APSIPA Transactions on Signal and Information Processing, 2023, 12, e5

Original Paper

TypeEA: Type-Associated Embedding for Knowledge Graph Entity Alignment

Xiou Ge[1*], Yun Cheng Wang[1], Bin Wang[2] and C.-C. Jay Kuo[1]

[1] *University of Southern California, Los Angeles, CA, USA*
[2] *National University of Singapore, Singapore*

ABSTRACT

Entity alignment is commonly used to link different knowledge graphs and augment facts about entities. The main objective is to identify the counterpart of a source entity in the target knowledge graph. Although the auxiliary information such as textual, visual, and temporal features was leveraged to improve the entity alignment performance in the past, the entity type information is rarely considered in existing entity alignment models. In this paper, we demonstrate that the entity type information, which is commonly available in knowledge graphs, is very helpful to knowledge graph alignment and propose a new method called the Type-associated Entity Alignment (TypeEA) accordingly. TypeEA exploits the entity type information to guide entity alignment models so that they can focus on entities with matching types. A type embedding model based on semantic matching is developed in TypeEA to capture the association between types in different knowledge graphs. Experimental results show that the proposed TypeEA consistently outperforms state-of-the-art baselines across all OpenEA entity alignment datasets with different experimental settings.

Keywords: Knowledge graph, entity alignment, type embeddings.

*Corresponding author: Xiou Ge, xiouge@usc.edu.

Received 27 June 2022; Revised 08 November 2022
ISSN 2048-7703; DOI 10.1561/116.00000139

1 Introduction

The entity type offers an important piece of side information. It indicates what class an entity belongs to. Besides, ontological structures between types allow us to group entities together at different levels of granularity. Intuitively, the entity type can improve the performance of entity alignment models since we do not need to align entities of mismatched types.

We use an example in Figure 1 to illustrate the underlying idea. Suppose "Home Alone" is the same entity to be aligned between DBpedia [1] and Wikidata [20] KGs. In DBpedia, the entity "Home Alone" has type labels such as "Creative Work", "Movie", etc. In Wikidata KG, entities with "Film" type should be ranked higher than entities with "Actor" type or "Film director" type, although these type labels are closely related concepts.

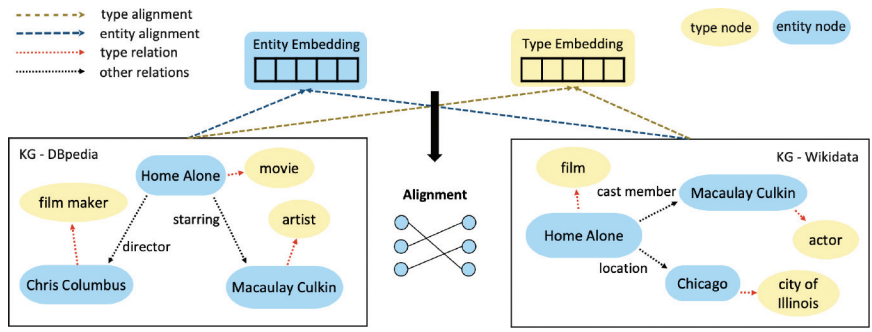


Figure 1: An Illustrative example of the idea behind the proposed TypeEA method.

After inspecting recent entity alignment models, we observe that a large fraction of errors of the predicted entity alignment pairs have mismatched types between entities. Since these predictions are unlikely to be the correct ones, such errors can be avoided by taking type information into consideration. We collect the statistics of the proportion of H@1 prediction errors due to mismatched types for different entity alignment models in Table 1. Based on the statistics, we could potentially reduce up to 30%–50% of the top 1 prediction errors when considering type information.

Prior to the deep learning era, type information has already been leveraged for entity alignment. For instance, PBA [32] is a partition-and-blocking-based

Table 1: Proportion of H@1 prediction errors due to mismatched types by different entity alignment methods for the D-W 15K V1 dataset.

Model	**MTransE**	**JAPE**	**BootEA**	**MultiKE**	**RDGCN**
Ratio	45.240%	42.678%	40.884%	34.588%	57.038%

alignment method that uses the type information as the blocking key. However, since different KGs have disjoint sets of type labels, solving the type resolution problem can be challenging. To address this problem, we propose a method to train an embedding model to capture the type association, and call it the Type-associated Entity Alignment (TypeEA) method. Seed alignments allow us to generate some associated type pairs. Based on them, we can train the TypeEA model to capture more associations. Hence, given the type information of an entity in the source KG, instead of performing rule-based blocking, we use the TypeEA model to identify its most relevant counterpart in the target KG automatically. In this work, we first show that the bilinear product embedding for the proposed TypeEA can capture the type association well. For entity alignment, we make the alignment ranking and decisions by considering the alignment score and the type association score jointly so that TypeEA can better focus on entities with matched types.

The main contributions of this work can be summarized as follows.

- We present a simple and low memory cost embedding model to capture the type association in different KGs and leverage this information to improve the performance of the entity alignment model. We use far fewer free parameters compared to complex models that use large pretrained neural models such as EVA [6] and BERT-INI [18].
- We prepare a type pair dataset for DBP v1.1 by querying the DBpedia English (EN), the DBpedia German (DE), the DBpedia French (FR), the Wikidata, and the YAGO public endpoint KGs. A subset of entity types is selected to learn high-quality type association embedding. The dataset is released to facilitate future research.
- We conduct extensive experiments on all entity alignment datasets in DBP v1.1, which contains cross-lingual and cross-KG alignment tasks. we observe a consistent improvement when combining TypeEA with different embedding-based entity alignment models.

2 Related Work

Entity alignment is a long-standing problem in KG research. Prior to embedding-based models, traditional methods align entities using strategies such as string similarity [10], schema similarity [11] and neighborhood similarity [5]. These methods are hardly applicable when the textual and ontological information is not uniform across different KGs.

Recently, several surveys on embedding-based entity alignment have been published with comprehensive codebases and sampled datasets [4, 16, 26, 28, 29]. These codebases integrate different entity alignment models together

so that fair performance comparison of different models on these datasets can be carried out. Based on the entity embedding techniques, embedding-based entity alignment models have two major categories [28]: Translation embedding-based methods and Graph Neural Networks (GNN)-based methods. They are reviewed below.

2.1 Translational-embedding-based Methods

MTransE [3], BootEA [15], JAPE [13], MultiKE [27], AttrE [19], and COTSAE [25] belong to this category. They use TransE [2] or a variant of TransE. MTransE proposes several score functions for alignment, including distance-based axis calibration, translation vectors, and linear transformations. BootEA learns a classifier through bootstrapping using the negative log-likelihood loss.

Auxiliary features such as entity attributes have also been extensively investigated. JAPE represents attribute features using the Skip-gram word embedding. AttrE represents attribute values through different character embedding aggregation strategies such as LSTM. MultiKE models the association between entity embedding and attribute embedding using CNNs. COTSAE uses a Pseudo Siamese Network to learn attribute predicate and value embedding. Apart from attribute features, visual features [6] generated from entity images using the ResNet is leveraged in EVA to overcome the bottleneck of very few alignment seeds in training. In this work, we propose to leverage the entity type features. Although a recent method, known as JTMEA [7], also considered the entity type, it was benchmarked with a few weaker and earlier baselines. We will demonstrate that the proposed TypeEA model can outperform stronger baselines with the help of type features.

2.2 GNN-based Methods

It is also possible to use graph neural networks to learn representations of entities. GCN-Align [21] uses graph convolutional networks (GCN) to embed both the structural and attribute information of two KGs in a common space with shared weight matrices. RDGCN [22] extends GCNs with highway gates to capture the neighborhood information and includes the relation information by the attentive interaction between a primal graph and a dual graph. Graph attention networks (GATs) are also explored. For example, NAEA [31] embedded the neighborhood information in addition to attribute relations and attribute values. A time-aware GNN based model is proposed in TEA-GNN [23] to handle the alignment of KGs with the temporal information. According to [16], BootEA and RDGCN are the top performing models on different tasks of the DBP v1.1 dataset. In this work, we add the type information to these models and verify whether TypeEA can outperform previous best models.

In addition, researchers leverage the modeling capability of different neural networks and design a hybrid framework. RoadEA [14] consists of an attribute encoder and a relation encoder to aggregate entity attributes or relational neighbors using attention mechanisms for entity representation. Adaptive embedding fusion is achieved through a gated mechanism to unify the representation space. EMGCN [8] is an unsupervised entity alignment framework that captures the relation-based correlation between entities using a multi-order GCN and incorporates the attribute-based correlation via a translation machine. A late-fusion mechanism is used to combine the information together to enhance the final alignment result.

3 The TypeEA Method

To perform entity alignment, the proposed TypeEA method consists of two parts: (1) how to effectively train the type association embedding and (2) how to select the subset of entity types to learn the representation. Then, we integrate the trained type association embedding with the state-of-the-art entity alignment models to correct the type mismatch problem in their models.

3.1 Problem Formulation

Let $\mathcal{G} = (\mathcal{E}, \mathcal{R}, \mathcal{L}, \mathcal{T})$ denotes a knowledge graph where $\mathcal{E}$, $\mathcal{R}$, and $\mathcal{L}$ represents a set of all entities, relations, and type labels, respectively. $\mathcal{T}$ denotes a set of all relation triples $\{(h, r, t)|h, t \in \mathcal{E}, r \in \mathcal{R}\}$.

To align entities in two KGs, denoted by

$$\mathcal{G}_1 = (\mathcal{E}_1, \mathcal{R}_1, \mathcal{L}_1, \mathcal{T}_1), \tag{1}$$

$$\mathcal{G}_2 = (\mathcal{E}_2, \mathcal{R}_2, \mathcal{L}_2, \mathcal{T}_2), \tag{2}$$

we need to identify all pairs of equivalent entities

$$\psi = \{(e_1, e_2)|e_1 \in \mathcal{E}_1, e_2 \in \mathcal{E}_2\} \tag{3}$$

from two KGs. Seed entity pairs are often given in entity alignment datasets. Since entity type labels are available from KG queries, we can infer label pairs

$$\phi = \{(l_1, l_2)|l_1 \in \mathcal{L}_1, l_2 \in \mathcal{L}_2\} \tag{4}$$

from the entity pair set ψ. Our goal is to design embedding models to encode the type information and investigate whether the type embedding can improve the entity alignment performance.

3.2 Type Acquisition

One contribution of this work is to add the type label information of entities to existing datasets. The DBpedia (EN) KG has the most abundant type labels for each entity. However, there are two challenges in choosing an appropriate subset of type labels for modeling. First, many of the type labels are acquired from different sources and are often redundant. Second, a large number of type labels are too fine-grained. With a limited amount of seed entity pairs, it is difficult to generate enough type label pairs to train type embedding well. To solve this problem, we obtain non-overlapping subsets of types and their association pairs for both source and target KGs. Details on type information acquisition are discussed in Section 4.1. We train and evaluate the type association embedding using the type pairs dataset.

3.3 Type Association Embedding

The goal of training the type association embedding is to model the relationship between type labels from two KGs. Since the type sets for two KGs are disjoint, we essentially use the type association embedding to align the types from two KGs before aligning the entities. Source and target entities whose type labels can generate higher type association scores are more likely to be aligned. To model the type association, we adopt two scoring functions: (1) the cosine similarity and (2) the bilinear product.

Cosine Similarity. We first experiment with the cosine similarity as the score function to capture the association between types. This can be written in form of

$$f_{\text{type}}(u, v) = \cos(\mathbf{u}, \mathbf{v}) = \frac{\mathbf{u}^{\mathbf{T}}\mathbf{v}}{\|\mathbf{u}\|\|\mathbf{v}\|}, \tag{5}$$

where u and v denote two types associated with two different KGs. The use of the cosine similarity score function is intuitive since the goal is to have entity types that frequently appear in a type pair to have higher scores while those pairs that never appear before have lower scores. However, the cosine similarity score does not generate satisfactory results in retrieving the most relevant types in practice. We perform experiments on type label pairs for D-W 15K V1/V2 dataset and show the results in Table 3.

Bilinear Product. Since the cosine similarity measure is not effective in modeling the relationship between types, we propose a more expressive bilinear product score function to model type association. Figure 2 provides an illustration for the logic behind the proposed scoring function. The semantic-matching-based score is defined as

$$f_{\text{type}}(u, v) = \mathbf{u}^{\mathbf{T}}\mathbf{W}\mathbf{v}, \tag{6}$$

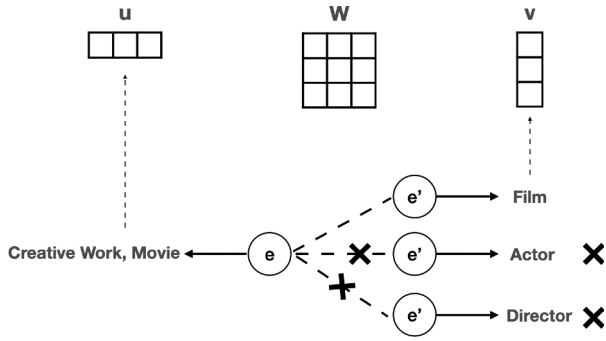


Figure 2: Illustration of using the type association embedding for identifying relevant candidates in entity alignment, where **u** and **v** denote two type embeddings in two KGs, respectively, and **W** denotes their association embedding using the bilinear product.

where u and v denote the type of one KG1 entity and the type of one KG2 entity, respectively. Also, $\mathbf{u} \in \mathbb{R}^m$ and $\mathbf{v} \in \mathbb{R}^n$ denote the learnable representations of u and v in the type space, respectively. Similar to RESCAL [9] and DistMult [24], we construct a learnable embedding matrix, $\mathbf{W} \in \mathbb{R}^{m \times n}$, which is shared among all type pairs. Both the type embedding and the shared weight matrix are uniformly initialized.

We find that the self-adversarial negative sampling strategies introduced in RotatE [17] are particularly useful in learning the type association embedding parameters. The objective function in learning these parameters is set to

$$O_{\mathrm{T}} = -\log \sigma(f_{\mathrm{type}}(u, v) - \gamma) - \sum_{i=1}^{n} p(u, v_i') \log \sigma(f_{\mathrm{type}}(u, v_i') - \gamma), \quad (7)$$

where γ is a fixed margin hyper-parameter, (u, v_i') is the i-th negative type pair, and $p(u, v_i')$ is the probability of drawing negative type pair (u, v_i'). Given a corrupted type pair (u, v_i'), the sampling distribution can be written as

$$p(u, v_j' | \{(u, v_i)\}) = \frac{\exp \alpha f_{\mathrm{type}}(u, v_j')}{\sum_i \exp \alpha f_{\mathrm{type}}(u, v_i')}. \quad (8)$$

This self-adversarial negative sampling scheme has two advantages. First, hard negative samples are more likely to be chosen for training. The embedding model can be fine-tuned more effectively by hard negative examples than easy negative samples. Second, since hard negative samples carry a higher weight in the objective function, their loss is given more attention in optimization. The performance on predicting the associated type pairs for various datasets is given in Table 3.

3.4 Entity Representation and Alignment

We experiment on three model types with different feature representations below.

Translation Embedding. Translation-based EA techniques mainly use the translation embedding model to extract structural features for entities and relations. The well known TransE scoring function is defined as

$$f_{\text{triple}}(h, r, t) = \|\mathbf{h} + \mathbf{r} - \mathbf{t}\|, \tag{9}$$

where $\mathbf{h}$, $\mathbf{r}$, $\mathbf{t}$ are the low dimensional space representation for the head entity, the relation, and the tail entity of a triple, respectively. To pull entity vectors from KG1 and KG2 into a unified space, we generate new triples by swapping aligned entities in the corresponding triples. For example, given an aligned entity pair (e_1, e_2), where e_1 and e_2 comes from KG1 and KG2 respectively, we can generate the following set of triples:

$$\begin{aligned}\mathcal{T}_{Gen} = &\{(e_2, r, t)|(e_1, r, t) \in \mathcal{T}_1\} \cup \{(h, r, e_2)|(h, r, e_1) \in \mathcal{T}_1\} \\ &\cup \{(e_2, r, t)|(e_1, r, t) \in \mathcal{T}_2\} \cup \{(h, r, e_2)|(h, r, e_1) \in \mathcal{T}_2\}.\end{aligned} \tag{10}$$

Moreover, instead of using the max-margin loss function adopted by the original TransE model, we use the *limit-based loss function* [30] to optimize the embedding. The loss function can be expressed as:

$$\begin{aligned}O_{\text{e}} = &\sum_{(h,r,t)\in\mathcal{T}_r} \max(0, [f_{\text{triple}}(h, r, t) - \gamma_1]) + \\ &\beta_1 \sum_{(h',r',t')\in\mathcal{T}'_r} \max(0, [\gamma_2 - f_{\text{triple}}(h', r', t')]),\end{aligned} \tag{11}$$

where $\mathcal{T}_r = \mathcal{T}_1 \cup \mathcal{T}_2 \cup \mathcal{T}_{Gen}$, and $\mathcal{T}'_r$ contains all corrupted triples generated by uniform negative sampling. Based on the learned entity representation, an alignment module is further proposed and trained to identify the counterpart of a target entity in the other KG. Among different alignment modules, Bootstrapping [15] is one of the best performing strategy. In our experiment, we also include the Bootstrapping strategy to facilitate the alignment. In particular, we minimize the following cross-entropy objective:

$$O_{\text{a}} = -\sum_{e_1\in\mathcal{E}_1} \sum_{e_2\in\mathcal{E}_2} \mathbb{1}_{e_1}(e_2) \log \pi(e_2|e_1; \Theta), \tag{12}$$

where $\mathbb{1}_{e_1}(e_2)$ is a indicator function that denotes the labeling probability of entity e_1 and $\pi(e_2|e_1; \Theta)$ is the function that computes the likelihood of labeling the counterpart of entity e_1 as e_2, given embedding parameters Θ obtained from TransE. In our experiment, the cosine similarity is used as

the similarity function. The alignment decision is made using the following function

$$f_{\text{align}}(e_i, e_j) \propto \pi(e_2|e_1; \Theta) = \cos(\mathbf{e}_i, \mathbf{e}_j). \tag{13}$$

Attribute Auxiliary Features. In this line of work, auxiliary features such as textual information from entity names and numeric attributes from entity property laterals such as "date", "age" are leveraged to improve entity alignment performance. These auxiliary features were not considered in learning the structural embedding. Yet, they provide important information for identifying matching entities. In our experiments, we choose MultiKE [27], which uses auxiliary features, as one of our baselines and verify if our type association method can improve the performance of MultiKE. In the baseline, pre-trained word and character embeddings are used to encode entity names. To model the attribute-value information, separate embedding matrices are trained for attribute labels and values, respectively. To learn the attribute label and value embedding, we use the following scoring function

$$f(e, a, v) = \|\mathbf{e} - \text{CNN}([\mathbf{a}\|\mathbf{v}])\|, \tag{14}$$

where e, a, v represent entity, attribute label, and attribute value, respectively, in a attribute lateral triple. $[\mathbf{a}\|\mathbf{v}]$ denotes the concatenation of attribute label and attribute value vectors. The concatenated feature vector is passed into a Convolutional Neural Network (CNN) and the error between the resulting vector and the entity vector $\mathbf{e}$ is minimized. We use the logistic based objective function to optimize the model

$$O_{\text{v}} = \sum_{(e,a,v)\in\mathcal{T}_v} \log(1 + \exp(f(e, a, v))), \tag{15}$$

where $\mathcal{T}_v$ is a set of all attribute triples. The obtained structural, textual, and attribute embeddings are combined to form the representation of entities. Alignment inferences are performed through nearest neighbor search.

Graph Neural Networks. Another approach is to use graph convolutional networks (GCNs) to represent the entity. The message passing process in GCNs can be formulated as

$$\mathbf{H}^{(l+1)} = \sigma(\tilde{\mathbf{D}}^{-\frac{1}{2}}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-\frac{1}{2}}\mathbf{H}^{(l)}\mathbf{W}^{(l)}), \tag{16}$$

where $\tilde{\mathbf{A}} = A + I$ is the adjacency matrix of the graph, I is the identity matrix that denotes the self connection, $\tilde{\mathbf{D}}$ is a diagonal matrix of node degrees where $\tilde{\mathbf{D}}_{ii} = \sum_j \tilde{\mathbf{A}}_{ij}$, $\mathbf{W}^{(l)}$ is the weight matrix at the l-th layer to be optimized, and $\mathbf{H}^{(l)}$ is the node embedding at the l-th layer. The particular variant of the

GCN-based approach adopted as a baseline in our experiment is the RDGCN [22]. RDGCN uses a coupled GCN to incorporate the relation information through attentive interactions between the original graph and its dual relation graph. A max-margin loss is used to optimize the model. We can obtain the vector representation of each node from the output layer of the RDGCN and compute the alignment score as

$$f_{\text{align}}(e_i, e_j) \propto 1 - \|\mathbf{e}_i - \mathbf{e}_j\|. \tag{17}$$

3.5 Inference

During inference, we take both the type score and the alignment score into account. Suppose e_i is the source entity. To infer the matching target entity, e_j, we choose the entity that maximizes the linear combination of the type score and the alignment score. Mathematically, we have

$$\begin{aligned} \hat{e}_j = &\arg\max_{e_j \in E} \lambda \cdot f_{\text{type}}(L(e_i), L(e_j)) \\ &+ (1 - \lambda) \cdot f_{\text{align}}(e_i, e_j), \end{aligned} \tag{18}$$

where $\lambda \in [0, 1]$ is the balancing parameter, and $L(\cdot)$ is a type label lookup function for the entity.

4 Experiments

4.1 Datasets

We perform experiments on the DBP v1.1 entity alignment datasets from Sun *et al.* [16] that includes both cross-KB and cross-lingual settings. To be specific, under the cross-KB setting, there are D-W and D-Y which denote DBpedia-Wikidata and DBpedia-YAGO, respectively. Under the cross-lingual settings, there are EN-FR and EN-DE which denote DBpedia English-DBpedia French and DBpedia English-DBepdia German, respectively. For each of the above tasks, there are also variants with different size: 15 k and 100 k, and variants of sparse (V1) and dense (V2) subgraphs. These datasets were generated using a method called iterative degree-based sampling (IDS). The detailed statistics of the DBP v1.1 dataset can be found in the original paper by Sun *et al.* [16].

We obtain the type data by querying the DBpedia[1] [1], Wikidata[2] [20], and YAGO[3] [12] public endpoint using SPARQL queries. From all the type labels obtained from queries, we select a subset of type labels in order to get reliable type embedding. For example, to make the alignment between DBpedia and

[1] https://dbpedia.org/sparql
[2] https://query.wikidata.org/
[3] https://yago-knowledge.org/sparql

Wikidata, we only use type labels that have "http://www.wikidata.org/" prefix and filter out other type labels. For DBpedia to YAGO alignment, we use only type labels with "http://dbpedia.org/" prefix. For DBpedia EN to DE and EN to FR, we use labels with prefixes "http://schema.org/", "http://dbpedia.org/ontology", "http://de.dbpedia.org/", and "http://fr.dbpedia.org/". Statistics of the type pair datasets are shown in Table 2. Specifically, the **Train**, **Valid**, and **Test** columns indicate the number of type pairs in training, validation, and testing sets, respectively. The **KG1** and **KG2** columns indicate the number of distinct type in KG1 and KG2 respectively.

4.2 Implementation Details

Model Configuration. We set the type embedding dimension $m = 200$ and $n = 200$ for the source and target KGs respectively, and the type pair batch size is set to 4096. To train type embedding, we use the Adam optimizer with the learning rate $\eta = 1e - 4$. We set the batch size to 1024, the number of negative sample to 256, the sampling temperature $\alpha = 1$, the margin parameter $\gamma = 24$. The parameter of entity alignment baseline models are kept the same as provided in OpenEA [16]. We use a server with Intel(R) Xeon(R) E5-2620 CPU and Nvidia Quadro M6000 GPU to run all of our experiments.

Evaluation Metrics. Following the convention, we use Hits@k and Mean Reciprocal Rank (MRR) as our evaluation metrics to evaluate the performance of both type association embedding and the entity alignment models. Hits@k is the proportion of ground truth entity appears in the top-k candidate list. Higher Hits@k and MRR imply better performance of the model. To evaluate the performance of type association embedding, we also include the Mean Rank (MR) metric.

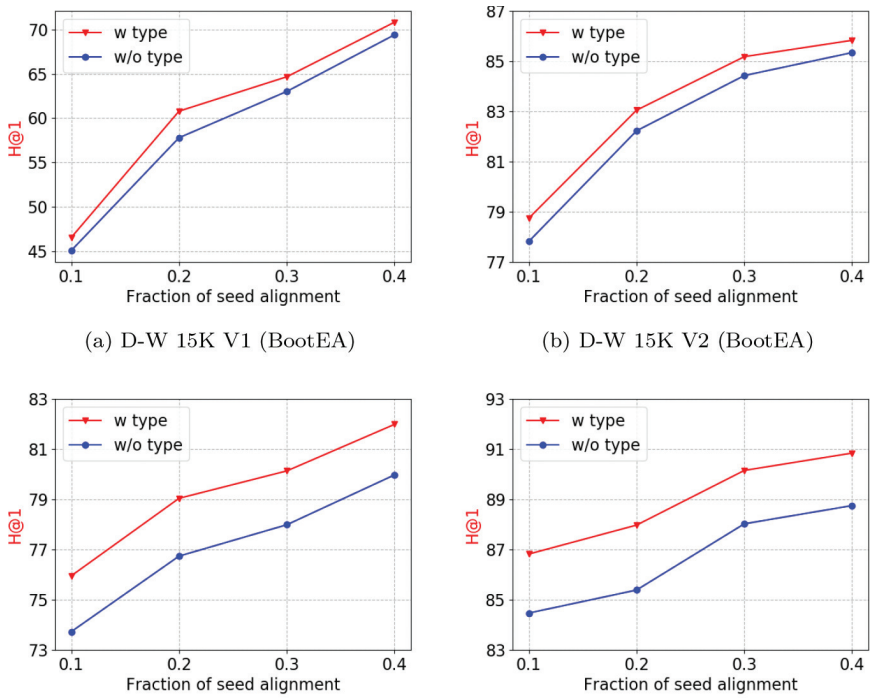
Baselines. To evaluate the performance improvement contributed by TypeEA, we compare it with 7 highly-cited strong baseline models. According to the results in Sun *et al.* [16], BootEA [15], MultiKE [27] and RDGCN [22] are the top performers across different datasets settings in DBP v1.1.

4.3 Results

Type Association Embedding. As shown in Table 3, our proposed bilinear product type embedding model consistently achieves good results of predicting the associated types in the target KG across all the datasets and settings. The Hits@k scores are all above 90 and the MRR scores are all above 0.9. This means that our proposed model can predict most relevant associated types accurately and reliably. The bilinear product score is more effective than the cosine similarity score. One possible reason behind it is that the shared

embedding matrix $\mathbf{W} \in \mathbb{R}^{m \times n}$ has more modeling power and improves the expressiveness of the model.

Entity Alignment. Tables 5, 6, and 7 present a comprehensive performance comparison of our proposed TypeEA with previous best baseline models for both cross-KG (D-W) and cross-lingual (EN-FR) EA datasets, for both small (15k) and large (100k) subgraphs, and under both sparse (V1) and dense (V2) sampling settings. In particular, TypeEA-B, TypeEA-R, and TypeEA-M denote the results generated from applying type association embedding to baseline model BootEA, RDGCN, and MultiKE respectively. We use the given split where the train, valid, and test set have 20%, 10%, and 70% of entity pairs respectively. We observe consistent performance improvement as compared to previous results. Among all, the most performance improvement is observed for EN-FR 15K V1 and EN-FR 15K V2 datasets where the Hit@1 scores are improved by 4.16 and 4.44 respectively.



(a) D-W 15K V1 (BootEA)

(b) D-W 15K V2 (BootEA)

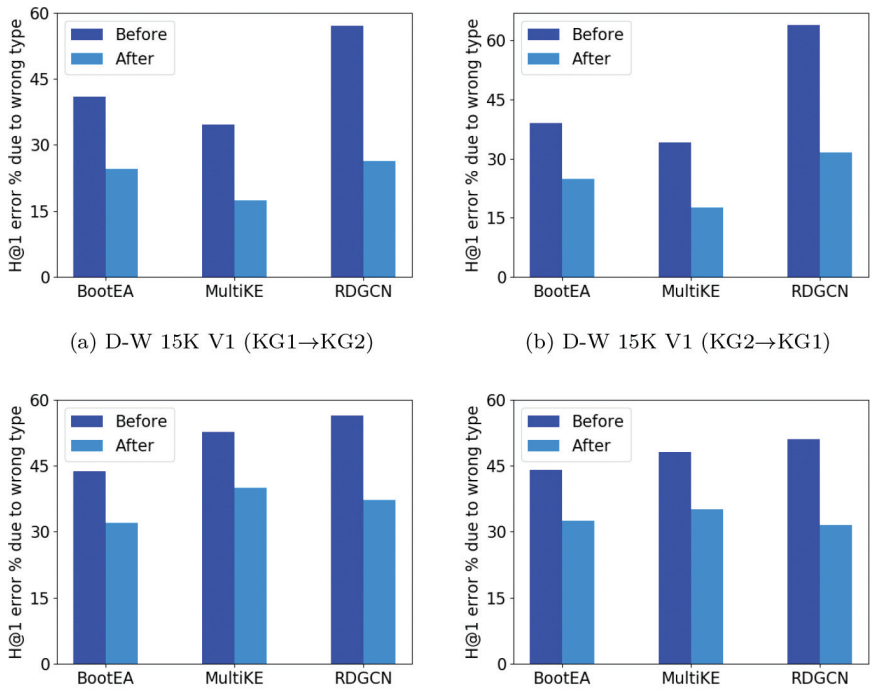
(c) EN-FR 15K V1 (RDGCN)

(d) EN-FR 15K V2 (RDGCN)

Figure 3: Comparison of the H@1 entity alignment performance with or without the type information for different datasets and models as a function of the fraction of seed alignment.

Table 4 shows an entity alignment example from D-Y 15K V1 dataset comparing the ranking of candidate entities. In this example, the source entity in DBpedia is "Ed. Weinberger" and we are trying to find its counterpart in YAGO. Without using the type information, the baseline model BootEA makes a few erroneous predictions in the top candidate list. Among the incorrect predictions, many have mismatched types such as "TV Series", "Movie" and they are not the matching candidate that we are looking for. The ground-truth target has relatively low ranks. After applying the type information, entities with wrong type labels are ranked lower in the predictions. This confirms our intuition that predicting entities with mismatched type is indeed a problem of the baseline models. With the help of type association embedding, the ground-truth target can be ranked higher in the final alignment predictions.

In Figure 3, we show plots of entity alignment accuracy as a function of alignment seed fraction. We conduct experiments when the fraction of seed is $\{0.1, 0.2, 0.3, 0.4\}$ respectively and observe consistent improvement even if only



(a) D-W 15K V1 (KG1→KG2)

(b) D-W 15K V1 (KG2→KG1)

(c) EN-FR 15K V1 (KG1→KG2)

(d) EN-FR 15K V1 (KG2→KG1)

Figure 4: Comparison of the H@1 entity alignment performance with or without the type information for different datasets and models as a function of the fraction of seed alignment.

10% of alignment seeds are provided. When we have more alignment seeds, the advantage of adding the type information seems to diminish perhaps because the entity alignment models are making fewer mistakes that can be corrected using type information when predicting the counterpart of a target entity.

In Figure 4, we compare the fraction of H@1 prediction errors due to mismatched type labels for three baseline models: BootEA, MultiKE, RDGCN. In particular, we run experiments on D-W 15K V1 (Figure 4 (a) and (b)) and EN-FR 15K V1 (Figure 4 (c) and (d)) dataset. We show the error reduction for alignment for both directions: from KG1 to KG2 and from KG2 to KG1. We observe that our TypeEA approach is effective in reducing the error for all three baseline models. Largest reduction is observed for the RDGCN baseline, perhaps because RDGCN makes the highest percentage of type mismatch errors.

5 Conclusion and Future Work

In this paper, we present a new approach called Type-Associated Entity Alignment (TypeEA) for helping entity alignment model decisions. We experiment with different scoring functions for modeling TypeEA and find that the bilinear product is the best for capturing the type association. We also employ the self-adversarial negative sampling strategy which is very effective in learning the embedding. We integrate type associated embedding with entity alignment models and demonstrate better alignment performance on DBP v1.1 dataset. Moreover, we collect and prepare entity type label pairs datasets complementary to all sub-datasets of DBP v1.1 so that the type association embedding can be learned. One limitation of our work is that the subset of types is still heuristically selected for training reliable type association embedding. In the future, we will further investigate how to use embedding to model more diverse entity types and a more complex relationship in entity type pairs.

Table 2: The statistics of type association pairs for various datasets.

	V1 (Sparse)					V2 (Dense)				
Dataset	Train	Valid	Test	KG1	KG2	Train	Valid	Test	KG1	KG2
D-W 15K	2,609	1,328	9,273	101	1,185	2,696	1,350	9,435	51	563
D-W 100K	17,313	8,680	60,844	163	3,883	17,995	8,915	62,620	104	2,682
D-Y 15K	2,884	1,437	10,062	387	407	2,903	1,472	10,178	178	75
D-Y 100K	19,138	9,532	66,777	1,117	1,306	19,162	9,565	66,961	690	791
EN-DE 15K	2,884	1,455	10,189	497	103	2,877	1,452	10,176	284	53
EN-DE 100K	19,466	9,777	68,257	1619	149	19,386	9,705	68,005	1,179	109
EN-FR 15K	2,511	1,246	8,799	565	208	2,594	1,302	9,166	340	115
EN-FR 100K	16,090	8,003	56,236	1,757	335	16,660	8,259	58,199	1,422	290

Table 3: Ranking of associated type pair prediction.

	V1 (Sparse)					V2 (Dense)				
Dataset	MRR	MR	H@1	H@3	H@10	MRR	MR	H@1	H@3	H@10
D-W 15K (cos)	0.312	42.38	0	50.39	86.14	0.368	67.47	2.35	56.68	91.73
D-W 15K	0.915	55.14	91.44	91.52	91.63	0.957	13.96	95.68	95.68	95.71
D-W 100K	0.953	97.50	95.31	95.33	95.34	0.970	41.65	97.01	97.03	97.05
D-Y 15K	0.944	17.21	94.16	94.44	94.97	0.985	2.15	98.26	98.85	98.94
D-Y 100K	0.966	29.22	96.54	96.61	96.79	0.981	10.25	98.03	98.10	98.22
EN-DE 15K	0.957	6.52	95.26	95.72	96.67	0.978	2.29	97.58	97.90	98.54
EN-DE 100K	0.977	13.48	97.54	97.73	98.11	0.983	6.13	98.16	98.34	98.68
EN-FR 15K	0.930	18.28	92.62	93.08	93.75	0.964	5.76	96.05	96.41	97.04
EN-FR 100K	0.962	23.90	95.90	96.24	96.85	0.969	16.71	96.71	96.98	97.40

Note: The first row shows the results of preliminary experiments using the cosine similarity score function while results in all other rows are generated using the bilinear product score.

Table 4: An example from D-Y 15K V1 illustrating the advantage of using type for alignment.

	w/o Type		w Type	
Rank	Entity	Type	Entity	Type
1	Chris Hayward	Human	Chris Hayward	Human
2	Lou Grant (TV series)	TV Series	Lou Grant	N/A
3	Lou Grant	N/A	James Coco	Human
4	The Munsters	TV Series	**Ed. Weinberger**	Human
5	Mr. Smith (TV series)	TV Series	Carol Sobieski	Human
6	The Toy (1982 film)	Movie	Teresa Ganzel	Human
...	...		...	
10	**Ed. Weinberger**	Human	Ernest Kinoy	Human

Note: The source entity **Ed. Weinberger** has type label in DBpedia. The target entities candidates together with their corresponding types in YAGO knowledge graph are listed by their ranks.

Table 5: Comparison of entity alignment performance of TypeEA with baselines for cross-KG (DBpedia to Wikidata) alignment with 15k entities.

	D-W 15K V1			D-W 15K V2		
Models	MRR	H@1	H@5	MRR	H@1	H@5
MTransE	0.352	25.45	46.03	0.365	25.84	48.03
JAPE	0.339	24.32	44.42	0.364	25.71	48.06
GCNAlign	0.467	37.33	58.26	0.618	51.27	74.95
AttrE	0.383	30.18	46.91	0.586	48.99	69.32
BootEA	0.655	57.8	75.01	<u>0.87</u>	<u>82.22</u>	<u>92.45</u>
RDGCN	0.587	51.83	67.26	0.678	61.71	75.36
MultiKE	0.483	42.27	54.30	0.574	50.08	64.86
TypeEA-B	**0.681**	**60.79**	**77.05**	**0.889**	**83.04**	**94.01**
TypeEA-R	<u>0.656</u>	<u>59.04</u>	<u>73.80</u>	0.729	66.75	80.91
TypeEA-M	0.522	45.53	58.85	0.612	53.41	70.26

Table 6: Comparison of entity alignment performance of TypeEA with baselines for cross-lingual (English to French) alignment with 15k entities.

	EN-FR 15K V1			EN-FR 15K V2		
Models	MRR	H@1	H@5	MRR	H@1	H@5
MTransE	0.35	24.6	46.67	0.34	24.47	44.04
JAPE	0.374	26.66	49.96	0.404	29.44	52.65
GCNAlign	0.446	33.45	57.91	0.545	41.89	70.17
AttrE	0.558	46.90	66.05	0.651	55.61	76.71
BootEA	0.597	50.31	71.02	0.747	66.13	85.41
RDGCN	0.799	75.45	85.25	0.881	84.84	91.87
MultiKE	0.776	74.2	81.26	0.884	86.13	90.85
TypeEA-B	0.643	54.28	76.91	0.909	88.21	93.93
TypeEA-R	**0.832**	79.44	**87.91**	**0.930**	**90.57**	**95.91**
TypeEA-M	0.827	**79.61**	86.08	0.908	89.07	92.91

Table 7: Comparison of entity alignment performance of TypeEA with baselines for cross-lingual (English to French) alignment with 100k entities.

	EN-FR 100K V1			EN-FR 100K V2		
Models	MRR	H@1	H@5	MRR	H@1	H@5
MTransE	0.203	13.74	26.46	0.131	8.60	16.95
JAPE	0.243	16.92	31.20	0.183	12.35	23.94
GCNAlign	0.321	23.14	41.33	0.351	25.76	45.21
AttrE	0.509	42.96	59.73	0.541	45.70	63.59
BootEA	0.475	39.03	56.30	0.715	63.97	80.52
RDGCN	0.682	63.81	72.96	0.751	71.77	79.03
MultiKE	0.654	62.85	67.94	0.669	64.21	69.52
TypeEA-B	0.483	40.65	58.21	0.722	65.56	82.42
TypeEA-R	0.689	65.65	**74.51**	**0.758**	**73.62**	**80.57**
TypeEA-M	**0.701**	**67.52**	72.84	0.701	67.38	72.85

References

[1] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. Ives, "DBpedia: A Nucleus for a Web of Open Data," in *SWJ*, 2007.

[2] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, and O. Yakhnenko, "Translating Embeddings for Modeling Multi-relational Data," *NeurIPS*, 2013.

[3] M. Chen, Y. Tian, M. Yang, and C. Zaniolo, "Multilingual Knowledge Graph Embeddings for Cross-lingual Knowledge Alignment," in *IJCAI*, 2017.

[4] N. Fanourakis, V. Efthymiou, D. Kotzinos, and V. Christophides, "Knowledge Graph Embedding Methods for Entity Alignment: An experimental review," *arXiv preprint arXiv:2203.09280*, 2022.

[5] S. Lacoste-Julien, K. Palla, A. Davies, G. Kasneci, T. Graepel, and Z. Ghahramani, "Sigma: Simple Greedy Matching for Aligning Large Knowledge Bases," in *KDD*, 2013.

[6] F. Liu, M. Chen, D. Roth, and N. Collier, "Visual Pivoting for (Unsupervised) Entity Alignment," in *AAAI*, 2021.

[7] G. Lu, L. Zhang, M. Jin, P. Li, and X. Huang, "Entity Alignment via Knowledge Embedding and Type Matching Constraints for Knowledge Graph Inference," *Journal of Ambient Intelligence and Humanized Computing*, 13, 2022, 5199–209.

[8] T. T. Nguyen, T. T. Huynh, H. Yin, V. Van Tong, D. Sakong, B. Zheng, and Q. V. H. Nguyen, "Entity Alignment for Knowledge Graphs with Multi-order Convolutional Networks," *IEEE Transactions on Knowledge and Data Engineering*, 34(9), 2020, 4201–14.

[9] M. Nickel, V. Tresp, and H.-P. Kriegel, "A Three-way Model for Collective Learning on Multi-relational Data," in *ICML*, 2011.

[10] Y. Raimond, C. Sutton, and M. B. Sandler, "Automatic Interlinking of Music Datasets on the Semantic Web," in *LDOW*, 2008.

[11] F. M. Suchanek, S. Abiteboul, and P. Senellart, "Paris: Probabilistic Alignment of Relations, Instances, and Schema," *PVLDB*, 2011.

[12] F. M. Suchanek, G. Kasneci, and G. Weikum, "YAGO: A Core of Semantic Knowledge," in *WWW*, 2007.

[13] Z. Sun, W. Hu, and C. Li, "Cross-lingual Entity Alignment via Joint Attribute-Preserving Embedding," in *ISWC*, 2017.

[14] Z. Sun, W. Hu, C. Wang, Y. Wang, and Y. Qu, "Revisiting Embedding Based Entity Alignment: A Robust and Adaptive Method," *IEEE Transactions on Knowledge and Data Engineering*, 2022, 1–14.

[15] Z. Sun, W. Hu, Q. Zhang, and Y. Qu, "Bootstrapping Entity Alignment with Knowledge Graph Embedding," in *IJCAI*, 2018.

[16] Z. Sun, Q. Zhang, W. Hu, C. Wang, M. Chen, F. Akrami, and C. Li, "A Benchmarking Study of Embedding-based Entity Alignment for Knowledge Graphs," *Proceedings of the VLDB Endowment*, 13(11), 2020, 2326–40.

[17] Z. Sun, Z.-H. Deng, J.-Y. Nie, and J. Tang, "Rotate: Knowledge Graph Embedding by Relational Rotation in Complex Space," in *ICLR*, 2019.

[18] X. Tang, J. Zhang, B. Chen, Y. Yang, H. Chen, and C. Li, "BERT-INT: A BERT-Based Interaction Model for Knowledge Graph Alignment," in *IJCAI*, 2020.

[19] B. D. Trisedya, J. Qi, and R. Zhang, "Entity Alignment between Knowledge Graphs Using Attribute Embeddings," in *AAAI*, 2019.

[20] D. Vrandečić and M. Krötzsch, "Wikidata: A Free collaborative Knowledgebase," *Communications of the ACM*, 57(10), 2014, 78–85.
[21] Z. Wang, Q. Lv, X. Lan, and Y. Zhang, "Cross-lingual Knowledge Graph Alignment via Graph Convolutional Networks," in *EMNLP*, 2018.
[22] Y. Wu, X. Liu, Y. Feng, Z. Wang, R. Yan, and D. Zhao, "Relation-aware Entity Alignment for Heterogeneous Knowledge Graphs," in *IJCAI*, 2019.
[23] C. Xu, F. Su, and J. Lehmann, "Time-aware Graph Neural Network for Entity Alignment Between Temporal Knowledge Graphs," in *EMNLP*, 2021.
[24] B. Yang, W.-t. Yih, X. He, J. Gao, and L. Deng, "Embedding Entities and Relations for Learning and Inference in Knowledge Bases," in *ICLR*, 2015.
[25] K. Yang, S. Liu, J. Zhao, Y. Wang, and B. Xie, "Cotsae: Co-Training of Structure and Attribute Embeddings for Entity Alignment," in *AAAI*, 2020.
[26] K. Zeng, C. Li, L. Hou, J. Li, and L. Feng, "A Comprehensive Survey of Entity Alignment for Knowledge Graphs," *AI Open*, 2, 2021, 1–13.
[27] Q. Zhang, Z. Sun, W. Hu, M. Chen, L. Guo, and Y. Qu, "Multi-View Knowledge Graph Embedding for Entity Alignment," in *IJCAI*, 2019.
[28] R. Zhang, B. D. Trisedya, M. Li, Y. Jiang, and J. Qi, "A Benchmark and Comprehensive Survey on Knowledge Graph Entity Alignment via Representation Learning," *The VLDB Journal*, 31, 2022, 1143–68.
[29] X. Zhao, W. Zeng, J. Tang, W. Wang, and F. Suchanek, "An Experimental Study of State-of-the-art Entity Alignment Approaches," *IEEE TKDEIEEE Transactions on Knowledge and Data Engineering*, 34(6), 2020, 2610–25.
[30] X. Zhou, Q. Zhu, P. Liu, and L. Guo, "Learning Knowledge Embeddings by Combining Limit-based Scoring Loss," in *CIKM*, 2017.
[31] Q. Zhu, X. Zhou, J. Wu, J. Tan, and L. Guo, "Neighborhood-aware Attentional Representation for Multilingual Knowledge Graphs," in *IJCAI*, 2019.
[32] Y. Zhuang, G. Li, Z. Zhong, and J. Feng, "PBA: Partition and Blocking based Alignment for Large Knowledge Bases," in *DASFAA*, 2016.