ASTRA: Adaptive Structure-Aware Post-Hoc Alignment of Knowledge Graph Embeddings

Duygu Ekinci[1], N'Dah Jean Kouagou[1], Shivam Sharma[1], Albert Khomich[1], Mohamed Ahmed Sherif[1], and Axel-Cyrille Ngonga Ngomo[1]

Data Science Group, Heinz Nixdorf Institute, Paderborn University, Germany

Abstract. Many existing knowledge graph embedding (KGE) algorithms learn vector representations of entities on a single knowledge graph (KG), which can be leveraged for downstream tasks such as link prediction, entity typing, and multi-hop query answering. However, when embeddings are trained separately on different KGs, even using the same KGE model, they are located in separate vector spaces, which makes them unsuitable for cross-KG tasks such as entity disambiguation, nearest neighbor search, or cross-domain recommendation. One common strategy to address this challenge is to fuse multiple KGs into a unified KG prior to KGE training; however, this approach introduces substantial computational overhead and becomes infeasible for large-scale KGs due to hardware limitations. To overcome these limitations, we propose AstrA (Adaptive Structure-Aware Post-Hoc Alignment), a model-agnostic framework that aligns independently trained KGE without requiring KGE retraining or KG fusion. By learning adaptive structure-aware nonlinear transformations, AstrA preserves the original domain-specific semantics while enabling consistent cross-KG inference. Experiments on benchmark datasets for entity alignment and link prediction show that AstrA substantially improves post-hoc alignment quality while producing embeddings that remain usable for cross-KG inference.

Keywords: Knowledge Graph Embeddings, Entity Alignment, Orthogonal Procrustes, Neural Representation Alignment.

1 Introduction

Knowledge graph embedding (KGE) models have been used for many applications [9] including link prediction, entity resolution, and entity classification. In recent years, numerous methods have been proposed to generate KGEs, e.g., TransE [1], DistMult [31], ComplEx [27], DualE [3], and DeCal [25]. Although the same embedding model is applied across different knowledge graphs (KGs), independent training on each graph often results in disjoint vector spaces. Therefore, equivalent entities across these KGs are represented differently, which limits the applicability of such embeddings beyond the KGs on which they are trained [14].

Joint-training entity alignment approaches compute aligned embeddings for multilingual KGs by leveraging available `owl:sameAs` links between equivalent

entities [5,21,28]. However, joint training on large-scale KGs such as the `DBpedia-Wikidata` datasets [15], which comprise approximately 180 million entities, 15 thousand relations, and 1.2 billion triples, poses significant computational challenges, which would be further exacerbated when dealing with even larger KGs. In our experiments on DGX nodes (8× A100 GPUs, 320 GB) and standard GPU nodes (4× A100, 160 GB), training on a 22.4 TB KG exceeds hardware capacities and job-time limits. These results underscore that training on fused KGs requires significant memory and computational resources, posing challenges for many practical applications.

To address these challenges, we propose AstrA, a model-agnostic framework for post-hoc alignment of embeddings generated from separately trained KGs. Unlike previous methods that train a single embedding model over fused KGs [15] or rely on joint-training [18,10], AstrA post-hoc aligns independently trained embeddings, generated using the same KGE model into a shared space through non-linear transformations. This approach provides an adaptable alignment method by dynamically learning the transformation function, which is more effective than static linear transformations like *Procrustes* alignment [19].

AstrA supports both entity alignment and link prediction within a unified framework. For entity alignment, it learns structure-aware transformations that preserve cross-KG semantic equivalence and is evaluated using top-k nearest neighbor retrieval. For link prediction, alignment is incorporated into an iterative fine-tuning process where relational feedback refines the transformation parameters, enabling cross-KG inference without merged training.

2 Background and Notation

Knowledge Graph Embeddings and Alignment. A KG can be regarded as a set of triples $G = \{\langle e^{(i)}, r^{(i)}, e^{(i)} \rangle\}_{i=1}^{N} \subseteq \mathcal{E} \times R \times \mathcal{E}$, where $\mathcal{E}$ and R denote the sets of all entities and all relations, respectively [13]. KGE methods learn vector representations of entities within a single KG, which support downstream tasks such as link prediction, entity typing, and multi-hop query answering. To support cross-KG inference, embeddings from two separate KGs can be aligned such that equivalent entities connected by `owl:sameAs` links are represented in the same vector space. We refer to the entity sets of two different KGs, G_1 and G_2, to be aligned by $\mathcal{E}_1$ and $\mathcal{E}_2$, respectively. KGE models produce embedding matrices $\mathbf{E}_1, \mathbf{R_1} = \Psi(G_1)$ and $\mathbf{E}_2, \mathbf{R_2} = \Psi(G_2)$, where Ψ is an embedding model, $\mathbf{E}_1$ and $\mathbf{E}_2$ are entity embeddings, and $\mathbf{R}_1$ and $\mathbf{R}_2$ are relation embeddings of G_1 and G_2, respectively. Let $\mathcal{A} = \{(e_1, e_2), (...), \ldots\} \subseteq \mathcal{E}_1 \times \mathcal{E}_2$ be the set of aligned entity pairs linked via `owl:sameAs` between G_1 and G_2. For each pair $(e_1, e_2) \in \mathcal{A}$, embedding-based entity alignment approaches operate on the embeddings corresponding to the aligned entities $\mathbf{e}_1$ and $\mathbf{e}_2$. We denote the initial embeddings of all aligned entities between G_1 and G_2 by $\mathbf{E}_1^{\mathcal{A}}$ and $\mathbf{E}_2^{\mathcal{A}}$, respectively. The final aligned embeddings produced by our proposed approach AstrA on G_1 and G_2 are referred to as $\mathbf{E}_1^*$ and $\mathbf{E}_2^*$, respectively.

3 Related Work

Numerous KGE models have been proposed and are frequently combined with alignment strategies to enable integration and reasoning across KGs [5,33,6]. Foundational models such as TransE [1] represent relations as translations in the embedding space but struggle with complex relation patterns. To address these limitations, bilinear models such as DistMult [31] and ComplEx [27] model relational interactions through multiplicative scoring functions, enabling the representation of symmetric and asymmetric relations. More expressive algebraic approaches have also been proposed. DualE [3] employs dual quaternions to capture both rotational and translational patterns, while DeCal [25] generalizes several embedding models using degenerate Clifford algebras to represent more complex relational semantics.

Building upon some of these embedding models, various alignment strategies have been proposed that incorporate structural patterns, attribute correlations, and textual descriptions using supervised, semi-supervised, or unsupervised techniques. Many of these approaches employ a *joint-training entity alignment*, where embeddings for different KGs are learned simultaneously under alignment supervision [2,4,5,33,6]. MTransE [5] extends TransE to multilingual KGs by training separate models per language and jointly optimizing with an alignment loss. MTransE supports multiple alignment variants, including distance-based and linear transformation-based mappings, and improves cross-lingual and monolingual tasks. IPTransE [33] jointly trains embeddings using TransE, aligning entities using parameter sharing and iteratively expanding alignment pairs. Another method, KDCoE [6], co-trains two models, one over KG structure and the other over entity descriptions, enabling mutual reinforcement through nearest-neighbor alignment. JAPE [21] merges the two graphs utilizing seed alignments and embeds them jointly using TransE while refining entity positions through attribute correlations. Similarly, Graph Convolutional Network (GCN)-based methods [28] apply separate GCNs per KG but optimize both using alignment loss over seed pairs. The attribute embedding-based approach [26] also constructs a unified graph after predicate alignment and jointly embeds structural and attribute-based information using character-level attribute embeddings. Another approach applies bootstrapping methods that progressively improve alignment scores during the training phase and incorporate attribute embeddings to improve entity matching [22]. More recent work explores graph neural network (GNN) architectures to model cross-graph dependencies, optimizing shared alignment objectives over seed entity pairs [28,29,30]. Other approaches emphasize attribute-level interactions or bootstrapping strategies to compensate for weak structural correspondence [26,22,32]. A common characteristic of these approaches is their reliance on a specific embedding model, where alignment constraints are incorporated directly into the training process. This results in tightly coupled representations that are often model-dependent and less flexible for post-hoc alignment. Moreover, since the embeddings are learned jointly, they are tightly linked to the specific alignment pairs used during training. Introduc-

ing new `owl:sameAs` links requires retraining to preserve embedding consistency, which increases computational demand.

In contrast to joint embedding training approaches, some studies have explored *post-hoc embedding alignment* techniques, i.e., align embeddings after they are independently trained. These methods aim to avoid the computational complexity of joint training, particularly for large or heterogeneous KGs. *Wasserstein Procrustes* [11] combines optimal transport theory with the classical Procrustes method to jointly estimate a transformation and matching between embedding spaces, while *Orthogonal Procrustes Analysis (OPA)* [17] solves for the optimal rotation. However, these methods assume a linear transformation between embedding spaces. When the embeddings are produced by independently trained models on structurally different KGs, this assumption can be too restrictive.

Both *Joint-training entity alignment* approaches and linear post-hoc alignment methods have demonstrated promising results. However, these methods often depend on a specific embedding model and typically require retraining when new alignment links (e.g., `owl:sameAs`) are introduced. Furthermore, linear alignment methods fail to capture the non-linear alignment patterns observed in the experiments. These limitations underscore the need for a more flexible, model-independent post-hoc embedding alignment approach that supports both linear and non-linear transformations without requiring retraining the embeddings.

4 Approach

We propose AstrA, an adaptive structure-aware post-hoc alignment framework that aligns entity embeddings from independently trained KGs (Figure 1). By operating directly on pretrained embeddings rather than performing costly, often impractical KG fusion prior to training, AstrA reduces computational cost and improves scalability for large KGs. As a baseline, we consider *Orthogonal Procrustes Alignment (OPA)* [8], a widely used linear alignment method based on seed correspondences. However, its linearity limits its ability to capture complex alignment patterns; in contrast, AstrA learns adaptive non-linear mappings that project multiple embedding spaces into a shared latent space while preserving KG-specific structural properties.

4.1 Baseline Approach: Orthogonal Procrustes Alignment (OPA)

As a baseline, we consider OPA [8], a linear baseline post-hoc alignment method. OPA finds the optimal orthogonal transformation R minimizing the *Frobenius norm* [19] between aligned embeddings from two spaces: $\min_R \|\mathbf{E}_1 R - \mathbf{E}_2\|_F^2$, where $\mathbf{E}_1$ and $\mathbf{E}_2$ are matrices of matched entity embeddings (e.g., obtained via `owl:sameAs` links or generated by a link discovery system such as Limes [16]). The approach defines the matrix product, $S = \mathbf{E}_1^\top \mathbf{E}_2$, where S represents the correlation between the two KGs. The *singular value decomposition* is then applied to the matrix S, resulting in $S = VDW^T$, where V and W are orthogonal

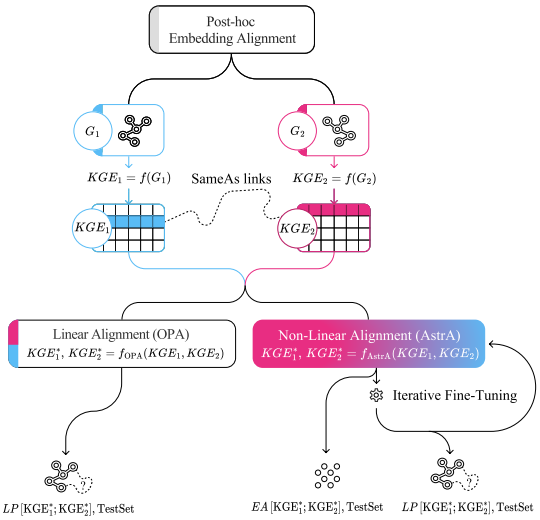

Fig. 1: Post-hoc embedding alignment approaches (OPA vs. AstrA).

matrices, and D is a diagonal matrix with singular values. $R = VW^T$ gives the optimal rotation matrix, where R is an orthogonal matrix that minimizes alignment errors. OPA learns an orthogonal linear transformation between embedding spaces, making it interpretable but limiting its ability to capture non-linear patterns. Despite its limitations, several embedding alignment techniques adopt OPA to learn a linear transformation that aligns embeddings across different vector spaces [7]. Moreover, OPA has been applied to align node embeddings in dynamic graphs, improving tasks such as link prediction and graph reconstruction [24]. However, OPA's potential as a post-hoc embedding alignment method for KGE remains underexplored.

4.2 Proposed Approach: ASTRA

To overcome the limitations of OPA, we propose AstrA (Figure 2), a nonlinear post-hoc alignment framework that operates on independently trained KGE models without requiring KG fusion or retraining. Unlike joint-training alignment approaches, AstrA aligns fixed pretrained embeddings by optimizing only the alignment components.

Problem Formulation. Given two KGs G_1 and G_2 with pretrained entity embeddings $\mathbf{E}_1$ and $\mathbf{E}_2$ Section 2, our goal is to learn a transformation $f : \mathbb{R}^d \to \mathbb{R}^d$ that maps both embedding spaces into a shared latent space. In this space, embeddings of aligned entities $(e_1, e_2) \in \mathcal{A}$ should be close while preserving the structural information encoded in the original embeddings. To achieve this, AstrA introduces three key components: (i) relational structural propagation to incorporate graph context, (ii) adaptive gated fusion to balance structural and

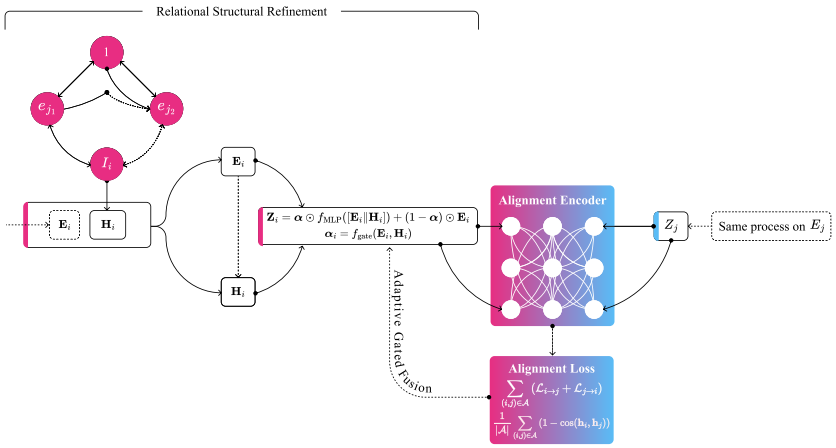

Fig. 2: Overview of the AstrA post-hoc alignment framework.

semantic information, and (iii) a shared non-linear projection that aligns the embedding spaces (Figure 2), where E_i and E_j denote the pretrained embeddings of entities from G_1 and G_2, respectively. We describe each component in more detail below.

(i) Relational Structural Propagation. Starting from the pretrained embeddings, we construct a merged relational graph that contains the entities and relations from both KGs. A *Relational Graph Convolutional Network* (R-GCN)[20], denoted as f_{RGCN}, is applied to this graph in order to propagate structural information between neighboring entities. The pretrained embeddings serve as input features, and the encoder produces structurally refined representations $\mathbf{H} = f_{\text{RGCN}}(\mathbf{E}, G)$, where $\mathbf{E}$ represents the merged pretrained embeddings and G denotes the relational graph. The R-GCN aggregates relation-specific messages from neighboring entities, allowing the model to incorporate relational context into the representations. Unlike typical R-GCN setups used for link prediction, the network here is not trained to reconstruct triples and does not learn embeddings from scratch. Instead, it acts as a structural encoder that enriches the pretrained embeddings. Formally, the representation of entity i at layer $l{+}1$ is updated as $h_i^{(l+1)} = \sigma\left(W_0^{(l)} h_i^{(l)} + \sum_{r\in\mathcal{R}} \sum_{j\in\mathcal{N}_i^r} \frac{1}{|\mathcal{N}_i^r|} W_r^{(l)} h_j^{(l)}\right)$, where $\mathcal{N}_i^r$ denotes the set of neighbors connected to i through relation r, $W_r^{(l)}$ are relation-specific transformation matrices, and σ is a non-linear activation function. By stacking multiple layers, the model is able to incorporate information from increasingly distant nodes in the graph. In our implementation, two R-GCN layers are used to derive the structural representations.

(ii) Adaptive Gated Fusion. The structural representations obtained from the R-GCN provides relational context, but relying on relational context alone may

distort the original semantic information encoded in the pretrained embeddings. To integrate these two sources of information, we introduce a learnable gating mechanism that combines the original entity embeddings with their structurally refined representations. Thus, the gating coefficient is computed individually for each entity and determines how much structural information to incorporate. Formally, the gating coefficients are computed as $\boldsymbol{\alpha} = \sigma\left(g_\phi([\mathbf{E}\|\mathbf{H}])\right)$, where $[\mathbf{E}\|\mathbf{H}]$ denotes the concatenation of the original embeddings and the structural representations, g_ϕ is a *multi-layer perceptron* (MLP) composed of fully connected layers and non-linear activations. σ is the sigmoid activation. The final fused representation is then obtained as $\mathbf{T} = f_{\text{MLP}}([\mathbf{E}\|\mathbf{H}]), \quad \mathbf{Z} = \boldsymbol{\alpha} \odot \mathbf{T} + (1 - \boldsymbol{\alpha}) \odot \mathbf{E}$. where $f_{\text{MLP}}(\cdot)$ denotes a two-layer MLP applied to the concatenation of the original embeddings and structural representations, and $\odot$ denotes element-wise multiplication. In this way, the model first learns a joint structural-semantic representation before adaptively combining it with the original embeddings.

(iii) Alignment Encoder. The fused representations are then mapped into a shared latent space so that embeddings from both KGs can be directly compared. This step is implemented using two-layer *feed-forward neural network* (FNN) with a *residual connection*. Each fused embedding $\mathbf{z}$ is first transformed through two fully connected layers with GELU activation [12]: $\mathbf{z}' = W_2 \cdot \text{GELU}(W_1\mathbf{z})$. The transformed representation is then combined with the original input through a residual interpolation: $\tilde{\mathbf{z}} = (1-\sigma(\beta))\mathbf{z} + \sigma(\beta)\mathbf{z}'$, where β is a learnable parameter and σ represents the sigmoid function. The learnable parameter β controls the influence of the transformed embedding on the final representation. During training, the projection network is optimized jointly with the fusion module using a *symmetric margin-based alignment loss* over seed entity pairs (see Eq. 1). The model is optimized iteratively using the alignment objective. During backpropagation, gradients update both the projection network and the fusion gate, adjusting the contribution of structural and pretrained embedding information to the final entity representations.

Symmetric Hard-Negative Margin Loss. Let $\tilde{\mathbf{z}}_i \in \mathbb{R}^d$ and $\tilde{\mathbf{z}}_j \in \mathbb{R}^d$ denote the projected embeddings of aligned entity pair $(i, j) \in \mathcal{A}$. All embeddings are ℓ_2-normalized. We compute cosine similarity as $cos(i, j) = \tilde{\mathbf{z}}_i^\top \tilde{\mathbf{z}}_j$. For each positive pair (i, j), we select the top-k hardest negatives based on similarity and compute their average similarity: $\overline{cos}(i) = \frac{1}{k} \sum_{j' \in \mathcal{N}_k(i)} cos(i, j')$, where $\mathcal{N}_k(i)$ denotes the set of k most similar non-matching entities to i. The margin ranking loss from source to target is defined as:

$$\mathcal{L}_{S \to T} = \frac{1}{|\mathcal{A}|} \sum_{(i,j) \in \mathcal{A}} \max\left(0, \overline{cos}(i) + \gamma - cos(i, j)\right), \tag{1}$$

where γ is the margin hyperparameter. To enforce bidirectional consistency, we define the symmetric loss as $\mathcal{L}_{align} = \mathcal{L}_{S \to T} + \mathcal{L}_{T \to S}$.

Algorithm 1: Structure-Aware Adaptive Alignment (AstrA)

Input : $\mathcal{A}$: *Set of aligned entity pairs*, $\mathbf{E}$: *Merged pretrained entity embeddings*, G: *Merged relational graph*, γ: *Margin parameter*
Model : RGCN encoder f_{RGCN}, Fusion layer with gate g_ϕ, Shared projection network f_θ
Output: $\mathbf{E}^*$: *Aligned entity embeddings*

for *each training epoch* **do**
  Compute structural representations:
  $\mathbf{H} \leftarrow f_{\text{RGCN}}(\mathbf{E}, G)$
  foreach *mini-batch* $\mathcal{B} \subset \mathcal{A}$ **do**
    Retrieve batch embeddings:
    $\mathbf{E}_\mathcal{B} \leftarrow \mathbf{E}[\mathcal{B}]$
    $\mathbf{H}_\mathcal{B} \leftarrow \mathbf{H}[\mathcal{B}]$
    Adaptive gated fusion:
    $\boldsymbol{\alpha} \leftarrow \sigma(g_\phi([\mathbf{E}_\mathcal{B} \| \mathbf{H}_\mathcal{B}]))$
    $\mathbf{T}_\mathcal{B} \leftarrow f_{\text{MLP}}([\mathbf{E}_\mathcal{B} \| \mathbf{H}_\mathcal{B}])$
    $\mathbf{Z}_\mathcal{B} \leftarrow \boldsymbol{\alpha} \odot \mathbf{T}_\mathcal{B} + (1 - \boldsymbol{\alpha}) \odot \mathbf{E}_\mathcal{B}$
    Project fused embeddings into shared alignment space:
    $\tilde{\mathbf{Z}}_\mathcal{B} \leftarrow f_\theta(\mathbf{Z}_\mathcal{B})$
    Compute symmetric hard-negative alignment loss:
    $\mathcal{L}_{align} \leftarrow \mathcal{L}_{S \to T} + \mathcal{L}_{T \to S}$
    $\mathcal{L} \leftarrow \mathcal{L}_{align}$
    Update model parameters via backpropagation
Compute final aligned embeddings:
$\mathbf{E}^* \leftarrow f_\theta(\mathbf{Z})$ applied to all entities
return $\mathbf{E}^*$

5 Experiments

In this study, we evaluate our proposed approach, AstrA, on two key tasks: Entity alignment and link prediction. The evaluation follows the standard OpenEA evaluation protocol[1], and we report the metric MRR and Hits@K. We intentionally do not compare AstrA with joint-training entity alignment approaches (e.g., JAPE, MTransE, KDCoE). These methods integrate alignment supervision directly into the embedding learning process and require retraining of the underlying KGE models. In contrast, AstrA assumes independently trained embeddings and performs alignment strictly as a post-hoc adaptation step. Therefore, we focus on baseline that operates under the same post-hoc alignment assumption, enabling a fair comparison of alignment strategies applied to fixed pretrained embeddings. To systematically evaluate these aspects, we formulate the following research questions:

[1] OpenEA repository: `https://github.com/nju-websoft/OpenEA`

Table 1: Statistics of the DBpedia EN-FR and EN-DE datasets. The final three columns indicate the number of owl:sameAs links used for training, validation, and testing.

Dataset	KG	$\|E\|$	$\|R\|$	$\|G\|$	Train	Val.	Test
EN–FR 15K	EN	15,000	267	39,342	3,000	1,500	10,500
	FR	15,000	210	40,864			
EN–FR 100K	EN	100,000	400	278,646	20,000	10,000	70,000
	FR	100,000	300	258,285			
EN–DE 15K	EN	15,000	215	47,676	3,000	1,500	10,500
	DE	15,000	131	50,419			
EN–DE 100K	EN	100,000	381	335,359	20,000	10,000	10,500
	DE	100,000	196	336,240			

Q_1. Does AstrA achieve stronger entity alignment performance than a linear baseline, and remain effective across embeddings produced by different KGE models?
Q_2. Does AstrA achieve accurate link prediction while scaling to larger KGs and generalizing across cross-lingual settings and beyond?
Q_3. How important are the structural components of AstrA, including the R-GCN encoder and the adaptive fusion gate, for alignment and link prediction performance?

Datasets. We employ two benchmark datasets for evaluation: (1) The first set includes four cross-lingual datasets EN-FR-15K, EN-FR-100K, EN-DE-15K, and EN-DE-100K, which are widely used for cross-lingual entity alignment. These datasets contain $15,000$ to $100,000$ aligned entities between the English, French, and German editions of DBpedia. They are created explicitly for entity alignment tasks and are available from the OpenEA repository[1]. Table 1 shows the statistics of these datasets. We employ the same training, validating, and testing splits as in past studies. (2) The second set contains the DBpedia-Wikidata dataset which originates from the *Universal KGEs* paper [15]. To construct the dataset, 1% of DBpedia entities with owl:sameAs links to Wikidata are randomly selected. For each selected entity, its 1-hop neighborhood and associated relation types are extracted. Corresponding subsets of the dataset are then generated based on these entities. Table 2 summarizes the dataset statistics. Standard link prediction benchmarks (e.g., FB15k-237 or YAGO15k) are not used because they lack owl:sameAs links between distinct KGs required by our alignment framework.

Experiment Setup. The R-GCN encoder, fusion module, and alignment encoder are trained jointly using the Adam optimizer with a learning rate of 10^{-3}. Entity and relation embeddings have dimensionality $d = 256$. Training is performed using mini-batches of aligned entity pairs, and hard negative mining is applied within each batch. The alignment model is trained for 60 epochs. For

Table 2: Statistics of the `DBpedia-Wikidata` dataset. $\overline{D}$: Average entity degree.

Dataset	$\lvert E\rvert$	$\lvert R\rvert$	$\lvert G\rvert$	`owl:sameAs`	$\overline{D}$
DBpedia$_{train}$	31,116	392	69,667	22,102	4.48
DBpedia$_{test}$	15,602	279	15,374	10,471	1.97
Wikidata$_{train}$	72,058	707	235,814	22,102	6.55
Wikidata$_{test}$	41,137	465	53,761	10,471	2.61
MERGE$_{train}$	81,836	1,099	305,481	22,102	7.47

link prediction evaluation, aligned entity embeddings are injected into the original KGE model together with the original relation embeddings and fine-tuned for 20 epochs. Model selection for both alignment and link prediction is based on the *Mean Reciprocal Rank* (MRR) computed on the validation set.

Entity Alignment Evaluation. Following the OpenEA protocol[1], the goal is to identify equivalent entities across KGs via nearest neighbor search in the embedding space. Formally, $\mathcal{A} \subseteq \mathcal{E}_1 \times \mathcal{E}_2$ denotes the set of aligned entity pairs, and $\mathbf{E}_1^{\mathcal{A}}, \mathbf{E}_2^{\mathcal{A}} \subset \mathbb{R}^d$ are the corresponding embeddings of aligned entities from $\mathcal{E}_1$ and $\mathcal{E}_2$, respectively. Given a source entity embedding $\mathbf{e}_1 \in \mathbf{E}_1^{\mathcal{A}}$, all candidate embeddings in $\mathbf{E}_2^{\mathcal{A}}$ are ranked based on similarity (e.g., cosine similarity). We report the MRR and Hits@K as: $\text{MRR}_{\text{align}} = \frac{1}{|\mathcal{A}|} \sum_{i=1}^{|\mathcal{A}|} \frac{1}{\text{rank}(\mathbf{e}_2^{(i)} | \mathbf{e}_1^{(i)})}$. and $\text{Hits@}K_{\text{align}} = \frac{1}{|\mathcal{A}|} \sum_{i=1}^{|\mathcal{A}|} \mathbb{1}\left[\text{rank}(\mathbf{e}_2^{(i)} \mid \mathbf{e}_1^{(i)}) \leq K\right]$, where $\mathbb{1}$ is the indicator function which returns 1 if its input evaluates to true and 0 otherwise.

Link Prediction Evaluation. Aligned entity embeddings obtained via AstrA are injected into the original KGE model with original relation embeddings. The model is then fine-tuned using its specific `k_vs_all_score()`[2] function, as defined for that model architecture. Fine-tuning is performed iteratively, where in each cycle a randomly sampled subset of 10% of the training triples is used to support generalization. Thus, aligned embeddings are refined to maintain their structural alignment while enhancing task-specific performance. To assess whether the aligned embeddings preserve the original graph structures, we perform link prediction on a set of test triples $\mathcal{G}_{test} \subseteq \mathcal{E} \times \mathcal{R} \times \mathcal{E}$. A KGE model is initialized with aligned entity and relation embeddings $\mathbf{E}^*$ and $\mathbf{R}^*$, and evaluated on the test triples. We report MRR and Hits@K based on head and tail entity predictions:

$$\text{MRR}_{\text{lp}} = \frac{1}{2|\mathcal{G}_{test}|} \sum_{(h,r,t) \in \mathcal{G}_{test}} \left(\frac{1}{\text{rank}(t \mid h, r)} + \frac{1}{\text{rank}(h \mid r, t)} \right),$$

$$\text{Hits@}K_{\text{lp}} = \frac{1}{2|\mathcal{G}_{test}|} \sum_{(h,r,t) \in \mathcal{G}_{test}} \Big(\mathbb{1}\left[\text{rank}(t \,|\, h, r) \leq K\right] + \mathbb{1}\left[\text{rank}(h \,|\, r, t) \leq K\right] \Big).$$

[2] k-vs-all scoring ranks all entities in the KG as candidates for the missing entity in a triple.

6 Results and Discussion

In this section, we present and analyze the results of our experiments, structured around the research questions introduced in Section 5. We report entity alignment and link prediction performance, comparing AstrA with baseline method across various datasets.

Q_1: AstrA entity alignment quality vs. baseline alignment method. To assess whether AstrA offers a practical alternative to the linear baseline (OPA) and scales across different embedding architectures, we compare AstrA with OPA using embeddings from multiple KGE models. Table 3 presents entity alignment results across two cross-lingual datasets. To ensure a fair comparison, we use the `DBpedia EN-FR` and `EN-DE` 15K benchmark datasets provided by [23], and the setup described in Section 5. We evaluate alignment performance using 5-fold cross-validation, and report the average performance with standard deviation across the five runs. Evaluation metrics include *Hits@K* (for $K = 1, 5, 10, 50$) and MRR. In addition to the full AstrA model and the OPA baseline, the table also includes the variant AstrA\RGCN, which removes the structural encoder and is used as an ablation of the structural component. The impact of this variant is analyzed in detail in Q_3. As shown in Table 3, AstrA clearly outperforms the linear OPA baseline across various embedding models, improving H@1 by approximately 0.10-0.14 absolute points and increasing MRR by over five times on average. This pattern is consistent across the different embedding models, including TransE, ComplEx, DualE, and Ckeci. This demonstrates that AstrA's alignment behavior remains stable across different embedding architectures, indicating that the proposed alignment transformation generalizes well across different embedding geometries. Slight variations in alignment performance across models and datasets suggest that the characteristics of the underlying embeddings influence the resulting alignment quality. This addresses Q_1 by showing that AstrA provides a post-hoc alignment alternative when KGE retraining is infeasible, outperforming a linear baseline across embeddings from different KGE models.

Q_2: AstrA link prediction performance under increasing KG scale. To address Q_2, we conduct experiments to evaluate whether AstrA can generate aligned embeddings that enable accurate link prediction while scaling to larger KGs and across different alignment settings. We benchmark AstrA against the linear alignment baseline OPA (see Section 4.1) to assess its effectiveness in preserving structural semantics after alignment. Table 4 reports the average results of the 10-fold link prediction experiments on the `DBpedia EN-FR` and `EN-DE` datasets with two sizes (15K and 100K), as well as results on the `DBpedia-Wikidata` dataset. In addition to the OPA baseline and the full AstrA model, we also report results for the variant AstrA\RGCN, which excludes the structural encoder. This variant is included to support the ablation analysis discussed in Q_3. For the `DBpedia-Wikidata` dataset, we report results using the predefined train/test split introduced in [15]; therefore, standard deviations are not

Table 3: Average 5-Fold Entity Alignment Results on `DBpedia EN-FR` and `EN-DE` 15K using different embedding models and post-hoc alignment methods.

Dataset	Model	Method	H@1	H@5	H@10	H@50	MRR
EN–FR	TransE	OPA	0.003 ±0.000	0.019 ±0.008	0.039 ±0.014	0.190 ±0.024	0.033 ±0.002
		AstrA\RGCN	0.027 ±0.002	0.099 ±0.003	0.186 ±0.003	0.561 ±0.004	0.094 ±0.002
		AstrA	**0.103 ±0.005**	**0.244 ±0.009**	**0.323 ±0.011**	**0.543 ±0.016**	**0.183 ±0.012**
	ComplEx	OPA	0.002 ±0.001	0.009 ±0.003	0.021 ±0.004	0.162 ±0.012	0.019 ±0.002
		AstrA\RGCN	0.014 ±0.002	0.062 ±0.004	0.121 ±0.005	0.432 ±0.006	0.061 ±0.003
		AstrA	**0.099 ±0.004**	**0.246 ±0.007**	**0.331 ±0.009**	**0.577 ±0.014**	**0.182 ±0.011**
	DualE	OPA	0.003 ±0.001	0.011 ±0.003	0.027 ±0.004	0.171 ±0.013	0.022 ±0.002
		AstrA\RGCN	0.018 ±0.002	0.074 ±0.005	0.137 ±0.006	0.455 ±0.007	0.068 ±0.003
		AstrA	**0.110 ±0.005**	**0.265 ±0.007**	**0.357 ±0.010**	**0.577 ±0.013**	**0.197 ±0.011**
	Ckeci	OPA	0.009 ±0.002	0.031 ±0.005	0.067 ±0.007	0.298 ±0.012	0.051 ±0.003
		AstrA\RGCN	0.073 ±0.003	0.241 ±0.006	0.349 ±0.007	0.684 ±0.008	0.168 ±0.004
		AstrA	**0.147 ±0.006**	**0.311 ±0.008**	**0.402 ±0.011**	**0.604 ±0.016**	**0.238 ±0.013**
EN–DE	TransE	OPA	0.006 ±0.004	0.018 ±0.008	0.042 ±0.008	0.246 ±0.015	0.036 ±0.004
		AstrA\RGCN	0.058 ±0.001	0.214 ±0.005	0.337 ±0.004	0.678 ±0.004	0.150 ±0.002
		AstrA	**0.135 ±0.004**	**0.283 ±0.007**	**0.361 ±0.008**	**0.524 ±0.013**	**0.215 ±0.011**
	ComplEx	OPA	0.003 ±0.001	0.012 ±0.004	0.028 ±0.005	0.183 ±0.013	0.024 ±0.002
		AstrA\RGCN	0.032 ±0.002	0.128 ±0.005	0.214 ±0.006	0.544 ±0.007	0.098 ±0.003
		AstrA	**0.121 ±0.006**	**0.269 ±0.010**	**0.352 ±0.012**	**0.561 ±0.018**	**0.203 ±0.009**
	DualE	OPA	0.004 ±0.001	0.014 ±0.003	0.031 ±0.005	0.198 ±0.012	0.027 ±0.002
		AstrA\RGCN	0.041 ±0.002	0.152 ±0.005	0.245 ±0.006	0.593 ±0.007	0.114 ±0.003
		AstrA	**0.129 ±0.005**	**0.279 ±0.011**	**0.368 ±0.013**	**0.571 ±0.017**	**0.208 ±0.010**
	Ckeci	OPA	0.007 ±0.002	0.028 ±0.004	0.059 ±0.006	0.276 ±0.013	0.047 ±0.003
		AstrA\RGCN	0.065 ±0.003	0.221 ±0.005	0.336 ±0.006	0.662 ±0.008	0.156 ±0.004
		AstrA	**0.134 ±0.005**	**0.297 ±0.008**	**0.389 ±0.010**	**0.582 ±0.015**	**0.222 ±0.012**

reported. Across all datasets and evaluation metrics, AstrA consistently outperforms OPA by a large margin. On the `EN-FR` 15K dataset, AstrA achieves an MRR of 0.326, compared to 0.011 for OPA, while on the `EN-DE` 15K dataset it reaches an MRR of 0.347, compared to 0.014 for OPA. Although link prediction performance decreases when the graph size increases from 15K to 100K entities, AstrA maintains a clear advantage over the linear baseline. On the `EN-FR` 100K dataset, AstrA achieves an MRR of 0.152 compared to 0.004 for OPA, demonstrating robustness under increasing KG scale. To further evaluate generalization beyond cross-lingual alignment scenarios, we also report results on the cross-domain `DBpedia-Wikidata` dataset. Despite the structural differences between these two KGs, AstrA achieves an MRR of 0.243, substantially outperforming the OPA baseline (0.006). Overall, these results demonstrate that AstrA achieves accurate link prediction while scaling to large KGs and generalizing not only across cross-lingual settings but also to general-purpose datasets, thereby addressing our research question Q_2.

Q_3: Contribution of structural components in AstrA. To evaluate the contribution of the structural components within AstrA, we conduct an ablation study by removing the R-GCN encoder (AstrA\RGCN). This variant performs alignment directly on the pretrained embeddings without structural refinement. As shown in Tables 3 and 4, removing the R-GCN consistently reduces performance

Table 4: Link prediction results on the `DBpedia EN-FR`, `EN-DE` (15K and 100K) and `DBpedia-Wikidata` datasets. Results on EN–FR and EN–DE are reported as the average of 10-fold evaluation, while DBpedia–Wikidata uses the predefined split from [15].

Dataset	Size	Method	H@1	H@3	H@10	MRR
EN–FR	15K	OPA	0.007 ± 0.002	0.013 ± 0.003	0.018 ± 0.004	0.011 ± 0.003
		AstrA\RGCN	0.137 ± 0.032	0.256 ± 0.033	0.423 ± 0.041	0.243 ± 0.034
		AstrA	**0.189 ± 0.048**	**0.390 ± 0.072**	**0.610 ± 0.094**	**0.326 ± 0.061**
	100K	OPA	0.002 ± 0.001	0.004 ± 0.001	0.007 ± 0.002	0.004 ± 0.001
		AstrA\RGCN	0.062 ± 0.017	0.136 ± 0.029	0.223 ± 0.037	0.117 ± 0.024
		AstrA	**0.083 ± 0.011**	**0.148 ± 0.016**	**0.236 ± 0.021**	**0.152 ± 0.014**
EN–DE	15K	OPA	0.009 ± 0.003	0.017 ± 0.004	0.025 ± 0.005	0.014 ± 0.004
		AstrA\RGCN	0.136 ± 0.024	0.297 ± 0.028	0.387 ± 0.032	0.236 ± 0.028
		AstrA	**0.203 ± 0.045**	**0.421 ± 0.073**	**0.638 ± 0.088**	**0.347 ± 0.059**
	100K	OPA	0.007 ± 0.002	0.013 ± 0.003	0.018 ± 0.004	0.011 ± 0.003
		AstrA\RGCN	0.070 ± 0.018	0.150 ± 0.030	0.240 ± 0.035	0.125 ± 0.026
		AstrA	**0.091 ± 0.013**	**0.159 ± 0.017**	**0.248 ± 0.021**	**0.161 ± 0.015**
DBpedia–Wikidata	82K	OPA	0.004	0.008	0.014	0.006
		AstrA\RGCN	0.172	0.302	0.431	0.263
		AstrA	**0.195**	**0.266**	**0.327**	**0.243**

across both entity alignment and link prediction tasks. Across the evaluated embedding models, the AstrA\RGCN variant achieves lower alignment accuracy than the full AstrA model, indicating that structural propagation improves cross-KG alignment. A similar trend is observed for link prediction, where the ablated variant generally underperforms the full model across different datasets and graph scales. These results suggest that incorporating structural information helps preserve relational semantics after alignment.

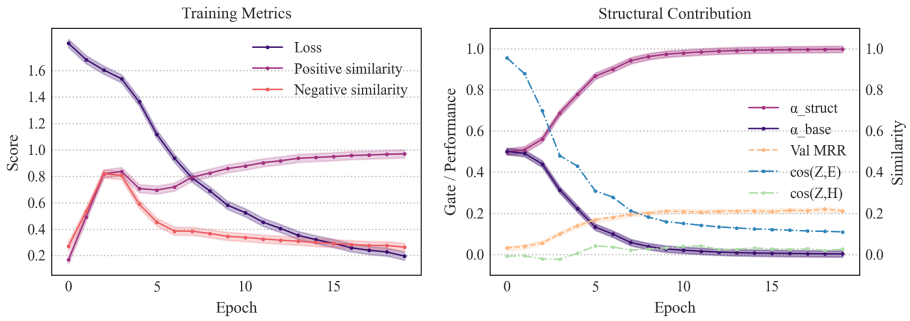

Fig. 3: Training dynamics of the adaptive fusion mechanism in AstrA. The figure shows the evolution of the structural gate, base weight, cosine similarities, and validation MRR during training.

We further analyze the adaptive gating coefficient α, which balances the original embeddings and the structurally refined representations. Figure 3 demon-

strates that during training, the structural weight α_{struct} increases while the base weight α_{base} (the original embedding weights) decreases. At the same time, cosine similarity between the fused representation Z and the structural embeddings increases, whereas similarity to the original embeddings decreases. This trend indicates that the model relies more on structural signals, which correlates with the observed improvement in validation MRR and highlights the importance of structural propagation in AstrA, thereby addressing Q_3.

7 Conclusion

We presented AstrA, a structure-aware post-hoc alignment framework for independently trained KGEs that avoids KG fusion and embedding retraining. Experiments across multiple datasets show that AstrA consistently outperforms linear alignment baselines and produces embeddings suitable for both entity alignment and link prediction. In the future, we will focus on further improving the performance of AstrA when using more expressive embedding models. While AstrA already incorporates architecture-aware alignment by leveraging each models own scoring function, more specialized strategies may further enhance performance for complex-valued or dual/quaternion-based embeddings.

Reproducibility. Embeddings were generated using the DICE Embedding Framework[3]. The source code is available from an anonymous GitHub repository[4]. We use the publicly available `DBpedia EN-FR` and `EN-DE` datasets from the OpenEA benchmark[5] (`EN-FR-15K-V1`, `EN-FR-100K-V1`, `EN-DE-15K-V1`, `EN-DE-100K-V1`). The `DBpedia-Wikidata` dataset is available on Zenodo[6].

References

1. Bordes, A., Usunier, N., García-Durán, A., Weston, J., Yakhnenko, O.: Translating embeddings for modeling multi-relational data. Advances in Neural Information Processing Systems **26**, 2787–2795 (2013)
2. Cao, Y., Wang, Z., Li, J., Liu, Z.: Multi-channel graph neural network for entity alignment. In: Annual Meeting of the Association for Computational Linguistics (ACL) (2019)
3. Cao, Z., Xu, Q., Yang, Z., Cao, X., Huang, Q.: Dual quaternion knowledge graph embeddings. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 35, pp. 3761–3769. AAAI Press (2021), available at: https://ojs.aaai.org/index.php/AAAI/article/view/16850
4. Chen, M., Sun, Z., Zhou, B., Sun, Y., Wang, W., Zaniolo, C.: Multilingual knowledge graph completion with self-supervised adaptive graph alignment. In: Proceedings of the Web Conference (WWW) (2020)

[3] `https://github.com/dice-group/dice-embeddings`
[4] `https://github.com/anon-kgalignment/ASTRA`
[5] `https://www.dropbox.com/scl/fi/lo69wjm1f37qiik59kmg8/OpenEA_dataset_v1.1.zip?rlkey=tr2nv5qykkdwzci43wmx43a7b&e=1&dl=0`
[6] `https://zenodo.org/records/7566020`

5. Chen, M., Tian, Y., Yang, M., Zaniolo, C.: Multilingual knowledge graph embeddings for cross-lingual knowledge alignment. In: Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI). pp. 1511–1517 (2017)
6. Chen, M., Tian, Y., Yang, M., Zaniolo, C.: Co-training embeddings of knowledge graphs and entity descriptions for cross-lingual entity alignment. In: Proceedings of the 27th International Joint Conference on Artificial Intelligence (IJCAI) (2018)
7. Chen, X., Heimann, M., Vahedian, F., Koutra, D.: CONEAlign: Consistent Network Alignment with Proximity-Preserving Node Embedding. In: Proceedings of the 29th ACM International Conference on Information & Knowledge Management (CIKM '20). pp. 1985–1988. Association for Computing Machinery, New York, NY, USA (2020). https://doi.org/10.1145/3340531.3412136
8. Crosilla, F., Beinat, A., Fusiello, A., Maset, E., Visintini, D.: Orthogonal Procrustes Analysis, pp. 7–28. Springer International Publishing, Cham (2019). https://doi.org/10.1007/978-3-030-11760-3_2
9. Dai, Y., Approaches, a., benchmarks on Knowledge Graph, S., Xiong, N.N., Guo, W.: Approaches, applications and benchmarks. on knowledge graph embedding: Approaches, applications and benchmarks. Electronics **9**(5), 750 (2020). https://doi.org/10.3390/electronics9050750
10. Fanourakis, N., Efthymiou, V., Kotzinos, D., Christophides, V.: Knowledge graph embedding methods for entity alignment: An experimental review. Data Mining and Knowledge Discovery (2023). https://doi.org/10.48550/arXiv.2203.09280
11. Grave, E., Joulin, A., Berthet, Q., et al.: Unsupervised alignment of embeddings with wasserstein procrustes. arXiv preprint arXiv:1805.11222 (2018)
12. Hendrycks, D., Gimpel, K.: Gaussian error linear units (gelus). arXiv preprint arXiv:1606.08415 (2016), https://doi.org/10.48550/arXiv.1606.08415
13. Hogan, A., Blomqvist, E., Cochez, M., d'Amato, C., de Melo, G., Gutierrez, C., Kirrane, S., Neumaier, S., Polleres, A., Sequeda, J.F., et al.: Knowledge graphs. ACM Computing Surveys (CSUR) **54**(4), 1–37 (2021)
14. Jain, N., Kalo, J.C., Balke, W.T., Krestel, R.: Do embeddings actually capture knowledge graph semantics? In: Proceedings of the 18th Extended Semantic Web Conference (ESWC). Lecture Notes in Computer Science, vol. 12731. Springer (2021)
15. Kouagou, N.J., Demir, C., Heindorf, S., Zahera, H.M., Li, J., Wilke, A., Ngomo, A.C.N.: Universal knowledge graph embeddings: Merging multiple kgs for comprehensive entity representations. In: Proceedings of the Web Conference 2024 (WWW '24). ACM, Singapore (2024)
16. Ngonga Ngomo, A.C., Sherif, M.A., Georgala, K., Hassan, M., DreSSler, K., Lyko, K., Obraczka, D., Soru, T.: LIMES - A Framework for Link Discovery on the Semantic Web. KI-Künstliche Intelligenz, German Journal of Artificial Intelligence - Organ des Fachbereichs "Künstliche Intelligenz" der Gesellschaft für Informatik e.V. (2021), https://papers.dice-research.org/2021/KI_LIMES/public.pdf
17. Peng, X., Chen, G., Lin, C., Stevenson, M.: Highly efficient knowledge graph embedding learning with orthogonal procrustes analysis. arXiv preprint arXiv:2104.04676 (2021)
18. Quan, Mao, Z., Wang, B., Guo, L.: Knowledge graph embedding: A survey of approaches and applications. IEEE Transactions on Knowledge and Data Engineering **29**(12), 2724–2743 (2017). https://doi.org/10.3390/electronics9050750
19. Schenemann, P.H.: A generalized solution of the orthogonal procrustes problem. Psychometrika **31**(1), 1–22 (1966). https://doi.org/10.1007/BF02289451

20. Schlichtkrull, M., Kipf, T.N., Bloem, P., van den Berg, R., Titov, I., Welling, M.: Modeling relational data with graph convolutional networks. arXiv preprint arXiv:1703.06103 (2017). https://doi.org/10.48550/arXiv.1703.06103, https://arxiv.org/abs/1703.06103
21. Sun, Z., Hu, W., Li, C.: Cross-lingual entity alignment via joint attribute-preserving embedding. In: The Semantic Web ISWC 2017. Lecture Notes in Computer Science, vol. 10587, pp. 628–644. Springer (2017)
22. Sun, Z., Hu, W., Zhang, Q., Qu, Y.: Bootstrapping entity alignment with knowledge graph embedding. In: International Joint Conference on Artificial Intelligence (IJCAI) (2018)
23. Sun, Z., Zhang, Q., Hu, W., Wang, C., Chen, M., Akrami, F., Li, C.: A benchmarking study of embedding-based entity alignment for knowledge graphs. Proceedings of the VLDB Endowment **13**(11), 2326–2340 (2020). https://doi.org/10.14778/3407790.3407828
24. Tagowski, K., Bielak, P., Kajdanowicz, T.: Embedding alignment methods in dynamic networks. In: Proceedings of ICCS 2021. Springer, Wroclaw University of Science and Technology, Poland (2021)
25. Teyou, L.M.K., Demir, C., Ngomo, A.C.N.: Embedding knowledge graphs in degenerate clifford algebras. In: Proceedings of the 27th European Conference on Artificial Intelligence (ECAI). EurAI (2024), available at: https://papers.dice-research.org/2024/ECAI_DeCaL/public.pdf
26. Trisedya, B.D., Qi, J., Zhang, R., Schockaert, S.: Entity alignment between knowledge graphs using attribute embeddings. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 33, pp. 297–304 (2019)
27. Trouillon, T., Welbl, J., Riedel, S., Gaussier, E., Bouchard, G.: Complex embeddings for simple link prediction. In: International Conference on Machine Learning (ICML). pp. 2071–2080 (2016)
28. Wang, Z., Lv, Q., Lan, X., Zhang, Y.: Cross-lingual knowledge graph alignment via graph convolutional networks. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP). pp. 349–357. Association for Computational Linguistics (2018)
29. Wu, Y., Liu, X., Feng, Y., Wang, Z., Yan, R., Zhao, D.: Relation-aware entity alignment for heterogeneous knowledge graphs. In: Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence (IJCAI) (2019)
30. Xin, K., Sun, Z., Hua, W., Hu, W., Zhou, X.: Informed multi-context entity alignment. In: Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining (WSDM) (2022). https://doi.org/10.1145/3488560.3498523
31. Yang, B., Yih, W.t., He, X., Gao, J., Deng, L.: Embedding entities and relations for learning and inference in knowledge bases. arXiv preprint arXiv:1412.6575 (2014)
32. Yang, L., Cheng, J., Xu, C., Wang, X., Li, J., Zhang, F.: Attr-int: A simple and effective entity alignment framework for heterogeneous knowledge graphs. arXiv preprint arXiv:2410.13409 (2024)
33. Zhu, H., Xie, R., Liu, Z., Sun, M.: Iterative entity alignment via joint knowledge embeddings. In: IJCAI. pp. 4258–4264. ijcai.org (2017)