

Large-scale Entity Alignment in Knowledge Graphs Using Language Models

Ningxin Chen¹, Zhichun Wang^{1,2*}

¹School of Artificial Intelligence, Beijing Normal University, Beijing 100875, China

²Engineering Research Center of Intelligent Technology and Educational Application, Ministry of Education, Beijing 100875, China

Keywords: Knowledge graph; Pre-trained language model; Entity alignment; Large-scale entity alignment; Dense retrieval

Citation: Chen N.X., Wang Z.C.: Large-scale Entity Alignment in Knowledge Graphs Using Language Models. Data Intelligence, 2026, Vol. 8, Art. No.: 2026r05. DOI: <https://doi.org/10.3724/2096-7004.di.2025.0074>

ABSTRACT

Entity alignment (EA) is crucial for knowledge fusion and integration, as it aims to match equivalent entities across different KGs. Recently, many neural-based EA methods have been proposed, focusing on developing various graph representation learning models to match entities in vector spaces. However, most real-world KGs are large-scale and contain rich structural and attribute information about entities, presenting challenges for current approaches designed primarily for small- and medium-sized KGs. To address the challenges of large-scale EA, this paper introduces a simple, effective, and scalable method based on language models. Our approach first leverages the capabilities of language models to encode entities' multi-view information into low-dimensional embeddings, identifying potential aligned entity pairs with high similarity. These candidates are then re-ranked using a global matching algorithm to produce the final alignments. Experimental results show that our method achieves state-of-the-art performance on real-world large-scale EA datasets, with superior accuracy and efficiency compared to existing methods.

1. INTRODUCTION

Knowledge Graphs (KGs) represent structured information of entities in various domains. Well-known examples of KGs include YAGO [1], DBpedia [2], Freebase [3], and WikiData [4]. Information of entities are represented as triples (*subject*, *predicate*, *object*) in KGs, which facilitates machines to handle domain knowledge. Most published KGs are heterogeneous because they are built from different data sources

*Corresponding author: Zhichun Wang (E-mail: zcwang@bnu.edu.cn, ORCID: 0000-0001-8797-7065).

and use different terminologies. To integrate knowledge in separate KGs, it is essential to perform Entity Alignment (EA), which aims to discover equivalent entities in different KGs.

To precisely align entities between KGs, embedding-based EA has attracted research interests, which first embed entities in low-dimensional vector spaces, and then discover alignments based on the distance of embeddings. There are mainly two paradigms of KG embedding, translation-based methods and Graph Neural Network (GNN) based methods: translation-based methods learn entity embeddings using TransE or its extended models, including MTransE [5], JAPE [6], and BootEA [7], etc; GNN-based methods learn neighborhood-aware representations of entities by aggregating features of their neighbors, including GCN-Align [8], MuGNN [9], and AliNet [10], etc. The above two methods mainly utilizing structure information, to further improve results, some approaches explore entities' attribute information to enhance the entity embeddings by using character or word embedding techniques, which includes MultiKE [11], AttrGNN [12] and CEA [13], etc. Most recently, pre-trained language models (PLMs) have also been used to encode the attribute values, such as in BERT-INT [14], SDEA [15]. Some approaches explored Large Language Models (LLMs) to match entities, including AutoAlign [16], LLMEA [17], and ChatEA [18], etc.

Although much progress has been achieved, we find that there are still several challenges to real-world EA problems.

Challenge 1: most real-world KGs are large-scale. In most existing approaches, entity embeddings are parameters to be learned. Embedding models have to take the whole KG as inputs, the computation and storage complexities increase as the number of entities increases. Therefore, most of the existing approaches can only run on small and medium-sized KGs, they cannot work well on large-scale EA tasks. Some approaches have been proposed to partition large KGs into sub-graphs, and then use existing models to learn entity embeddings and get alignments of sub-graphs, including LargeEA [19], LIME [20], DivEA [21] and LargeGNN [22]. These approaches divide a large EA task into sub-tasks to reduce the number of pair-wise entity comparisons, but the number of total parameters (entity embeddings) does not decrease. These approaches have to access all entities during training.

Challenge 2: entity relations and attributes are difficult to be encoded in a unified way. Real-world KGs contain both relation and attribute triples, and most existing approaches use different techniques to encode two kinds of triples. For example, MultiKE [11] generates vector representations of attributes based on pre-trained word embeddings of literals in the attribute values, while structure information is modeled by KG embedding model TransE. AttrGNN [12] uses BERT to encode literal features in attributes, and uses GNNs to encode structure information. There lacks a unified and effective method to model different types of information in KGs.

Challenge 3: seed alignments are usually difficult or expensive to obtain. Most existing approaches assume there are certain number of seed alignments. Usually, 30% of entities in two KGs are assumed to be pre-aligned in previous work. Seed alignments are used as training samples and anchors to link two KGs. However, annotating seed alignments requires massive manual work, which may not be available in practice. Many previous approaches do not work without sufficient seed alignments.

To deal with the above challenges, we formalize EA as an entity retrieval problem, and propose a Language Model based large-scale Entity Alignment (**LMEA**) approach. Given two KGs, a source KG and a target KG, LMEA first encodes attribute and structure information using pre-trained language models. Entities in the target KG are indexed based on multi-view encodings, candidate alignments are obtained by retrieving top-k similar target entities with source entities. To further improve the alignment results, a re-ranking model is designed to adjust the similarities by optimizing a global matching objective. Comparing with the existing approaches, our approach LMEA has the following advantages:

- LMEA is inductive learning approach so that it does not need to access all the entities in two KGs during training; only a small number of seed alignments are needed to train the retriever and re-ranker in LMEA.
- LMEA adopts the dense retrieval framework, entity embeddings are generated and indexed for efficient similarity search. Therefore, LMEA works well on large-scale EA problems.
- Experiments on real-world datasets show that LMEA can achieve comparable results with baselines on small datasets, LMEA outperforms baselines significantly while running on mid-size and large-scale KGs.

The rest of this paper is organized as follows: Section 2 formalizes the entity alignment problem, Section 3 describes our proposed approach, Section 4 presents the evaluation results, Section 5 discusses some related work.

2. PROBLEM FORMULATION

Knowledge Graph (KG). KGs represent structural information about entities as triples $\langle s, p, o \rangle$. A triple can be relation or attribute: relation triple describes relation between entities, and attribute triple describes attribute of an entity. In this work, we consider both triples in KGs. Formally, we represent a KG as $G = (E, R, A, V, T)$, where E, R, A and V are sets of entities, relations, attributes, and literal values; $T \subseteq (E \times E) \cup (E \times A \times V)$ is the sets of triples.

Entity Alignment. Given two KGs G_s and G_t , and a set of pre-aligned entity pairs $S = \{(u, v) | u \in G_s, v \in G_t, u \equiv v\}$ ($\equiv$ denotes equivalence), the task of entity alignment is to find new equivalent entity pairs between G_s and G_t .

Entity Alignment as Entity Retrieval. In this work, we formalize entity alignment as an entity retrieval task. Given a source KG G_s and a target KG G_t , entity retrieval aims to return a ranked list of k most similar entities in G_t for each entity $s \in G_s$, i.e. $L(s) = [t_1, t_2, \dots, t_k]$. The top-ranked entity t_1 is considered as be equivalent to the source entity s .

3. METHODOLOGY

Figure 1 shows the overall framework of our proposed approach. Given two KGs to be aligned, LMEA works in two main stages: nearest entity retrieval and global matching re-ranking. In the nearest entity retrieval stage, LMEA adopts a multi-view language model based architecture to generate multiple representations of each entity; entities in the source KG are used as queries to retrieve similar indexes in the target KG. In the global matching re-ranking stage, LMEA employs optimal transport technique to re-rank the candidate target entities for each source entity.

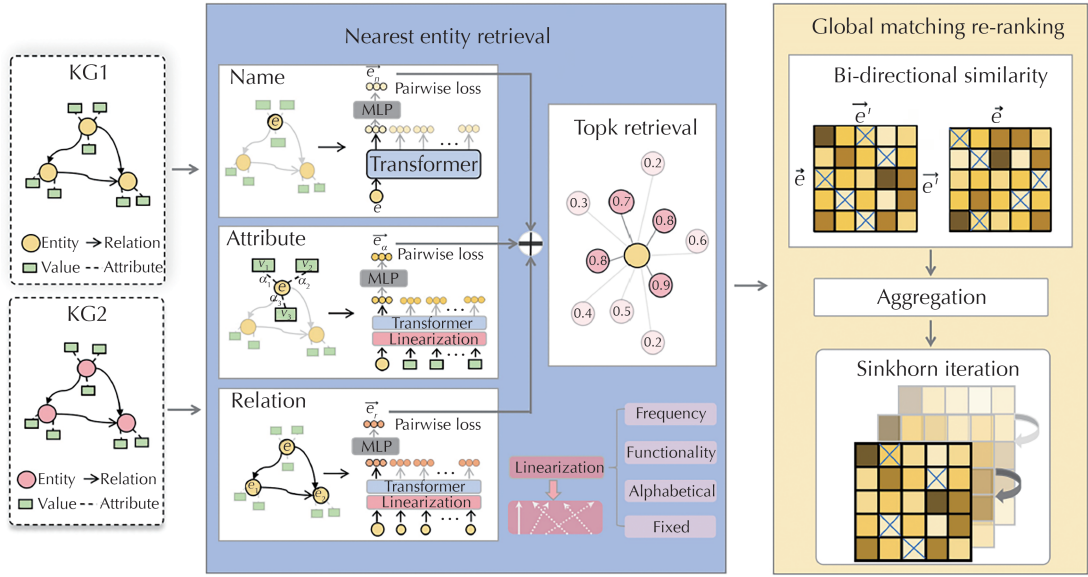


Figure 1. The LMEA framework.

3.1 Nearest Entity Retrieval

This section introduces the details of Nearest Entity Retrieval. To achieve accurate entity retrieval, PLMs are leveraged in our approach to encode entities into dense vectors, and the similarities of entities are computed using their vectors:

$$f(s, t) = \text{sim}(\phi(s), \psi(t)) \quad (1)$$

where $\phi(\cdot) \in \mathbb{R}^d$ and $\psi(\cdot) \in \mathbb{R}^d$ are PLM-based encoders mapping the source and target entities into d -dimensional vector space, respectively. In this work, we use the same encoder for source and target entities, and use dot-product for computing the similarity of entities. Therefore, the similarity is redefined as:

$$f(s, t) = \phi(s) \cdot \phi(t) \quad (2)$$

There are different types of information about entities in a KG (i.e. names, attributes, relations), single vector representation cannot well encode multi-aspect semantics of an entity. To fully explore all kinds of information for precisely EA, our approach encodes different information and generates multi-view representations of each entity, including vectors of entity name, attributes, and relations.

3.1.1 Entity Feature Encoding

Information of entities are described using triples, including relational ones and attributive ones. Previous EA approaches employed different techniques to encode entities' attributive and relational features. In this work, we use a unified way to represent all the available features of entities. Triples of entities are first linearized into sequences of tokens, which are then processed by PLM-based encoders to get their vector representations. More specifically, we generate three sequences for each entity, which correspond to its name feature, relation feature, and attribute feature, respectively. In the following, we introduce how the input sequences are built.

Entity Name. Name of an entity is an important clue for determining whether two entities are equivalent. Let e be the literal name of an entity, we construct input of name feature as:

$$S_e^{name} = \{[CLS], e, [SEP]\}$$

Entity Relations. Let $\mathcal{N}(e) = \{(r_i, e_i)\}$ be a set of neighbors and associated relations of entity e , where e_i is an entity and r_i is the relation connecting two entities. Here, we consider only one-hop neighbors for efficiency when dealing with large-scale KGs.

$$S_e^{rel} = \{[CLS], e, [SEP], r_1, e_1, \dots, r_j, e_j, [SEP]\}$$

Entity Attributes. Let $\mathcal{A}(e) = \{(a_i, v_i)\}$ be a set of attributes and values, where v_i is the value of a_i of e . The input of attribute feature is:

$$S_e^{att} = \{[CLS], e, [SEP], a_1, v_1, \dots, a_i, v_i, [SEP]\}$$

During the linearization of triples, attributes and relations are ranked in descending order of their functionalities [23], a metric that measures the importance of relations in matching entities in ontologies. We extend this functionality to attributes as well. After converting entities' triples to sequences of tokens, sequences are then fed to language model based encoders. Let $w \in \{name, att, rel\}$ represent a specific view, entity representation is generated by feeding the hidden state of [CLS] from encoder to a MLP layer:

$$\phi_w(e) = \text{MLP}(\text{PLM}_{\text{CLS}}(S_e^w)) \quad (3)$$

3.1.2 Triple Ordering Methods

When converting an entity's relational and attributive triples into input sequences for language model encoding, the order in which these triples are arranged can significantly influence the resulting

representations and, consequently, the alignment performance. To systematically study this effect, we design and implement four different triple ordering strategies. These methods determine the order of relations and attributes when constructing the input sequences for each entity.

Let an entity e have a set of neighboring relations and entities $\mathcal{N}(e) = \{(r_i, n_i)\}$, and a set of attributes and values $\mathcal{A}(e) = \{(a_i, v_i)\}$. The four ordering strategies are as follows:

- **Frequency:** This method sorts relations or attributes by their frequency of occurrence across the entire KG. Relations (or attributes) that appear more frequently are placed earlier in the sequence. Specifically, we compute the global frequency of each relation and attribute and sort them in descending order before linearizing them into the input.
- **Functionality:** This method sorts attributes or relations in descending order of their functionality. Functionality is a measure of to what extent a relation or an attribute is a function one. The functionality of a relation r is defined as

$$\text{fun}(r) = \frac{|\{s \mid (s, r, o) \in T_{\text{rel}}\}|}{|\{(s, o) \mid (s, r, o) \in T_{\text{rel}}\}|} \quad (4)$$

where T_{rel} is the set of relational triples in the KG. Following similar idea, we define the functionality of an attribute a as:

$$\text{fun}(a) = \frac{|\{s \mid (s, a, v) \in T_{\text{att}}\}|}{|\{(s, v) \mid (s, a, v) \in T_{\text{att}}\}|} \quad (5)$$

where T_{att} is the set of attributive triples in the KG. Higher functionality implies a more discriminative feature. Relations and attributes are sorted in descending order of functionality before inclusion in the input.

- **Alphabetical:** This strategy sorts the relation and attribute names alphabetically. It serves as a naive baseline that does not depend on global statistics or data distribution.
- **Fixed:** This method randomly generates a fixed permutation of all relation and attribute names at the start of training. This permutation is then consistently applied to every entity in the dataset. The goal is to test the robustness of the model to arbitrary but consistent orderings.

After ordering, the selected relations and attributes are tokenized into sequences and encoded using PLMs. For example, the relation sequence is constructed as:

$$S_e^{\text{rel}} = [\text{CLS}], e, [\text{SEP}], r_1, n_1, r_2, n_2, \dots, r_j, n_j, [\text{SEP}]$$

and similarly for the attribute sequence.

In our experiments (see Section 4.3.3), we evaluate all four strategies and find that functionality-based and frequency-based orderings lead to more consistent and robust alignment performance across datasets. Therefore, we adopt the **functionality-based ordering** as the default configuration in our final model.

3.1.3 Model Training

To get the optimal dense representations of entities so that equivalent entities have higher similarities than inequivalent entities, we use known alignments to train the encoders in our approach. The parameters in the PLM together with the MLP layer in a encoder are optimized in the training process. Concretely, given a set of seed alignments $S = \{(u, v) | u \in G_s, v \in G_t, u \equiv v\}$, each encoder is trained by minimizing the following loss function:

$$\mathcal{L} = \sum_{(u_i, v_i) \in S} \sum_{v'_i \in N_i} \max\{\gamma - f(u_i, v_i) + f(u_i, v'_i), 0\} \quad (6)$$

where $\gamma > 0$ is a margin hyper-parameter, $f(u, v) = \mathbf{u} \cdot \mathbf{v}$, and N_i is the set of negative target entities for u_i . Negative target entities are randomly selected from G_t .

Our approach uses three PLM-based encoders for encoding name, relation, and attribute features of entities. In the training process, we train these encoders separately, which do not share parameters. During the training, seed alignments are used as positive samples, while negative samples are not explicitly available. We have to sample negative samples for each positive one. As studied in the previous approaches [24], negative sampling methods highly influence the final results of EA. The simplest method is to uniformly sample negative entities from all the possible negative ones. A better method is to select hard negative samples in the training process, which can improve the retriever's capability of discriminating between equivalent and inequivalent entities. In this work, we select negative samples for each source entity from the top entities in the retrieval results before training the encoders. Negative samples are generated before training and keep the same during the training process. This method allows the model to focus on hard negative samples while maintaining efficiency.

3.1.4 Entity Retrieval

The multi-view entity encoders generate three vector representations, denoted as $\phi_w(e)$, $w \in \{\text{name}, \text{att}, \text{rel}\}$. To combine alignment information from multi-views, our method generates a fused representation $\phi(e)$ by concatenating vectors of three views:

$$\phi(e) = [\phi_{\text{name}}(e) || \phi_{\text{att}}(e) || \phi_{\text{rel}}(e)] \quad (7)$$

Using the fused feature from multiple views, our approach builds entity index using Faiss^①. Faiss is a library for efficient similarity search and clustering of dense vectors, which implements k-nearest neighbor search method on GPU proposed in [25]. With the Faiss index, we can effectively search the top-k nearest entities in the target KG G_t for each entity in the source KG G_s . Here we use $L(e) = [t_1, t_2, \dots, t_k]$ denote the set of nearest entities for $e \in G_s$, where $t_i \in G_t$ ($i = 1, 2, \dots, k$).

^① <https://github.com/facebookresearch/faiss>.

3.2 Global Matching Re-ranking

The results of nearest entity retrieval only consider the local similarities between entities, without taking the global matching constraint into account, i.e. one entity in the source KG can only match at most one entity in the target KG. If we only consider the similarities between entity pairs and select the closest one as the alignment result, the one-to-one matching rule may not be satisfied. In this work, we design a global matching re-ranker for the alignment results. The re-ranker first performs bidirectional entity retrieval to build a fused similarity matrix of two KGs, and then applies Sinkhorn algorithm on the similarity matrix to obtain one-to-one entity alignments.

3.2.1 Bi-directional Similarity Fusion

Using entities from the source knowledge graph (KG) as queries, we retrieve the top- k nearest entities from the target KG. By computing similarities between each source entity e and each candidate target entity $t \in L(e)$, we can then construct a similarity matrix $\mathbf{S} \in \mathbb{R}^{|E_s| \times |E_t|}$, where E_s and E_t are the sets of entities in the two KGs, and S_{ij} represents the similarity between the i -th entity in E_s and the j -th entity in E_t . We retrieve only the top- k nearest entities for each source entity, and set the similarities between the source entity and target entities not in the top- k to 0. As a result, the similarity matrix is sparse.

To further improve alignment, we also perform entity retrieval in the opposite direction, using target entities as queries to find corresponding source entities. This generates an additional similarity matrix $\mathbf{S}' \in \mathbb{R}^{|E_t| \times |E_s|}$. To integrate the similarities from both matrices $\mathbf{S}$ and $\mathbf{S}'$, we compute an aggregated similarity matrix as follows:

$$\mathbf{M} = \frac{1}{2}(\mathbf{S} + \mathbf{S}') \quad (8)$$

3.2.2 Reranking by Sinkhorn Algorithm

As EA is a one-to-one matching task, previous studies [26-27] consider it as an assignment problem:

$$\begin{aligned} \text{Maximize} \quad & Z = \sum_{i=1}^{|E_s|} \sum_{j=1}^{|E_t|} x_{ij} M_{ij} \\ \text{s.t.} \quad & \begin{cases} \sum_{j=1}^{|E_t|} x_{ij} = 1, & \forall i \in \{1, 2, \dots, |E_s|\} \\ \sum_{i=1}^{|E_s|} x_{ij} = 1, & \forall j \in \{1, 2, \dots, |E_t|\} \\ x_{ij} \in \{0, 1\} \end{cases} \end{aligned} \quad (9)$$

where M is the aggregated similarity matrix, and x_{ij} indicates whether the i -th entity in E_s matches the j -th entity in E_t . a permutation matrix denoting the alignment. The permutation matrix E_t is a square binary matrix in which each row and each column contains exactly one entry equal to 1, while all remaining

entries are 0. To solve this assignment problem, we apply the Sinkhorn algorithm [28] on the aggregated similarity matrix, which iteratively performs both row-normalization and column-normalization of the matrix. Sinkhorn algorithm guarantees the one-to-one mapping rules in a soft manner. When Sinkhorn algorithm obtain the convergence result, a doubly-stochastic matrix is generated.

$$\mathbf{D} = \text{Sinkhorn}(\mathbf{M}) \quad (10)$$

The element D_{ij} in the doubly-stochastic matrix is taken as the matching score of entity pair (e_i, e_j) , where $e_i \in E_s$ and $e_j \in E_t$. To generate the final alignments, the target entity with the highest matching score is selected for each source entity.

4. EXPERIMENTS

This section presents the experimental results. First, we describe the experimental settings. Next, we provide the main results of the experiments. Finally, we conduct several in-depth analyses of our approach.

4.1 Experiment Settings

4.1.1 Datasets

We use three groups of datasets in the experiments, which contain 15K, 100K, and more than 1M entity pairs, respectively. These datasets are widely used in related studies, and they are representative of small, medium, and large scale datasets.

- SRPRS [29]. It contains two cross-lingual datasets, which are built from DBpedia. Each dataset has 15,000 aligned entity pairs. SRPRS datasets contain both relation and attribute triples.
- IDS100K [30]. It contains two cross-lingual datasets which are also build from DBpedia. Each dataset has 100k aligned entity pairs. IDS100K datasets also contain both relation and attribute triples.
- DBP1M [19]. It contains two large-scale cross-lingual datasets extracted from DBpedia. There are more than 1 million aligned entity pairs. DBP1M datasets only contain relation triples.

Table 1 summarizes the statistics of the datasets.

4.1.2 Implementation Details

We implement our approach using PyTorch^② and Transformers^③. We use multi-lingual BERT^④ for cross-lingual entity alignment. In our approach, the dimension of vector representations in each view is 768, the number of negative samples is 2. During the training, we randomly choose 30% alignments from training set as validation set. We set the maximum training epoch to 20, and select the best model based on the

② <https://pytorch.org>.

③ <https://huggingface.co/docs/transformers>.

④ <https://huggingface.co/google-bert/bert-base-multilingual-cased>.

Table 1. Statistics of the datasets used in experiments.

Datasets		Entity	Alignment	Rel.	Rel.Tri.	Attr.	Val.	Attr.Tri.
SRPRS EN-FR	EN	15000	15000	296	36508	580	62098	70750
	FR	15000		177	33532	415	37414	56344
SRPRS EN-DE	EN	15000	15000	244	38363	354	57298	62715
	DE	15000		120	37377	193	39021	142506
IDS EN- FR-100K	EN	100000	100000	516	309607	819	366608	497729
	FR	100000		300	258285	519	228356	426670
IDS EN- DE-100K	EN	100000	100000	410	335359	517	473939	552750
	DE	100000		196	336240	252	326166	716615
DBP EN- FR-1M	EN	1211270	488563	771	8150656	0	0	0
	FR	1176865		400	4268719	0	0	0
DBP EN- DE-1M	EN	1160306	418436	633	7922555	0	0	0
	DE	849646		247	3000489	0	0	0

Hits@1 on the validation set. All the experiments are run on a workstation with an Intel Xeon CPU @ 2.50GHz and 128GB RAM, a GPU with 24GB VRAM.

The results of LargeEA, LIME, and LightEA on IDS100K and DBP1M are reproduced using their source codes. In their original versions, literal information (names, attributes) are encoded using pre-trained word embeddings. According to our experiments, using BERT as encoder will significantly improve the results. Therefore, we replace pre-trained word embeddings in these approaches with BERT for fare comparison.

One of the advantages of our approach LMEA is that it needs only small number of seed alignments to train the retriever and reranker. For all the datasets, LMEA only uses 3k aligned entity pairs as training data. For large-scale KGs as in DBP1M, training data is only 0.3% of the whole dataset.

4.1.3 Evaluation metrics.

We use Hits@k, MRR and Running Time for evaluation.

Hits@k metric evaluates the performance of aligning entities between two knowledge graphs. It measures the proportion of source entities for which the correct target entity is ranked within the top-k results, based on their matching scores.

$$\text{Hits@k} = \frac{1}{|E_s|} \sum_{e_i \in E_s} \mathbb{I}(\text{rank}(\text{GT}(e_i)) \leq k) \quad (11)$$

where $|E_s|$ is the total number of source entities in the dataset; $\mathbb{I}(\text{rank}(\text{GT}(e_i)) \leq k)$ is an indicator function that takes the value 1 if the rank of the ground-truth aligned target entity $\text{GT}(e_i)$ for the source entity e_i is within the top k , and 0 otherwise.

Mean Reciprocal Rank (MRR) evaluates the quality of alignment predictions by considering the rank of the correct target entity for each source entity. It is defined as the average of the reciprocal ranks of the ground-truth target entities. Let E_s and E_t be the sets of source and target entities, respectively. For a source entity $e_i \in E_s$, let $\text{rank}(\text{GT}(e_i))$ denote the rank of its ground-truth aligned target entity $\text{GT}(e_i) \in E_t$, based on the predicted matching scores. The MRR is calculated as:

$$\text{MRR} = \frac{1}{|E_s|} \sum_{e_i \in E_s} \frac{1}{\text{rank}(\text{GT}(e_i))} \quad (12)$$

Running Time measures the computational efficiency of an entity alignment model when aligning entities between two large-scale KGs. This metric is crucial for evaluating the practicality of the model when dealing with large-scale KGs. Running time refers to the total amount of time a model, including model training, similarity computation and final results generation.

4.1.4 Baselines.

We compare our approach with three groups of entity alignment approaches, including embedding-based approaches, LLM-based approaches, and approaches designed for Large-scale entity alignment tasks.

Embedding-based approaches: (1) MTransE [5] encodes entities and relations of two KGs using TransE, and provides transitions for each embedding vector to its counterparts in other spaces. (2) GCN-Align [8] uses GCN to generate entity embeddings to align entities in different KGs. (3) RDGCN [31] uses relation-aware dual GCNs to incorporate relation information into entity embeddings for entity alignment. (4) GMNN [32] first constructs local sub-graphs of entities and then solves entity alignment as a graph matching problem. (5) EASY [33] first performs name-based entity alignment, and then uses a structure-based refinement strategy to iteratively correct the misaligned entities.

PLM-based approaches: (1) AttrGNN [12] encodes attribute triples and relation triples in a unified network, and learn importance of each attributes and values dynamically. (2) BERT-INT [14] uses BERT to encode entities' names and descriptions, and computes the interactions between neighbors to align entities. (3) TEA [34] transforms triples of two entities into textual sequences and treats the alignment problem as a PLM-based text entailment problem. (4) SDEA [15] uses BERT to encode attribute information of entities, and employs BiGRU to incorporate relation information into embeddings.

Large-scale EA approaches: (1) LargeEA [19] partitions the source and target KGs using METIS and constructs entity alignment blocks. (2) LIME [20] also partitions large KGs into small sub-graphs, it uses a bidirectional partitioning method to guarantee the recall of seed mappings in each subtask. (3) LightEA [27] employs label propagation algorithm to effectively align KGs, it can handle large-scale EA because of its non-neural framework.

We do not include recent LLM-based methods such as LLMEA [17], ChatEA [18], and LLM4EA [35] in our comparisons. These approaches utilize LLMs for reasoning over entity candidates and often rely on API-based or pairwise input paradigms that require extensive computational resources. While promising in accuracy on small datasets, they scale poorly when applied to large KGs due to the exponential increase in candidate pairs and API call overhead. As our focus is on large-scale, efficient EA, such methods are not directly comparable. Moreover, most of them depend on candidate filtering techniques (e.g., from conventional EA methods) before applying the LLM, and the actual contribution of the LLM component remains limited when considering end-to-end scalability.

In contrast, our method LMEA is designed to balance accuracy and efficiency: it uses pre-trained language models for feature encoding in a retrieval-based framework, supports fast vector search with FAISS, and achieves strong performance even with limited training data. This makes LMEA more suitable for real-world large-scale EA tasks.

4.2 Main Results

This section reports the experimental results of LMEA on SRPRS, IDS100K, and DBP1M, and the best results are in boldface, the second best with underlines. The encoder used in entity feature encoding is the only part that requires training. To ensure time efficiency on large datasets, we standardized the training set sizes across all three datasets, with 3K samples for training and 1.5K for validation.

4.2.1 Results on SRPRS

Table 2 presents the experimental results on the SRPRS datasets. The best performance in each column is highlighted in bold, while the second-best is underlined. We compare our approach against both embedding-based methods (MTransE, GCN-Align, RDGCN, GMNN, and EASY) and PLM-based methods (AttrGNN, BERT-INT, TEA, and SDEA). The results for the embedding-based methods are taken from previously published papers. Since the PLM-based methods do not have reported results on SRPRS, we ran experiments using their official source codes to obtain these results.

The results demonstrate that PLM-based approaches generally outperform traditional embedding-based methods. Compared to these approaches, our method achieves the best performance on the SRPRS EN-FR dataset, attaining the highest Hits@1, Hits@10, and MRR scores. For the SRPRS EN-DE dataset, our method ranks third in Hits@1 and MRR, and second in Hits@10.

It is worth noting that our approach generates entity embeddings without involving any entity interactions, ensuring efficiency when handling large-scale knowledge graphs (KGs). In contrast, methods such as SDEA, BERT-INT, and TEA employ more complex processes to generate entity embeddings, which contributes to their top-ranked performance on the SRPRS datasets. However, these methods face efficiency challenges when scaling to larger KGs.

Table 2. Overall EA results on SRPRS datasets.

Methods	SRPRS EN-FR			SRPRS EN-DE		
	Hits@1	Hits@10	MRR	Hits@1	Hits@10	MRR
MTransE	0.213	0.447	0.290	0.107	0.248	0.160
GCN-Align	0.243	0.522	0.340	0.385	0.600	0.460
GMNN	0.574	0.646	0.602	0.681	0.748	0.710
RDGCN	0.672	0.767	0.710	0.779	0.886	0.820
EASY	0.965	0.989	0.970	0.974	0.992	0.980
SDEA	0.966	0.986	0.970	0.968	0.989	0.980
BERT-INT	0.963	0.968	0.965	<u>0.984</u>	0.986	<u>0.986</u>
TEA-MLM	<u>0.977</u>	<u>0.990</u>	<u>0.982</u>	0.987	0.997	0.990
LMEA	0.983	0.995	0.988	0.972	<u>0.990</u>	0.970

4.2.2 Results on IDS100K

Table 3 presents the results on the IDS100K datasets. We compare LMEA with methods specifically designed for large-scale entity alignment tasks, including LIME [20], LightEA [27], and LargeEA [19], as well as the aforementioned PLM-based approaches. For LightEA, we use the LightEA-L version which uses the pre-trained word embeddings of translated entity names as the inputs. In addition to Hits@k and MRR, we report the Running Time, as efficiency is a critical factor in large-scale entity alignment.

Table 3. Overall EA results on IDS100K datasets.

Methods	IDS100K EN-FR				IDS100K EN-DE			
	Hits@1	Hits@10	MRR	Time	Hits@1	Hits@10	MRR	Time
SDEA	0.923	0.961	0.938	3h08m	0.930	<u>0.975</u>	0.947	1h41m
BERT-INT	<u>0.949</u>	0.956	<u>0.952</u>	3h05m	<u>0.936</u>	0.941	0.938	3h18m
TEA-MLM	0.947	<u>0.975</u>	<u>0.952</u>	7h01m	0.949	0.978	0.959	7h10m
LargeEA	0.758	0.844	0.796	17m33s	0.757	0.846	0.797	<u>18m12s</u>
LIME	0.932	<u>0.975</u>	0.948	32m03s	0.931	<u>0.975</u>	<u>0.949</u>	33m25s
LightEA-L	0.893	0.965	0.919	07m54s	0.907	0.965	0.929	07m36s
LMEA	0.956	0.978	0.964	<u>15m30s</u>	0.929	0.957	0.939	21m30s

Among PLM-based methods, while they achieve higher Hits values, they exhibit lower efficiency. This is because PLM-based methods often perform pairwise comparisons among relations and attributes, which enhances precision but also increases computational time as the number of entities grows.

Among all large-scale alignment approaches, LMEA achieves the best results on the EN-FR dataset and ranks third in Hits@1 on the EN-DE dataset. Regarding efficiency, LightEA is the fastest, as it uses training-free label propagation. Our approach, LMEA, ranks second and third in running time across the two datasets, balancing performance and efficiency effectively.

4.2.3 Results on DBP1M

Table 4 presents the results on the DBP1M datasets, the largest among the three dataset groups. Due to the scale of DBP1M, embedding-based and PLM-based methods are not applicable [22]. Therefore, we only compared our approach to large-scale entity alignment approaches, including LargeEA, LIME and LightEA.

Table 4. Overall EA results on DBP1M datasets.

Method	DBP1M EN-FR				DBP1M EN-DE			
	Hits@1	Hits@10	MRR	Time	Hits@1	Hits@10	MRR	Time
LargeEA	0.759	0.816	0.784	2h45m	0.784	0.832	0.805	2h32m
LIME	<u>0.799</u>	<u>0.858</u>	<u>0.832</u>	3h15m	<u>0.799</u>	<u>0.860</u>	<u>0.834</u>	2h53m
LightEA-L	0.657	0.833	0.720	1h32m	0.683	0.838	0.738	1h24m
LMEA	0.920	0.963	0.940	<u>2h30m</u>	0.915	0.954	0.930	<u>2h25m</u>

According to the results, our approach LMEA achieves the best Hits@1, Hits@10, and MRR on both DBP1M EN-FR and EN-DE datasets. Specifically, LMEA outperforms the second-best approach, LIME, by 12.1% and 11.6% in Hits@1 on the two datasets, respectively.

In terms of efficiency, LightEA remains the fastest method. LMEA ranks second in running time, demonstrating greater efficiency than LargeEA and LIME while maintaining superior performance.

4.2.4 Discussion on Alignment Errors

Based on the experimental results presented above, we observe that while the alignment model performs well in many cases, there are still instances of misalignment. We identify three key factors that contribute to these failures: name ambiguity, attribute inconsistency, and sparse relational context. When all of these factors are present, our model struggles to correctly identify the true alignments.

A typical failure case involves the source entity `Neighbors_(1920_film)`[Ⓢ] and its target entity `La_Voisine_de_Malec`[Ⓢ] in the SRPRS EN-FR dataset. These two entities have significantly different names—“Neighbors” vs. “La Voisine de Malec”—introducing lexical ambiguity. In terms of attributes, the source entity includes `imdbId:0011508` and a runtime value of “1080.0”, which are either missing or inconsistent in the target entity. Additionally, in terms of relations, the two entities are weakly connected within their respective knowledge graphs, with only 7 and 5 neighbors, respectively. As a result, the model fails to find significant similarities in names, attributes, or relations, leading to errors in identifying the true alignments.

4.3 In-depth Analysis

4.3.1 Impact of Hyperparameters on Model Performance

To assess the sensitivity of our model to hyperparameters, specifically the learning rate and batch size, we perform a grid search across a range of configurations and evaluate their impact on the Hit@1 performance on SRPRS datasets. The results are summarized in Table 5. Our default setting, with a learning rate of 1×10^{-5} and a batch size of 32, achieves the best performance, with a Hit@1 of 0.961.

Table 5. Impact of learning rate and batch size on model performance (SRPRS datasets).

Batch Size \ Learning Rate	5×10^{-6}	1×10^{-5}	5×10^{-5}
16	0.948	0.954	0.932
32	0.956	0.961	0.944
64	0.949	0.957	0.936

We observe that smaller learning rates (e.g., 5×10^{-6}) result in slightly slower convergence and marginally lower performance across all batch sizes. Conversely, a larger learning rate (e.g., 5×10^{-5}) tends to degrade performance, especially with smaller batch sizes. This suggests that higher learning rates may introduce instability during optimization, leading to suboptimal results.

Regarding batch size, increasing it beyond 32 does not lead to significant improvements and, in some cases, slightly hurts performance. This may be due to the reduced frequency of weight updates and the diminishing effect of gradient noise smoothing as the batch size grows. These findings indicate that a moderate learning rate, combined with a balanced batch size, provides the optimal trade-off between convergence stability and generalization performance.

[Ⓢ] [http://dbpedia.org/resource/Neighbors_\(1920_film\)](http://dbpedia.org/resource/Neighbors_(1920_film)).

[Ⓢ] http://fr.dbpedia.org/resource/La_Voisine_de_Malec.

4.3.2 Effectiveness of Components in LMEA

LMEA comprises three feature encoders and a re-ranker. To evaluate their effectiveness, we analyzed LMEA's performance using individual encoders across all datasets. Two sets of experiments were conducted: one with LMEA operating without the re-ranker and another with the re-ranker included. Table 6 presents the results. $LMEA_{name}$, $LMEA_{attribute}$, and $LMEA_{relation}$ represent the variations of LMEA using individual encoders, while $LMEA_{full}$ refers to the full version of LMEA that integrates all three encoders.

Table 6. Ablation study.

Method	SRPRS		IDS100K		DBP1M	
	EN-FR	EN-DE	EN-FR	EN-DE	EN-FR	EN-DE
Without Re-ranker						
$LMEA_{name}$	0.918	0.945	0.889	0.864	0.845	0.855
$LMEA_{attribute}$	0.930	0.922	0.901	0.882	/	/
$LMEA_{relation}$	0.952	0.941	0.921	0.914	0.872	0.869
$LMEA_{full}$	0.961	0.953	0.930	0.921	0.901	0.897
With Re-ranker						
$LMEA_{name}$	0.952	0.971	0.905	0.879	0.873	0.875
$LMEA_{attribute}$	0.960	0.955	0.916	0.902	/	/
$LMEA_{relation}$	0.977	0.970	0.943	0.930	0.899	0.892
$LMEA_{full}$	0.983	0.977	0.956	0.929	0.920	0.915

When operating without the re-ranker, among the variations, $LMEA_{relation}$ achieves the best performance, followed by $LMEA_{attribute}$, and then $LMEA_{name}$. The full version, $LMEA_{full}$, surpasses all individual variations across all datasets when the three encoders are combined. Similar results are observed when the re-ranker is included.

From the results, we can conclude that: (1) All three encoders, utilizing different features, are effective for entity alignment. (2) The relative importance of the features is Relation > Attribute > Name. (3) The re-ranker significantly enhances performance.

4.3.3 Efficiency Analysis

Efficiency is crucial when handling large-scale entity alignment. To better understand the time consumption of LMEA, we recorded the running time for different stages of its workflow.

- **Training:** This stage involves training the encoders for entity features.
- **Tokenizing:** This stage converts the input entity features into sequences of tokens.
- **Embedding:** This stage generates embeddings from the tokenized input.
- **Searching:** This stage conducts nearest-neighbor entity searches using FAISS [25].

During training, each entity in one graph is paired with a positive alignment and n negative samples. This results in a training complexity of $O(E_1 \times (1 + n))$. After training, all $E_1 + E_2$ entities are encoded through forward passes, with a complexity of $O(E_1 + E_2)$. For candidate selection, we perform a top- k similarity search using FAISS. Assuming approximate indexing, the retrieval cost is $O(E_1 \times \log E_2 + E_1 \times k \times d)$, where d is the embedding dimension. The resulting bidirectional sparse similarity matrices are merged via summation, and both time and space complexity are bounded by $O((E_1 + E_2) \times k)$. Overall, our design balances encoding precision with efficient similarity computation, making it suitable for large-scale entity alignment.

Figure 2 illustrates the time distribution of LMEA across these stages when applied to the DBP1M dataset. The results show that encoder training is the most time-intensive process, accounting for approximately 45% and 48% of the total time on the EN-FR and EN-DE datasets, respectively. The embedding stage ranks second, consuming about 37% to 38% of the time. Tokenization accounts for around 13% to 14%, while the searching stage is the most efficient, requiring only about 2% of the total time.

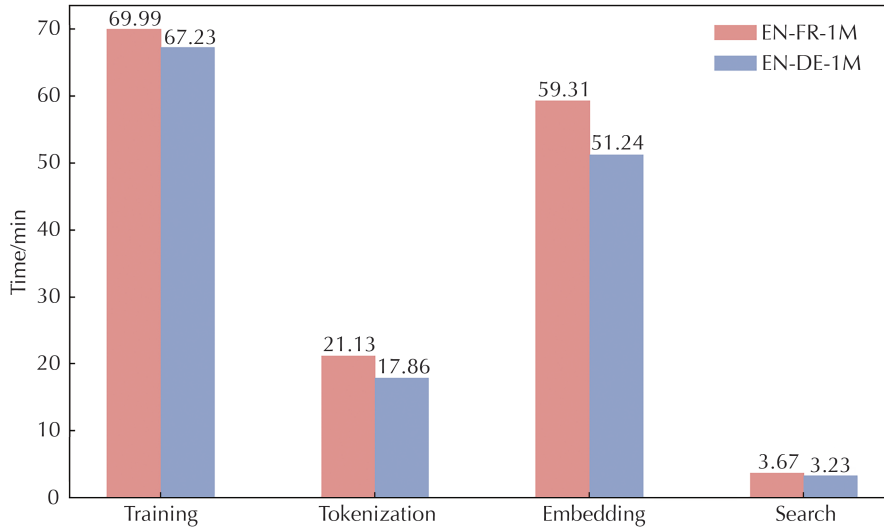


Figure 2. Running time analysis of LMEA on DBP1M datasets (minutes).

The experimental results provide valuable insights into maintaining efficiency in large-scale entity alignment. The training time largely depends on the number of training samples used in the training stage. As training is the most time-consuming stage, it is essential to keep the training data at an appropriate size. The time required for tokenization and embedding depends on the number of entities and the length of

their input features. To further reduce the time spent on these two stages, a triple selection method can be employed to identify the necessary entity triples for alignment, thereby selecting only a relevant subset of triples from the entities' input features.

4.3.4 Effectiveness of Triple Ordering Methods

We introduced four different triple ordering methods in Section 3.1.1. To evaluate their impact on alignment results, we conducted experiments comparing these methods. Figure 3 illustrates the performance of LMEA using different ordering methods on the SRPRS EN-FR and EN-DE datasets. The results indicate that the Frequency and Functionality methods consistently perform well across both datasets. In contrast, the Alphabetical and Fixed methods show strong performance on the SRPRS EN-FR dataset but perform poorly on the SRPRS EN-DE dataset. These findings demonstrate that the Frequency and Functionality methods are more robust across diverse alignment tasks. Accordingly, we adopt the Functionality method for triple ordering in the experiments presented in Section 4.2.

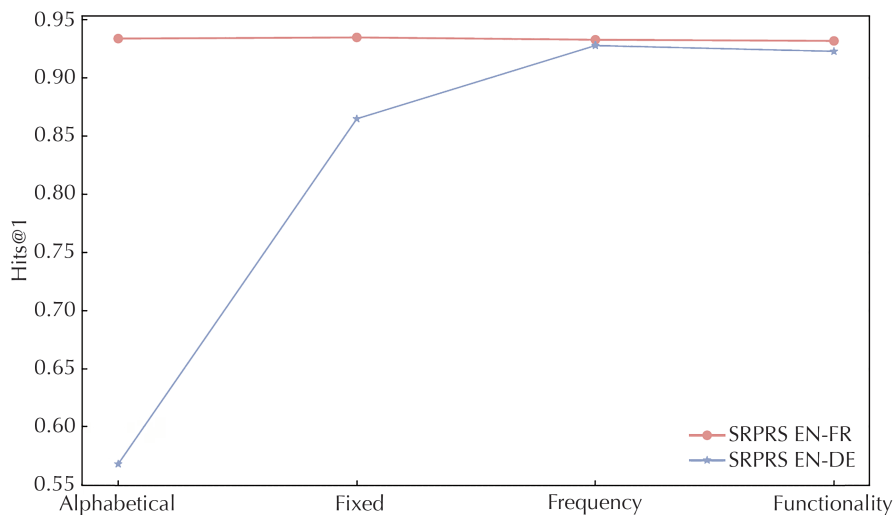


Figure 3. Comparison of results on SRPRS EN-FR dataset and SRPRS EN-DE datasets with different triple ordering methods.

4.3.5 Impact of Negative Samples

During the training of feature encoders for the entity retriever, we observed that the number of negative samples has a significant impact on alignment performance. To determine the optimal number of negative samples, we tested different ratios of negative to positive samples, ranging from 1 to 6. Figure 4 illustrates the results of LMEA trained with varying numbers of negative samples on the SRPRS EN-FR dataset. The encoders for entity names, relations, and attributes were trained separately, and the Hits@1 results on the testing datasets are presented in Figure 4.

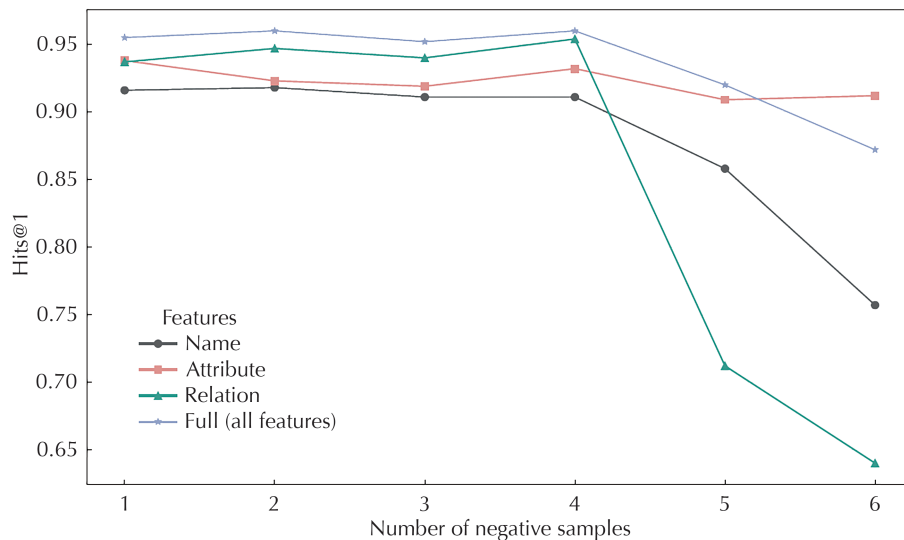


Figure 4. Alignment results based on different choices of negative sample numbers on three encoding channels of SRPRS EN-FR dataset.

The findings suggest that the optimal number of negative samples is between 1 and 4, as performance declines significantly when more than 4 negative samples are used. Based on this observation, we set the number of negative samples to 2 for all encoders in LMEA, consistent with prior studies such as BERT-INT [14].

4.3.6 Impact of Seed Alignments

Seed alignments are crucial for training the encoders in the entity retriever, affecting both the efficiency and effectiveness of entity alignment (EA). To evaluate their impact, we tested LMEA with varying numbers of seed alignments on the EN-FR-100K dataset. Figure 5 compares the Hits@1 performance of LMEA using individual encoders and the full set of encoders with different numbers of seed alignments.

The results show that as the number of seed alignments increases from 1K to 5K, the performance of LMEA and its variations improves slightly, both with and without the re-ranker. However, as the number of seed alignments increases from 5K to 20K, the improvements are not significant and consistent. For LMEA with name encoder, its performances even drop when 20k seed alignments are used. LMEA with full features also exhibits similar performance variations as the seed alignments increase.

Figure 6 illustrates the running time of LMEA with different numbers of seed alignments. It reveals that as the number of seed alignments grows, the running time of LMEA increases significantly, whether using a single feature or full features.

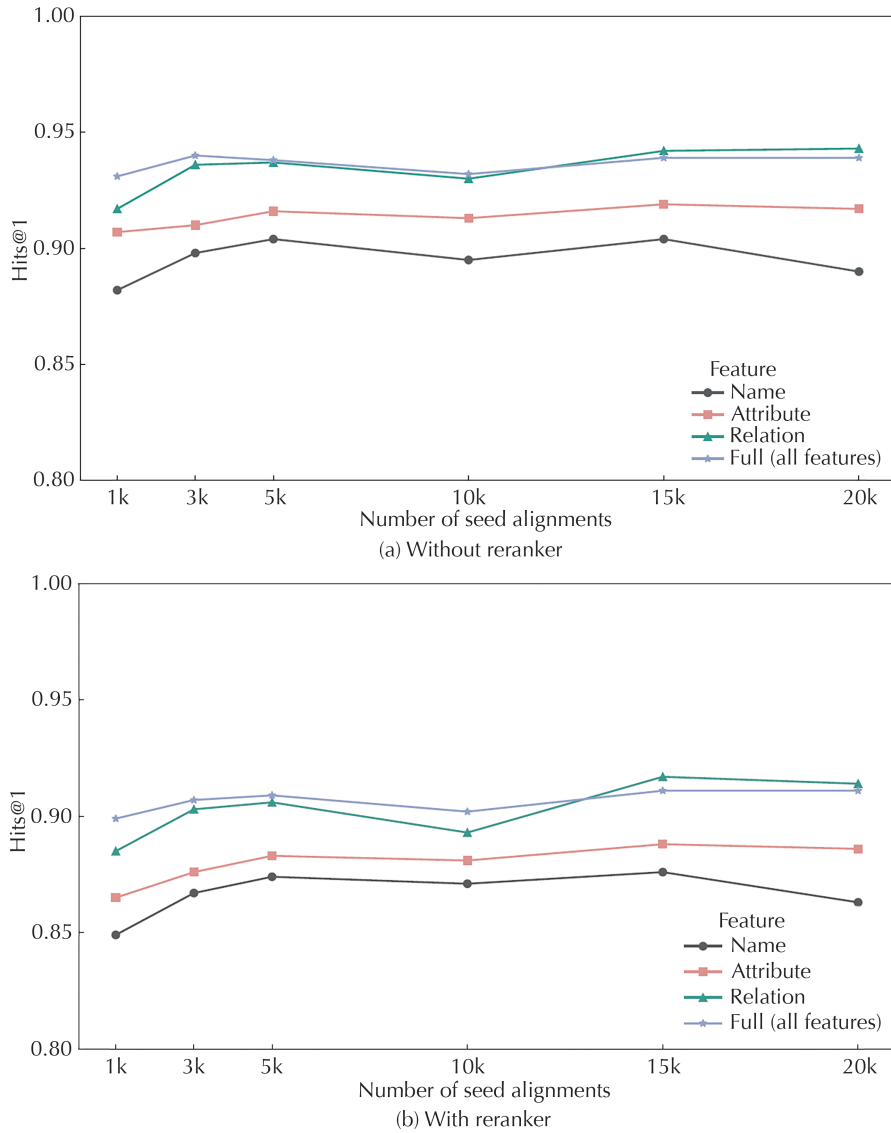


Figure 5. Results of LMEA using different number of seed alignments.

Based on the above observations, we trained encoders in LMEA with 3K seed alignments across all datasets, including those with 15K, 100K and 1M alignments. As for large-scale datasets like 100K and 1M, seed alignments account only 3% and 0.3% of the whole datasets.

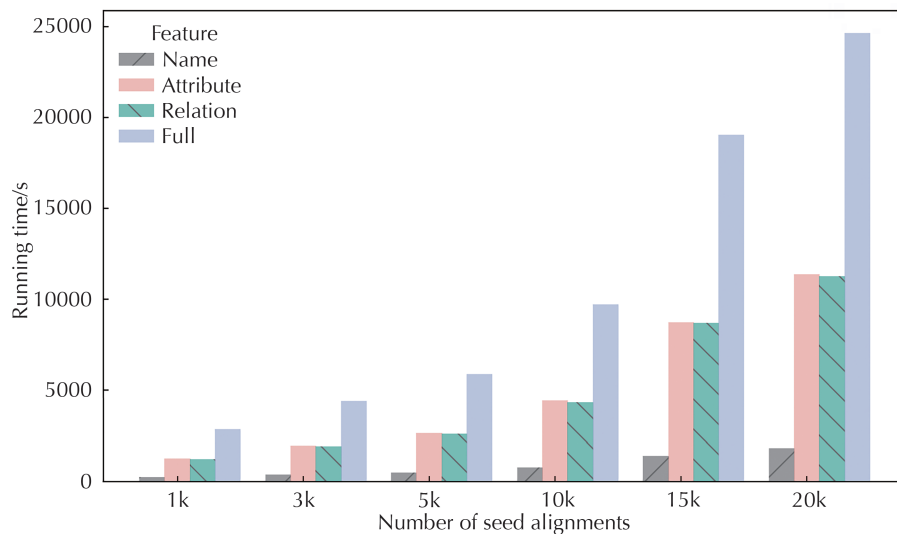


Figure 6. Running time of LMEA using different number of seed alignments.

5. RELATED WORK

5.1 Embedding-based EA

Embedding-based EA methods typically use TransE or graph neural networks to generate embeddings, which are then used to identify equivalent entities. Early approaches focused primarily on structural information for alignment, including TransE-based methods such as MTransE [5], IPTransE [36], and BootEA [24], as well as GNN-based methods like MuGNN [9], NAEA [37], RDGCN [31], and AliNet [10]. To enhance performance, several approaches incorporate entity attributes or names from the KGs. For example, JAPE [6] employs a Skip-Gram model to embed attributes by capturing their correlations within KGs. GCN-Align [8] integrates attribute information into embeddings using graph convolutional networks (GCNs). MultiKE [11] adopts a unified framework to learn embeddings from the perspectives of entity names, relations, and attributes. Similarly, CEA [13] combines structural, semantic, and string features of entities, dynamically weighting these features for improved alignment.

5.2 Language Model-based EA

With the success of pre-trained language models in various tasks, several methods have leveraged language models to capture the semantic information of entities for knowledge graph (KG) alignment. AttrGNN [12] utilizes BERT to encode entity attributes by processing each attribute and its value separately, then applying a graph attention network to compute their weighted average. BERT-INT [14] encodes entity names, descriptions, attributes, and values using a language model. It calculates entity matching scores through pairwise interactions between neighbors and attributes, but its interaction-based approach is computationally expensive, limiting its scalability to large KGs. SDEA [15] fine-tunes

BERT to encode attribute values into attribute embeddings, which are then passed through a BiGRU to generate relation embeddings for entities. TEA [34] organizes triples alphabetically by relations and attributes to form sequences and employs a textual entailment framework for alignment. It uses PLMs to predict entailment probabilities for entity pairs, but its pairwise input structure hinders scalability for large-scale KGs. AttrGNN, BERT-INT, and SDEA rely on BERT to encode entity attribute information and use GNNs to integrate relational data into entity embeddings. In contrast, our approach directly employs a language model to encode both attributes and relations simultaneously. Although TEA encodes attributes and relations in a similar way, it needs to process every possible entity pairs to get the alignment results, making it unsuitable for large-scale EA tasks.

Most recently, Large Language Models (LLMs) have also been explored for EA tasks. AutoAlign [16] extracts attribute character embeddings and predicate-proximity-graph embeddings using large language models. LLMEA [17] combines knowledge from KGs and LLMs to predict entity alignments. ChatEA [18] uses Simple-HHEA [38] to generate alignment candidates and then leverages the reasoning capabilities of LLMs to predict the final results. LLM4EA [35] uses LLMs to generate pseudo-labels of entity pairs, and then employs a probabilistic reasoning method to refine the labels. Seg-Align [39] uses SDEA to get candidate alignments, and then segments candidates, selecting suitable ones for processing by LLMs. LLM-based EA approaches exploit the reasoning abilities of LLMs to enhance entity alignment prediction. However, due to the vast number of potential alignments, these methods rely on existing EA techniques to generate alignment candidates, from which LLMs are used to select the final results. The improvements contributed by LLMs in these approaches are, however, somewhat limited.

5.3 Large-scale EA

To address the challenges of aligning large-scale knowledge graphs, several methods have been introduced in recent years. LargeEA [19] partitions large KGs into subgraphs using a mini-batch generation technique. It calculates name similarities between entities using embeddings from BERT and combines these with structural similarities to achieve complementary alignment results. LIME [20] employs a seed-oriented bidirectional partitioning strategy, where alignment results from subgraphs are aggregated to produce the final alignment outcomes. DivEA [21] adopts a progressive counterpart discovery approach to generate subgraphs with high coverage of potential alignments. LargeGNN [22] offers a scalable solution by training a GNN on subgraphs in mini-batches and performing EA using approximate k-nearest neighbor searches across the entire entity embedding space.

6. CONCLUSION

In this paper, we propose a language model based approach LMEA for large-scale entity alignment in knowledge graphs. LMEA leverages the ability of language models to encode entities' multi-view information, and employs a retrieval-reranking scheme to generate the final alignments. Experimental results show that language models can effectively encode different features of entities for the alignment

task. The proposed retrieval-reranking scheme in LMEA works effectively and efficiently in handling large-scale knowledge graphs. The experimental comparisons show that our approach LMEA can achieve the state-of-the-art results on large-scale entity alignment tasks.

AUTHOR CONTRIBUTIONS

Ningxin Chen: proposed the research problems; designed the research framework; performed the research; collected and analyzed the data; wrote and revised the manuscript.

Zhichun Wang: proposed the research problems; designed the research framework; wrote and revised the manuscript.

ACKNOWLEDGEMENTS

This work was supported by the National Natural Science Foundation of China (No. 62276026).

REFERENCES

- [1] T. Rebele, F. M. Suchanek, J. Hoffart, J. A. Biega, E. Kuzey, and G. Weikum, “Yago: A multilingual knowledge base from wikipedia, wordnet, and geonames,” in *Proceedings of the Fifteenth International Semantic Web Conference (ISWC2016)*, 2016, pp. 177–185.
- [2] C. Bizer, J. Lehmann, G. Kobilarov, S. Auer, C. Becker, R. Cyganiak, and S. Hellmann, “Dbpedia-a crystallization point for the web of data,” *Web Semantics: Science, Services and Agents on the World Wide Web*, vol. 7, no. 3, pp. 154–165, 2009.
- [3] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor, “Freebase: a collaboratively created graph database for structuring human knowledge,” in *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*. ACM, 2008, pp. 1247–1250.
- [4] D. Vrandečić and M. Krötzsch, “Wikidata: a free collaborative knowledgebase,” *Commun. ACM*, vol. 57, no. 10, p. 78–85, sep 2014. [Online]. Available: <https://doi.org/10.1145/2629489>.
- [5] M. Chen, Y. Tian, M. Yang, and C. Zaniolo, “Multilingual knowledge graph embeddings for cross-lingual knowledge alignment,” in *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence (AAAI2017)*, 2017, pp. 1511–1517.
- [6] Z. Sun, W. Hu, and C. Li, “Cross-lingual entity alignment via joint attribute-preserving embedding,” in *Proceedings of the Sixteenth International Semantic Web Conference (ISWC2017)*, 2017, pp. 628–644.
- [7] Z. Sun, W. Hu, Q. Zhang, and Y. Qu, “Bootstrapping entity alignment with knowledge graph embedding,” in *International Joint Conference on Artificial Intelligence*, 2018, pp. 4396–4402.
- [8] Z. Wang, Q. Lv, X. Lan, and Y. Zhang, “Cross-lingual knowledge graph alignment via graph convolutional networks,” in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP2018)*, 2018, pp. 349–357.
- [9] Y. Cao, Z. Liu, C. Li, Z. Liu, J. Li, and T.-S. Chua, “Multi-channel graph neural network for entity alignment,” in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL2019)*, Jul. 2019, pp. 1452–1461.

- [10] Z. Sun, C. Wang, W. Hu, M. Chen, J. Dai, W. Zhang, Y. Qu, "Knowledge graph alignment network with gated multi-hop neighborhood aggregation," in *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence (AAAI2020)*, 2020. vol. 34(01), pp. 222–229.
- [11] Q. Zhang, Z. Sun, W. Hu, M. Chen, L. Guo, and Y. Qu, "Multi-view knowledge graph embedding for entity alignment," in *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence (IJCAI2019)*, 7 2019, pp. 5429–5435.
- [12] Z. Liu, Y. Cao, L. Pan, J. Li, Z. Liu, and T.-S. Chua, "Exploring and evaluating attributes, values, and structures for entity alignment," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, B. Webber, T. Cohn, Y. He, and Y. Liu, Eds. Online: Association for Computational Linguistics, Nov. 2020, pp. 6355–6364. [Online]. Available: <https://aclanthology.org/2020.emnlp-main.515>.
- [13] W. Zeng, X. Zhao, J. Tang, and X. Lin, "Collective entity alignment via adaptive features," in *2020 IEEE 36th International Conference on Data Engineering (ICDE2020)*, 2020, pp. 1870–1873.
- [14] X. Tang, J. Zhang, B. Chen, Y. Yang, H. Chen, and C. Li, "Bert-int: A bert-based interaction model for knowledge graph alignment," in *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*, ser. IJCAI'20, 2021, pp. 3174–3180.
- [15] Z. Zhong, M. Zhang, J. Fan, and C. Dou, "Semantics Driven Embedding Learning for Effective Entity Alignment," in *2022 IEEE 38th International Conference on Data Engineering (ICDE)*, May 2022, pp. 2127–2140, ISSN: 2375–026X. [Online]. Available: <https://ieeexplore.ieee.org/document/9835701>.
- [16] R. Zhang, Y. Su, B. D. Trisedya, X. Zhao, M. Yang, H. Cheng, and J. Qi, "Autoalign: Fully automatic and effective knowledge graph alignment enabled by large language models," *IEEE Transactions on Knowledge and Data Engineering*, pp. 1–14, 2023.
- [17] L. Yang, H. Chen, X. Wang, J. Yang, F.-Y. Wang, and H. Liu, "Two heads are better than one: Integrating knowledge from knowledge graphs and large language models for entity alignment," *ArXiv*, vol. abs/2401.16960, 2024.
- [18] X. Jiang, Y. Shen, Z. Shi, C. Xu, W. Li, Z. Li, J. Guo, H. Shen, and Y. Wang, "Unlocking the power of large language models for entity alignment," in *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, L.-W. Ku, A. Martins, and V. Srikumar, Eds. Bangkok, Thailand: Association for Computational Linguistics, Aug. 2024, pp. 7566–7583. [Online]. Available: <https://aclanthology.org/2024.acl-long.408/>.
- [19] C. Ge, X. Liu, L. Chen, Y. Gao, and B. Zheng, "Largeea: Aligning entities for large-scale knowledge graphs," *Proc. VLDB Endow.*, vol. 15, no. 2, pp. 237–245, oct 2021. [Online]. Available: <https://doi.org/10.14778/3489496.3489504>.
- [20] W. Zeng, X. Zhao, X. Li, J. Tang, and W. Wang, "On entity alignment at scale," *The VLDB Journal*, vol. 31, no. 5, pp. 1009–1033, sep 2022. [Online]. Available: <https://doi.org/10.1007/s00778-021-00703-3>.
- [21] B. Liu, W. Hua, G. Zuccon, G. Zhao, and X. Zhang, "High-quality task division for large-scale entity alignment," *ArXiv*, vol. abs/2208.10366, 2022.
- [22] K. Xin, Z. Sun, W. Hua, W. Hu, J. Qu, and X. Zhou, "Large-scale entity alignment via knowledge graph merging, partitioning and embedding," *ArXiv*, vol. abs/2208.11125, 2022.
- [23] F. M. Suchanek, S. Abiteboul, and P. Senellart, "Paris: probabilistic alignment of relations, instances, and schema," *Proc. VLDB Endow.*, vol. 5, no. 3, p. 157–168, Nov. 2011. [Online]. Available: <https://doi.org/10.14778/2078331.2078332>.
- [24] Z. Sun, W. Hu, Q. Zhang, and Y. Qu, "Bootstrapping entity alignment with knowledge graph embedding," in *Proceedings of the Twenty-Seventh international joint conference on Artificial Intelligence (IJCAI2018)*, 2018, pp. 4396–4402.

- [25] J. Johnson, M. Douze, and H. Jégou, “Billion-scale similarity search with GPUs,” *IEEE Transactions on Big Data*, vol. 7, no. 3, pp. 535–547, 2019.
- [26] X. Mao, W. Wang, Y. Wu, and M. Lan, “From alignment to assignment: Frustratingly simple unsupervised entity alignment,” in *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, M.-F. Moens, X. Huang, L. Specia, and S. W.-t. Yih, Eds. Online and Punta Cana, Dominican Republic: Association for Computational Linguistics, Nov. 2021, pp. 2843–2853. [Online]. Available: <https://aclanthology.org/2021.emnlp-main.226>.
- [27] X. Mao, W. Wang, and M. Lan, “LightEA: A scalable, robust, and interpretable entity alignment framework via three-view label propagation,” in *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*. Abu Dhabi, United Arab Emirates: Association for Computational Linguistics, Dec. 2022, pp. 825–838. [Online]. Available: <https://aclanthology.org/2022.emnlp-main.52>.
- [28] M. Cuturi, “Sinkhorn distances: lightspeed computation of optimal transport,” in *Proceedings of the 26th International Conference on Neural Information Processing Systems-Volume 2*, ser. NIPS’13. Red Hook, NY, USA: Curran Associates Inc., 2013, pp. 2292–2300.
- [29] L. Guo, Z. Sun, and W. Hu, “Learning to exploit long-term relational dependencies in knowledge graphs,” in *International Conference on Machine Learning*, PMLR 97: 2505–2514.
- [30] Z. Sun, Q. Zhang, W. Hu, C. Wang, M. Chen, F. Akrami, and C. Li, “A benchmarking study of embedding-based entity alignment for knowledge graphs,” *Proceedings of the VLDB Endowment*, vol. 13, no. 11, pp. 2326–2340, 2020. [Online]. Available: <http://www.vldb.org/pvldb/vol13/p2326-sun.pdf>.
- [31] Y. Wu, X. Liu, Y. Feng, Z. Wang, R. Yan, and D. Zhao, “Relation-aware entity alignment for heterogeneous knowledge graphs,” in *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence (IJCAI2019)*, 7 2019, pp. 5278–5284.
- [32] K. Xu, L. Wang, M. Yu, Y. Feng, Y. Song, Z. Wang, and D. Yu, “Cross-lingual knowledge graph alignment via graph matching neural network,” in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, A. Korhonen, D. Traum, and L. Màrquez, Eds. Florence, Italy: Association for Computational Linguistics, Jul. 2019, pp. 3156–3161. [Online]. Available: <https://aclanthology.org/P19-1304>.
- [33] C. Ge, X. Liu, L. Chen, B. Zheng, and Y. Gao, “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*, ser. SIGIR’21. New York, NY, USA: Association for Computing Machinery, 2021, p. 777–786. [Online]. Available: <https://doi.org/10.1145/3404835.3462870>.
- [34] Y. Zhao, Y. Wu, X. Cai, Y. Zhang, H. Zhang, and X. Yuan, “From Alignment to Entailment: A Unified Textual Entailment Framework for Entity Alignment,” in *Findings of the Association for Computational Linguistics: ACL 2023*, A. Rogers, J. Boyd-Graber, and N. Okazaki, Eds. Toronto, Canada: Association for Computational Linguistics, Jul. 2023, pp. 8795–8806. [Online]. Available: <https://aclanthology.org/2023.findings-acl.559>.
- [35] S. Chen, Q. Zhang, J. Dong, W. Hua, Q. Li, and X. Huang, “Entity alignment with noisy annotations from large language models,” in *Proceedings of the Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. [Online]. Available: <https://arxiv.org/abs/2405.16806>.
- [36] H. Zhu, R. Xie, Z. Liu, and M. Sun, “Iterative entity alignment via joint knowledge embeddings,” in *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence (IJCAI2017)*, 2017, pp. 4258–4264.
- [37] Q. Zhu, X. Zhou, J. Wu, J. Tan, and L. Guo, “Neighborhood-aware attentional representation for multilingual knowledge graphs,” in *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence (IJCAI2019)*, 7 2019, pp. 1943–1949.

- [38] X. Jiang, C. Xu, Y. Shen, Y. Wang, F. Su, Z. Shi, F. Sun, Z. Li, J. Guo, and H. Shen, "Toward practical entity alignment method design: Insights from new highly heterogeneous knowledge graph datasets," in *Proceedings of the ACM on Web Conference 2024*, ser. WWW'24. New York, NY, USA: Association for Computing Machinery, 2024, pp. 2325–2336. [Online]. Available: <https://doi.org/10.1145/3589334.3645720>.
- [39] L. Yang, J. Cheng, and F. Zhang, "Advancing cross-lingual entity alignment with large language models: Tailored sample segmentation and zero-shot prompts," in *Findings of the Association for Computational Linguistics: EMNLP 2024*, Y. Al-Onaizan, M. Bansal, and Y.-N. Chen, Eds. Miami, Florida, USA: Association for Computational Linguistics, Nov. 2024, pp. 8122–8138. [Online]. Available: <https://aclanthology.org/2024.findings-emnlp.475/>.

AUTHOR BIOGRAPHY



Ningxin Chen is currently a master student at School of Artificial Intelligence, Beijing Normal University. She received her bachelor degree from Beijing Normal University in 2023. Her research interests include knowledge graph, and language models.



Zhichun Wang is currently an associate professor at the School of Artificial Intelligence, Beijing Normal University. He received his Ph.D. degree from Tianjin University in 2010, and worked as post-doctoral researcher in Tsinghua University from 2011 to 2012. His research interests include large language models and knowledge graphs.