal{L}_{\text{ITC}}(\tilde{\mathbf{H}},\mathbf{H}) = \sum_{i=1}^{D}||\tilde{\mathbf{H}} - \mathbf{H}^{(i)}||_F^2, \quad (19)$$

where $\mathbf{H} = \bigcup_{i=1}^{D}\mathbf{H}^{(i)}$. The goal is to maximize the agreement between the combined embeddings and the view-specific embeddings in a unified embedding space.

Algorithm 1: Combined training process of MultiKE

```
   Input: $\mathcal{G}_a, \mathcal{G}_b$, word embeddings, max epochs $Q$
1  Train literal embeddings and get the name embeddings;
2  for $q = 1, 2, \dots, Q$ do
3      Minimize $\mathcal{L}(\Theta^{(2)})$ under the relation view;
4      Minimize $\mathcal{L}(\Theta^{(3)})$ under the attribute view;
5      if in-training combination is used then
6          Minimize $\mathcal{L}_{\text{ITC}}(\tilde{\mathbf{H}}, \mathbf{H})$;
7      Minimize $\mathcal{L}_{\text{CE}}(\Theta^{(2)})$ and $\mathcal{L}_{\text{CE}}(\Theta^{(3)})$;
8      Update soft alignment $\mathcal{S}_{\text{rel}}$ and $\mathcal{S}_{\text{attr}}$;
9      Minimize $\mathcal{L}_{\text{CRA}}(\Theta^{(2)})$ and $\mathcal{L}_{\text{CRA}}(\Theta^{(3)})$;
10 if in-training combination is not used then
11     Run weighted view averaging or shared space learning;
12 Find entity alignment in $\tilde{\mathbf{H}}$ by nearest-neighbor search;
```

5.4 Combined Training Process

The complete training process is shown in Algorithm 1. The parameters are initialized with Xavier initializer and the loss functions are optimized with AdaGrad. We first train literal embeddings based on pre-train word embeddings [Mikolov *et al.*, 2018] and character embeddings. Thus, we can directly obtain entity name embeddings. Then, we train (and combine, if in-training combination is used) embeddings from other views and perform the cross-KG entity, relation and attribute identity inference alternately. The soft alignment of relations and attributes is also updated iteratively during training. Finally, we combine the multiple view-specific entity embeddings if in-training combination is not used, and find entity alignment by the nearest-neighbor search. Note that, the name embeddings are retrieved from literal embeddings and they would not be updated in the follow-up training process.

6 Experiments

In this section, we report our experiments on MultiKE. The source code is accessible online[2].

6.1 Datasets

In our experiments, we reused two datasets, namely DBP-WD and DBP-YG, recently proposed in [Sun *et al.*, 2018]. The two datasets were sampled from DBpedia, Wikidata and YAGO3, each of which contains 100 thousand aligned entity pairs, i.e., reference entity alignment. Each dataset provides 30% reference entity alignment as seed and leaves the remaining for evaluating entity alignment performance.

6.2 Comparative Methods

We compared MultiKE with seven recent embedding-based entity alignment methods, namely MTransE, IPTransE, JAPE, BootEA, KDCoE, GCN-Align and AttrE, which have been described in Section 2. We also extended TransD, HolE and ConvE for entity alignment on behalf of the three kinds of KG embedding models, because we found that their results were most competitive in our preliminary experiments. We merged the two KGs in a dataset as one by letting the two entities in each seed entity alignment have the same embedding and used these extended models to learn embeddings. We denote the MultiKE variants that employ weighted view averaging, shared space learning and in-training combination by MultiKE-WVA, MultiKE-SSL and MultiKE-ITC, respectively.

6.3 Experimental Settings

The following hyper-parameters were used in the experiments. Each training took $Q = 200$ epochs with learning rate 0.001. For the relation view embedding, 10 negative facts were sampled for each real relation fact. For the attribute view embedding, the number of filters was 2 and the convolution kernel size was 2×4 (i.e., $c = 4$). The activation function for the autoencoder and CNN was $\tanh(\cdot)$. For the relation and attribute identity inference, we set $\alpha_1 = 0.6, \alpha_2 = 0.4$ and $\eta = 0.9$. The embedding dimension d was set to 75 for all the comparative methods. We chose Hits@k ($k = 1, 10$), mean rank (MR) and mean reciprocal rank (MRR) as the evaluation metrics. Higher Hits@k and MRR scores as well as lower MR scores indicate better performance. Note that, Hits@1 should be more preferable, and it is equivalent to precision widely-used in conventional entity alignment.

6.4 Entity Alignment Results and Analyses

Table 1 shows the comparison results of MultiKE and other embedding-based entity alignment methods. We found that MultiKE significantly outperforms the others on all the metrics across the two datasets. For example, on DBP-WD, MultiKE-SSL achieved a Hits@1 score of 91.86% with 17.07% absolute gain compared to BootEA (the second best method). This is because MultiKE explores the multi-view features of entity identities, while the others like MTransE, IPTransE and BootEA only learn from relation facts. As compared with AttrE, which leverages literals for entity alignment, MultiKE also achieved superior results (e.g., 52.9% improvement of Hits@1 on DBP-WD). This is due to the fact that the pre-trained word embeddings in MultiKE can better capture the semantic similarity of literals than the character embeddings used in AttrE. Furthermore, AttrE embeds entities and literals in a unified space by TransE, which would fall short of handling the diversity of attribute values. Our three variants all achieved similar results. Note that the performance of MultiKE-SSL slightly decreased on DBP-YG. We think that this is because DBP-YG has stronger relation and attribute heterogeneity. For example, their relation numbers are 302 vs. 31, and the attribute numbers are 334 vs. 23, respectively. This hinders the combination by shared space learning.

Effectiveness of views. Table 2 depicts the entity alignment results based on view-specific entity embeddings when trained independently or by in-training combination. The three views all contributed to entity alignment, especially the name view. We owe it to the proposed literal embeddings, which can capture the semantic similarity of entity names. With in-training combination, the relation and attribute views benefited from the name view and also each other, thus their results improved a lot. As name embeddings are fixed, entity alignment results in the name view are the same in different combinations. This experiment indicated that entity names and word embeddings have great potentials for capturing the entity similarity.

[2]https://github.com/nju-websoft/MultiKE

Features	Methods	DBP-WD				DBP-YG			
		Hits@1	Hits@10	MR	MRR	Hits@1	Hits@10	MR	MRR
Relation only	MTransE	28.12†	51.95†	656	0.363†	25.15†	49.29†	512	0.334†
	IPTransE	34.85†	63.84†	265	0.447†	29.74†	55.76†	158	0.386†
	BootEA	74.79†	89.84†	109	0.801†	76.10†	89.44†	34	0.808†
	GCN-Align	47.70	75.96	1,988	0.577	60.05	84.14	299	0.686
	TransD	36.20	65.08	152	0.456	33.49	59.72	114	0.421
	HolE	22.26	45.22	811	0.289	25.04	48.36	629	0.327
	ConvE	40.31	62.78	1,429	0.483	50.27	73.56	837	0.582
Rel. + Attr.	JAPE	31.84†	58.88†	266	0.411†	23.57†	48.41†	189	0.320†
Rel. + Textual desc.	KDCoE	57.19	69.53	182	0.618	42.71	48.30	137	0.446
Rel. + Literals	AttrE	38.96	66.77	142	0.487	23.24	42.70	706	0.300
Multiple views	MultiKE-WVA	90.42	94.59	**22**	0.921	85.92	94.99	**19**	0.891
	MultiKE-SSL	**91.86**	**96.26**	39	**0.935**	82.35	93.30	21	0.862
	MultiKE-ITC	91.45	95.19	114	0.928	**88.03**	**95.32**	35	**0.906**

"†" indicates that the results were taken from [Sun *et al.*, 2018]. Other results were produced using their source code.

Table 1: Comparison with existing embedding-based entity alignment methods

		DBP-WD			DBP-YG		
		Hits@1	MR	MRR	Hits@1	MR	MRR
Indep.	Name view	84.30	1,108	0.871	83.60	1,294	0.841
	Rel. view	54.24	169	0.635	59.89	58	0.680
	Attr. view	17.32	7,688	0.215	51.36	1,966	0.558
ITC	Rel. view	70.22	34	0.769	68.57	47	0.751
	Attr. view	61.13	6,774	0.648	62.52	1,001	0.688

Table 2: Results of entity alignment under independent views

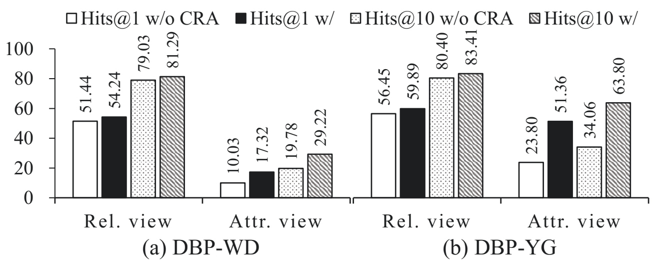

Figure 3: Results under the relation and attribute views, with or without cross-KG relation or attribute identity inference.

Effectiveness of cross-KG inference. Here, we examined the effectiveness of the proposed relation and attribute identity inference. We additionally performed KG embedding under the relation and attribute views without optimizing $\mathcal{L}_{\text{CRA}}(\Theta^{(2)})$ and $\mathcal{L}_{\text{CRA}}(\Theta^{(3)})$, respectively. We denote such variants by "w/o CRA". The entity alignment results under the relation and attribute views are shown in Figure 3. We observed that the relation and attribute identity inference brought improvement to entity alignment, especially in the attribute view. This is because the attribute heterogeneity is weaker than relation heterogeneity between different KGs. The soft attribute alignment was more accurate and thus contributed more.

Analysis of Unsupervised Entity Alignment. This task aims to align entities without seed entity alignment. As found in [Sun *et al.*, 2017; Wang *et al.*, 2018b], seed entity alignment is vital to learn KG embeddings from relation facts. We also encountered the same issue that the relation view would fall short if no seed entity alignment is given. However, as shown in Table 3, MultiKE-ITC still achieved acceptable results, thanks to the special in-training combination. The name view does not rely on seed alignment as supervision, while the relation and attribute views can benefit from it during training.

	DBP-WD			DBP-YG		
	Hits@1	MR	MRR	Hits@1	MR	MRR
Rel. view	67.59	301	0.726	12.71	3,504	0.174
Attr. view	49.08	986	0.551	57.78	701	0.630
MultiKE-ITC	83.98	421	0.866	58.46	732	0.636

Table 3: Results of unsupervised entity alignment with MultiKE-ITC

	DBP-WD			DBP-YG		
	Prec.	Recall	F1-score	Prec.	Recall	F1-score
LogMap	86.61	80.68	83.54	82.71	83.73	83.22
MultiKE	**91.45**	**91.45**	**91.45**	**88.03**	**88.03**	**88.03**

Table 4: Comparison with LogMap

This experiment revealed that MultiKE has good robustness and can alleviate the reliance on seed alignment.

Comparison with Conventional Methods. We compared MultiKE with a famous and open-source conventional entity alignment method LogMap (version 2.4) [Jiménez-Ruiz *et al.*, 2012]. In Table 4, we depict the results measured by the conventional precision, recall and F1-score. For embedding-based entity alignment, recall and F1-score are equal to precision, because the embedding-based methods always return a list of candidates for every input entity. LogMap is very competitive. In fact, it outperformed other comparative methods in Table 1. MultiKE achieved better results than LogMap, which again demonstrated its effectiveness and practicability.

7 Conclusion and Future Work

In this paper, we proposed a multi-view KG embedding framework for entity alignment, which learns entity embeddings from three representative views of KGs. We introduced two cross-KG training methods for alignment inference. We also designed three kinds of strategies to combine view-specific embeddings together. Our experiments on two real-world datasets demonstrated the effectiveness of our framework. In future work, we plan to investigate more feasible views (e.g., entity types) and study cross-lingual entity alignment.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (Nos. 61872172, 61772264) and the Collaborative Innovation Center of Novel Software Technology and Industrialization.

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 *NIPS*, pages 2787–2795, 2013.

[Cheatham and Hitzler, 2014] Michelle Cheatham and Pascal Hitzler. The properties of property alignment. In *ISWC Workshop on Ontology Matching*, 2014.

[Chen *et al.*, 2017] Muhao Chen, Yingtao Tian, Mohan Yang, and Carlo Zaniolo. Multilingual knowledge graph embeddings for cross-lingual knowledge alignment. In *IJCAI*, pages 1511–1517, 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 *IJCAI*, pages 3998–4004, 2018.

[Clark *et al.*, 2018] Kevin Clark, Minh-Thang Luong, Christopher D. Manning, and Quoc V. Le. Semi-supervised sequence modeling with cross-view training. In *EMNLP*, pages 1914–1925, 2018.

[Dettmers *et al.*, 2018] Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. Convolutional 2D knowledge graph embeddings. In *AAAI*, pages 1811–1818, 2018.

[Isele and Bizer, 2013] Robert Isele and Christian Bizer. Active learning of expressive linkage rules using genetic programming. *Journal of Web Semantics*, 23:2–15, 2013.

[Ji *et al.*, 2015] Guoliang Ji, Shizhu He, Liheng Xu, Kang Liu, and Jun Zhao. Knowledge graph embedding via dynamic mapping matrix. In *ACL*, pages 687–696, 2015.

[Jiménez-Ruiz *et al.*, 2012] Ernesto Jiménez-Ruiz, Bernardo Cuenca Grau, Yujiao Zhou, and Ian Horrocks. Large-scale interactive ontology matching: Algorithms and implementation. In *ECAI*, pages 444–449, 2012.

[Li *et al.*, 2016] Yingming Li, Ming Yang, and Zhongfei Zhang. Multi-view representation learning: A survey from shallow methods to deep methods. *CoRR*, abs/1610.01206, 2016.

[Lin *et al.*, 2015] Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. Learning entity and relation embeddings for knowledge graph completion. In *AAAI*, pages 2181–2187, 2015.

[Mikolov *et al.*, 2013] Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. In *NIPS*, pages 3111–3119, 2013.

[Mikolov *et al.*, 2018] Tomas Mikolov, Edouard Grave, Piotr Bojanowski, Christian Puhrsch, and Armand Joulin. Advances in pre-training distributed word representations. In *LREC*, 2018.

[Nickel *et al.*, 2016] Maximilian Nickel, Lorenzo Rosasco, and Tomaso A. Poggio. Holographic embeddings of knowledge graphs. In *AAAI*, pages 1955–1961, 2016.

[Qu *et al.*, 2017] Meng Qu, Jian Tang, Jingbo Shang, Xiang Ren, Ming Zhang, and Jiawei Han. An attention-based collaboration framework for multi-view network representation learning. In *CIKM*, pages 1767–1776, 2017.

[Schlichtkrull *et al.*, 2018] Michael Sejr Schlichtkrull, Thomas N. Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. Modeling relational data with graph convolutional networks. In *ESWC*, pages 593–607, 2018.

[Shi and Weninger, 2017] Baoxu Shi and Tim Weninger. ProjE: Embedding projection for knowledge graph completion. In *AAAI*, pages 1236–1242, 2017.

[Suchanek *et al.*, 2012] Fabian M. Suchanek, Serge Abiteboul, and Pierre Senellart. PARIS: Probabilistic alignment of relations, instances, and schema. *PVLDB*, 5(3):157–168, 2012.

[Sun *et al.*, 2017] Zequn Sun, Wei Hu, and Chengkai Li. Cross-lingual entity alignment via joint attribute-preserving embedding. In *ISWC*, pages 628–644, 2017.

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

[Trouillon *et al.*, 2016] Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. Complex embeddings for simple link prediction. In *ICML*, pages 2071–2080, 2016.

[Trsedya *et al.*, 2019] Bayu Distiawan Trsedya, Jianzhong Qi, and Rui Zhang. Entity alignment between knowledge graphs using attribute embeddings. In *AAAI*, 2019.

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

[Wang *et al.*, 2018a] Yueyang Wang, Liang Hu, Yueting Zhuang, and Fei Wu. Intra-view and inter-view attention for multi-view network embedding. In *PCM*, pages 201–211, 2018.

[Wang *et al.*, 2018b] Zhichun Wang, Qingsong Lv, Xiaohan Lan, and Yu Zhang. Cross-lingual knowledge graph alignment via graph convolutional networks. In *EMNLP*, pages 349–357, 2018.

[Wu *et al.*, 2018] Ledell Yu Wu, Adam Fisch, Sumit Chopra, Keith Adams, Antoine Bordes, and Jason Weston. Starspace: Embed all the things! In *AAAI*, pages 5569–5577, 2018.

[Yang *et al.*, 2015] Bishan Yang, Wen tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. Embedding entities and relations for learning and inference in knowledge bases. In *ICLR*, 2015.

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