

# OntoEA: Ontology-guided Entity Alignment via Joint Knowledge Graph Embedding

Yuejia Xiang<sup>1,2\*</sup>, Ziheng Zhang<sup>1\*</sup>, Jiaoyan Chen<sup>3</sup>, Xi Chen<sup>1,2†</sup>, Zhenxi Lin<sup>1</sup>, Yefeng Zheng<sup>1</sup>

<sup>1</sup>Tencent Jarvis Lab, Shenzhen, China

<sup>2</sup>Platform and Content Group, Tencent, Shenzhen, China

<sup>3</sup>Department of Computer Science, University of Oxford, UK

{yuejiaxiang, zihengzhang}@tencent.com, jiaoyan.chen@cs.ox.ac.uk  
{jasonxchen, chalerislin, yefengzheng}@tencent.com

## Abstract

Semantic embedding has been widely investigated for aligning knowledge graph (KG) entities. Current methods have explored and utilized the graph structure, the entity names, and attributes, but ignore the ontology (or ontological schema) which contains critical meta information such as classes and their membership relationships with entities. In this paper, we propose an ontology-guided entity alignment method named OntoEA, where both KGs and their ontologies are jointly embedded, and the class hierarchy and the class disjointness are utilized to avoid false mappings. Extensive experiments on seven public and industrial benchmarks have demonstrated the state-of-the-art performance of OntoEA and the effectiveness of the ontologies.

## 1 Introduction

Knowledge graphs (KGs) that are composed of entities and facts in the RDF form (i.e., RDF triples) are of vital importance in various applications, such as search engines and personal assistants (Hogan et al., 2020). Although there have been several large-scale KGs, the content of one individual KG is often incomplete, especially in supporting some domain-specific applications such as clinical AI assistants. As these KGs are developed separately, they are usually heterogeneous and supplementary to each other. Thus it becomes urgently needed to align multiple KGs (i.e., matching the equivalent elements) to fully explore their usability.

Recently, a few embedding-based methods have been proposed for entity alignment (EA) (Sun et al., 2020b; Zhang et al., 2020; Qi et al., 2021), in which entities are embedded into vectors and the equivalent entities are determined via calculating the similarity of their vectors. They extend the embedding of one KG to the embedding of multiple

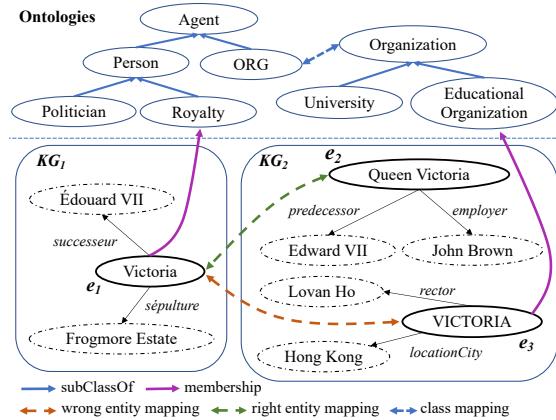


Figure 1: A class conflict example in entity alignment.

KGs into one vector space by, for instance, a transformation matrix, which is learnt from annotated mappings (a.k.a. seed mappings). They also utilize the entity names and attributes besides the graph structures for better performance. These methods, however, all ignore the ontology (or ontological schema) which is an important part of many KGs such as DBpedia (Auer et al., 2008) and Wikidata (Vrandečić and Krötzsch, 2014) as meta information for higher quality and usability. The ontology typically contains hierarchical classes and properties, and optionally defines some logical constraints such as the class disjointness, the property domain and range (Horrocks, 2008). Meanwhile, the KG usually clarifies the membership relationship between entities and classes.

The entity mappings by the above methods that do not exploit the ontologies may induce some class conflicts. Considering the example in Fig. 1, *Victoria* in KG<sub>1</sub> is often incorrectly aligned to *VICTORIA* in KG<sub>2</sub> by many embedding-based methods, but they belong to two potentially disjointed classes, namely *Person* and *Organization*. We find such class conflicted mappings are quite common in the wrongly predicted mappings. Considering the EN-FR-15K-V1 benchmark by (Sun et al., 2020b), 42.2% and 55.7% of the wrongly pre-

\*Equal contribution

†Corresponding author

dicted mappings are class conflicted when running BootEA (Sun et al., 2018) and RSN4EA (Guo et al., 2019), respectively. These false positive mappings could be avoided if their ontologies are considered.

In this study, we propose an ontology-guided entity alignment method named OntoEA, which enriches the embedded semantics and avoids wrong mappings by exploring the class conflicts. OntoEA can work well in two contexts: (*i*) one ontology is shared across the to-be-aligned KGs; (*ii*) the ontologies of the to-be-aligned KGs are separated. The first context seems to follow a strict assumption but is actually quite common. For example, DBpedia has multilingual KGs (or versions) that share the same ontology, and in industrial scenarios, it is preferred to create a common ontology and then construct different KGs from different sources. The second context can be transformed into the first context by pre-aligning the ontologies, using existing ontology alignment systems such as PARIS (Suchanek et al., 2011) and LogMap (Jiménez-Ruiz et al., 2012), and/or cost-sensitive human intervention.

There are two challenges to utilize the ontology. First, it is difficult to embed two KGs together with the ontology, and as far as we know, there are currently no such solutions. Second, the class conflicts, indicated by the class disjointness in the ontology, are not all explicitly defined. Most of them should be learned from the KGs and they actually vary from KG to KG. For example, two classes *Human* and *Animal* are often disjointed in artwork KGs, but *Human* may belong to *Animal* in biological KGs. To address these challenges, we develop a joint embedding method that includes five modules for embedding the KGs, the ontology, the class conflicts, the membership relationships, and the seed mappings, respectively. Specifically, we develop a class conflict matrix (CCM) to represent different kinds of class conflicts, including those explicitly defined as disjoint, those indicated by the class hierarchy (e.g., sibling classes likely to be disjointed) and those plausible deduced from the KGs.

To the best of our knowledge, OntoEA is the first to utilize the ontology and the embedding for KG alignment. Extensive experiments on seven benchmarks have verified the effectiveness of the ontology guidance with both one shared ontology or two separated ontologies. OntoEA consistently outperforms the state-of-the-art baselines on all the benchmarks. For example, it on average achieves

over 35% higher Hits@1, Hits@5, and MRR than the best baseline on MED-BBK-9K — a new and challenging industrial benchmark. Last but not least, we extend the current benchmarks with ontologies and membership relationships. The source code and benchmarks are publicly accessible at <https://github.com/ZihengZZH/OntoEA>.

## 2 Methodology

### 2.1 Problem Statement

A KG is denoted as  $G = (E, R, T)$ , where  $E, R, T$  are the sets of entities, relations and triples, respectively. Each triple  $(h, r, t) \in T$ , includes a head entity  $h \in E$ , a relation  $r \in R$ , and a tail entity  $t \in E$ . Their embeddings are denoted as  $\mathbf{h}, \mathbf{r}, \mathbf{t}$ , respectively. For two to-be-aligned KGs  $G_i = (E_i, R_i, T_i)$  and  $G_j = (E_j, R_j, T_j)$ , an entity mapping, denoted as  $m = (e_i, e_j)$  where  $e_i \in E_i, e_j \in E_j$ , indicates that  $e_i$  and  $e_j$  refer to the same real-world object. The entity alignment (EA) task aims to find all the mappings  $M$  between  $E_i$  and  $E_j$ , where we assume a small set of known entity mappings (or seed mappings)  $M_s$  are given.

Each KG is assumed to be associated with an ontology, which contains hierarchical classes and optionally disjoint constraints between classes. For simplicity, we consider the classes as entities in KG and the subsumption relationships between classes, often known as *rdfs:subClassOf*, as relations in KG. The simplified ontology is therefore regarded as a graph. For KGs  $G_i$  and  $G_j$ , their associated ontologies are represented as  $O_i = (C_i, H_i)$  and  $O_j = (C_j, H_j)$ , respectively. We simply denote both  $O_i$  and  $O_j$  as  $O = (C, H)$  if  $G_i$  and  $G_j$  share one ontology (i.e.,  $O_i \equiv O_j$ ); otherwise we denote the merged ontology after ontology alignment as  $O = (C, H)$ . Note  $C_i, C_j$  and  $C$  are the class sets, while  $H_i, H_j$  and  $H$  are the triple sets with only the *subClassOf* relation. Furthermore, the membership relationships, which link the entities and the corresponding classes, are denoted as  $B_i$  and  $B_j$ , and e.g.,  $B_i$  links  $G_i$  and  $C$  via  $b_i = (e_i, c)$  where  $e_i \in E_i$  and  $c \in C$ . The class and membership embeddings are denoted as  $\mathbf{c}$  and  $\mathbf{b}$ , respectively.

### 2.2 OntoEA Framework

As shown in Fig. 2, OntoEA includes five modules: (*i*) *entity embedding* which embeds each KG into a separate embedding space; (*ii*) *ontology embedding* which embeds the class hierarchical structure with a non-linear transformation; (*iii*) *conflict loss*

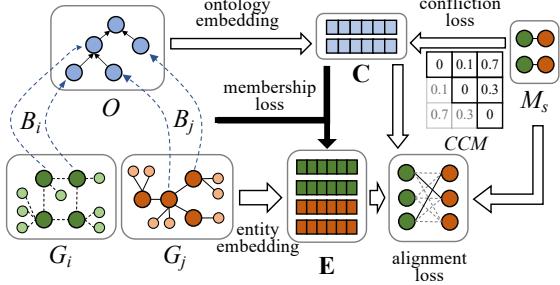


Figure 2: The framework of OntoEA.

which incorporates all potential class conflicts in the embeddings with the CCM; (iv) *membership loss* which incorporates the membership relationships and enables the joint learning of the entity embedding and the ontology embedding; (v) *alignment loss* which bridges the embedding spaces of the to-be-aligned KGs via seed mappings. An iterative co-training strategy is proposed to incorporate these five modules and the entity mappings are predicted with a pre-defined similarity measure between the joint embeddings.

### 2.3 Entity Embedding

To embed the KGs  $G_i$  and  $G_j$ , we adopt the translation-based method TransE (Bordes et al., 2013) which interprets a triple  $(h, r, t)$  as a translation by the relation  $r$  from  $h$  to  $t$ . The margin-based loss of  $(h, r, t)$  is defined as  $f_e(h, t) = \|\mathbf{h} + \mathbf{r} - \mathbf{t}\|_2$ , where  $\|\cdot\|_2$  denotes the L2 norm. Besides, we extend the loss with a limit-based scoring loss (Zhou et al., 2017) to ensure the discrimination between the positive and negative triples and also lower scores for positive triples:

$$\mathcal{L}_E = \sum_{(h, t) \in T} \sum_{(h', t') \in T'} \{[\gamma_e^1 + f_e(h, t) - f_e(h', t')]_+ + \alpha_e [f_e(h, t) - \gamma_e^2]_+\}, \quad (1)$$

where  $[\cdot]_+$  denotes the function  $f(x) = \max(0, x)$ , hyperparameters  $\gamma_e^1$  and  $\gamma_e^2$  control the margins,  $\alpha_e$  balances the margin-based loss and the limit-based loss, and  $T'$  is the set of the negative triples with each triple sampled using the  $\epsilon$ -truncated uniform negative sampling strategy (Sun et al., 2018) to distinguish two similar triples.

It is worth noting that this study focuses on the EA task and the joint embedding challenge, and TransE is chosen due to its simplicity and efficiency. Our OntoEA framework is open to other advanced KG embedding methods, such as GCN (Wang et al., 2018) and RSN (Guo et al., 2019).

### 2.4 Ontology Embedding

The shared or merged ontology  $O = (C, H)$  is composed of triples with the *subClassOf* relation. For simplicity, a triple  $(c_h, r, c_t)$  is written as a class pair  $(c_h, c_t)$  in  $H$  where  $r := \text{subClassOf}$ . Inspired by Hao et al. (2019) on embedding the hierarchical graph structure, we calculate the scoring loss of  $(c_h, c_t)$  with a non-linear transformation,  $f_o(c_h, c_t) = \|\tanh(\mathbf{W}_o c_h + \mathbf{b}_o) - c_t\|_2$ , where  $\mathbf{W}_o \in \mathcal{R}^{d_o \times d_o}$  and  $\mathbf{b}_o \in \mathcal{R}^{d_o}$  are the learnable parameters, and  $d_o$  denotes the ontology embedding dimension. This tends to encode each class as a sphere and each subclass as a vector in the same semantic space after the non-linear transformation, and the relative positions are employed to model the relations between class and its subclass.

Similarly, we adopt the margin-based and limit-based loss for training:

$$\mathcal{L}_O = \sum_{(c_h, c_t) \in H} \sum_{(c'_h, c'_t) \in H'} \{[\gamma_o^1 + f_o(c_h, c_t) - f_o(c'_h, c'_t)]_+ + \alpha_o [f_o(c_h, c_t) - \gamma_o^2]_+\}, \quad (2)$$

where  $H'$  denotes the set of negative class pairs with each pair sampled by replacing  $c_h$  or  $c_t$  following the uniform negative sampling strategy (Bordes et al., 2013), while  $\gamma_o^1$ ,  $\gamma_o^2$  and  $\alpha_o$  are hyperparameters similar to  $\gamma_e^1$ ,  $\gamma_e^2$  and  $\alpha_e$  in Eq. 1.

Note that we do not directly utilize TransE in the ontology embedding, because the *subClassOf* relation is transitive (e.g., we can infer (*Royalty*, *subClassOf*, *Agent*) via (*Royalty*, *subClassOf*, *Person*) and (*Person*, *subClassOf*, *Agent*) as shown in Fig. 1), which can lead to one-to-many and many-to-one mappings (or triples) in the ontology, and TransE cannot well address the relation with such transitive property (Lv et al., 2018).

### 2.5 Confliction Loss

We use a class conflict matrix (CCM) to represent the inter-class conflicts that are either explicitly defined by class disjointness or implicitly discovered from the entities. Within the CCM, the entry on the  $i^{\text{th}}$  row and  $j^{\text{th}}$  column, denoted as  $m_{i,j}$ , represents the conflict degree between class  $c_i$  and class  $c_j$ . For one ontology, the CCM is a squared and symmetric matrix, and we only maintain the upper triangular for higher efficiency. Given two classes  $c_i$  and  $c_j$ ,  $m_{i,j} \in [0, 1]$  is calculated as follows. First, we set  $m_{i,j} = 0$  if  $c_i \equiv c_j$ , which ensures each class does not conflict with itself. Second,  $c_i$  and

$c_j$  are regarded as fully conflicted, i.e.,  $m_{i,j} = 1$ , if they are declared as disjointed by the ontology.<sup>1</sup> Third,  $c_i$  and  $c_j$  are regarded as not conflicted, i.e.,  $m_{i,j} = 0$ , if they have at least one common member (entity) or some of their members are given as seed mapping (i.e., there exists  $(e_i, e_j)$  in  $M_s$  such that  $e_i$  belongs to  $c_i$  and  $e_j$  belongs to  $c_j$ ). Note the above three conditions are matched sequentially, and the computation of  $m_{i,j}$  is finished if any condition is met. Finally, if none of the three conditions is met, we follow the principle that the farther two classes are separated in the tree-like class hierarchy structure, the lower semantic similarity and higher conflict degree the two classes have (Mumtaz and Giese, 2020). To calculate the distance between  $c_i$  and  $c_j$ , we use the set of classes passed by routing from  $c_i$  and  $c_j$  to the root class, denoted as  $S(c_i)$  and  $S(c_j)$ , respectively, and then adopt the ratio of the intersection of  $S(c_i)$  and  $S(c_j)$ . Accordingly, we have  $m_{i,j} = 1 - \frac{|S(c_i) \cap S(c_j)|}{|S(c_i) \cup S(c_j)|}$  where  $|\cdot|$  denotes the set cardinality.

For the implicitly discovered conflicts, we assume the small conflict degree between two classes if their embeddings are similar (i.e., high cosine similarity). Thus we calculate another cosine similarity-based class conflict degree:  $d_{cos}(c_i, c_j) = 1 - \cos(\mathbf{c}_i, \mathbf{c}_j)$ , where  $\mathbf{c}_i$  and  $\mathbf{c}_j$  represent the embeddings of  $c_i$  and  $c_j$ . We propose to minimize the following negative log-likelihood loss to incorporate the class conflicts represented by CCM into the class embeddings,

$$\mathcal{L}_C = - \sum_{c_i \in C} \sum_{c_j \in C} m_{i,j} \log d_{cos}(c_i, c_j). \quad (3)$$

## 2.6 Membership Loss

We develop the membership embedding module to utilize the membership relationships,  $B_i$  and  $B_j$ , to associate the KG embedding spaces with the ontology embedding space, which is regarded as enhancing the KG embeddings with the ontology semantics. Given one membership relationship  $b = (e, c)$ , we utilize a non-linear transformation to map the entity embeddings to the ontology embedding space, and we calculate the scoring loss as  $f_m(e, c) = \|\tanh(\mathbf{W}_m \mathbf{e} + \mathbf{b}_m) - \mathbf{c}\|_2$  where  $\mathbf{W}_m \in \mathcal{R}^{d_e \times d_o}$  and  $\mathbf{b}_m \in \mathcal{R}^{d_o}$  are learnable parameters,  $d_e$  and  $d_o$  denote the dimension of KG embedding and ontology embedding, respectively.

<sup>1</sup>For ontologies by Web Ontology Language (OWL), the class disjointness is common and is usually defined as a constraint by the built-in relation *owl:disjointWith*.

Similarly, we minimize the following margin-based and limit-based loss to model all the membership relationships:

$$\begin{aligned} \mathcal{L}_M = & \sum_{(e, c) \in B} \sum_{(e', c') \in B'} \{[\gamma_m^1 + f_m(e, c) - f_m(e', c')] + \\ & + \alpha_m [f_m(e, c) - \gamma_m^2]_+\}, \end{aligned} \quad (4)$$

where  $B'$  denotes the set of negative membership relationships with each relationship created by replacing the class  $c$  following the uniform negative sampling strategy (Bordes et al., 2013), and  $\gamma_m^1$ ,  $\gamma_m^2$  and  $\alpha_m$  are hyperparameters similar to  $\gamma_e^1$ ,  $\gamma_e^2$  and  $\alpha_e$  in Eq. 1.

## 2.7 Alignment Loss

For the alignment embedding module, OntoEA utilizes the seed mappings  $M_s$  to bridge the embedding spaces of  $G_i$  and  $G_j$  such that the equivalent mappings between two cross-KG entities can be calculated via some distance metrics, such as cosine similarity. Given a seed mapping  $m = (e_i, e_j)$ , its score is calculated as  $f_a(e_i, e_j) = \|\mathbf{W}_a \mathbf{e}_i - \mathbf{e}_j\|_2$  where  $\mathbf{W}_a \in \mathcal{R}^{d_e \times d_e}$  is a learnable translation matrix, and the training loss is defined as,

$$\mathcal{L}_A = \sum_{(e_i, e_j) \in M_s} f_a(e_i, e_j). \quad (5)$$

## 2.8 Iterative Co-Training and Prediction

To incorporate the aforementioned modules and obtain the embeddings of  $G_i$ ,  $G_j$  and  $O$ , we can directly minimize the following loss:

$$\mathcal{L} = \mathcal{L}_E + \mathcal{L}_O + \lambda_1 \mathcal{L}_C + \lambda_2 \mathcal{L}_M + \lambda_3 \mathcal{L}_A, \quad (6)$$

where  $\lambda_1$ ,  $\lambda_2$ , and  $\lambda_3$  are hyperparameters that balance the losses of conflict embedding, membership embedding and alignment embedding, respectively. Instead of directly optimizing  $\mathcal{L}$ , we use an iterative co-training strategy in OntoEA to reduce model complexity and accelerate model convergence. At each iteration, OntoEA first optimizes  $\mathcal{L}_E$  and  $\mathcal{L}_O$  independently, then sequentially optimizes  $\mathcal{L}_C$  and  $\mathcal{L}_M$ , and finally optimizes  $\mathcal{L}_A$ . The iteration stops until some stopping criterion on the validation set is met.

With the embeddings of  $G_i$ ,  $G_j$  and  $O$ , we calculate the entity mappings with the cosine similarity. Given two entities  $e_i \in G_i$  and  $e_j \in G_j$ , the weighted similarity score is calculated as:

$$sim(e_i, e_j) = \beta \cos(\mathbf{e}_i, \mathbf{e}_j) + (1 - \beta) \cos(\mathbf{c}_i, \mathbf{c}_j), \quad (7)$$

where hyperparameter  $\beta \in [0, 1]$  balances the similarities of entity embeddings and class embeddings. It is possible that one entity has multiple classes declared; for example, *Victoria* in Fig. 1 can be an entity of *Royalty* and another class like *Female Leader*. In this case we calculate the average of the embeddings of all the declared classes as the class embedding ( $c_i$  or  $c_j$ ).<sup>2</sup> In the prediction, for each entity in  $G_i$  to be aligned, we rank all candidate entities in  $G_j$  by their weighted similarity scores.

### 3 Benchmarks

#### 3.1 Benchmark Details

For comprehensive evaluation, we adopt six popular EA benchmarks including EN-FR-15K-V1/V2, EN-DE-15K-V1/V2 and D-W-15K-V1/V2 from Sun et al. (2020b), and an industrial benchmark MED-BBK-9K from Zhang et al. (2020). Note that for the cross-lingual benchmarks within DBpedia (i.e., EN-FR and EN-DE), the to-be-aligned KGs share one ontology, while for the other benchmarks, the to-be-aligned KGs have different ontologies that need to be aligned beforehand.

The benchmarks themselves do not include ontologies, and we thus extract and append an ontology for each KG, which includes the class structure (i.e., *rdfs:subClassOf* relationships) and membership relationships. Each benchmark therefore contains two KGs, their ontologies, and associated membership relationships. The benchmark statistics are shown in Table 1 where “-15K” and “9K” in the benchmark names are omitted. For the KGs of DBpedia, we use the DBpedia ontology and the membership relationships from the DBpedia SPARQL endpoint<sup>3</sup> by querying the classes of each entity with *rdfs:type*. Note we also utilize the defined class disjointness constraints by *owl:disjointWith* for initializing CCM. For the two Wikidata KGs in D-W-V1/V2, we extract their ontologies and membership relationships from the Wikidata SPARQL endpoint<sup>4</sup> using queries with *rdfs:subClassOf* and *rdfs:type*. For the KGs of the industrial benchmark, we use domain knowledge (with the assistance of medical experts) to construct a shared, small-scale, but high-quality ontology, and the corresponding membership relationships.

<sup>2</sup>Note that we do not consider the inferred classes, such as *Person* for *Victoria* in Fig. 1.

<sup>3</sup><http://dbpedia.org/sparql>

<sup>4</sup><https://query.wikidata.org/sparql>

Table 1: Statistics of the benchmarks.

	Dataset	KG	Ontology <sup>b</sup>		Membership <sup>c</sup>	
			#Cls.	#Trs.	#Links	#Roots
<i>share-O<sup>a</sup></i>	EN-FR-V1	EN/FR	189	755	15,000	639
	EN-FR-V2	EN/FR	104	755	15,000	533
	EN-DE-V1	EN/DE	175	755	15,000	155
	EN-DE-V2	EN/DE	86	755	15,000	165
	MED-BBK	MED BBK	11 11	10 10	9,162 9,162	86 3,362
<i>not-share-O</i>	D-W-V1	DB WK	172 140	755 695	15,000 15,000	306 342
	D-W-V2	DB WK	71 68	755 695	15,000 15,000	463 418

<sup>a</sup> “*share-O*” means the to-be-aligned KGs share one ontology while “*not-share-O*” means the to-be-aligned KGs have different ontologies.

<sup>b</sup> “#Cls.” denotes the number of classes and “#Trs.” denotes the number of *rdfs:subClassOf* relation triples.

<sup>c</sup> “#Roots” denotes the number of entities that have no *rdf:type* property and are linked to the root class.

<sup>d</sup> See Table 6 in the Appendix for more benchmark statistics.

#### 3.2 Ontology Alignment

For the benchmarks whose KGs do not share one ontology, we first align the ontologies. We adopt two ontology alignment methods: *manual annotation* and *alignment system*. In the first method, we employ five annotators to annotate class mappings for each ontology pair and the classes annotated by more than three annotators are adopted. It is worth noting that the manual annotation of class mappings is worthwhile and often adopted in KG construction and curation because of the high quality and relatively small scale in comparison with the entity mappings. In the second method, we apply a state-of-the-art ontology alignment system named PARIS<sup>5</sup> and some ad-hoc pre-processing and post-processing for automatic class mapping computation. Please see Appendix C for more details on ontology alignment and merging. For the D-W benchmarks, we considered both methods and compared their performance (see Table 4).

### 4 Experiments

#### 4.1 Experimental Setup

We compare OntoEA against state-of-the-art EA models including four translation-based models, MTransE (Chen et al., 2017); JAPE (Sun et al., 2017); SEA (Pei et al., 2019); BootEA (Sun et al., 2018), two graph neural network based models, GCNAlign (Wang et al., 2018); AliNet (Sun et al., 2020a), and one recurrent neural network based model, RSN4EA (Guo et al., 2018). We also compare OntoEA against some state-of-the-art models that additionally utilize entity surface information

<sup>5</sup><http://webdam.inria.fr/paris/>

Table 2: Overall results of OntoEA and the baselines.

Models		EN-FR-15K-V1			EN-FR-15K-V2			D-W-15K-V1			D-W-15K-V2			MED-BBK-9K		
		H@1	H@5	MRR	H@1	H@5	MRR	H@1	H@5	MRR	H@1	H@5	MRR	H@1	H@5	MRR
w/o SI	MTransE	.247	.467	.351	.240	.436	.336	.259	.461	.354	.271	.490	.376	.004	.014	.012
	JAPE	.262	.497	.372	.292	.524	.402	.250	.457	.348	.262	.484	.368	.003	.009	.009
	SEA	.280	.530	.397	.360	.651	.494	.360	.572	.458	.567	.770	.660	.199	.375	.287
	GCNAlign	.338	.589	.451	.414	.698	.542	.364	.580	.461	.506	.743	.612	.065	.153	.117
	BootEA	<u>.507</u>	<u>.718</u>	<u>.603</u>	<b>.660</b>	<u>.850</u>	<u>.745</u>	<u>.572</u>	<u>.744</u>	<u>.649</u>	<b>.821</b>	<u>.926</u>	<u>.867</u>	<u>.307</u>	<u>.495</u>	<u>.399</u>
	RSN4EA	.393	.595	.487	.579	.759	.662	.441	.615	.521	.723	.854	.782	.195	.311	.253
	AliNet	.258	.437	.339	.359	.569	.453	.270	.403	.331	.522	.698	.601	.017	.042	.033
	OntoEA	<b>.566</b>	<b>.818</b>	<b>.678</b>	<u>.654</u>	<b>.891</b>	<b>.757</b>	<b>.591</b>	<b>.808</b>	<b>.688</b>	<u>.814</u>	<b>.950</b>	<b>.873</b>	<u>.343</u>	<b>.546</b>	<b>.440</b>
Improv. best %		11.6	13.9	12.4	-0.9	4.8	1.6	3.3	8.6	6.0	-0.9	2.6	0.7	11.7	10.3	10.3
w/ SI	AttrE	.481	.671	.569	.535	.746	.631	.299	.467	.381	.489	.695	.585	.194	.363	.279
	MultiKE	.749	.819	.782	<u>.864</u>	.909	<u>.885</u>	.411	.521	.468	.495	.646	.569	.213	.367	.289
	RDGCN	<u>.755</u>	<u>.854</u>	<u>.800</u>	.847	<u>.919</u>	.880	<u>.515</u>	<u>.669</u>	<u>.584</u>	<u>.623</u>	<u>.757</u>	<u>.684</u>	<u>.306</u>	<u>.425</u>	<u>.365</u>
	OntoEA	<b>.797</b>	<b>.871</b>	<b>.832</b>	<b>.901</b>	<b>.981</b>	<b>.931</b>	<b>.553</b>	<b>.787</b>	<b>.656</b>	<b>.795</b>	<b>.943</b>	<b>.860</b>	<u>.517</u>	<b>.703</b>	<b>.604</b>
Improv. best %		5.6	2.0	4.0	4.3	6.7	5.2	7.4	17.6	12.3	27.6	24.6	25.7	68.9	65.4	65.5

(SI) (i.e., entity names) including AttrE (Trisedya et al., 2019), MultiKE (Zhang et al., 2019) and RDGCN (Wu et al., 2019). Since using and not using SI are usually regarded as two different evaluation contexts (Guo et al., 2019), we separately evaluate OntoEA without (w/o) and with (w/) SI. Note that OntoEA with SI is implemented by a simple but effective strategy which initializes the embeddings of the translation-based models in Section 2.3 and Section 2.4 (i.e.,  $h$ ,  $r$ ,  $t$  and  $c$ ) with the pre-trained word embeddings of their names. As in Sun et al. (2020b), we use the unified multilingual word embeddings fastText (Bojanowski et al., 2017) for the cross-lingual benchmarks.<sup>6</sup> All the results of AliNet, and the results of MTransE, JAEP, SEA and GCNAlign on MED-BBK-9K are reproduced locally; while the other baseline results are taken from Sun et al. (2020b) and Zhang et al. (2020). We follow the same train (20%), validation (10%) and test (70%) splits as Sun et al. (2020b) and Zhang et al. (2020).

We implement OntoEA upon the open source library OpenEA<sup>7</sup>. We use the AdaGrad optimizer with a learning rate of 0.01. The batch sizes for entity embedding and ontology embedding are set to 4500 and 64, respectively, and their dimensions are both set to 300. The weight hyperparameters are set as  $\lambda_1 = 1$ ,  $\lambda_2 = 1$ ,  $\lambda_3 = 5$  and  $\beta = 0.5$ , and other hyperparameters are set as  $\gamma_x^1 = 0.01$ ,  $\gamma_x^2 = 2.0$  and  $\alpha_x = 0.2$  for all the losses where  $x \in \{e, o, m\}$ . These hyperparamters are tuned w.r.t. the MRR on the validation set. Please see Appendix B for more implementation details.

<sup>6</sup>The word embeddings are publicly available at <https://fasttext.cc/docs/en/crawl-vectors.html>.

<sup>7</sup><https://github.com/nju-websoft/OpenEA>

With the embeddings we use the nearest neighbor search with cross-domain similarity local scaling (Lample et al., 2018) to calculate the entity matching of each to-be-aligned entity. In the evaluation, we rank matching candidates of each to-be-aligned entity and calculate the metrics of Hits@1 (H@1), Hits@5 (H@5) and mean reciprocal rank (MRR). In the following tables, the best (second best resp.) result is **bolded** (underlined resp.).

## 4.2 Overall Results

Table 2 reports the overall results of OntoEA and the baselines.<sup>8</sup> Overall, in both contexts of using and not using SI, OntoEA performs the best across all the benchmarks on all the metrics, except for H@1 on EN-DE-15K-V2 and D-W-15K-V2.

**Without SI.** Regarding the benchmarks that share one ontology, OntoEA achieves at least 10% higher performance in all the metrics over the best baseline BootEA on EN-FR-15K-V1 which has sparse KG structures. On EN-FR-15K-V2 with dense KG structures, OntoEA achieves higher H@5 and MRR but competitive H@1 in comparison with the best baseline. For the D-W benchmarks whose KGs have different ontologies, we have a similar finding: the outperformance of OntoEA is more significant on the benchmark with sparse KG structures (i.e., D-W-15K-V1) than on the benchmark with dense KG structures (i.e., D-W-15K-V2). On the one hand, dense KG structures indicate that more information can be utilized with better performance, and relatively the additional positive impact by the ontology becomes more limited. On the other hand, the ontologies of both

<sup>8</sup>Results on the two EN-DE benchmarks, which lead to similar findings as the EN-FR benchmarks, are shown in Table 7 in the Appendix.

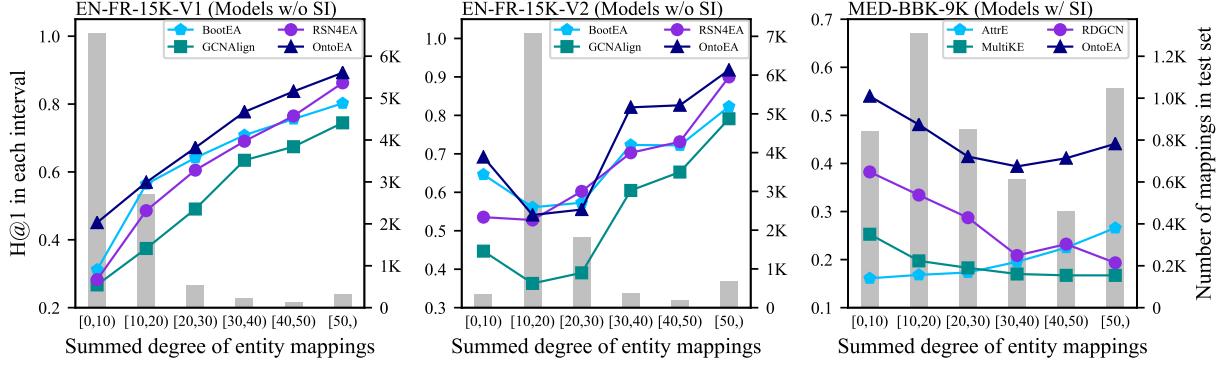


Figure 3: Results of OntoEA and some baselines on test entity mappings with different summed degrees.

EN-FR-15K-V2 and D-W-15K-V2 coincidentally have fewer classes (see Table 1), which would also lead to a lower impact of the ontology guidance. Regarding the industrial MED-BBK-9K, which is quite new and has shown more challenging w.r.t. these baselines (Zhang et al., 2020), OntoEA outperforms the best baseline by more than 10% in all the metrics. Although its ontology has a limited scale, the ontology is of high quality as it is specifically created with the domain knowledge.

**With SI.** OntoEA shows very promising overall results as OntoEA without SI. It outperforms all the baselines on all the benchmarks, with the improvements over the best baseline ranging from 2% to 68.9%. Regarding the industrial MED-BBK-9K, the performance improvement of OntoEA is especially significant with more than 60% in all the metrics. As aforementioned, this benchmark is challenging to these baselines and it has high-quality ontologies. It is worth mentioning that on the D-W benchmarks, the involvement of SI deteriorates the performance of OntoEA and the best baseline, while the positive impact of the ontologies (i.e., the performance gain of OntoEA over the best baseline) becomes more significant.

### 4.3 Ablation Studies

**Model Component Analysis.** To investigate different components in OntoEA, we compare OntoEA variants without the CCM loss (w/o  $\mathcal{L}_C$ ), without the membership relationship loss (w/o  $\mathcal{L}_M$ ), and without the ontology (w/o Onto.). The experimental results of OntoEA w/o SI on EN-FR-15K-V1/V2 are reported in Table 3, which shows that  $\mathcal{L}_M$  contributes more to OntoEA than  $\mathcal{L}_C$  as removing it leads to a larger performance drop. The results OntoEA w/ SI show similar findings and are thus omitted. As expected, the removal of the ontology deteriorates performance the most.

All these results verify the significant role of the ontologies and their associated class conflicts and membership relationships.

Table 3: Results of OntoEA w/o SI and its variants with the removal of different components.

Models	EN-FR-15K-V1			EN-FR-15K-V2		
	H@1	H@5	MRR	H@1	H@5	MRR
OntoEA	<b>.566</b>	<b>.818</b>	<b>.678</b>	<b>.654</b>	<b>.891</b>	<b>.757</b>
w/o $\mathcal{L}_C$	.520	.781	.636	.589	.845	.701
w/o $\mathcal{L}_M$	.481	.750	.601	.549	.810	.665
w/o Onto.	.430	.698	.551	.545	.814	.664

**Results on Different Test Entity Mappings.** We split the test mappings according to the summed degree, i.e.,  $\deg(e_i, e_j) := \deg(e_i) + \deg(e_j)$  where  $(e_i, e_j)$  are the entity mappings. We select three different experimental settings (benchmarks and usage of SI) and Fig. 3 shows the results of OntoEA and some competitive baselines on different degree intervals. On EN-FR-15K-V1, OntoEA outperforms the baselines on all the intervals and the performance gap reaches the highest on [0,10). On EN-FR-15K-V2 with dense KG structures, OntoEA performs close to or slightly worse than BootEA on intervals [10,20) and [20,30) but outperforms all the baselines on the other intervals. On MED-BBK-9K, OntoEA performs much better than the baselines on all the intervals. All these observations are consistent with our findings from the overall results in Table 2 and also verify the effectiveness of OntoEA on different test entity mappings.

**Analysis of Class Conflicts.** This part analyses the predicted mappings that have class conflicts, with the results shown in Fig. 4. Note that the class conflict ratio of a method is the rate of its false positive mappings with class conflicts among all its false positive mappings. On EN-FR-15K-V1 and EN-FR-15K-V2, OntoEA significantly decreases the class conflict ratio to 3% and 0.3%, respectively,

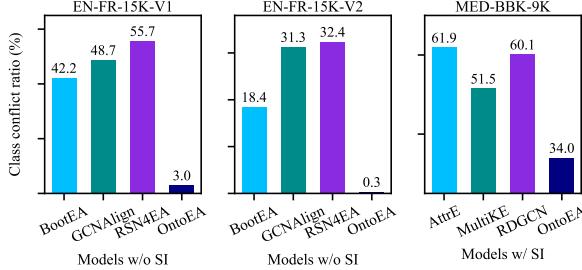


Figure 4: The class conflict ratios.

much lower than those of the baselines. On MED-BBK-9K, OntoEA reduces the class conflict ratio to 34.0% while MultiKE and RDGCN have 51.5% and 60.1%, respectively. The above observations illustrate that OntoEA has effectively implemented our motivation of using the ontologies and class conflicts to avoid false positive mappings.

**Analysis of Ontology Alignment Methods.** As the case that the to-be-aligned KGs have different ontologies is common, we analyse the impact of different ontology alignment methods proposed in Sec. 3.2, with the results in Table 4. We find that OntoEA with manual annotation achieves better results than OntoEA using the PARIS based alignment system method; the latter, however, still has competitive or better results in comparison with the best baseline (i.e., BootEA or RDGCN). This also provides motivations for the future research and industrial deployment work on iterative ontology alignment to keep a balance between the annotation cost and the alignment quality.

Table 4: Results of OntoEA using different ontology alignment methods where “-M” denotes the manual annotation while “-A” denotes the alignment system.

	Models	D-W-15K-V1			D-W-15K-V2		
		H@1	H@5	MRR	H@1	H@5	MRR
w/o SI	OntoEA-M	.591	.808	.688	.814	.950	.873
	OntoEA-A	.572	.795	.673	.762	.907	.827
	BootEA	.572	.744	.649	.821	.926	.867
w/ SI	OntoEA-M	.553	.787	.656	.795	.943	.860
	OntoEA-A	.510	.758	.621	.737	.893	.808
	RDGCN	.515	.669	.584	.623	.757	.684

## 5 Related Work

**Embedding-based KG Alignment.** These methods can be categorised into translation-based, GNN-based and RNN-based. The translation-based methods mainly rely upon some translation-based KG embedding models; for example, MTransE adopts TransE and various mapping losses to align cross-lingual KGs (Chen et al., 2017). To utilize more useful information in the KG, JAPE embeds the to-

be-aligned KGs into one unified vector space via leveraging attribute correlations (Sun et al., 2017); AttrE exploits attributes by generating attribute character embeddings (Trisedya et al., 2019); and MultiKE makes full use of entity names, entity attributes and graph structures via multi-view learning (Zhang et al., 2019). The GNN-based methods are typically based on GNN variants for KG embedding, such as Graph Convolutional Networks (GCNs) Wang et al. (2018), dual-primal GCNs Sun et al. (2020a), and GCNs with attentive aggregation Sun et al. (2020a). As an RNN-based method, RSN4EA generates biased random walks on KGs and learns the embeddings by a recurrent skipping network (Guo et al., 2019). Some work has explored a semi-supervised learning setting, and for instance, BootEA (Sun et al., 2018) adopts the bootstrapping strategy to iteratively append new likely mappings during the learning process.

**KG Embedding with Ontology.** Only a few attempts have been made towards the KG embedding with ontology. The most relevant work is JOIE (Hao et al., 2019) which embeds the KG and the ontology in two embedding spaces and enables them to enhance each other by the cross-view links i.e., membership relationships. Some ontology embedding methods such as OWL2Vec\* (Chen et al., 2021a) can be extended to jointly embed KG plus ontology since the KG can be regarded as an assertion part (ABox) of the ontology. However, as far as we know, there are currently no embedding methods that support two KGs along with their ontologies, let alone utilize the ontologies to augment embedding-based KG alignment.

## 6 Conclusions

This paper presented a novel method OntoEA to augment embedding-based entity alignment with the ontology. OntoEA enriches the semantics of KG embeddings by jointly learning the KGs, the ontology, the membership relationships and the seed mappings, and utilizes the potential class conflicts to avoid false mappings. The evaluation on multiple benchmarks has showed that OntoEA could achieve state-of-the-art performance and reduce the false positive mappings with class conflicts. For future work, we plan to utilize other ontology semantics besides the class hierarchy and class disjointness constraints (Chen et al., 2021b). We also plan to extend the entity embedding module with other advanced KG embedding methods.

## References

- Sören Auer, Chris Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. 2008. DBpedia: A nucleus for a web of open data. In *Proceedings of the 6th International Semantic Web Conference*, volume 4825 of *Lecture Notes in Computer Science*, pages 722–735. Springer.
- Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.
- Antoine Bordes, Nicolas Usunier, Alberto García-Durán, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *27th Annual Conference on Neural Information Processing Systems 2013*, pages 2787–2795.
- Jiaoyan Chen, Pan Hu, Ernesto Jiménez-Ruiz, Ole Magnus Holter, Denvar Antonyrajah, and Ian Horrocks. 2021a. OWL2Vec\*: Embedding of OWL ontologies. *Machine Learning*.
- Jiaoyan Chen, Ernesto Jimenez-Ruiz, Ian Horrocks, Denvar Antonyrajah, Ali Hadian, and Jaehun Lee. 2021b. Augmenting ontology alignment by semantic embedding and distant supervision. In *Proceedings of the 18th Extended Semantic Web Conference*, pages 1–16, Heraklion, Greece.
- Muhao Chen, Yingtao Tian, Mohan Yang, and Carlo Zaniolo. 2017. Multilingual knowledge graph embeddings for cross-lingual knowledge alignment. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017*, pages 1511–1517. ijcai.org.
- Lingbing Guo, Zequn Sun, Ermei Cao, and Wei Hu. 2018. Recurrent skipping networks for entity alignment. *CoRR*, abs/1811.02318.
- Lingbing Guo, Zequn Sun, and Wei Hu. 2019. Learning to exploit long-term relational dependencies in knowledge graphs. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019*, volume 97 of *Proceedings of Machine Learning Research*, pages 2505–2514. PMLR.
- Junheng Hao, Muhao Chen, Wenchao Yu, Yizhou Sun, and Wei Wang. 2019. Universal representation learning of knowledge bases by jointly embedding instances and ontological concepts. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2019*, pages 1709–1719. ACM.
- Aidan Hogan, Eva Blomqvist, Michael Cochez, Claudia d’Amato, Gerard de Melo, Claudio Gutierrez, José Emilio Labra Gayo, Sabrina Kirrane, Sebastian Neumaier, Axel Polleres, Roberto Navigli, Axel Cyrille Ngonga Ngomo, Sabbir M. Rashid, Anisa Rula, Lukas Schmelzeisen, Juan Sequeda, Steffen Staab, and Antoine Zimmermann. 2020. Knowledge graphs. *CoRR*, abs/2003.02320.
- Ian Horrocks. 2008. Ontologies and the semantic web. *Communications of the ACM*, 51(12):58–67.
- Ernesto Jiménez-Ruiz, Bernardo Cuenca Grau, Yujiao Zhou, and Ian Horrocks. 2012. Large-scale interactive ontology matching: algorithms and implementation. In *Proceedings of the 20th European Conference on Artificial Intelligence*, pages 444–449.
- Guillaume Lample, Alexis Conneau, Marc’Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. 2018. Word translation without parallel data. In *International Conference on Learning Representations*.
- Xin Lv, Lei Hou, Juanzi Li, and Zhiyuan Liu. 2018. Differentiating concepts and instances for knowledge graph embedding. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1971–1979, Brussels, Belgium. Association for Computational Linguistics.
- Summaya Mumtaz and M. Giese. 2020. Frequency-based vs. knowledge-based similarity measures for categorical data. In *AAAI Spring Symposium: Combining Machine Learning with Knowledge Engineering*.
- Shichao Pei, Lu Yu, Robert Hoehndorf, and Xiangliang Zhang. 2019. Semi-supervised entity alignment via knowledge graph embedding with awareness of degree difference. In *The World Wide Web Conference, WWW 2019*, pages 3130–3136. ACM.
- Zhiyuan Qi, Ziheng Zhang, Jiaoyan Chen, Xi Chen, Yuejia Xiang, Ningyu Zhang, and Yefeng Zheng. 2021. Unsupervised knowledge graph alignment by probabilistic reasoning and semantic embedding. *arXiv preprint arXiv:2105.05596*.
- Fabian M. Suchanek, Serge Abiteboul, and Pierre Senellart. 2011. PARIS: Probabilistic alignment of relations, instances, and schema. *Proc. VLDB Endow.*, 5(3):157–168.
- Zequn Sun, Wei Hu, and Chengkai Li. 2017. Cross-lingual entity alignment via joint attribute-preserving embedding. In *International Semantic Web Conference*, volume 10587 of *Lecture Notes in Computer Science*, pages 628–644. Springer.
- Zequn Sun, Wei Hu, Qingheng Zhang, and Yuzhong Qu. 2018. Bootstrapping entity alignment with knowledge graph embedding. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018*, pages 4396–4402. ijcai.org.
- Zequn Sun, Chengming Wang, Wei Hu, Muhao Chen, Jian Dai, Wei Zhang, and Yuzhong Qu. 2020a. Knowledge graph alignment network with gated multi-hop neighborhood aggregation. In *Proceedings of the Conference on Artificial Intelligence*, pages 222–229. AAAI Press.

- Zequn Sun, Qingheng Zhang, Wei Hu, Chengming Wang, Muha Chen, Farahnaz Akrami, and Chengkai Li. 2020b. A benchmarking study of embedding-based entity alignment for knowledge graphs. *Proc. VLDB Endow.*, 13(11):2326–2340.
- Bayu Distiawan Trisedya, Jianzhong Qi, and Rui Zhang. 2019. Entity alignment between knowledge graphs using attribute embeddings. In *Proceedings of the Conference on Artificial Intelligence*, pages 297–304. AAAI Press.
- Denny Vrandečić and Markus Krötzsch. 2014. Wikidata: A free collaborative knowledgebase. *Communications of the ACM*, 57(10):78–85.
- Zhichun Wang, Qingsong Lv, Xiaohan Lan, and Yu Zhang. 2018. Cross-lingual knowledge graph alignment via graph convolutional networks. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 349–357, Brussels, Belgium. Association for Computational Linguistics.
- Yuting Wu, Xiao Liu, Yansong Feng, Zheng Wang, Rui Yan, and Dongyan Zhao. 2019. Relation-aware entity alignment for heterogeneous knowledge graphs. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019*, pages 5278–5284. ijcai.org.
- Qingheng Zhang, Zequn Sun, Wei Hu, Muha Chen, Lingbing Guo, and Yuzhong Qu. 2019. Multi-view knowledge graph embedding for entity alignment. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019*, pages 5429–5435. ijcai.org.
- Ziheng Zhang, Jiaoyan Chen, Xi Chen, Hualuo Liu, Bo Liu, YueJia Xiang, and Yefeng Zheng. 2020. An industry evaluation of embedding-based entity alignment. In *Proceedings of the 28th International Conference on Computational Linguistics: Industry Track*, pages 179–189, Online. International Committee on Computational Linguistics.
- Xiaofei Zhou, Qiannan Zhu, Ping Liu, and Li Guo. 2017. Learning knowledge embeddings by combining limit-based scoring loss. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, CIKM 2017*, pages 1009–1018. ACM.

## Appendix

### A More Benchmark Details

Table 6 shows the full statistics of all the benchmarks used in the experiments, including four cross-lingual benchmarks and two cross-KG benchmarks from OpenEA (Sun et al., 2020b), and one recent industrial cross-KG benchmark from Zhang et al. (2020). We report the benchmark statistics from three perspectives: the ontologies, the KGs and the membership relationships. The benchmarks are divided as previous work that 20%, 10% and 70% of all the entity mappings are used as training, validation and test sets, respectively (Sun et al., 2020b; Zhang et al., 2020).

### B More Implementation Details

Our experiments are conducted on a workstation with an Intel Xeon E5 2.40 GHz CPU, 128 GB memory, an NVIDIA Tesla M40 GPU, and CentOS 7.2. Search spaces of the hyperparameters used in OntoEA are listed in Table 5. We apply a grid search strategy for the hyperparameter configuration that performs the best on the validation set with respect to mean reciprocal rank (MRR).

Table 5: Search spaces of the hyperparameters.

Hyperparameters	Search spaces
learning rate	{1e-3, 5e-3, 1e-2, 5e-2}
batch size ( <i>entity</i> )	{4000, 4500, 5000}
batch size ( <i>ontology</i> )	{32, 64, 128}
$\gamma_x^1$ in Eqs. (1) (2) (4) *	{0.01, 0.02, 0.03}
$\gamma_x^2$ in Eqs. (1) (2) (4) *	{1.0, 2.0, 3.0}
$\alpha_x$ in Eqs. (1) (2) (4) *	{0.1, 0.2, 0.3}
$\lambda_1$ in Eq. (6)	{0, 1, 2, 3, 4, 5}
$\lambda_2$ in Eq. (6)	{0, 1, 2, 3, 4, 5}
$\lambda_3$ in Eq. (6)	{0, 1, 2, 3, 4, 5}
$\beta$ in Eq. (7)	{0.3, 0.4, 0.5, 0.6, 0.7}

\* where  $x \in \{e, o, m\}$ .

## C Ontology Alignment Methods

In this section, we present the details of the two ontology alignment methods: *manual annotation* and *alignment system*.

### C.1 Manual Annotation

We refer to crowdsourcing for manual annotation of the class mappings by the following steps which are presented with the example of the DBpedia ontology and the Wikidata ontology in the D-W benchmarks. First, we recruit five volunteers with at least undergraduate education as the annotators

and provide them the two to-be-aligned ontologies. The DBpedia ontology ( $O_D$ ) contains a total of 755 classes while the Wikidata ontology ( $O_W$ ) contains 695 classes. Second, for each class in  $O_D$ , denoted as  $c_D$ , each annotator is required to consider as more classes in  $O_W$  as possible and find the equivalent class mapping  $(c_D, c_W)$ , where  $c_W$  denotes a class in  $O_W$ . Note that the annotators are allowed to utilize her/his knowledge and accelerate the annotation by searching for class  $c_W \in O_W$ , using some friendly ontology management and accessing software such as Protégé. After all five annotators finish the work, only the class mappings that are labelled by more than three annotators are accepted as the final class mappings, denoted as  $M_C$ .

With the class mappings, we need to further merge the two ontologies as one shared ontology which can then be processed by OntoEA. To this end, our solution is to build new membership relationships between the entities of the Wikidata KG and the classes of the DBpedia ontology. For each original membership relationship within the Wikidata KG and ontology, we replace its class  $c_W$  with  $c_D$  if  $(c_D, c_W) \in M_C$ , or with the root class *owl:Thing* otherwise.

### C.2 Alignment System

#### Algorithm 1 Alignment System Method

---

**Require:** Ontologies  $O_D$  and  $O_W$ , threshold  $\lambda$

- 1: Build directed graphs  $D_D$  and  $D_W$  from  $O_D$  and  $O_W$ , respectively, with the relation of *rdfs:subClassOf*
- 2: Obtain equivalent class pairs  $P_C$  from pre-defined *owl:equivalentClass* in  $O_D$  and  $O_W$
- 3: Generate  $D'$  via connecting  $D_D$  and  $D_W$  with  $P_C$
- 4: Define  $\phi(c_1, c_2)$  as the length of directed walk from  $c_1$  to  $c_2$  where  $c_1, c_2 \in D'$
- 5: Initialize empty mapping sets  $M_C$  and  $M'_C$
- 6: **for** each class  $c_W$  in  $O_W$  **do**
- 7:     Route  $D'$  starting from  $c_W$  and output routed class set  $S = \{c\}$  where  $c \in D'$
- 8:     **if**  $S = \emptyset$  **then**
- 9:         Append  $(c_W, owl:Thing)$  to  $M_C$
- 10:     **else**
- 11:         Find a class  $c$  in  $S$  such that for any  $c'$  in  $S$   $\phi(c, owl:Thing) \geq \phi(c', owl:Thing)$
- 12:         Append  $(c_W, c)$  to  $M_C$
- 13:     **end if**
- 14: **end for**
- 15: Perform PARIS on  $O_D$  and  $O_W$
- 16: Append  $M'_C$  from PARIS with threshold  $\lambda$
- 17: **for** each class mapping  $(c_w, c_d)$  in  $M'_C$  **do**
- 18:     **if**  $(c_w, c'_d) \in M'_C$  and  $c'_d \neq c_d$  **then**
- 19:         Remove  $(c_w, c_d)$  from  $M_C$
- 20:         Append  $(c_w, c'_d)$  to  $M_C$
- 21:     **end if**
- 22: **end for**
- 23: **return**  $M_C$

---

Table 6: Full statistics of all the benchmarks.

Dataset		KG	Ontology <sup>b</sup>		Knowledge Graph				Membership <sup>c</sup>	
#Cls.	#Sub tr.		#Rel.	#Attr.	#Rel tr.	#Attr tr.	#Links	#Roots		
<i>share-O<sup>a</sup></i>	EN-FR-15K-V1	EN FR	189	755	267 210	308 404	47,334 40,864	73,121 67,167	15,000	639
	EN-FR-15K-V2	EN FR	104	755	193 166	189 221	96,318 80,112	66,899 68,779	15,000	533
	EN-DE-15K-V1	EN DE	175	755	215 131	286 194	47,676 50,419	83,775 156,150	15,000	155
	EN-DE-15K-V2	EN DE	86	755	169 96	171 116	84,867 92,632	81,998 186,335	15,000	165
	MED-BBK-9K	MED BBK	11 11	10 10	32 20	19 21	158,357 50,307	11,467 44,987	9,162 9,162	86 3,362
<i>not-share-O</i>	D-W-15K-V1	DB WK	172 140	755 695	248 269	342 649	38,265 42,746	68,258 138,246	15,000 15,000	306 342
	D-W-15K-V2	DB WK	71 68	755 695	167 121	175 457	73,983 83,365	66,813 175,686	15,000 15,000	463 418

<sup>a</sup> “*share-O*” means the to-be-aligned KGs share one ontology while “*not-share-O*” means the to-be-aligned KGs have different ontologies.

<sup>b</sup> “#Cls.” denotes the number of classes and “#Trs.” denotes the number of *rdfs:subClassOf* relation triples.

<sup>c</sup> “#Roots” denotes the number of entities that have no *rdf:type* property and are linked to the root class.

Table 7: Overall results of OntoEA and the baselines on the EN-DE benchmarks.

Models	EN-DE-15K-V1			EN-DE-15K-V2		
	H@1	H@5	MRR	H@1	H@5	MRR
w/o SI	MTransE	.307	.518	.407	.193	.352
	JAPE	.288	.512	.394	.167	.329
	SEA	.530	.718	.617	.606	.779
	GCNAlign	.481	.679	.571	.534	.717
	BootEA	.675	.820	.740	.833	.912
	RSN4EA	.587	.752	.662	.791	.890
	AliNet	.243	.353	.295	.169	.247
	OntoEA	<b>.742</b>	<b>.897</b>	<b>.812</b>	<b>.864</b>	<b>.955</b>
Improv. best %		9.9	9.4	9.7	3.7	4.7
w/ SI	AttrE	.517	.687	.597	.650	.816
	MultiKE	.756	.809	.782	.755	.813
	RDGCN	.830	.895	.859	.833	.891
	OntoEA	<b>.850</b>	<b>.906</b>	<b>.877</b>	<b>.914</b>	<b>.977</b>
Improv. best %		2.4	1.2	2.1	9.7	9.7

The automatic method based on the ontology alignment system PARIS and some ad-hoc pre-processing and post-processing is shown in Algorithm 1. The pre-processing from Line 1 to Line 13 obtains class mappings via finding the most fine-grained mappings (i.e., the farthest class node in the directed graph). Since PARIS generally outputs reliable class mappings, the post-processing from

Line 16 to Line 21 updates class mappings with those output from PARIS.

In the implementation we set PARIS hyperparameter  $\lambda$  to 0.4. The *owl:equivalentClass* relationship across the DBpedia ontology and the Wikidata ontology is acquired via DBpedia and Wikidata SPARQL endpoints. After getting  $M_C$ , we merge the two ontologies as introduced in the method of manual annotation.

## D More Experimental Results

Table 7 reports the overall results on the cross-lingual benchmarks EN-DE-15K-V1 and EN-DE-15K-V2. We have similar findings in these two EN-DE benchmarks as in the EN-FR benchmarks that OntoEA consistently outperform baselines, w/o SI and w/ SI. On EN-DE-15K-V1, OntoEA generally outperforms competitive models by up to 9.9% in H@1 and the performance gap is larger in the models w/o SI than in the models w/ SI. On the dense benchmark (V2), OntoEA shows limited performance gain (about 4%) compared with the models w/o SI but much larger performance gain (more than 9%) compared with the models w/ SI.