7-2021

Make it easy: An effective end-to-end entity alignment framework

Congcong GE

Xiaoze LIU

Lu Chen CHEN

Baihua ZHENG
Singapore Management University, bhzheng@smu.edu.sg

Yunjun GAO

Make It Easy: An Effective End-to-End Entity Alignment Framework

Congcong Ge[†], Xiaoze Liu[†], Lu Chen[†], Baihua Zheng[§], and Yunjun Gao[†♯]
[†]College of Computer Science, Zhejiang University, Hangzhou, China
[§]School of Computing and Information Systems, Singapore Management University, Singapore
[♯]Alibaba–Zhejiang University Joint Institute of Frontier Technologies, Hangzhou, China
{[1]gcc, [2]xiaoze, [3]luchen, [5]gaoyj}@zju.edu.cn [4]bhzheng@smu.edu.sg

ABSTRACT

Entity alignment (EA) is a prerequisite for enlarging the coverage of a unified knowledge graph. Previous EA approaches either restrain the performance due to inadequate information utilization or need labor-intensive pre-processing to get external or reliable information to perform the EA task. This paper proposes EASY, an effective end-to-end EA framework, which is able to (i) remove the labor-intensive pre-processing by fully discovering the name information provided by the entities themselves; and (ii) jointly fuse the features captured by the names of entities and the structural information of the graph to improve the EA results. Specifically, EASY first introduces NEAP, a highly effective name-based entity alignment procedure, to obtain an initial alignment that has reasonable accuracy and meanwhile does not require much memory consumption or any complex training process. Then, EASY invokes SRS, a novel structure-based refinement strategy, to iteratively correct the misaligned entities generated by NEAP to further enhance the entity alignment. Extensive experiments demonstrate the superiority of our proposed EASY with significant improvement against 13 existing state-of-the-art competitors.

CCS CONCEPTS

• **Computing methodologies** → **Knowledge representation and reasoning**; *Semantic networks.*

KEYWORDS

Entity alignment; Entity name; Graph structure; Iterative training

ACM Reference Format:
Congcong Ge[†], Xiaoze Liu[†], Lu Chen[†], Baihua Zheng[§], and Yunjun Gao[†♯]. 2021. Make It Easy: An Effective End-to-End Entity Alignment Framework. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '21), July 11–15, 2021, Virtual Event, Canada.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3404835.3462870

SIGIR '21, July 11–15, 2021, Virtual Event, Canada.

ACM ISBN 978-1-4503-8037-9/21/07…$15.00
https://doi.org/10.1145/3404835.3462870

1 INTRODUCTION

Knowledge graph (KG) is a widely used human-knowledge representation, which consists of entities and their rich relationships. Increasingly, companies and institutions turn to construct knowledge graphs (KGs) to facilitate various knowledge-driven applications, such as question answering [4, 39] and recommendation systems [53]. Despite the large scale of existing mainstream knowledge graphs (e.g., Yago [22], Freebase [3], and DBpedia [1]), they are still highly incomplete and thus restrict the quality of knowledge-driven applications. Integrating knowledge graphs from different sources provides an effective way to expand and enrich knowledge graphs. Entity alignment (EA) [8], which aims to identify entities residing at different KGs that actually refer to the same real-world objects, is a prerequisite for knowledge graph integration.

Recently, embedding has become an increasingly powerful tool to encode entities into low-dimensional vectors that are able to retain their semantic information. To this end, various embedding-based EA methods have emerged. Those approaches typically consist of three steps: (i) specifying pre-aligned entities as *seed alignment*; (ii) training an EA model guided by the seed alignment; and (iii) aligning the remaining entities based on the well-trained EA model. The majority of embedding-based EA approaches *purely* rely on graph structure to align equivalent entities [8, 32, 34, 36]. The expressive power of the graph structure is closely relevant to the graph isomorphism [10]. Nevertheless, real-life KGs are always heterogeneous, which significantly limits the ability of the graph structure in the task of entity alignment [52]. To further improve the quality of entity alignment, other studies exploit auxiliary information, such as attributes [21, 40, 43], entity names [10, 49, 52], descriptions [7, 38, 49, 54] and images [20].

Although the auxiliary information mentioned above has improved the performance of EA, there is still a big room for improvement because of the following two challenges. (i) **Labor-intensive pre-processing.** The pre-aligned seed annotations are labor-intensive and/or require expert involvement. Besides, some auxiliary information needs extra efforts to be applied in the EA task, e.g., extracting each entity's images or descriptions incurs additional overhead. In addition, certain auxiliary information might not be available at every entity (e.g., some entities do not have image information), which further limits its usability. Last but not the least, recent work [50] has also reported the challenge of using attribute information due to a large percent of noise. (ii) **Insufficient name information discovery.** Contrary to attributes, descriptions, and image information, entity name is an important type of auxiliary

information that does not require extra efforts to collect. Since every entity has its own name, we would like to highlight that it is easier and more effective to utilize entity names to facilitate EA, as compared with other types of auxiliary information. Nonetheless, existing methods [10, 21, 51, 55] lack in-depth exploration of the rich semantic information captured by the entity names, and hence, they restrict the accuracy of EA, as described in Example 1.1.

Example 1.1. Existing methods tend to utilize *max-pooling* or *average-pooling* to generate embeddings via pre-trained language models (e.g., fastText [2] and BERT [9]) to represent entities. Concretely, each entity name can be treated as a series of tokens. For each entity, max-pooling first assigns each token an embedding with fixed-dimension. It then selects the maximal value in each dimension among all the relevant embedded tokens to form a new embedding representing the entity. We argue that this may destroy the semantic information since most values inferior to the maximum are still useful but they are simply ignored in this strategy. Similarly, average-pooling uses the average embedding of all the tokens to represent the entity. However, it is well-known that certain tokens in an entity are more important than others. Take an entity e_1 called "*Blues de Saint Louis*" in a French KG (KG_{FR}) as an example. It can be split into the following four tokens, i.e., *Blues*, *de*, *Saint*, and *Louis*. Here, the token "*de*" (a common preposition in French) is less important and thus is negligible in e_1.

To address these two challenges, we propose an end-to-end entity alignment framework, termed as EASY. It consists of two components: (i) a name-based entity alignment procedure called NEAP; and (ii) a structure-based refinement strategy called SRS. First, NEAP is presented to obtain the initial entity alignment based on the semantic information captured by the names of entities. It first characterizes entities by leveraging the *local features* of tokens and then generates an initial EA according to the *global features* between the tokenized entities from two different KGs. Second, SRS aims to refine the alignment by correcting the misaligned entities caused by NEAP in an iterative manner. In each iteration, SRS first uses a *confident seeds generator* based on a newly presented concept of *graph matching discrepancy* to automatically generate seed alignment to guide the entity representation leaning via a structure-based EA model. The learned entity representations can be used to compute the structural similarities between entities from different KGs. Then, SRS adjusts the misaligned entities by considering both the semantic similarity and structural similarity between entities. We summarize the key contributions of this paper as follows:

- *Flexible EA framework.* We propose a novel end-to-end entity alignment framework, i.e., EASY, which requires *zero* labor-intensive pre-processing. To the best of our knowledge, it is the first end-to-end framework that can be easily integrated with any structure-based EA model.
- *Lightweight EA Initialization.* We present NEAP to align entities by discovering both local and global features of entity names. NEAP greatly reduces memory consumption without sacrificing the expressive power of entities. It can achieve desirable EA results without any complex training process.
- *Confident EA Refinement.* We propose SRS, a novel structure-based refinement strategy, which learns the structural features of entities by a *confident seeds generator* guided EA model and absorbs the name features of entities captured by NEAP. The fuse of both structural and name information can improve the entity alignment produced by NEAP.
- *Extensive experiments.* We conduct comprehensive experimental evaluation on cross-lingual EA tasks against state-of-the-art approaches. Considerable experimental results demonstrate the superiority of EASY.

Organization. The rest of the paper is organized as follows. Section 2 reviews the related work. Section 3 presents the technical details of our proposed EASY. Section 4 reports the experimental results and our findings. Finally, Section 5 concludes the paper.

2 RELATED WORK

Entity alignment (EA) is one of the most fundamental tasks for enlarging the coverage of a unified knowledge graph. Early techniques exploit hand-crafted features [22], crowdsourcing [41, 58], and OWL-based equivalence reasoning [14] to address entity alignment problems. However, those methods cannot effectively align entities from heterogeneous KGs, which are very common in real life since most KGs are independently created. Besides, they incur expensive monetary costs for involving humans in EA.

Recently, embedding techniques have been proposed to convert heterogeneous data into semantic vectors and have proven their effectiveness in the EA task [8, 32, 36]. Techniques on embedding-based EA are highly related to the graph structures of KGs. According to how the KG structures are captured, existing approaches can be clustered into two categories, namely, *Translational-based EA* and *GNN-based EA*. The first category [8, 19, 26, 28, 34, 35, 40, 56] incorporates the translational KG embedding models (such as TransE [5] and its variants) to ensure the aligned entities to have relative close embeddings in the semantic vector space. The second category [6, 18, 23, 32, 36, 42, 44, 57] learns the entity embeddings by aggregating the neighbors' information of entities.

Many EA approaches purely rely on graph structures of KGs guided by pre-aligned seeds [6, 8, 11, 18, 26, 28, 30, 32, 34–36, 56, 57]. However, Zeng et al. [52] revealed that most real-life KGs are sparse with limited structural information, and hence, the structural-dominated methods cannot yield satisfactory alignment results in real-life KGs. To overcome this limitation, some methods jointly exploit the graph structure and *the auxiliary information*, such as entity names [10, 21, 23–25, 33, 38, 45–49, 52, 54], descriptions [7, 38, 49, 54], images [20], and entity attributes [16, 21, 33, 38, 40, 42, 43, 49, 50, 54]. The studies based on images, descriptions, or attributes are either labor-intensive or error-prone and thus restrict the scope of their applications. Though it is promising to use the name information of entities, the existing studies may destroy the semantic information captured by the name of each individual entity as illustrated in Example 1.1.

Motivated by the limitations and challenges faced by existing EA approaches, our EASY is designed to use name information of entities in a much more effective manner when performing EA task. It is worth mentioning that among all auxiliary-information-based EA methods, the closest to the newly proposed EASY are three *unsupervised* models that do not require any manually annotated seed, i.e., EVA [20], MRAEA [23], and RREA [24]. Specifically, EVA still requires labor-intensive images collection in advance for

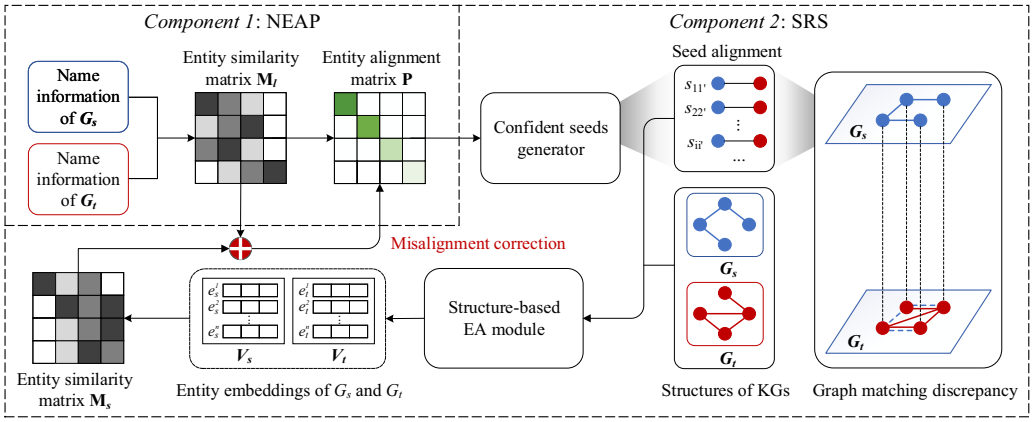


Figure 1: Illustration of our proposed EASY framework

generating pseudo seeds, while our work requires *zero* human intervention and performs EA in an end-to-end process. MRAEA and RREA treat the entity pairs that can be mutually translated as pseudo seeds and accumulate the seeds in iterations. We will demonstrate that the process of seed accumulation is error-prone in the experiments to be presented in Section 4.4.1. However, our EASY includes a novel *confident seeds generator* that enables more accurate seeds generation automatically. Last but not the least, unlike the above mentioned unsupervised models that aim to design newly structure-based EA models, the purpose of EASY is to construct a flexible EA framework that can be easily integrated with any existing structure-based EA models. We will show that EASY achieves the state-of-the-art EA results in the experiments to be presented in Section 4.1.

3 EASY FRAMEWORK

In this section, we first present some preliminaries of our work, including the formalization of EA and some background techniques; we then describe our end-to-end EA framework called EASY in detail. Figure 1 illustrates the EASY architecture, which includes two components: a *name-based entity alignment procedure* (NEAP) followed by a *structure-based refinement strategy* (SRS).

3.1 Preliminaries

Problem Statement. A knowledge graph (KG) can be denoted as $G = (\mathcal{E}, \mathcal{R}, \mathcal{T})$, where $\mathcal{E}$ is the set of entities, $\mathcal{R}$ is the set of relations with regard to entities, and $\mathcal{T}$ is the set of triples. Given two entities $e_i, e_j \in \mathcal{E}$ with a relation $r \in \mathcal{R}$ flowing from e_i to e_j, a triple $t = (e_i, r, e_j) \in \mathcal{T}$ represents the relationship between them. The entity alignment task aims to find a 1-to-1 matching of entities from a source KG $G_s = (\mathcal{E}_s, \mathcal{R}_s, \mathcal{T}_s)$ to a target KG $G_t = (\mathcal{E}_t, \mathcal{R}_t, \mathcal{T}_t)$ [37]. Formally, an entity alignment matrix $\boldsymbol{P} = \{0,1\}^{|\mathcal{E}_s| \times |\mathcal{E}_t|}$, w.l.o.g. $|\mathcal{E}_s| \leq |\mathcal{E}_t|$. Its element $P_{ii'}$ is such that $P_{ii'} = 1$ iff $e_s^i \equiv e_t^{i'}$. Here, $e_s^i \in \mathcal{E}_s$, $e_t^{i'} \in \mathcal{E}_t$, and $\equiv$ means an equivalence relation.

Graph Matching. Generally, the structure-based entity alignment method are highly relevant to the graph matching problem. Given two KGs G_s and G_t, graph matching aims to find a correspondence between their entities possessing similar neighborhood structures. A generic formulation of the graph matching consists of finding an optimal correspondence matrix by minimizing the difference between distributions of two KGs, defined as:

$$dis\,(G_s, G_t) = \min_{\boldsymbol{P}} \left(\sum_{e_s^i, e_s^j \in \mathcal{E}_s} \sum_{e_t^{i'}, e_t^{j'} \in \mathcal{E}_t} D(c_{ij}^s, c_{i'j'}^t) P_{ii'} P_{jj'} \right) \tag{1}$$

where $\boldsymbol{C} = [c_{ij}] \in \{0,1\}^{|\mathcal{E}| \times |\mathcal{E}|}$ is the adjacency matrix of a KG $G = (\mathcal{E}, \mathcal{R}, \mathcal{T})$. For simplicity, we treat each KG as a unweighted graph, that is, $c_{ij} = 1$ iff $\exists r \in \mathcal{R}$ such that $(e_i, r, e_j) \in \mathcal{T}$. $D(c_{ij}^s, c_{i'j'}^t) = |c_{ij}^s - c_{i'j'}^t|$ is an element-wise distance function. $\boldsymbol{P}$ is the optimal transport (w.r.t. entity alignment matrix) between two KGs, and its element $P_{ii'}$ represents whether $e_i \in \mathcal{E}_s$ and $e_{i'} \in \mathcal{E}_t$ are aligned. Note that, to enforce the 1-to-1 matching, $\boldsymbol{P}$ is defined to satisify $\forall e_s^i \in \mathcal{E}_s, \sum_{e_t^{i'} \in \mathcal{E}_t} P_{ii'} = 1$, and $\forall e_t^{i'} \in \mathcal{E}_t, \sum_{e_s^i \in \mathcal{E}_s} P_{ii'} \leq 1$.

3.2 Name-based Entity Alignment Procedure

Aligning entities from KGs without labor-intensive pre-processing is not trivial. In particular, it is challenging to propose an explicit loss function to guide the learning without any pre-aligned seed annotation. The rich semantic information contained in the names of entities inspires us to employ the entity names for aligning entities directly without any complex training process.

Recall that the name information has not been fully explored in the existing EA approaches. To this end, we design a simple but highly effective *name-based entity alignment procedure* (NEAP), which considers both *local features* and *global features* when performing the alignment. The local features of each entity refer to characterize each entity with tokenized name information. The global features aim to evaluate the semantic similarity between entities from two KGs. Next, we give the formal definitions of local features and global features, respectively.

Local Features. For each entity e_i, its name can be characterized as a series of tokens. A token can be a word or a sub-word. We denote the set of tokens corresponding to the name of e_i as $T_i = \{\tau_1, \tau_2, ..., \tau_j, ..., \tau_{|T_i|}\}$, where τ_j is the j-th token and $|T_i|$ is the number of tokens contained by the name of e_i. As mentioned earlier, every token plays a different role in representing the entity's name features. In order to describe the comprehensive local features of each entity, we assign different weights to different tokens for each entity. We use *term frequency–inverse document*

frequency (TF-IDF) (denoted as w), a well-known statistical measure, to reflect how important a token τ is to entity e_i. Formally, $w_{\tau,T_i} = \frac{\text{count}(\tau)}{|T_i|} \times \log \frac{|\mathcal{E}|}{1+\sum_{j=1}^{|\mathcal{E}|} I(\tau,T_j)}$, $I(\tau, T_j)$ denotes whether τ is contained in another entity e_j, i.e., $I(\tau, T_j) = 1$ when $\tau \in T_j$, otherwise $I(\tau, T_j) = 0$. Specifically, $\frac{\text{count}(\tau)}{|T_i|}$ reflects the frequency of τ appearing in e_i. The higher the frequency, the more important the token. $\log \frac{|\mathcal{E}|}{1+\sum_{i=1}^{|\mathcal{E}|} I(\tau,T_i)}$ reflects the universality of a token. The more common the token (that is, many entities have the token in their names), the lower the importance. Thus, the collection of local features of entities in a KG can be denoted as a matrix $\boldsymbol{E} \in [0,1]^{|\mathcal{E}|\times|T|}$. Take Figure 2 as an example. Given two entities e_s^1 in a KG_{FR} and $e_t^{1'}$ in a KG_{EN}, e_s^1 called "Blues de Saint Louis" can be split into four tokens, i.e., *Blues, de, Saint, Louis*. Similarly, $e_t^{1'}$ named "St. Louis Blues" can be split into three tokens, i.e., *St., Louis, Blues*. The local features of e_s^1 and $e_t^{1'}$ stored in the matrix $\boldsymbol{E_s}$ and $\boldsymbol{E_s}$ are highlight in the red dotted frame, respectively. The importance of each token is reflected by the color darkness. The darker the color, the higher the importance.

Global Features. Local features allow us to characterize each entity based on a list of tokens that appear in the name of the entity. Thereafter, we are ready to construct global features that try to capture the semantic similarities between entities from G_s and entities from G_t. According to the problem statement in Section 3.1, each entity in G_s has its equivalent entity in G_t. Besides, every entity can be treated as a set of tokens. Therefore, we propose to estimate entities' semantic similarities by measuring the semantic similarities between their relevant tokens. Specifically, we first represent tokens collected from two KGs as semantic vectors in a unified embedding space by a pre-trained language model, and then use certain similarity measures (e.g., cosine similarity is employed in this work) to quantify the similarity between tokens. Thus, the similarity between tokens from different KGs is denoted as $\boldsymbol{W_{s,t}} \in [0,1]^{|T_s|\times|T_t|}$, where $|T_s|$ and $|T_t|$ represent the total number of tokens of G_s and G_t, respectively. Given two tokens $\tau_s^i \in G_s$ and $\tau_t^j \in G_t$, the element stored in the i-th row and the j-th column of the matrix $\boldsymbol{W_{s,t}}$ indicates their similarity. As the objective of maintaining the global features is to capture the similarities between tokens from two KGs to facilitate the EA task, we develop to store the similarities of pairs of tokens that are very similar but ignore the scores of pairs of tokens that are different. To this end, we decide to sparsify $\boldsymbol{W_{s,t}}$ by only retaining the top-k correspondences, and reducing the required memory footprint from $O(|T_s| \times |T_t|)$ to $O(k|T_s|)$, where k is a small constant and $k \ll |T_s|$. In our work, we set $k = 1$ as it is found to be the optimal one via our experiments. Revisit our example shown in Figure 2. Circular nodes represent the tokens, and there are in total 12 (i.e., 4×3) token pairs. In $\boldsymbol{W_{s,t}}$, only the similarity scores of 3 token pairs (those connected via dotted lines, e.g, ("Saint", "St.")) are stored.

By considering both local and global features mentioned above, we can construct the name-based entity similarity matrix $\boldsymbol{M_l}$ between G_s and G_t via Equation (2):

$$\boldsymbol{M_l} = Sinkhorn(\hat{\boldsymbol{M}}_{\boldsymbol{l}}) = Sinkhorn(\boldsymbol{E_s W_{s,t} E_t^{\top}}) \tag{2}$$

where $\boldsymbol{E_s} \in [0,1]^{|\mathcal{E}_s|\times|T_s|}$, and $\boldsymbol{E_t} \in [0,1]^{|\mathcal{E}_t|\times|T_t|}$. *Sinkhorn* [31] is a normalization function used to get rectangular doubly-stochastic

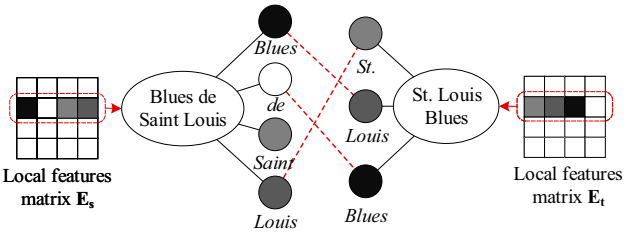


Figure 2: A toy example of NEAP

correspondence matrices fulfilling $\forall e_s^i \in \mathcal{E}_s$, $\sum_{e_t^{i'} \in \mathcal{E}_t} M_{ii'} = 1$ and $\forall e_t^{i'} \in \mathcal{E}_t$, $\sum_{e_s^i \in \mathcal{E}_s} M_{ii'} \leq 1$.

Thereby, we can get the name-based entity alignment matrix $\boldsymbol{P_{(0)}}$ between G_s and G_t derived from the similarity matrix $\boldsymbol{M_l}$. To enforce the 1-to-1 mapping, each element $P_{ii'} \in \boldsymbol{P_{(0)}}$ is given by:

$$P_{ii'} = \begin{cases} 1, & i' = argmax(M_i) \\ 0, & i' \neq argmax(M_i) \end{cases} \tag{3}$$

Discussion. The proposed NEAP significantly saves memory footprint without sacrificing the expressive power of entities. It is mainly attributed by the sparsity of $\boldsymbol{M_l}$, which is composed of three matrices, including two matrices corresponding to entities' local features (i.e., $\boldsymbol{E_s}$ and $\boldsymbol{E_t}$) and one matrix related to the global features (i.e., $\boldsymbol{W_{s,t}}$). Specifically, we assign TF-IDF for obtaining different importance of tokens to each entity. For each entity, only a small number of tokens are associated with it. Thus, both $\boldsymbol{E_s}$ and $\boldsymbol{E_t}$ are sparse matrices in nature. Besides, $\boldsymbol{W_{s,t}}$ is a sparse matrix that only retains the similarities between the most similar tokens. We will confirm the sparsity of the matrix $\boldsymbol{M_l}$ in Section 4.3.3.

3.3 Structure-based Refinement Strategy

Name information can help EA task to a certain degree. However, name information from different KGs might not be perfectly matched, especially in cross-lingual scenarios. In other words, misalignment inevitably exists in the initial alignment generated by NEAP. As highlighted by existing work [10, 36], the rich structural information of the KG also contributes to EA task. Intuitively, we expect to enhance the EA results by complementing the name information with structural information. A straightforward method is to discover entity pairs based on the name information as pseudo alignment seeds for training a structure-based EA model. Nonetheless, it is still challenging due to the presence of misaligned entity pairs generated by NEAP based on the name information.

To correct the misaligned entities in NEAP, we present a structure-based refinement strategy (SRS), which iteratively (i) provides a novel *confident seeds generator* to generate pseudo seed alignment; (ii) feeds the pseudo seeds into a *structure-based EA model* to learn entity embeddings; and (iii) obtains the structure-based entity similarities according to the learned embeddings and then mixes them with the name-based entity similarities from NEAP for *correcting the misaligned entities*. Next, we detail the three steps.

(i) Confident seeds generator. Under the assumption that each entity of the source KG is equivalent to a unique entity of the target KG, we expect graphs G_s and G_t to have similar graph structures. Specifically, for each equivalent entity pair $(e_s^i, e_t^{i'})$ where $e_s^i \in G_s$ and $e_t^{i'} \in G_t$, we expect the neighborhood of e_s^i should be the same

as that of $e_t^{i'}$. Nevertheless, the structures of G_s and G_t might appear differently due to the misaligned entities from NEAP and the heterogeneity between them. Apparently, the confidence of an entity pair $(e_s^i, e_t^{i'})$ is highly related to their neighbors' structural difference. We introduce *graph matching discrepancy* (GMD) to denote the neighborhood difference between two entities. The smaller the discrepancy between the neighborhoods of e_s^i and $e_t^{i'}$, the higher the confidence that they form a correct alignment.

Inspired by confidence-based techniques [27, 34], we propose to give each entity pair an approximate confidence and select the entity pairs with high confidence as seeds. In view of this, we first formulate GMD to represent the confidence of each entity pair and then illustrate how to generate confident seeds using GMD.

Recall that the EA task considering the structure of KGs can be abstracted as a graph matching problem. It aims to find the optimal entity alignment matrix by minimizing the structural difference between the source and the target KGs. Conversely, given a certain entity alignment matrix $\boldsymbol{P}$, it is able to compute the structural difference between two KGs. That is, derived from Equation (1), we can quantify GMD $d_{ii'}$ between every entity pair $(e_s^i, e_t^{i'})$ based on the given entity alignment matrix $\boldsymbol{P}$. Formally, $d_{ii'}(\boldsymbol{C_s}, \boldsymbol{C_t}, \boldsymbol{P}) = \sum_j^{K_i} \sum_{j'}^{K_{i'}} |c_{ij}^s - c_{i'j'}^t| P_{ii'} P_{jj'}$, where K_i and $K_{i'}$ denote the number of neighbors of e_s^i and that of $e_t^{i'}$, respectively. However, it fails to effectively reflect the impact of GMD on different entity pairs. We use the following example to depict the drawback.

Example 3.1. Assume that there are two entity pairs, denoted as $pair_1$ and $pair_2$. The entities in $pair_1$ are both connected with many neighbors; while the entities in $pair_2$ are connected to less than three other entities, i.e., they both are long-tail entities [11]. Only one edge difference exists between the neighborhood graph structure of the two entities in both pairs. Obviously, GMD of $pair_1$ and that of $pair_2$ are both 1. The first pair $pair_1$ is likely to be a correct alignment because of the small GMD. Nevertheless, the second pair may not be correctly aligned, even though their GMD is small too. This is because it's much easier for long-tail entities to have similar neighborhood structures, and unfortunately nearly half of the entities in real-life KGs are long-tail [52]. In other words, a small GMD, in the current form, is not necessarily a good indicator to tell that the alignment of a pair of two long-tail entities is correct.

To tackle the drawback, we re-formulate the equation of GMD as follows:

$$d_{ii'}(\boldsymbol{C_s}, \boldsymbol{C_t}, \boldsymbol{P}) = \frac{\sum_j^{K_i} \sum_{j'}^{K_{i'}} |c_{ij}^s - c_{i'j'}^t| P_{ii'} P_{jj'}}{\sum_j^{K_i} c_{ij}^s P_{ii'} P_{jj'} + \sum_{j'}^{K_{i'}} c_{i'j'}^t + \epsilon} \tag{4}$$

where the denominator is used to bound $d_{ii'}$ in the range of $[0, 1]$; and $\epsilon > 0$ is used to prevent the denominator from being zero.

After acquiring GMD for all entity pairs based on Equation (4), we have a collection of entity pairs, each of which has its confidence (w.r.t. GMD) to describe its likelihood. We only consider entity pairs with GMD no larger than a specified threshold $\bar{d}$ as seeds. Here, $\bar{d}$ denotes the average GMD of all entity pairs. We will justify the setting of $\bar{d}$ for pruning unreliable seeds in Section 4.4.2.

(ii) Structure-based EA model. Then, a structure-based EA model takes in the seed alignment produced by our proposed seeds generator to learn entity embeddings and provide a basis for obtaining the subsequent structure-based entity similarities. As the structure-based EA model has been extensively studied, we can adopt a state-of-the-art model (e.g.,RREA [24] in our work) and learn the entity embeddings according to the following loss function:

$$L = \sum\nolimits_{(e_s^i, e_t^{i'}) \in SP} \max\left(dist\left(e_s^i, e_t^{i'}\right) - dist\left(e_s^{i^*}, e_t^{i'^*}\right) + \lambda, 0\right) \tag{5}$$

where $e_s^{i^*}$ and $e_t^{i'^*}$ represent the negative pair of e_s^i and $e_t^{i'}$ which are generated by nearest neighbor sampling [34]. In the training process, we adopt the same setting as RREA [24] to use Manhattan distance to compute $dist(\cdot, \cdot)$. Note that, users have the flexibility to apply any EA model in our proposed EASY framework.

(iii) Misaligned Entities Adjustment. Both structural and name information are essential for entities. They characterize entities from two different aspects. Intuitively, jointly utilizing structural and name information is of great significance to facilitate the EA task. Hence, we introduce a hybrid similarity matrix $\boldsymbol{M}_h$ that captures both structural and name information, defined as $\boldsymbol{M}_h = \boldsymbol{M_s} + \alpha \boldsymbol{M_n}$, where $\boldsymbol{M_s}$ denotes the structure-based entity similarity matrix computed by the cosine similarity between the learned entity embeddings V_s from G_s and V_t from G_t. α is a hyper-parameter controlling the contribution of the name information to $\boldsymbol{M}_h$. $\boldsymbol{M_n}$ represents the name-based entity similarity matrix. Inspired by CEA [51], which claims that both semantic-level and string-level name information can describe entities from different perspectives, we set $\boldsymbol{M_n} = \boldsymbol{M_l} + \boldsymbol{M_e}$. Here, $\boldsymbol{M_l}$ is the semantic-level similarity matrix generated by NEAP, and $\boldsymbol{M_e}$ is the string-level similarity matrix computed by the Levenshtein distance between entities.

SRS. We are ready to introduce the details of SRS. Algorithm 1 presents its pseudo-code. It takes as inputs a name-based entity similarity matrix $\boldsymbol{M_n}$ and an entity alignment matrix $\boldsymbol{P_{(0)}}$. During the refinement process, we iteratively adjust the hybrid entity similarity matrix $\boldsymbol{M_h}$ and the entity alignment matrix $\boldsymbol{P}$ with the purpose of aligning entities more accurately. First of all, SRS initializes $\boldsymbol{M_h}$ with $\boldsymbol{M_n}$ for capturing the name similarity between entities from the two KGs. Also, it sets $\boldsymbol{P}$ to the name-based entity alignment matrix $\boldsymbol{P_{(0)}}$ (Line 1). Next, SRS calculates the *graph matching discrepancy*, and uses *confident seed generator* to generate a set of seed alignment SP (Lines 4-6). Then, it trains the structure-based

Algorithm 1: Structure-based Refinement Strategy (SRS)

Input: two KGs G_s and G_t with their name-based similarity matrix $\boldsymbol{M_n}$ and entity matching matrix $\boldsymbol{P_{(0)}}$
Output: the final entity matching matrix $\boldsymbol{P}$

1 $\boldsymbol{M_h} \longleftarrow \boldsymbol{M_n}$; $\boldsymbol{P} \longleftarrow \boldsymbol{P_{(0)}}$
2 $N_{it} \longleftarrow$ the maximum number of iterations
3 **foreach** $k \in \{1, 2, ..., N_{it}\}$ **do**
4 $\quad \boldsymbol{C_s}, \boldsymbol{C_t} \longleftarrow$ adjacency matrices of G_s, G_t
5 $\quad$ compute $d_{ii'}(\boldsymbol{C_s}, \boldsymbol{C_t}, \boldsymbol{P})$ for each entity pair via $\boldsymbol{P}$
6 $\quad$ get entity pairs SP based on the *confident seeds generator*
7 $\quad \boldsymbol{V_s}, \boldsymbol{V_t} \longleftarrow$ train the *structure-based EA model*
8 $\quad \boldsymbol{M_s} \longleftarrow Sim(\boldsymbol{V_s}, \boldsymbol{V_t})$; $\boldsymbol{M_h} \longleftarrow \boldsymbol{M_s} + \alpha \boldsymbol{M_n}$
9 $\quad \boldsymbol{P} \longleftarrow$ find an entity alignment matrix based on $\boldsymbol{M_h}$
10 $\quad$ mitigate the structural difference between G_s and G_t
11 **return** $\boldsymbol{P}$

Table 1: Statistics of the datasets used in experiments

Datasets		#Entities	#Rels.	#Triples
$\text{DBP15K}_{\text{ZH–EN}}$	Chinese	66,469	2,830	153,929
	English	98,125	2,317	237,674
$\text{DBP15K}_{\text{JA–EN}}$	Japanese	65,744	2,043	164,373
	English	95,680	2,096	233,319
$\text{DBP15K}_{\text{FR–EN}}$	French	66,858	1,379	192,191
	English	105,889	2,209	278,590
$\text{SRPRS}_{\text{EN–FR}}$	English	15,000	221	36,508
	French	15,000	177	33,532
$\text{SRPRS}_{\text{EN–DE}}$	English	15,000	225	38,281
	German	15,000	118	37,069

EA model with seed alignment SP, and obtains the embeddings $\boldsymbol{V_s}$ and $\boldsymbol{V_t}$ of entities $\mathcal{E}_s$ and $\mathcal{E}_t$ respectively (Line 7). Thereafter, a structure-based similarity matrix $\boldsymbol{M_s}$ can be obtained by computing the cosine similarities between $\boldsymbol{V_s}$ and $\boldsymbol{V_t}$, denoted as $Sim(\boldsymbol{V_s}, \boldsymbol{V_t})$. Then, SRS iteratively adjusts $\boldsymbol{M_h}$ according to $\boldsymbol{M_s}$ and $\boldsymbol{M_n}$ (Line 8). Besides, an entity alignment matrix $\boldsymbol{P}$ is derived from the current $\boldsymbol{M_h}$ by using Equation (3) (Line 9). Considering that similar graph structures are conducive to entity alignment, SRS uses IKGC, a KG completion method proposed in [52], to mitigate the structural difference between G_s and G_t based on the adjusted similarity matrix $\boldsymbol{M_h}$ (Line 10). This refinement strategy lasts for N_{it} rounds, and returns the final entity matching matrix P (Line 11).

Discussion. Existing EA methods, although also using iterative learning strategy for enhancing the EA results, only rely on the features that are fitted by the corresponding models to generate pseudo seeds. The sole consideration of fitted features causes the loss of actual information to a certain extent. Different from existing ER methods, SRS generates seeds by incorporating the actual name information with the structural features learned by the EA model. Consequently, it is able to achieve higher accuracy, as to be verified by our experimental study in Section 4.4.

4 EXPERIMENTS

Datasets. We use two frequently utilized and representative datasets DBP15K [33] and SRPRS [11] for evaluation. *DBP15K* is the most widely-adopted EA dataset, which is extracted from DBpedia and consists of three settings, i.e., ZH-EN (Chinese-English), JA-EN (Japanese-English), and FR-EN (French-English). Recent work [11, 37] points out that DBP15K contains much more high-degree entities than real-world KGs do. *SRPRS*, constructed by Guo et al. [11], is a dedicated KG that is sampled from real-world KGs (i.e., DBpedia, Wikidata, and YAGO) to simulate real-life distributions of entities and their relationships. Table 1 lists the detailed statistics.

Evaluation metric. *Hits@N* ($N = 1, 10$) and *Mean Reciprocal Rank (MRR)* are used as the evaluation metrics. We use cross-domain similarity local scaling (CSLS) [17] to post-process the entity similarity matrix $\boldsymbol{M_h}$, following the common practice adopted by lots of relate studies [20, 24, 36]. $K = 10$ is a parameter used for defining local neighbourhood of CSLS. In particular, Hits@1 represents the accuracy of alignment results in the final entity alignment matrix $\boldsymbol{P}$ (derived from the final hybrid similarity matrix $\boldsymbol{M_h}$). Hits@10 denotes the proportion of correctly aligned entities in the top-10 ranks, which can be obtained from $\boldsymbol{M_h}$. MRR is the average of the reciprocal ranks of the correctly aligned entities, where reciprocal rank reports the mean rank of the correct alignment derived from $\boldsymbol{M_h}$. Note that, higher Hit@N and MRR indicate better EA performance. **Bold** digits in tables come from the best EA method.

Implementation details. We conduct our proposed EASY[1] without using any annotated alignment seeds. In NEAP, the pre-trained *BERT* is utilized to obtain token embeddings by default unless explicitly specified. We also employ *fastText* for further analyzing the performance of our presented NEAP and SRS. (i) For getting the BERT-based token embeddings, following Liu et al. [21], we first use the pre-trained *BERT-base-cased*[2] to obtain a sequence of hidden states, each of which indicates a token's embedding associated with an entity name. Considering that a token may appear in different entity names and thus contribute to multiple embeddings, we then average all the embeddings to obtain a fixed length vector to represent the token. (ii) For getting the fastText-based token embeddings, we concatenate the *aligned word vectors*[3] [15] provided in different languages in cross-lingual benchmarks for each token. In addition, following many recent studies [21, 23, 24, 33, 43], we utilize Google Translate to translate entity names to English before executing NEAP for *DBP15K* dataset, which contains big linguistic barriers. In contrast, since *SRPRS* dataset has small barriers in linguistics, we, following [52], directly use entity names without translation. In SRS, we use RMSProp [12] as the gradient optimization algorithm for the structure-based EA model (i.e., RREA by default), and train the model for 30 epochs in each iteration. We set the learning rate to 0.005, $N_{it} = 20$, $\alpha = 0.2$, and $\lambda = 3$. All experiments were conducted on a personal computer with an Intel Core i9-10900K CPU, an NVIDIA GeForce RTX3090 GPU, and 128GB memory. The programs were all implemented in Python.

Competitors. We compare the performance of our proposed EASY framework against 13 state-of-the-art EA methods. The competitors can be classified into two categories, i.e., *graph-structure-dominated EA methods* and *auxiliary-information-powered EA methods*.

The former refers to the group that solely relies on structural information for aligning entities, including (i) MTransE [8], which learns the multilingual knowledge graph structure for EA according to TransE (a well-known KG embedding model); (ii) MuGNN [6], which learns alignment-oriented embeddings by a multi-channel graph neural network; (iii) RSNs [11], which employs recurrent neural networks to represent the sequence of entities and relations for entity alignment; and (iv) BootEA [34], which refines the entity alignment results by an iterative training process.

The latter incorporates the group that uses auxiliary information to improve the performance of graph-structure-based EA approaches. In this group, three state-of-the-art *unsupervised* methods are most relevant to our work, i.e., (i) EVA [20], the first work using images to collect pseudo seeds and learn entity alignment; (ii) MRAEA [23], the unsupervised version that performs better than its supervised version, which first generates a set of pseudo seeds according to the Google Translate results and then learns cross-lingual entity embeddings by considering the entity's neighbors and

[1] The source code is available at https://github.com/ZJU-DBL/EASY
[2] https://github.com/huggingface/transformers
[3] https://fasttext.cc/docs/en/aligned-vectors.html

Table 2: Overall results of EA with and without using auxiliary information (AI for short)

	Methods	$DBP15K_{ZH-EN}$			$DBP15K_{JA-EN}$			$DBP15K_{FR-EN}$			$SRPRS_{EN-FR}$			$SRPRS_{EN-DE}$		
		H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10	MRR
w/o AI	MTransE	20.9	51.2	0.31	25.1	57.2	0.36	24.7	57.7	0.36	21.3	44.7	0.29	10.7	24.8	0.16
	MuGNN	47.0	83.5	0.59	48.3	85.6	0.61	49.1	86.7	0.62	13.1	34.2	0.20	24.5	43.1	0.31
	RSNs	58.0	81.1	0.66	57.4	79.9	0.65	61.2	84.1	0.69	35.0	63.6	0.44	48.4	72.9	0.57
	BootEA	61.4	84.1	0.69	57.3	82.9	0.66	58.5	84.5	0.68	36.5	64.9	0.46	50.3	73.2	0.58
w/ AI	JAPE	41.4	74.1	0.53	36.5	69.5	0.48	31.8	66.8	0.44	24.1	54.4	0.34	26.8	54.7	0.36
	GCN-Align	43.4	76.2	0.55	42.7	76.2	0.54	41.1	77.2	0.53	29.6	59.2	0.40	42.8	66.2	0.51
	DAT*	54.5	64.9	0.58	58.8	66.4	0.62	64.3	68.0	0.66	75.8	89.9	0.81	87.6	95.5	0.90
	DGMC*	80.1	87.5	0.83	84.8	89.7	0.86	93.3	96.0	0.94	86.9	89.0	0.88	87.0	90.0	0.88
	CEA	78.7	–	–	86.3	–	–	97.2	–	–	96.2	–	–	97.1	–	–
	EPEA*	88.5	95.3	0.91	92.4	96.9	0.94	95.5	98.6	0.97	–	–	–	–	–	–
	EVA°*	75.2	89.5	0.80	73.7	89.0	0.79	73.1	90.9	0.79	–	–	–	–	–	–
	MRAEA°*	77.8	83.2	–	88.9	92.7	–	95.0	97.0	–	81.6	92.4	0.86	84.7	93.6	0.88
	RREA°*	82.2	96.4	–	91.8	97.8	–	96.3	99.2	–	82.2	92.5	0.85	85.3	93.8	0.89
	EASY (GCN-Align)	81.4	92.4	0.85	89.4	96.5	0.92	96.7	99.2	0.98	94.0	97.3	0.95	95.0	97.9	0.96
	EASY (RREA)	**89.8**	**97.9**	**0.93**	**94.3**	**99.0**	**0.96**	**98.0**	**99.8**	**0.99**	**96.5**	**98.9**	**0.97**	**97.4**	**99.2**	**0.98**

[1] The results of *-marked methods are obtained from their original papers. Underline indicates results from our re-implementation with their publicly available source code and data. The rest are from [55], a recent experimental study of state-of-the-art EA approaches. "°" denotes the unsupervised EA methods most relevant to our work.

its connected relations' meta semantics; and (iii) RREA [24], which adopts the unsupervised setting from MRAEA to generate a set of pseudo seeds, and achieves the state-of-the-art performance in the unsupervised settings according to a newly proposed GNN model. To demonstrate that our newly proposed EASY even outperforms some *supervised* models, we also implement six recent *supervised* models in this group, including (i) JAPE [33], which exploits attribute correlations for entity alignment, based on the assumption that similar entities should have similar correlated attributes; (ii) GCN-Align [42], which aligns entities by learning the entity embeddings from both the structural and attribute information of entities via graph convolutional networks; (iii) DAT [52], which proposes a degree-aware co-attention mechanism to dynamically fuse name and structural signals for aligning entities in tail; (iv) DGMC [10], which is aimed towards reaching a neighborhood consensus between aligned entities, and entity name information is harnessed for initializing the model; (v) CEA [51], which formulates EA as a stable matching problem built upon a distance measure combining structural and entity name information at outcome level; and (vi) EPEA [43], which first extracts attribute features of entity-pairs and then propagates the features among the neighbors of entity pairs for learning aligned entities.

4.1 Main Results

Table 2 summarizes the entity alignment performance of EASY and its competitors on the two datasets. To demonstrate the flexibility of EASY, we provide two variants, i.e, (i) EASY using the state-of-the-art RREA as the structure-based EA model, denoted as EASY (RREA); and (ii) EASY with a classical structure-based EA model called GCN-Align, denoted as EASY (GCN-Align).

It is observed that EASY (RREA) outperforms the state-of-the-art results on all datasets. Specifically, compared with the first group that purely relies on graph structure, EASY brings about 28%-60% absolute improvement in H@1 over the best baseline. The superiority of EASY confirms that employing name information substantially promotes the EA results. Compared with the second group with auxiliary information involvement, EASY still gains about 2%-14% absolute improvement in H@1 over the best baseline. It is contributed by two reasons. First, EASY utilizes NEAP to discover the name information of entities. It captures more accurate name features for EA than other existing methods that also explore and study entity names. The corresponding experimental results and detailed analysis can be found in Section 4.3. Second, EASY includes SRS, which effectively fuses the name information and structural information to generate more reliable seeds than other iterative strategies, and thus, it further enhances the EA results. We will compare the performance of SRS and other existing iterative training strategies in Section 4.4. Besides, as expected, integrating existing EA models (including GCN-Align and RREA) into our framework can bring about 8%-64% improvement in H@1, compared to the EA results using GCN-Align and RREA alone.

Table 3: The results of ablation

Methods	$SRPRS_{EN-FR}$			$SRPRS_{EN-DE}$		
	H@1	H@10	MRR	H@1	H@10	MRR
EASY	**96.5**	**98.9**	**0.97**	**97.4**	**99.2**	**0.98**
EASY w/o NEAP	93.2	97.2	0.95	95.1	97.9	0.96
EASY w/o SRS	91.4	93.1	0.92	92.8	95.2	0.94
EASY w/o IL	95.5	98.1	0.96	97.0	99.1	0.98
EASY w/o CSLS	95.9	98.3	0.97	97.1	98.9	0.98
EASY w/o $\boldsymbol{M_e}$	95.5	98.3	0.97	95.5	98.0	0.96

4.2 Ablation Study

We conduct an ablation study on SRPRS dataset, which simulates the real KGs' distributions better than the DBP15K dataset. The results are reported in Table 3. By replacing NEAP component with a simple max-pooling strategy, the performance of EASY drops by 3.3% and 2.3% on H@1 (EASY vs. EASY w/o NEAP) on EN-FR dataset and EN-DE dataset, respectively. This shows that considering both *local* and *global* features of entities does capture more

Table 4: Analysis of NEAP

	Methods	$SRPRS_{EN-FR}$			$SRPRS_{EN-DE}$		
		H@1	H@10	MRR	H@1	H@10	MRR
	Levenshtein	84.2	89.4	0.86	85.9	92.1	0.88
fastText	Avg	58.1	66.8	0.61	62.6	77.6	0.68
fastText	CPM	81.7	86.8	0.84	82.8	87.5	0.85
fastText	NEAP w/o Sinkhorn	84.3	89.2	0.86	86.3	91.2	0.88
fastText	NEAP	**88.3**	**89.7**	**0.89**	**89.7**	**92.4**	**0.91**
BERT	Avg	81.5	85.5	0.83	80.4	85.0	0.82
BERT	NameBERT	86.1	90.5	0.88	86.9	92.0	0.89
BERT	NEAP w/o Sinkhorn	87.2	92.1	0.89	88.5	93.5	0.90
BERT	NEAP	**91.4**	**93.1**	**0.92**	**92.8**	**95.2**	**0.94**

Table 5: The sparsity analysis of NEAP

Methods	$SRPRS_{EN-FR}$			$SRPRS_{EN-DE}$		
	$\|\mathcal{M}_l\|_0$	$\|\mathcal{E}_s\|/\|\mathcal{E}_t\|$	R_{occ}	$\|\mathcal{M}_l\|_0$	$\|\mathcal{E}_s\|/\|\mathcal{E}_t\|$	R_{occ}
NEAP w/ f.	3,353,516	15,000	1.5%	1,641,575	15,000	0.7%
NEAP w/ b.	**625,907**	15,000	**0.3%**	**308,910**	15,000	**0.1%**

semantic information for EA. By removing the entire SRS component, EASY performs EA purely based on the name information provided by NEAP. The results drop by 5.1% and 4.6% on H@1 (EASY vs. EASY w/o SRS) on EN-FR dataset and EN-DE dataset, respectively. It verifies that iteratively combining both name and structural information can correct the misaligned entities and thus contributes to more accurate EA results. Iteration (IL) brings no more than 1% absolute improvement. This is because NEAP and SRS already allow our framework to capture fairly good alignment in the first iteration round, leaving smaller room for further improvement. By removing CSLS, the results show minor drops. Since CSLS is used to mitigate the hubness phenomenon in a dense space [20, 37], it cannot substantially improve EASY due to the sparsity of entities similarity computation. Besides, by replacing the entity similarity matrix using both semantic-level and string-level name information with that only involving semantic-level name information, the results drop no more than 1.9% on H@1 (EASY vs. EASY w/o $\boldsymbol{M_e}$). It demonstrates that, although involving different levels of name information can improve the EA results, only considering semantic-level information can also achieve satisfactory results.

4.3 NEAP Analysis

In this section, we further investigate the performance of NEAP by (i) using different pre-train language models to represent entities, and (ii) comparing NEAP with other name-based heuristics. We directly apply NEAP to align entities without the subsequent refinement strategy. All experiments in the following subsections are conducted on the SRPRS dataset.

4.3.1 Pre-trained Language Models. We explore the performance of NEAP by employing different pre-trained language models, i.e., fastText and BERT, with results reported in Table 4. The first observation is that BERT-based NEAP achieves the best EA results, contributed by the characteristics of BERT. BERT uses a large-scale corpus for accurately learning the semantic information of each token, and it achieves state-of-the-art performance in many NLP tasks. The second observation is that even using the relatively inferior fastText model, NEAP still surpasses the majority of existing methods whose performance is reported in Table 2. This confirms the superiority of using NEAP in EA.

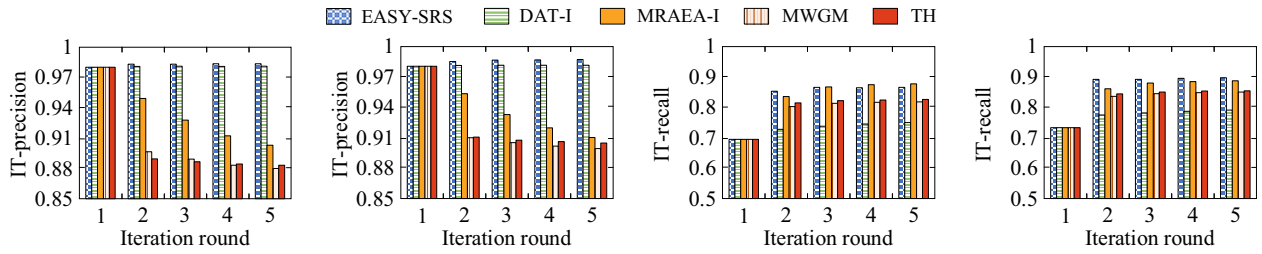
4.3.2 Name-based Heuristics. We also compare our proposed NEAP with five competitive name-based heuristics. Competitors represent entities in different ways, and derive the EA results according to the similarity between entities' names. Concretely, we use (i) *Levenshtein*, a string-based metric to measure the character similarity between two entities' name; (ii) *Avg*, which represents entities by averaging the corresponding token embeddings; and (iii) *CPM* [29], which concatenates different types of power mean word embeddings. A recent work DAT uses CPM to initialize the entity representations, and we use the settings reported in DAT to implement CPM; (iv) *NameBERT* [21], which applies the max-pooling strategy in a pre-trained BERT to obtain entity representations; and (v) *NEAP w/o Sinkhorn*, which obtains the name-based entity alignment directly without using the Sinkhorn process to optimize the alignment. Again, the results are depicted in Table 4. First, NEAP is observed to outperform all the competitors in all metrics. This validates that characterizing entities with different weighted tokens does capture more semantic features for EA. Second, even though we remove the Sinkhorn in NEAP, it still performs better than most of the existing heuristics. This further shows the superiority of NEAP that is simple but highly effective.

4.3.3 Sparsity Analysis. We explore the sparsity of NEAP by computing the memory footprint of the entity similarity matrix $\boldsymbol{M_l}$. It is known to all, only non-zero elements in a sparse matrix take up memory. We introduce a metric *occupation ratio*, denoted as R_{occ}, to quantify the memory footprint. R_{occ} is defined as $\frac{\|\boldsymbol{M_l}\|_0}{|\mathcal{E}_s|\times|\mathcal{E}_t|}$. Here, $\|\boldsymbol{M_l}\|_0$ represents the number of *non-zero* elements in $\boldsymbol{M_l}$, measuring the sparsity of this matrix [13]; $|\mathcal{E}_s| \times |\mathcal{E}_t|$ denotes the maximum occupation of the similarity matrix, where all elements are non-zero. We report the results by using fastText (f.) and BERT (b.) respectively in Table 5. We observe that compared to the dense similarity matrix where all elements are non-zero, R_{occ} is $\sim 0.1\%$ in the best case (i.e., using BERT for the EN-DE dataset) and no more than 1.5% in the worse case (i.e., using fastText for the EN-FR dataset). This demonstrates the high sparsity of NEAP.

4.4 SRS Analysis

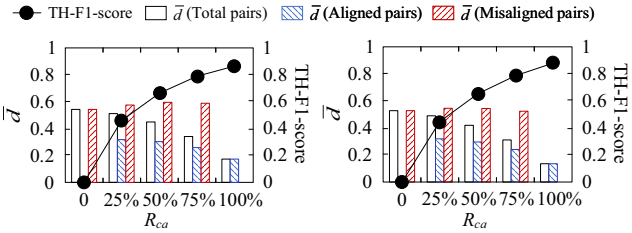
4.4.1 Iterative Strategy Variants. Next, we justify the effectiveness of our proposed SRS by comparing it with another four iterative strategies[4] that generate seeds in different ways, i.e., (i) DAT-I [52], where seed alignment requires the similarity between two entities in a seed to be the highest from both sides; (ii) MRAEA-I [23], a bi-directional iterative strategy that selects newly-aligned seeds in each iteration if and only if two entities from different KGs are mutually nearest neighbors of each other; (iii) TH [56], which collects entity pairs with similarity exceeding a given threshold as seeds in each iteration; and (iv) MWGM [34], which ensures the similarity within each selected seed alignment is above a given threshold and meanwhile guarantees the 1-to-1 labeling. We replace SRS with one

[4]Note that, in all iterative strategies, we apply the state-of-the-art structure-based EA model RREA to learn entity representations for fair comparisons.



(a) $\text{SRPRS}_{\text{EN-FR}}$ (b) $\text{SRPRS}_{\text{EN-DE}}$ (c) $\text{SRPRS}_{\text{EN-FR}}$ (d) $\text{SRPRS}_{\text{EN-DE}}$

Figure 3: Comparison results of iterative training strategies



(a) $\text{SRPRS}_{\text{EN-FR}}$ (b) $\text{SRPRS}_{\text{EN-DE}}$

Figure 4: The results of threshold analysis

of the above mentioned strategies under our framework, and tune the parameters according to their original papers. We utilize two metrics to measure the performance of different iterative strategies, i.e., *IT-precision* that is defined as the fraction of truly discovered seeds (that match the ground truth) over the total number of the generated seeds; and *IT-recall* that is defined as the fraction of truly discovered seeds over the total number of real entity alignment.

We plot their performance in Figure 3. We first observe that, in the first iteration, all the strategies achieve same precision and recall. This is because, seeds are only produced by NEAP in this iteration. As more iteration rounds are executed, IT-precision of SRS and DAT-I is observed to increase the values while that of other strategies drops. The reason is that, the initial seeds (w.r.t. the first round) achieve very high IT-precision values. Simply utilizing threshold or mutually nearest guarantee for seed generation is error-prone, accumulating erroneous seeds during the iteration. On the other hand, IT-recall of all the strategies is observed to increase its value as more iterations are performed. The reason is that the correct seeds generated in each round help train a reliable structure-based EA model, which can help identify more accurate seeds at the next iteration round. Last but not the least, it is observed that our proposed SRS is able to achieve higher precision and recall than other iterative strategies. This verifies that iteratively combining both name information and structural information contributes to more accurate alignment results. On the contrary, the existing iterative methods rely on the features that are fitted by the models to generate pseudo seeds and hence suffer from the loss of actual information to a certain extent.

4.4.2 Threshold Justification. We analyze the justification of leveraging the threshold (i.e., averaged graph matching discrepancy, denoted as $\bar{d}$) for identifying the misaligned entities. Let *correct alignment rate* (denoted as R_{ca}) be the proportion of correct aligned entities to the total number of alignments. We change R_{ca} from 0 to 100% by replacing the ground truth EA with misalignment (i.e., random permutation). We introduce a new metric *TH-F1-score* for evaluation. TH-F1-score is the harmonic mean between TH-precision (P) and TH-recall (R), i.e., $\frac{2 \cdot P \cdot R}{P+R}$. Here, P is defined as the fraction of truly discovered seeds (that match the ground truth) over the total number of the retained seeds by using the threshold $\bar{d}$ for filtering. R is defined as the fraction of truly discovered seeds over the total number of ground truth contained the current EA.

Figure 4 illustrates the performance of our *graph matching discrepancy* on different R_{ca}. First, the averaged graph matching discrepancy of (correctly) aligned entity pairs is always lower than that of misaligned entity pairs. This shows the effectiveness of using $\bar{d}$ to distinguish the correctly aligned pairs and misaligned pairs. Second, the averaged graph matching discrepancy of total entity pairs decreases as R_{ca} increases. This is because the more correct pairs, the smaller the graph matching discrepancy between KGs. Besides, as expected, TH-F1-score grows when R_{ca} ascends, since the fewer the misaligned pairs, the easier the filtering out the correct pair via the threshold $\bar{d}$. Furthermore, we can observe that when R_{ca} exceeds 60%, we can generate reliable seeds based on $\bar{d}$ (i.e., TH-F1-score>70%). Since our NEAP ensures that R_{ca} exceeds 60% in the initial entity alignment, it verifies the effectiveness of the threshold for selecting confident seeds in SRS.

5 CONCLUSIONS

In this paper, we explore an effective end-to-end EA framework EASY. It first aligns entities based on entities' name information without any complex training process via our presented NEAP. NEAP considers both *global features* and *local features* of entities' names, and greatly saves memory cost without sacrificing the expressive power of entities. Then, we present a novel iterative strategy SRS, which includes a novel concept of *graph matching discrepancy* and a confident seeds auto-generator for the guidance of fusing the features of both name and structural information to correct the misaligned entities in NEAP, and thus improves the EA results. Considerable experimental results on cross-lingual EA benchmarks demonstrate the superiority of EASY. In our current implementation, EASY does not consider attribute information (a popular type of auxiliary information) as it is error-prone. In the future, we would like to study the attribute information for EA.

ACKNOWLEDGMENTS

This work was supported by the NSFC under Grant No. 62025206 and 61972338. Lu Chen is the corresponding author of the work.

REFERENCES

[1] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary G. Ives. 2007. DBpedia: A Nucleus for a Web of Open Data. In *ISWC*. 722–735.

[2] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomás Mikolov. 2017. Enriching Word Vectors with Subword Information. *TACL* 5 (2017), 135–146.

[3] Kurt D. Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *SIGMOD*. 1247–1250.

[4] Antoine Bordes, Sumit Chopra, and Jason Weston. 2014. Question Answering with Subgraph Embeddings. In *EMNLP*. 615–620.

[5] Antoine Bordes, Nicolas Usunier, Alberto García-Durán, Jason Weston, and Oksana Yakhnenko. 2013. Translating Embeddings for Modeling Multi-relational Data. In *NIPS*. 2787–2795.

[6] Yixin Cao, Zhiyuan Liu, Chengjiang Li, Zhiyuan Liu, Juanzi Li, and Tat-Seng Chua. 2019. Multi-Channel Graph Neural Network for Entity Alignment. In *ACL*. 1452–1461.

[7] Muhao Chen, Yingtao Tian, Kai-Wei Chang, Steven Skiena, and Carlo Zaniolo. 2018. Co-training Embeddings of Knowledge Graphs and Entity Descriptions for Cross-lingual Entity Alignment. In *IJCAI*. 3998–4004.

[8] Muhao Chen, Yingtao Tian, Mohan Yang, and Carlo Zaniolo. 2017. Multilingual Knowledge Graph Embeddings for Cross-lingual Knowledge Alignment. In *IJCAI*. 1511–1517.

[9] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *NAACL-HLT*. 4171–4186.

[10] Matthias Fey, Jan Eric Lenssen, Christopher Morris, Jonathan Masci, and Nils M. Kriege. 2020. Deep Graph Matching Consensus. In *ICLR*.

[11] Lingbing Guo, Zequn Sun, and Wei Hu. 2019. Learning to Exploit Long-term Relational Dependencies in Knowledge Graphs. In *ICML*. 2505–2514.

[12] Geoffrey Hinton, Nitish Srivastava, and Kevin Swersky. 2012. Neural networks for machine learning lecture 6a overview of mini-batch gradient descent. (2012).

[13] Niall P. Hurley and Scott T. Rickard. 2009. Comparing measures of sparsity. *IEEE Trans. Inf. Theory* 55, 10 (2009), 4723–4741.

[14] Ernesto Jiménez-Ruiz and Bernardo Cuenca Grau. 2011. LogMap: Logic-Based and Scalable Ontology Matching. In *ISWC*. 273–288.

[15] Armand Joulin, Piotr Bojanowski, Tomás Mikolov, Hervé Jégou, and Edouard Grave. 2018. Loss in Translation: Learning Bilingual Word Mapping with a Retrieval Criterion. In *EMNLP*. 2979–2984.

[16] Chao Kong, Ming Gao, Chen Xu, Yunbin Fu, Weining Qian, and Aoying Zhou. 2019. EnAli: entity alignment across multiple heterogeneous data sources. *Frontiers Comput. Sci.* 13, 1 (2019), 157–169.

[17] Guillaume Lample, Alexis Conneau, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. 2018. Word translation without parallel data. In *ICLR*.

[18] Chengjiang Li, Yixin Cao, Lei Hou, Jiaxin Shi, Juanzi Li, and Tat-Seng Chua. 2019. Semi-supervised Entity Alignment via Joint Knowledge Embedding Model and Cross-graph Model. In *EMNLP*. 2723–2732.

[19] Xixun Lin, Hong Yang, Jia Wu, Chuan Zhou, and Bin Wang. 2019. Guiding Cross-lingual Entity Alignment via Adversarial Knowledge Embedding. In *ICDM*. 429–438.

[20] Fangyu Liu, Muhao Chen, Dan Roth, and Nigel Collier. 2020. Visual Pivoting for (Unsupervised) Entity Alignment. *arXiv preprint arXiv:2009.13603* (2020).

[21] Zhiyuan Liu, Yixin Cao, Liangming Pan, Juanzi Li, and Tat-Seng Chua. 2020. Exploring and Evaluating Attributes, Values, and Structures for Entity Alignment. In *EMNLP*. 6355–6364.

[22] Farzaneh Mahdisoltani, Joanna Biega, and Fabian M. Suchanek. 2015. YAGO3: A Knowledge Base from Multilingual Wikipedias. In *CIDR*.

[23] Xin Mao, Wenting Wang, Huimin Xu, Man Lan, and Yuanbin Wu. 2020. MRAEA: An Efficient and Robust Entity Alignment Approach for Cross-lingual Knowledge Graph. In *WSDM*. 420–428.

[24] Xin Mao, Wenting Wang, Huimin Xu, Yuanbin Wu, and Man Lan. 2020. Relational Reflection Entity Alignment. In *CIKM*. 1095–1104.

[25] Hao Nie, Xianpei Han, Le Sun, Chi Man Wong, Qiang Chen, Suhui Wu, and Wei Zhang. 2020. Global Structure and Local Semantics-Preserved Embeddings for Entity Alignment. In *IJCAI*. 3658–3664.

[26] Shichao Pei, Lu Yu, Robert Hoehndorf, and Xiangliang Zhang. 2019. Semi-Supervised Entity Alignment via Knowledge Graph Embedding with Awareness of Degree Difference. In *WWW*. 3130–3136.

[27] Shichao Pei, Lu Yu, Guoxian Yu, and Xiangliang Zhang. 2020. REA: Robust Cross-lingual Entity Alignment Between Knowledge Graphs. In *KDD*. 2175–2184.

[28] Shichao Pei, Lu Yu, and Xiangliang Zhang. 2019. Improving Cross-lingual Entity Alignment via Optimal Transport. In *IJCAI*. 3231–3237.

[29] Andreas Rücklé, Steffen Eger, Maxime Peyrard, and Iryna Gurevych. 2018. Concatenated power mean word embeddings as universal cross-lingual sentence representations. *arXiv preprint arXiv:1803.01400* (2018).

[30] Xiaofei Shi and Yanghua Xiao. 2019. Modeling Multi-mapping Relations for Precise Cross-lingual Entity Alignment. In *EMNLP*. 813–822.

[31] Richard Sinkhorn and Paul Knopp. 1967. Concerning nonnegative matrices and doubly stochastic matrices. *Pacific J. Math.* 21, 2 (1967), 343–348.

[32] Zequn Sun, Muhao Chen, Wei Hu, Chengming Wang, Jian Dai, and Wei Zhang. 2020. Knowledge Association with Hyperbolic Knowledge Graph Embeddings. In *EMNLP*. 5704–5716.

[33] Zequn Sun, Wei Hu, and Chengkai Li. 2017. Cross-Lingual Entity Alignment via Joint Attribute-Preserving Embedding. In *ISWC*. 628–644.

[34] Zequn Sun, Wei Hu, Qingheng Zhang, and Yuzhong Qu. 2018. Bootstrapping Entity Alignment with Knowledge Graph Embedding. In *IJCAI*. 4396–4402.

[35] Zequn Sun, JiaCheng Huang, Wei Hu, Muhao Chen, Lingbing Guo, and Yuzhong Qu. 2019. TransEdge: Translating Relation-Contextualized Embeddings for Knowledge Graphs. In *ISWC*. 612–629.

[36] Zequn Sun, Chengming Wang, Wei Hu, Muhao Chen, Jian Dai, Wei Zhang, and Yuzhong Qu. 2020. Knowledge Graph Alignment Network with Gated Multi-Hop Neighborhood Aggregation. In *AAAI*. 222–229.

[37] Zequn Sun, Qingheng Zhang, Wei Hu, Chengming Wang, Muhao Chen, Farahnaz Akrami, and Chengkai Li. 2020. A Benchmarking Study of Embedding-based Entity Alignment for Knowledge Graphs. *PVLDB* 13, 11 (2020), 2326–2340.

[38] Xiaobin Tang, Jing Zhang, Bo Chen, Yang Yang, Hong Chen, and Cuiping Li. 2020. BERT-INT: A BERT-based Interaction Model For Knowledge Graph Alignment. In *IJCAI*. 3174–3180.

[39] Peihao Tong, Qifan Zhang, and Junjie Yao. 2019. Leveraging Domain Context for Question Answering Over Knowledge Graph. *Data Sci. Eng.* 4, 4 (2019), 323–335.

[40] Bayu Distiawan Trisedya, Jianzhong Qi, and Rui Zhang. 2019. Entity Alignment between Knowledge Graphs Using Attribute Embeddings. In *AAAI*. 297–304.

[41] Denny Vrandecic and Markus Krötzsch. 2014. Wikidata: a free collaborative knowledgebase. *Commun. ACM* 57, 10 (2014), 78–85.

[42] Zhichun Wang, Qingsong Lv, Xiaohan Lan, and Yu Zhang. 2018. Cross-lingual Knowledge Graph Alignment via Graph Convolutional Networks. In *EMNLP*. 349–357.

[43] Zhichun Wang, Jinjian Yang, and Xiaoju Ye. 2020. Knowledge Graph Alignment with Entity-Pair Embedding. In *EMNLP*. 1672–1680.

[44] Yuting Wu, Xiao Liu, Yansong Feng, Zheng Wang, Rui Yan, and Dongyan Zhao. 2019. Relation-Aware Entity Alignment for Heterogeneous Knowledge Graphs. In *IJCAI*. 5278–5284.

[45] Yuting Wu, Xiao Liu, Yansong Feng, Zheng Wang, and Dongyan Zhao. 2019. Jointly Learning Entity and Relation Representations for Entity Alignment. In *EMNLP*. 240–249.

[46] Yuting Wu, Xiao Liu, Yansong Feng, Zheng Wang, and Dongyan Zhao. 2020. Neighborhood Matching Network for Entity Alignment. In *ACL*. 6477–6487.

[47] Kun Xu, Linfeng Song, Yansong Feng, Yan Song, and Dong Yu. 2020. Coordinated Reasoning for Cross-Lingual Knowledge Graph Alignment. In *AAAI*. 9354–9361.

[48] Kun Xu, Liwei Wang, Mo Yu, Yansong Feng, Yan Song, Zhiguo Wang, and Dong Yu. 2019. Cross-lingual Knowledge Graph Alignment via Graph Matching Neural Network. In *ACL*. 3156–3161.

[49] Hsiu-Wei Yang, Yanyan Zou, Peng Shi, Wei Lu, Jimmy Lin, and Xu Sun. 2019. Aligning Cross-Lingual Entities with Multi-Aspect Information. In *EMNLP*. 4430–4440.

[50] Kai Yang, Shaoqin Liu, Junfeng Zhao, Yasha Wang, and Bing Xie. 2020. COTSAE: CO-Training of Structure and Attribute Embeddings for Entity Alignment. In *AAAI*. 3025–3032.

[51] Weixin Zeng, Xiang Zhao, Jiuyang Tang, and Xuemin Lin. 2020. Collective Entity Alignment via Adaptive Features. In *ICDE*. 1870–1873.

[52] Weixin Zeng, Xiang Zhao, Wei Wang, Jiuyang Tang, and Zhen Tan. 2020. Degree-Aware Alignment for Entities in Tail. In *SIGIR*. 811–820.

[53] Fuzheng Zhang, Nicholas Jing Yuan, Defu Lian, Xing Xie, and Wei-Ying Ma. 2016. Collaborative Knowledge Base Embedding for Recommender Systems. In *SIGKDD*. 353–362.

[54] Qingheng Zhang, Zequn Sun, Wei Hu, Muhao Chen, Lingbing Guo, and Yuzhong Qu. 2019. Multi-view Knowledge Graph Embedding for Entity Alignment. In *IJCAI*. 5429–5435.

[55] Xiang Zhao, Weixin Zeng, Jiuyang Tang, Wei Wang, and Fabian M. Suchanek. 2020. An experimental study of state-of-the-art entity alignment approaches. *TKDE* 10 (2020).

[56] Hao Zhu, Ruobing Xie, Zhiyuan Liu, and Maosong Sun. 2017. Iterative Entity Alignment via Joint Knowledge Embeddings. In *IJCAI*. 4258–4264.

[57] Qiannan Zhu, Xiaofei Zhou, Jia Wu, Jianlong Tan, and Li Guo. 2019. Neighborhood-Aware Attentional Representation for Multilingual Knowledge Graphs. In *IJCAI*. 1943–1949.

[58] Yan Zhuang, Guoliang Li, Zhuojian Zhong, and Jianhua Feng. 2017. Hike: A Hybrid Human-Machine Method for Entity Alignment in Large-Scale Knowledge Bases. In *CIKM*. 1917–1926.