An Analysis of the Performance of Representation Learning Methods for Entity Alignment: Benchmark vs. Real-world Data

Ensiyeh Raoufi a,*, Bill Gates Happi b, Pierre Larmande b, François Scharffe a and Konstantin Todorov a

aLIRMM, University of Montpellier, CNRS, France
E-mails: ensiyeh.raoufi@lirmm.fr, francois.scharffe@lirmm.fr, konstantin.todorov@lirmm.fr
bDIADE, IRD, CIRAD, University of Montpellier, France
E-mails: bill.happi@ird.fr, pierre.larmande@ird.fr

Abstract.

Representation learning for Entity Alignment (EA) aims to map, across two Knowledge Graphs (KG), distinct entities that correspond to the same real-world object using an embedding space. The similarity of the entities can be measured based on the similarity of the learned embeddings, which serves as a proxy for that of the real-world objects. Although many embedding-based models show very good performance on certain synthetic benchmark datasets, benchmark overfitting limits the applicability of these methods in real-world scenarios where we deal with highly heterogeneous, incomplete, and domain-specific data. While there have been efforts to create benchmark datasets reflecting as much as possible real-world scenarios, there has been no comprehensive analysis and comparison between the performance of methods on synthetic benchmark and real-world heterogeneous datasets. In addition, most existing models report their performance by excluding from the alignment candidate search space entities that are not part of the validation data. This under-represents the knowledge and the data contained in the KGs, limiting the ability of these models to find new alignments in large-scale KGs. We analyze models with competitive performance on widely used synthetic benchmark datasets, such as the cross-lingual DBP15K. We compare the performance of the selected models on real-world heterogeneous datasets beyond DBP15K and we show that most of the current approaches are not effectively capable of discovering mappings between entities in the real world, due to the above-mentioned drawbacks. We compare the utilized methods from different aspects and measure joint semantic similarity and profiling properties of the KGs to explain the models’ performance drop on real-world datasets. Furthermore, we show how tuning the EA models by restricting the search space only to validation data affects the models’ performance and causes them to face generalization issues. By addressing practical challenges in applying EA models to heterogeneous datasets and providing valuable insights for future research, we signal the need for more robust solutions in real-world applications.

Keywords: Entity alignment, knowledge graphs, representation learning, knowledge graph heterogeneity

1. Introduction

Knowledge Graphs (KG) have emerged as powerful tools for a range of applications, including information retrieval, question answering, and federating data [1]. An entity in a knowledge graph refers to a distinct and identifiable concept, which can be a concrete object, an abstract idea, or even an event. In the context of a knowledge
graph, entities are represented as nodes, forming the building blocks of the graph. These nodes encapsulate entities such as people, places, or things. The relationships between entities in a knowledge graph are seen as edges, serving as connections that establish associations or interactions between the nodes. These edges convey various meanings, representing the attribute properties of the entities or relation properties between entities. Examples of attribute properties include "birth date", "genre" or "description" and examples of relation properties include "located in", "established by" or "worked with". Based on the given definition, (Person1, birth date, "1989-09-30") indicates an attribute triple, and (City1, located in, Country1) indicates a relation triple. Essentially, the combination of entities and their interconnecting relationships forms a structured representation of knowledge. Hence, knowledge graphs have been designed to store knowledge by organizing and interconnecting information in a way that facilitates semantic understanding of data and reasoning over it. They are widely used to represent and model information in various domains, including but not limited to the semantic web [2–4], cultural heritage [5–8], biomedicine [9–12], sociology [13–15], and data-driven industries [16, 17]. As data originates from diverse sources, it is dispersed across various knowledge graphs, presenting a variety of challenges, one of which is that of Entity Alignment (EA). EA is a process within the realm of entity alignment which is defined as the task of identifying and linking entities from a source KG to their counterparts in a target KG that refer to the same real-world object [18]. Entity alignment involves establishing correspondences or alignments between entities in different knowledge graphs that share similar or identical meanings. This facilitates tasks such as data integration, information retrieval, or entity disambiguation [19–21] across heterogeneous knowledge sources.

We use the term KG heterogeneity in the sense of [22], common in the fields of ontology and entity alignment. In a nutshell, it concerns any difference in the expression of a given piece of knowledge across two KGs (be it structural, syntactical, terminological, or other). Sticking to the data heterogeneity concept provided in [22] and algebraic properties of the graphs, in this paper, we denote differences in value and structural levels of the KGs in each dataset. By dataset, we mean an entity alignment dataset which is a pair source and target KGs to be linked with reference alignments. Reference or seed alignment is a manually curated set of correspondences or alignments (often together with a confidence score) between entities from the two different KGs. A synthetic benchmark dataset refers to a dataset that is generated by selecting multiple entities from a source Knowledge Graph (KG) that has no matches within itself. These selected entities are then matched with their corresponding counterparts in the target KG using a 1-to-1 assumption. In this paper, we use the benchmark dataset and synthetic benchmark dataset interchangeably. The pair of KGs in the datasets might differ in the graph’s structural properties such as size, degree distribution, etc. Also, each pair of aligned entities in two KGs may have descriptive and data quality heterogeneities, following [22].

Entity alignment techniques across KGs could be categorized into two main groups: (1) Non-embedding-based approaches apply user-crafted representations of entities and links to align the entities across the KGs based on similarity measures or logic axioms. This group of approaches prioritizes symbolic reasoning, logical inferences and linking specifications defined by domain experts to guide the alignment process. (2) Embedding-based approaches, in contrast, automatically represent the entities in a feature space and predict the alignments based on similarity metrics over the learnt embeddings. Embedding refers to representing an object as a vector in a continuous space based on a given number of constraints (e.g. close in meaning entities should have vectors that appear close in the embedding space).

Embedding-based EA models use an alignment module to infer alignment relationships between entities. While the embedding module represents each KG entity as a vector in a low-dimensional embedding space, the alignment module ensures that aligned entities are close together in a unique embedding space or learns a mapping between KGs with respect to the reference alignment. During the model’s training phase, by an iterative interaction between embedding and alignment modules, all entities of the two KGs would be embedded and, using the model, one can predict which entities will most likely align.

Utilizing non-embedding-based methods may be more suitable in scenarios where dealing with sparse or small datasets, or in situations where there are not much variety of problems such as predicate, class, or graph linking problems (see [23]). Real-world datasets often do not meet these ideal conditions and therefore, embedding-based approaches are more suitable. However, in this work, we used knowledge graphs represented in the RDF format, the research is not restricted to this format and applies to property graphs as well.

---

1 While this work was conducted with knowledge graphs represented in the RDF format, the research is not restricted to this format and applies to property graphs as well.
EA methods are more efficient for several reasons such as flexibility in cross-lingual alignment, scalability to large KGs, and global consistency in representations across the entire KG.

In this paper, we focus on analyzing the embedding-based EA models. As a baseline non-embeddings-based method, we include the results of the DLinker [24] system on our datasets, for reasons explained in Section 3. We collect the performance of several recent EA embedding-based models having state-of-the-art performance on synthetic benchmark datasets and analyze the capacity of these methods to produce high-quality alignments when they deal with heterogeneous data, showing a considerable drop in performance in these scenarios. We analyze and compare the real-world and synthetic benchmark datasets with respect to a set of dataset profiling features [25] studied and applied for the entity alignment task. [26] discussed that cutting-edge entity alignment models did not address the particular properties of data well because they prioritized genericity and automation. Indeed, the results of our study show how embedding-based models excel on certain synthetic benchmark datasets but encounter challenges in real-world situations due to neglecting to adequately consider the underlying data’s specific characteristics and inherent nature.

There have been numerous surveys and analysis studies conducted on entity alignment techniques [27–29]. Also, a variety of methods have been designed to align the entities of two KGs using graph embeddings-based approaches [30]. Several papers analyzed the performance of these representation learning-based entity alignment models [31–33] and compared them regarding their performance on certain benchmark datasets [27, 34–39]. Some research has been done on extracting more realistic EA benchmark datasets [34] from large knowledge bases like DBpedia [40]. In a different way to the cited studies, in this paper, we focus on analyzing the performance of embedding-based EA methods having different representation learning principles on real-world datasets together with presenting an in-depth analysis of the dataset features. Our main contributions are:

– Describing frameworks and comparison of EA methods having different embedding bases.
– Comparing features of the synthetic benchmark and real-world datasets from aspects related to entity alignment.
– Analyzing and explaining the performance drop of EA methods on real-world datasets in comparison to their performance on established synthetic benchmarks, presenting shreds of evidence and probable reasons to explain the observed gap in performance.

The findings of this study can guide future research in EA model development. Researchers and practitioners can use the insights gained from the analysis to enhance existing models, design new models, or develop strategies to improve alignment performance in the presence of real-world heterogeneous data.

The rest of the paper is structured as follows. In Section 2, we present a summary of other surveys over EA methods and how their research focus is different from ours. In Section 3, we describe the embedding-based EA methods that we select to analyze their performance and we prepare a comparison Table on the methods. In section 4, we analyze several relevant real-world datasets and present intuitions on how those are more heterogeneous compared to synthetic benchmark datasets. Our studies in Section 5 extend the limited research on reporting the performance of the existing EA models on real-world heterogeneous data, and investigate the reasons for the performance drop of different models on real-world datasets.

2. Related Work

Several studies have contributed to the understanding and advancement of KG embeddings and their applications such as link prediction, knowledge graph completion and reasoning, and entity alignment [1, 30, 41–43]. While [41] reveals sharp differences in the geometry of embeddings produced by various KG embedding methods, [44] introduces a multi-embedding interaction mechanism for analyzing KG embedding models like DistMult [45] and ComplEx [46]. The latter study unifies and generalizes these models, offering an intuitive perspective for effective use of them. The authors of [47] introduce a scalable and open-source Python library for multi-source knowledge graph embeddings. Supporting joint representation learning, it implemented 26 KG embedding models and 16 benchmark datasets. Moreover, [48] categorizes the existing KG Embedding (KGE) models based on representation spaces and discusses whether they have algebraic, geometric, or analytical structures. The focus of our empirical
study falling on embedding methods for the specific task of Entity Alignment, we describe in what follows work related to this task in detail.

There have been multiple surveys and experimental studies conducted on methods designed for entity alignment across knowledge graphs [36, 49, 50]. The study by Sun et al. [34] classifies EA methods from different aspects. As part of this study, they created an open-source toolkit, named OpenEA, designed for entity alignment using embedding-based methods. Focusing on embedding-based EA frameworks, the authors discuss their characteristics and functionalities, highlighting how these methods predict corresponding entities through nearest-neighbor searches among target entity embeddings. Two combination paradigms are outlined: one encoding KGs in independent spaces and learning a mapping using seed alignment, and another representing KGs in a unified space considering highly similar embeddings for aligned entities. The study underscores the incorporation of entity relation and attribute properties into embedding modules to enhance accuracy, categorizing relation embeddings into triple-based, path-based, and neighborhood-based groups. Attribute embedding, achieved through correlation or literal methods, is also explored for improving entity similarity measures.

Zhang et al. [35] offer a tutorial-style survey on entity alignment techniques using representation learning. They introduce a framework for understanding these techniques and additionally, they propose a new benchmark to address existing limitations. The framework highlights the importance of attribute triples and relation predicates. The study also explores entity alignment on large-scale knowledge graphs like Wikidata [51] and Freebase [52], providing insights and addressing limitations in existing benchmark datasets.

Jiang et al. [39] examine the performance of EA methods on highly heterogeneous KGs that differ in scale and structure, sharing fewer overlapping entities. In their study, they proposed two more realistic datasets called ICEWS-WIKI and ICEWS-YAGO that have been generated from two knowledge bases having highly different degree distributions. These new datasets deviate from the 1-to-1 assumption, where each entity in a KG must have a counterpart in the second KG.

In [27], Zeng et al. provide a brief overview of research in entity alignment, covering traditional methods, knowledge representation learning, and alignment based on representation learning in knowledge graphs. They emphasize that representation learning remains the mainstream approach. To address fairness issues in cutting-edge methods, they open-source a user-friendly KG entity alignment toolkit, organized in a modular form for reusability and scalability.

Overall, in these studies entity alignment KGE techniques are often categorized into two main groups: embedding-based and traditional methods [27, 34, 35]. Traditional EA approaches are based on user-extracted rules and OWL reasoning or use similarity computation based on symbolic features of entities. We refer to these approaches as non-embedding-based methods.

In this paper, our emphasis is on evaluating the performance of embedding-based EA methods with distinct representation learning principles on real-world and benchmark datasets. By placing a significant emphasis on real-world challenges, our study also stands out for its attention to data quality considerations. Data quality is generally conceived as fitness for use [25], that is, the ability of the data to meet the needs of the user given a particular use case. To allow for a fair comparison and to have evaluation criteria concerning the entity alignment task, we compute several structural and qualitative joint features of each pair of KGs in datasets. Hence, the singularity of our work with respect to related studies is that it goes beyond benchmark evaluations to address the practical challenges of applying EA models to heterogeneous datasets. The identified drop in performance serves as a valuable signal for the research community, indicating the need for more robust and adaptable entity alignment solutions in real-world applications.

3. Entity Alignment via Representation Learning Methods

As mentioned above, we consider two main groups of entity alignment methods: non-embedding-based and embedding-based methods. Since our focus falls on studying the latter, due to their wide use and high performance, we take one single example of the first group to use as a baseline for comparison. One of the best-performing systems
on the OAEI\(^2\) campaign from the non-embedding-based methods is LogMAP [53], receiving a Ten-Year Award for its significant impact from the International Semantic Web Conference a decade ago. However, as a representative method of the non-embedding-based group, we chose DLinker [24] because of its excellent results and the fact that it is developed in our team and thus facilitates further experiments. DLinker provides close enough results to other similar systems such as LogMap on several OAEI entity linking (instance matching) tracks to represent a good enough baseline. DLinker applies an average aggregation between the similarity measures derived from the instance objects calculated by the longest common sub-sequence algorithm [24].

Certain studies categorize embedding-based methods based on their use of semantic information to represent the KGs [54]. Other studies classify KG embedding methods based on whether they employ attribute or relation predicates for embedding, their alignment modules (whether they represent both of the KGs in the same embedding space or not), their learning strategy (supervised, semi-supervised, or unsupervised)[34, 36]. We observe that embedding-based EA methods differ in the way they embed the knowledge graphs and this difference in the framework of the embedding module is an important criterion for categorizing these methods. Based on recent studies [27, 34–36, 39, 54] and our analysis, we propose to classify the embedding-based EA models into four groups: (1) Translational, (2) GNN-based, (3) Graph Transformers-based, and (4) co-training-based.

Several entity alignment models such as MTransE [55] and IPTransE [56] have been designed by using translational techniques like TransE [57] for KG embedding and entity alignment across KGs [58]. In the translational model’s framework, a relation predicate is embedded as a translation vector from a head entity to a tail. Also, a variety of approaches like GCN-Align [59] and GMNN [60] employ GNNs [61–63] to represent the graphs and link them. These models use a GNN message-passing system to integrate the information of each entity’s neighbors. Following the successful application of the Transformer [64] model in representing sequences for the automatic translation task [65], several research works used Graph Transformers to represent the graphs [66–70]. Recently, models built on top of Transformers have been used to embed KGs for the task of entity alignment [71, 72]. Models using Graph Transformers (GT) adopt the self-attention mechanism from Transformers to represent entities in KGs. As a point of comparison between GNNs and Transformers, we should mention that Transformers use multi-head attention, treating the entire sequence as the local neighborhood, whereas standard GNNs aggregate features from local nodes [73]. The motivation behind applying Transformer architecture to GNNs, as in the Graph Transformers approaches, lies in overcoming the issue of information dispersion between distant elements in structural data [74]. Graph Transformers address the limitations of traditional GNNs by leveraging Transformers’ ability to capture long-term dependencies. By integrating GNNs and Transformers, GTs expand the receptive field of GNNs, effectively utilize graph structure information, and establish a collaborative framework where each module reinforces the other’s strengths [75].

Finally, there are methods that some papers mention as others [39] but we identify them and include them in a group having a common important characteristic which is applying a co-training of the two KGs’ embeddings [54, 76–78]. Methods in the three other groups, embed the two KGs separately and then try to minimize the distance between the embeddings of the aligned pairs of entities in the two KGs during the training and validation phases. Therefore, they must embed the whole KGs for their final prediction, which estimates the entity in the target KG that is closest (among all other target KG’s entities) to a specific entity in the source KG. KG’s co-training models instead simultaneously train and embed each pair of entities belonging to the source and target KG. The final predictions are based on a threshold, either automatically or manually set. If the final embeddings of a pair of entities belonging to the source and target KGs are closer than the threshold, then the entities are aligned together. These models do not need to embed entire KGs which this feature makes their inference more adaptable to unseen data. Inference refers to the process of applying a trained model to make predictions or decisions based on new data. The main framework of co-training results in a joint embedding of the pairs of entities belonging to the two different KGs. These methods might use translational or GNN or any other basic model to initially embed the entities but in contrast to the three other groups, these models can provide insights on the correlation between the features of entities belonging to two KGs, whether they are aligned or not. We refer to this group as the KGs co-training group.

\(^2\)https://oaei.ontologymatching.org/
Table 1

<table>
<thead>
<tr>
<th>Method</th>
<th>KG embedding approach</th>
<th>Best-evaluated benchmark dataset</th>
<th>Hit@1 on benchmark dataset</th>
<th>Input features</th>
</tr>
</thead>
<tbody>
<tr>
<td>MultiKE</td>
<td>Translational</td>
<td>DBP_WD_100K</td>
<td>0.918</td>
<td>Relation name</td>
</tr>
<tr>
<td>RDGCN</td>
<td>GNN</td>
<td>DBP15K</td>
<td>0.886 (on FR-EN)</td>
<td>Attr name/value</td>
</tr>
<tr>
<td>i-Align</td>
<td>Graph Transformer</td>
<td>DBP_YG_15K</td>
<td>0.912</td>
<td>Entity name</td>
</tr>
<tr>
<td>BERT-INT</td>
<td>KGs co-training</td>
<td>DBP15K</td>
<td>0.992 (on FR-EN)</td>
<td>Relation name</td>
</tr>
</tbody>
</table>

After analyzing many comparative studies on benchmark datasets, we focused on the following recently-proposed embedding-based EA methods to represent each of the four groups we outline above: MultiKE [45] translational model, RDGCN [79] GNN-based model, i-Align [72] GT-based model, and BERT-INT [76] co-training-based model. We choose these state-of-the-art models because they are scalable to run on real-world large KGs and have state-of-the-art performance on the benchmark dataset [39]. We give more detail on each of them in the following.

MultiKE augments what the number of relation triples by replacing the head and tail entities with their aligned instances. Then, it represents each entity and relation using a variant of TransE. To have the final EA predictions, it feeds the representations together with the encoded local names of entities and predicates to a Convolutional Neural Network (CNN) model [45].

BERT-INT uses the description or names/labels of the entities for its first phase of initial embedding of entities using a pre-trained BERT-based model and then, it creates a similarity matrix over the initial embeddings of each pair of training entities. Then, it creates a neighborhood similarity matrix for co-training each pair of entities in the candidate set. For co-training the KGs structure embedding, this method just relies on the direct neighborhood of entities. It then aggregates all the vectors obtained by the similarity matrices to represent each pair of entities and then finalizes the entity pair representations using a Multi-Layer Perceptron (MLP).

RDGCN leverages the information of relations into entity representations employing a two-step process. First, a dual relation graph is constructed based on the input KG (the context graph), which is nothing but the line graph of the context graph. In the dual graph, each node represents a type of relation and two nodes are connected together if they have a common head or tail in the main KG. Then, a graph attention mechanism (GAT) is applied to arouse interactions between the two graphs. The resulting vertex representations in the context graph are fed to GCN layers to capture the graph’s structural information through a message-passing system. In the last step, the obtained entity representations are used for aligning pairs of entities.

i-Align uses two transformer-based architectures to represent the entities based on their graph structures and textual attribute values. The model uses a graph encoder to aggregate the entities’ structural information that can also effectively handle large KGs. The model’s other transformer obtains the interaction between the entity attributes using the embeddings of attribute keys and values as inputs. i-Align provides explanations of the alignment results in the form of a set of the most influential attribute predicates and entity neighbors based on the attention weights of its two transformers.

We summarize the main properties of these four methods in Table 1, showing how the methods represent the KGs’ structures using different architectures. As we can see in Table 1, by having a Hit@1 of more than 88% for all the methods, the results of the MultiKE and i-Align methods outperform the baseline methods on the DBP_WD_100K [58] and DBP_YG_15K [35] benchmark datasets, respectively, while the performance of BERT-INT and RDGCN exceed previous EA methods on the DBP15K [31] dataset. All four methods use entity names for embedding as an extra, i-Align utilizes the attribute predicate’s names and values in its embedding procedure. To use the maximum descriptive information of entities, BERT-INT employs the entity’s descriptions instead of their names when such descriptions exist.

---

3The line graph of an undirected graph G is another graph that represents the adjacencies between edges of G. The line graph of the given graph G is constructed by making a node (vertex) instead of each edge in G. Then, for every two edges in G that have a vertex in common, we make an edge between their corresponding vertices in the line graph of G. See https://en.wikipedia.org/wiki/Line_graph
In the sequel, we extract the features of KGs in every dataset that we have used and compare the datasets from heterogeneity aspects. Then, we present an analysis of the performance of the selected methods alongside the results of the DLinker non-embedding-based EA system [24] on our datasets.

4. Datasets and Their Heterogeneity Aspects

We used several datasets having different heterogeneity aspects to test and compare the performances of the four embedding-based EA methods that we consider. In this section, after giving a summary of these datasets, we study the degrees of their heterogeneities using specific metrics that we introduce and describe. Analysing the datasets by the help of these metrics, we found them to be sufficiently diverse in terms of their heterogeneity aspects. Hence, we believe the analysis of the performance of the models on this particular collection of datasets would give us the adequate insights into the performance of the models when dealing with datasets with different heterogeneities.

4.1. Datasets

We proceed to present and analyze the datasets that we use in our experiments. We consider DBP15K, which is a benchmark dataset that a significant number of state-of-the-art methods report their results on [80]. DBP15K consists of three pairs of KGs that differ in the used language (French, Japanese, and Chinese). We pick the French-English dataset (DBP15K_{FR–EN}) as a sample of the whole. Furthermore, we consider SPIMBENCH, the OAEI reference instance matching dataset, which is much smaller than DBP15K but similar to it in several heterogeneity aspects that we will discuss later in this section.

Because heterogeneity of KGs has a broader meaning than linguistic differences [22] and also, benchmarks often present idealized scenarios with a limited set of relationships, controlled noise, and specific characteristics [81], we investigated two other real-world datasets: DOREMUS [5], and AGROLD [82] that differ from benchmarks in terms of the types of their heterogeneity. DOREMUS is a real-world music-related dataset consisting of three interconnected datasets that describe classical music works and the related events and entities. The data is multilingual and comes from catalogs and archives of three major French cultural institutions (Radio France, La Philharmonie de Paris, and the French National Library) [83]. AGROLD consolidates data relevant to the plant science community, including crops like rice, wheat, and arabidopsis [84]. With approximately 900 million triples, AGROLD is the result of annotating and integrating over 100 datasets from 15 diverse sources [82].

Additionally, we consider the ICEWS-WIKI and ICEWS-YAGO datasets that are proposed in [39] and have been generated from KGs having highly different degree distributions. These two datasets have been generated in a way so that they are highly heterogeneous in structure as compared to DBP15K_{FR–EN} (for full comparison we refer to [39]) and the source and target KGs in each of these two datasets have very different scales. To have an intuition of how large the datasets are and how much their KGs differ in scale, in Table 2, we have shown the size of the source KG and target KG in each dataset by #S and #T, respectively.

4.2. Evaluating Heterogeneity of Datasets

To analyze and compare dataset heterogeneities, we compute three statistical and two qualitative metrics for each pair of KGs in our datasets. We first show the degree distribution of KGs in these datasets in Figure 1. We calculate the percentage of the entities with respect to the degrees and visualize it for degrees of up to 10.

Looking at Figure 1, we can observe that the percentage of the nodes having the same degrees are similar in the pair of KGs in both DBP15K and SPIMBENCH datasets, i.e. each pair of KGs has almost the same degree distribution in these two datasets. However, this is not the case for DOREMUS, AGROLD, ICEWS-WIKI, and ICEWS-YAGO where we can see the percentage of the nodes having the same degree is different across the KGs. To statistically analyze the underlying distribution of degree sequences in each pair of KGs, we apply our first statistical metric which is the Jensen–Shannon divergence test.
Table 2
Comparing each two KGs for each dataset (all numbers indicate percentages).

<table>
<thead>
<tr>
<th>Dataset</th>
<th>JS divergence</th>
<th>Max difference in percentage of nodes</th>
<th>#S Sizes</th>
<th>#T Sizes</th>
<th>Size similarity</th>
<th>Reference alignment</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBP15K FR−EN</td>
<td>5.55</td>
<td>1.87</td>
<td>19661</td>
<td>19993</td>
<td>98.3</td>
<td>60.1</td>
</tr>
<tr>
<td>SPIMBENCH</td>
<td>4.41</td>
<td>2.45</td>
<td>2966</td>
<td>3082</td>
<td>96.2</td>
<td>36.6</td>
</tr>
<tr>
<td>DOREMUS</td>
<td>16.8</td>
<td>14.0</td>
<td>2057</td>
<td>1889</td>
<td>92.8</td>
<td>30.3</td>
</tr>
<tr>
<td>AGROLD</td>
<td>6.84</td>
<td>6.22</td>
<td>96117</td>
<td>51488</td>
<td>53.6</td>
<td>19.6</td>
</tr>
<tr>
<td>ICEWS-WIKI</td>
<td>36.1</td>
<td>6.21</td>
<td>11047</td>
<td>15831</td>
<td>69.78</td>
<td>-</td>
</tr>
<tr>
<td>ICEWS-YAGO</td>
<td>43.0</td>
<td>10.37</td>
<td>26863</td>
<td>22555</td>
<td>83.9</td>
<td>-</td>
</tr>
</tbody>
</table>

4.2.1. Jensen–Shannon Divergence
We found Jensen–Shannon (JS) divergence or JS distance [85] a suitable statistical metric that captures the amount of overlap between two distributions by using a bi-directional Kullback–Leibler (KL) [86] divergence. It gives us a number in the range \([0, 1]\), where the higher the number, the more divergent the distributions are. The results of our study on JS divergence and the other parameters that reflect similarities and heterogeneities of KGs in every dataset are given in Table 2. We explain the other parameters in the following.

Looking at the JS values in Table 2, we notice that the degree distributions of KGs in DBP15K FR−EN and SPIMBENCH are more similar than their counterparts in DOREMUS and AGROLD datasets. The JS test result also shows that there is a much higher level of heterogeneity in degree distributions of the ICEWS-WIKI and ICEWS-YAGO datasets than the other datasets. This is not surprising, since these two datasets have been generated to mimic the ICEWS, YAGO, and WIKI KGs having different degree distributions with similar JS ratios.

4.2.2. Maximum Difference in Percentage of Nodes w.r.t Node Degrees
To better recognize the differences in the KGs’ degree distributions, we calculate our second statistical metric which measures the maximum difference in the percentage of the entities with respect to the degrees in each pair of KGs. By looking at the second column of Table 2, we can see that the maximum difference in the percentage of the nodes across the KGs (w.r.t. the node degrees) in DOREMUS is much more than all other datasets. Also as we observed intuitively in Figure 1, the percentage of entities having the same degree in two KGs varies more in all datasets other than in DBP15K FR−EN and SPIMBENCH benchmark datasets.

4.2.3. Size Similarity
As a third statistical metric, we calculate the normalized difference between the number of entities in every pair of KGs, so we can compute the similarity in the size of KGs using Equation 1:

\[
\text{Size similarity} = 1 - \frac{|s(KG_1) - s(KG_2)|}{\max(s(KG_1), s(KG_2))},
\]

where \(s(KG_1)\) and \(s(KG_2)\) are the sizes (i.e. number of nodes/entities) of the two KGs in the dataset. We divide the absolute value of the difference in sizes of the KGs by the size of the larger KG. Nevertheless, none of these three statistical metrics are indicators of the textual properties of the entities.

Looking at the size similarity column in Table 2, we can observe that the size of KGs in benchmark datasets is almost the same, while in real-world cases the two KGs might include very different numbers of entities, as is the case for the AGROLD dataset. The size similarity of 53.6% for AGROLD indicates that the size of one of its KGs is almost twice as bigger as that of other KG. The two KGs in the AGROLD real-world dataset differ more strongly in
scale even than their counterparts in the two highly heterogeneous synthetically-generated datasets of ICEWS-WIKI and ICEWS-YAGO.

Analyzing the results of the three statistical features in Table 2, in combination with observing the degree distributions given in Figure 1, reveals the higher level of structural heterogeneity in KGs of DOREMUS and AGROLD as compared to the two synthetic benchmark datasets. Moreover, there is less JS distance between the degree distribution of KGs in benchmark datasets in comparison with the other datasets. Furthermore, in all non-benchmark datasets, ICEWS-YAGO dataset contains KGs having the least overlaps in their degree distributions, and AGROLD is the dataset that includes KGs having the most difference in scales.

4.2.4. Levenshtein Normalized Similarity

If we want to monitor how dataset heterogeneities affect each model’s performance, we need qualitative heterogeneity metrics especially, for approaches like BERT-INT, MultiKE, and i-Align that use the textual attribute values of the entities. Even RDGCN, instead of random initialization, uses a representation vector of entity name as the entity’s initial embedding. Hence, we need to have suitable metrics to compare our datasets from a data quality perspective. Since all of the four analyzed methods have been trained in a supervised manner, they all use some part (30%) of the reference alignment as their training data. Hence, the quality of data of the reference alignment affects the model’s performance directly.
As the first qualitative indicative, we get an average over the Levenshtein normalized similarity of the attribute values for all pairs of aligned entities in the reference alignment. Levenshtein distance or edit distance, is a measure of the similarity between two strings. It quantifies the minimum number of single-character edits (insertions, deletions, or substitutions) required to transform one string into the other [87]. The resulting distance is always in the interval \([0, 1]\). The similarity is computed by subtracting the normalized distance by 1. To analyze the quality of the data, we focus on the reference alignment, and for the first step, we compute the average over the Levenshtein normalized similarity of the attribute values of each pair of aligned entities (results in Table 2).

Because minor variations in the input data do not affect the performance of the language models in embedding the texts [88, 89] and regarding the fact that in EA methods that utilize the attribute values of entities (including three of our employed methods), a language model is used for initial embeddings of the entities [90, 91], we first lemmatized and stemmed all the words in each attribute value. Then, we compared all attribute values for each pair of entities in the reference alignment and we computed the maximum Levenshtein similarity between each pair of attribute values and we averaged all. Due to the multilingual nature of DBP15K\(_{FR-E\text{N}}\), we used Google Translate to translate French KG to be more fair in Levenshtein similarity results.

The Levenshtein results reported in Table 2 show that, even though the translated version of DBP15K French KG has some translation errors [79], the Levenshtein normalized similarity of aligned entities in the benchmark datasets is higher than in the real-world datasets. It can give us an intuition of better performance of the EA methods on the benchmark datasets, especially those that use more textual descriptions of the entities. Because the attribute triples have not been included in the ICEWS-WIKI and ICEWS-YAGO datasets, it is not possible to calculate the Levenshtein metric using the attribute values in these datasets.

4.2.5. Semantic Similarity

Although the normalized Levenshtein similarity can provide some insights into the textual similarity between the aligned entities in each dataset, it mainly relies on character-level differences and hence, does not take into account the semantic or contextual similarities between the pairs of entities. Hence, we need another metric to measure the semantic similarity between the aligned entities of KGs.

As mentioned, approaches using the entity or predicate features as input, usually utilize a language model to embed the entities. The studies show that the performance of these methods is relevant to the quality of the initial embeddings [34, 92]. Hence, we want to measure the similarity of two aligned KGs [93, 94] based on initial embeddings of entities in the reference alignment. Because language models capture the semantic similarity between words, we call this metric Entity Alignment Semantic Similarity, and calculate it according to Equation 2:

\[
\text{Semantic\_similarity}(\mathcal{KG}_1, \mathcal{KG}_2) = 1 - \frac{1}{s} \sum_{i=0}^{s} \text{Normalized}(\frac{\|v(e_i) - v(e'_i)\|}{\|\sum_{j=0}^{s} v(e_j) - v(e'_j)\|}),
\]

where \(s\) is the size of the set of seed alignments \(\mathcal{S} = \{(e_0, e'_0), ..., (e_s, e'_s)\}\), and \(v(e_i)\) is the initial representation vector of entity \(e_i\) that has been obtained by using a language model. In fact, we calculate the Euclidean relative distance of every entity by its paired entity in the reference alignment. We then normalize the amounts of relative semantic distances and take an average over them. By subtracting this number from 1, we obtain the EA semantic similarity. Notice that by the definition, having a lower amount of EA semantic similarity for a dataset indicates a higher level of semantic heterogeneity.

In the last column of Table 2, we report the result of calculating the semantic similarity of the datasets based on the initial embedding of attribute values of the aligned entities using a pre-trained multilingual BERT model. Note that this measurement is important when we analyze the performance of the embedding-based EA models which use the entities’ attribute values in their frameworks. As reflected by the semantic similarity metric results, the real-world datasets have a higher amount of semantic heterogeneity in comparison to the benchmark datasets.

The results of both the Levenshtein and EA semantic similarities in Table 2 measure the quality of the input data to the EA embedding-based models. It shows that from both the character-level and conceptual perspectives of benchmark datasets, aligned entities have more similar textual descriptions than those in real-world datasets.

To visualize how the semantic similarity in different synthetic benchmark and real-world datasets differs, we applied t-SNE [95] to reduce the dimension of the embedded vectors and visualize the initial embedding space.
of the entities in 2-dimensional space. t-SNE (t-distributed Stochastic Neighbor Embedding) is a dimensionality reduction technique commonly used for visualizing high-dimensional data in lower-dimensional space, typically 2D or 3D. In Figure 2, we visualize the entity embedding spaces of the DOREMUS and SPIMBENCH datasets that according to the analysis of Table 2 are samples of EA benchmark datasets having a low and a high level of EA semantic similarity, respectively.

![Figure 2. Reduced-dimension BERT-based Initial entity embeddings of SPIMBENCH and DOREMUS from left to right, respectively.](image)

Although the amount of the semantic similarity for the DBP15K dataset is much higher than SPIMBENCH, we visualized SPIMBENCH because it has much fewer entities and this similarity is more obvious in its visualization. The dark blue and red points represent the seed alignment of the KGs. Also, in Figure 2, each entity in the seed alignment is connected to its counterpart using grey lines. Looking at the grey lines that show the distance between the initial embeddings of the entities in reference alignment, one can easily recognize how far the entities are located in the DOREMUS real-world dataset. In SPIMBENCH, only two aligned entities have a long distance, and for the other aligned samples, the distance is much shorter than what we can see for the DOREMUS dataset. So, looking at this Figure, we can intuitively observe a higher level of semantic heterogeneity in the DOREMUS dataset compared to SPIMBENCH.

5. Comparative Analysis of the Methods Performance

In this section, we present the results of implementing and applying the selected EA models on the chosen datasets. We first explain the challenges of using the models on real-world and less well-known benchmark datasets and how we overcome these issues. Next, we discuss the evaluation metrics employed by the models and the experiments regarding the entity alignment task that we have done with the models. In the last part of the section, we provide an overview of how the models perform on both benchmark and real-world datasets. We also investigate how these performances relate to the dataset features we discuss in Section 4.

5.1. Datasets Preparation for Applying the EA Models

For each dataset, we have a file of the source KG, a file of the target KG, and a file containing the reference alignments in XML, turtle, or Ntriples format. To feed the data to each model, we prepare a series of files in certain names and formats, e.g. json, pkl, txt, or other, that the model recognizes and is capable of processing. There are some obstacles in preparing the proper inputs for the models. These issues are either related to the dataset design itself e.g. using blank nodes, or are related to the model input’s design e.g. when the model’s input samples are
unlabeled and we have no information about their contents. Even if we prepare the proper inputs to the models, there might be some runtime errors that happen exceptionally by having very minor changes in the contents of the input data, which makes us get into the procedure of validating the data. Data validation in an ML pipeline ensures that training data is error-free and accurate, preventing issues that could degrade model performance during deployment and safeguarding against errors introduced during data processing [96]. Hence, we should handle the data lifecycle of inputs to each embedding-based EA model [97], and address as many problems as we face to prepare the suitable data. After writing the codes to prepare the proper input files to the 4 models that we used and validating them for different benchmark and real-world data, we decided to share the codes on a GitHub page4 to pave the way for researchers to benefit from employing these methods on their specific datasets. We included all the links to the original models on our GitHub page.

Note that, despite the high heterogeneity aspects of CEWS-WIKI and ICEWS-Y AGO, which make them more similar to the real-world KGs, we do not run experiments on these datasets. The reason is that these two datasets lack the attribute triples and only contain the relation triples. Hence, the only model that we can employ for them is RDGCN. Because of the lack of possibility to obtain results comparable across all models for these two datasets, we leave them aside.

5.2. Evaluation Metrics

The results of the performance of the models on the datasets we have utilized are shown in Table 3 using Hit@K and Recall metrics. A Hit@K occurs when the recommended entity that matches the ground truth has been located in the first K positions of the ranked list of predictions. For the validation phase of embedding-based methods, since the model predicts a ranked list of candidates for each given entity, precision, recall, and F1-score are equal to Hits@1 [34]. Hence, we compare the Hit@1 results of embedding-based models with DLinker recall on each dataset and highlight the results of the method with the best performance on each dataset in bold. Recall is calculated as the ratio of true positive predictions to the sum of true positives and false negatives, indicating the proportion of actual positive instances that were correctly identified by the model.

5.3. Evaluation Tasks

5.3.1. Analyzing Performance of the Models on the Real-world and Benchmark datasets

The performance of BERT-INT on different datasets, as compared to other models, is given in Table 3. We can see that its performance on DBP15K and SPIMBENCH is competitive and even better than that of other methods (99.3% and 82.4% of Hit@1, respectively), while on other datasets like DOREMUS and AGROLD, the performance substantially drops (47.9% and 21.1% of Hit@1, respectively). Looking at Table 1, we can see that BERT-INT is the only method that uses all kinds of textual information related to each entity. Furthermore, based on Table 2, it can be observed that the performance of BERT-INT and the amount of EA semantic similarity of datasets are directly related. Considering the lower amount of EA qualitative features of the AGROLD and DOREMUS datasets compared to the benchmarks reported in the last two columns of Table 2, the performance drop of the BERT-INT model is understandable. This again highlights the importance of data description quality in determining BERT-INT’s performance.

Observing the RDGCN results in Table 3, we can see that its performance is more than 88% and 63% on DBP15K and SPIMBENCH, respectively, and it drops significantly in the real-world scenarios (close to 0% Hit@1). According to the given information in Table 1 with RDGCN being an EA approach that only uses graph structure besides entity names and in light of the 3 statistical metrics reported in Table 2 which indicate the real-world dataset’s higher structural heterogeneity, this is a reasonable result. Besides the results of Table 2, looking at Figure 1, we can see the long-tailed issue of AGROLD’s KGs. The long-tailed problem in graphs [98, 99] is described as an issue where a small number of nodes have a substantial number of neighbors, while the majority (referred to as tail nodes) have only a few neighbors [100]. GNNs used in RDGCN underestimate tail nodes during the training of the model and leads to a low-quality KG embedding [101]. Moreover, RDGCN uses word embedding models to produce the initial

4https://github.com/EnsiyehRaoufi/Create_Input_Data_to_EA_Models
Table 3
Recall/Hit@1 and Hit@10 of analyzed EA models on DBP15K, SPIMBENCH, DOREMUS, and AGROLD datasets (All numbers indicate percentages)

<table>
<thead>
<tr>
<th>Methods</th>
<th>BERT-INT</th>
<th>RDGCN</th>
<th>MultiKE</th>
<th>i-Align</th>
<th>DLinker</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Hit@1</td>
<td>Hit@10</td>
<td>Hit@1</td>
<td>Hit@10</td>
<td>Hit@1</td>
</tr>
<tr>
<td>DBP15K FR-EN</td>
<td>99.3</td>
<td>99.8</td>
<td>88.6</td>
<td>95.7</td>
<td>37.5</td>
</tr>
<tr>
<td>SPIMBENCH</td>
<td>82.4</td>
<td>82.4</td>
<td>63.4</td>
<td>86.8</td>
<td>57.1</td>
</tr>
<tr>
<td>DOREMUS</td>
<td>47.9</td>
<td>64.1</td>
<td>1.2</td>
<td>10.9</td>
<td>2.70</td>
</tr>
<tr>
<td>AGROLD</td>
<td>21.1</td>
<td>33.2</td>
<td>0.02</td>
<td>0.3</td>
<td>2.30</td>
</tr>
</tbody>
</table>

embeddings of the entities using entity names. Because the names (which are the last part of the entity URIs by the model’s default) of the musical works and the proteins/genes in DOREMUS and AGROLD have been defined by some IDs in their ontologies, entity initial embeddings would not also be able to guide the embedding module to a better result. All the aforementioned observations can explain the low performance of RDGCN on DOREMUS (1.2% of Hit@1) and its very weak performance (less than 1% of Hit@1) on AGROLD.

Although MultiKE outperforms several EA Translational-based methods [45] using multi-view KG embedding technique, this model overall performed the weakest among the employed embedding- and non-embedding-based models on our selected datasets. According to the results reported in Table 3, MultiKE’s performance drops for DOREMUS and AGROLD. In addition, we observed a higher level of structural and qualitative heterogeneities in DOREMUS and AGROLD than that of the benchmarks, presented in Table 2. Hence, by the fact that MultiKE relies on both the graph structure and textual information of entities and their attributes (refer to Table 1), the gap in the performance of this model on the benchmark and real-world datasets would be justified.

i-Align uses two Transformer encoders for text and graph embeddings. As Table 3 shows, it performs better on SPIMBENCH and DOREMUS (75.0% and 53.1% of Hit@1, respectively) rather than on DBP15K (26.6% of Hit@1) and its performance drops significantly when it comes to AGROLD (4.4% of Hit@1). We suspect that the reason for the model performing worse on DBP15K as compared to DOREMUS is the fact that only the first ten characters of the attribute values were considered, while the rest of the sequence was ignored by the textual transformer-based encoder. This occurs while it is crucial to retain the informative attribute descriptions included in the values, as we saw (by our experiment) how reducing them can significantly harm the performance by up to 19% in the case of the BERT-INT model.

As a baseline of non-embedding-based approaches, we used the DLinker method. Because this model fundamentally finds the longest common subsequence in the descriptions of a pair of entities belonging to two different KGs, it does not support entity alignment on the multilingual dataset of DBP15K FR-EN. Moreover, by looking at Table 3 and comparing the Hit@1 of the embedding-based EA models, DLinker shows the top performance on DOREMUS and AGROLD datasets, although this model’s performance still has room for improvements on AGROLD. This result shows that even though embedding-based models perform very well on some benchmark datasets (e.g. 99.3% of Hit@1 for BERT-INT on DBP15K FR-EN), but it seems that they overfit on the benchmark data. Therefore, using these models could not be efficient in producing the alignments in real-world heterogeneous datasets.

Hence, we observed how dataset features presented in Table 2 together with the results of our experiments in this section can justify the gap between the performance of the selected methods on benchmark and real-world datasets.

5.3.2. Analyzing Effectiveness of Inference by the Models

During our analysis, we examined the effectiveness of existing methods’ predictions on real-world data. For this purpose, we focused on the capabilities of models in the inference phase. To illustrate this problem with the models, we make use of Figure 2 where the dark-colored points represent the entities in the reference alignment of a dataset and the light-colored points represent all other entities in the two KGs.

Models like RDGCN and BERT-INT just consider a subspace of dark-colored points to seek alignment candidates for every given validation entity and ignore the rest of the space. The first issue of such under-representation in the candidates’ search space is that we cannot make sure if the method reports reliable results and if not, the second issue would be the inefficiency of these methods in predicting the aligned entities besides the validation set.
Table 4
Performance of BERT-INT and RDGCN models on the validation data considering the candidate search space extended to the whole KG space (All numbers indicate percentages)

<table>
<thead>
<tr>
<th>Dataset</th>
<th>BERT-INT</th>
<th>RDGCN</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Hit@1</td>
<td>Hit@10</td>
</tr>
<tr>
<td>DBP15KFRA</td>
<td>98.8</td>
<td>99.7</td>
</tr>
<tr>
<td>EN</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SPIMBENCH</td>
<td>78.1</td>
<td>78.5</td>
</tr>
<tr>
<td>DOREMUS</td>
<td>37.1</td>
<td>46.7</td>
</tr>
<tr>
<td>AGROLD</td>
<td>18.5</td>
<td>28.6</td>
</tr>
</tbody>
</table>

Table 4 shows the results of our experiments on the performance of two selected models on the validation data while we extend each KG1 entity’s candidate search space to include all entities in KG2. We visualized the results of this Table for Hit@1 metric in Figure 3.

As we can observe from Table 4, there are 7.7% and 34.6% reductions in the Hit@1 of RDGCN on DBP15KFRA-EN and SPIMBENCH datasets, respectively when we extend the search space and compare every entity with all other entities in the embedding space. These reductions are 0.5% and 4.3% for the same datasets, respectively, when it comes to using BERT-INT. Extending the candidates’ search space of DOREMUS and AGROLD causes Hit@1 of RDGCN to drop to zero. Furthermore, extending this space on DOREMUS and AGROLD causes Hit@1 of BERT-INT to drop by 10.8% and 2.6%, respectively. These results show that embedding-based EA models still need improvements to be able to find the aligned entities in the realistic search space of the knowledge graphs.

6. Conclusion and Future Work

In this work, we conducted an in-depth analysis of the features of several real-world datasets compared to popular benchmark datasets. Also, we presented an empirical study analyzing the performance of embedding-based EA models beyond test data and on real-world heterogeneous data. We observed that a number of entity alignment embedding-based models like BERT-INT [76] with a very strong performance for the task of entity alignment on the well-known DBP15K dataset, suffer a drop of performance on real-world data with heterogeneous textual properties. Also, while a few GNN-based models like RDGCN outperform other models on the benchmark DBP15K datasets [34], they show a very degraded performance when it comes to datasets with higher levels of structural
heterogeneity like DOREMUS and AGROLD. As future work, regarding the observed issues, in each model using the textual values of entities as input features, we will consider not only relying on lexico-semantic embeddings [27], but also on using rule-based KG embedding [102, 103]. One can consider injecting background knowledge [104] in models that mostly rely on the graph structure to enhance their performance on datasets that structurally are highly heterogeneous.

Since the heterogeneity between aligned KGs is not just limited to diversity in size and degree distribution, we considered calculating a metric to measure the semantic similarity over the reference alignments, which we observed would be well-representative of the resemblance in the semantics of two KGs, helping explain the performance issues. Then, by investigating the reasons for performance fluctuations of the EA models regarding the heterogeneities of real-world datasets, we can upgrade the models to better fit real-world scenarios.

Most of the existing embedding-based EA methods simplify the inference process [27] and use just a limited portion of the embedding space for the evaluation. This seems to be neither fair nor practical when it comes to using them to discover unseen alignments. To overcome the problems that occur by under-representing the data in KGs, we may think about effective ways to limit the search space and find the proper set of alignment candidates [105, 106] or to design embedding-based EA methods that represent the entities over entire KGs in a more efficient way to be able to find alignments in heterogeneous cases. We can go toward inductive learning i.e. to train models on two aligned KGs that are capable of predicting the matches between unseen entities in other KGs. By solving this issue, we believe the EA models will be capable of discovering a much greater number of alignments over each pair of knowledge graphs.

Acknowledgements

This work is (partially) supported by the French National Research Agency ANR DACE-DL project, grant number ANR-21-CE23-0019.

This work was granted access to the HPC resources of IDRIS under the allocation 2024-AD010315119R1 made by GENCI.

References


