SECEA: Self Configuring Matcher Framework For Entity And Knowledge Graph Alignment

Alexander Becker[1][0009-0003-8212-4647], Axel-Cyrille Ngonga Ngomo[1][0000-0001-7112-3516], and Mohamed Ahmed Sherif[1][0000-0002-9927-2203]

Data Science Group (DICE), Heinz Nixdorf Institute, Department of Computer Science, Paderborn University https://dice-research.org/
beckeral@campus.upb.de,axel.ngonga@upb.de, mohamed.sherif@upb.de

Abstract. Entity alignment is the task of identifying equivalent instances across different knowledge graphs (KGs). This idea is extended by KG alignment, which additionally involves discovering correspondences not only between entities, but also between relations and classes across the KGs. Existing approaches to entity and KG alignment often fail to fully exploit schema information: embedding-based methods tend to obscure it during representation learning, while others ignore it entirely and rely primarily on lexical similarities. In this paper, we propose SECEA, an unsupervised framework for entity and KG alignment. SECEA integrates multiple categories of matchers based on explainable matching rules and employs a dynamic selection mechanism that prioritizes the most coherent matchers with respect to the current alignment state. We evaluate our approach on widely used real-world benchmark datasets from the alignment literature. The results show that SECEA outperforms current state-of-the-art methods in class and property alignment within the KG alignment setting, as well as in instance alignment for the entity alignment task.

Keywords: Knowledge Graphs · Data Integration · Entity Alignment · Knowledge Graph Alignment

1 Introduction

Knowledge Graphs [18] (KGs) represent structured knowledge in the form of triples $\langle$*subject, relation, object*$\rangle$, enabling efficient access for both humans and machines. Within a single KG, entities are identified by Internationalized Resource Identifiers (IRIs). However, the same real-world entity is often assigned different IRIs across different KGs [24], since IRIs begin normally with a namespace defined by the KG's creator, which inherently differs across KGs. This becomes a problem due knowledge becoming scattered between KGs such as DBPEDIA [26], WIKIDATA [48], YAGO [39], and enterprise KGs [13]. As a result, it becomes difficult to consolidate and fully exploit all available information about a given entity. Hence, entity alignment is a crucial task for integrating data and improving the accuracy of downstream tasks like link prediction [35], Retrieval-Augmented Generation (RAG) for question answering [29], and

fact verification [25]. The same holds for the task of KG alignment [15], which extends entity alignment to also require a mapping between the schemas of KGs (i.e., classes and relations as defined in OWL [17]).

Recently, the *Tiered Iterative Matching* (TIM) architecture [2,4] has been proposed for KG alignment consisting of individual matchers, which create matches based on previous defined rules, and are separated into iterative matchers that run several times and bootstrap matchers that only run once. However, even though the architecture shows promising results, we identify multiple insufficiencies within this architecture: 1) The order of matchers has to be explicitly specified by the user; 2) Bootstrap matchers are only run one time even if their output can change depending on the time they are run; 3) Several matchers rely on a predefined set of relations indicating the name of elements (e.g., `rdfs:label`), which is not generally applicable for all KGs.

Consequently, we develop an approach to mitigate these problems by proposing a framework that 1) is capable of selecting the order of matchers dynamically without any user input; 2) only consists of repeatably running matchers so no division between bootstrap matchers and iterative matchers is needed; 3) dynamically identifies relations that specify the name/title of elements for any KG without requiring any user intervention. 4) Retains the explainability, scalability, unsupervised nature, and low resource requirements from TIM.

The main contributions of this paper are the following:

- We propose an unsupervised iterative matching framework that dynamically configures the ordering of matchers instead of relying on a predefined order as it is utilized by current state-of-the-art architectures.
- We present multiple matcher selection strategies and analyze their strengths and weaknesses within our framework.
- We run extensive experiments on multiple real-world datasets that show the superiority of our approach compared to other SOTA approaches while not relying on expensive hardware nor training data.

2 Related Work

Entity Alignment. Supervised entity alignment systems often rely on translational or GNN-based entity for each input KG [43] that get integrated using parameter swapping [42,27,59,57], parameter sharing [58,40,46], calibration [51,50,6,54,28,56,55], or transformations [8,7,33] between vector spaces. Other supervised neural approaches treat entity alignment as a multi-class classification problem where the model is optimized to produce a 1-to-1 mapping [53,12].

Unsupervised entity alignment systems often rely on literal attributes to align entities using lexical similarity measures [5,1,3], pretrained BERT models [34,52,45,44], or LLMs [19,22,9] to use high similarity pairs as seed matches for self-supervised learning, move pairs with high literal similarity close to each other in the embedding space, or directly align entities based on the computed similarity. Traditional non-neural entity alignment frameworks [30,47,20,38] strongly rely on string similarity measures or predefined rules that can be found by unsupervised heuristics, training data, or active learning techniques [32,37,31].

While most recent methods for entity alignment rely on embeddings or transformers in some form and therefore lose explainability, SECEA is explainable throughout the whole alignment process. Each match (or non-match) can be explained as the individual matchers employed by our framework are inherently explainable through their rule-based nature and the selection of matchers is determined directly based on the proposed matches from each matcher and the current alignment.

Knowledge Graph Alignment. The task of KG alignment extends the entity alignment task by also requiring matches between classes and properties. Similar to the approaches for entity alignment, traditional KG alignment approaches [20,36,11] heavily rely on string similarity measures to find correspondences between the input KGs. Recently, fully LLM-based methods have been proposed [23,16] that use LLMs to decide whether a pair of classes, properties, or instances has a high similarity and if a match should be created. Furthermore, phrase encoders and fuzzy logic have been applied to KG matching [34]. The most similar work to our approach is TIM [4], which combines iterative matchers and bootstrap matchers based on the KG structure to iteratively expand the mapping. However, as stated previously, this design comes with a few weaknesses, which we avoid in our proposed framework.

3 Preliminaries

Definition 1 (Knowledge Graph). *A Knowledge Graph is a set of triples $G \subseteq G_{\mathcal{E}} \times G_{\mathcal{R}} \times (G_{\mathcal{E}} \cup G_{\mathcal{L}})$ with $G_{\mathcal{E}}$ being the entities, $G_{\mathcal{R}}$ being the relations, and $G_{\mathcal{L}}$ being the literals in G. OWL [17] further refines the structure of a KG into classes G_C that represent concepts, properties G_P that are similar to relations, and instances G_I that are individuals and can be part of a class concept. In the rest of the paper, we refer to any class, property, instance, or literal as an element of a KG when the distinction between them is not relevant.*

Definition 2 (Entity Alignment). *Entity Alignment is the task of finding an alignment $A \subseteq S_I \times T_I$ between the instances of two knowledge graphs S (source) and T (target). The alignment should contain instance pairs that refer to the same real-world entity.*

Definition 3 (Knowledge Graph Alignment). *Knowledge Graph Alignment extends entity alignment to the OWL domain. For this task, the alignment A should contain class, property, and instance pairs; formally $A \subseteq (S_C \times T_C) \cup (S_P \times T_P) \cup (S_I \times T_I)$.*

Similar to most other works, we restrict the task to finding 1-to-1 alignments [43].

Definition 4 (Triple Equivalence). *Two triples $\langle s_h, s_r, s_t \rangle \in S$ and $\langle t_h, t_r, t_t \rangle \in T$ are equivalent (denoted $\langle s_h, s_r, s_t \rangle \cong \langle t_h, t_r, t_t \rangle$) iff the respective components of the triple are mapped to each other, i.e., $(s_h, t_h) \in A \wedge (s_r, t_r) \in A \wedge (s_t, t_t) \in A$. We also define partial triple equivalences based on only two of the components each to find out*

if two triples contain the same fact about different entities or relations. Formally,

$$\langle s_h, s_r, s_t\rangle \overset{HR}{\cong} \langle t_h, t_r, t_t\rangle \iff (s_h, t_h) \in A \wedge (s_r, t_r) \in A$$
$$\langle s_h, s_r, s_t\rangle \overset{HT}{\cong} \langle t_h, t_r, t_t\rangle \iff (s_h, t_h) \in A \wedge (s_t, t_t) \in A$$
$$\langle s_h, s_r, s_t\rangle \overset{RT}{\cong} \langle t_h, t_r, t_t\rangle \iff (s_r, t_r) \in A \wedge (s_t, t_t) \in A$$

Unique triple equivalences $\cong^*$ are equivalences that appear for exactly one pair:

$$\langle s_h, s_r, s_t\rangle \overset{HR}{\cong^*} \langle t_h, t_r, t_t\rangle \iff \langle s_h, s_r, s_t\rangle \overset{HR}{\cong} \langle t_h, t_r, t_t\rangle \wedge$$
$$\nexists_{s' \neq s_h} : \langle s', s_r, s_t\rangle \overset{HR}{\cong} \langle t_h, t_r, t_t\rangle \wedge \nexists_{t' \neq t_h} : \langle s_h, s_r, s_t\rangle \overset{HR}{\cong} \langle t', t_r, t_t\rangle$$

$\overset{HT}{\cong^*}$ *and* $\overset{RT}{\cong^*}$ *are defined analogously.*

4 Approach

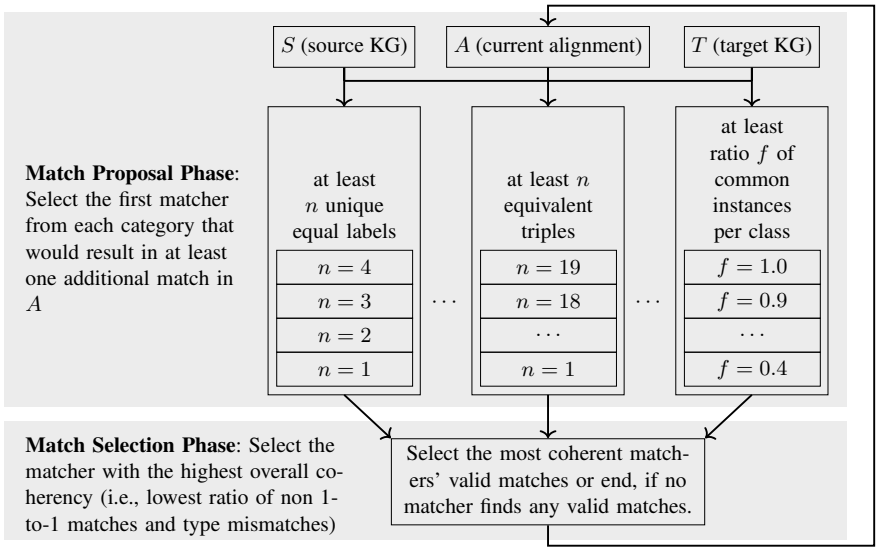

Fig. 1. An overview of the our proposed framework. In the first phase, within each category we compute the most restrictive matcher that proposes at least one match that could be added to the current alignment A. In the second phase, the most coherent matcher is selected and it's matcher are applied to A. This repeats until no matcher finds any possible matches.

SECEA is built upon multiple categories of individual rule based matchers, e.g., matchers that create matches based on identical labels, triple equivalences, or unique

triple equivalences. Within each matcher category, there is a logical order of matchers by their restrictiveness, i.e., a matcher that requires two triple equivalences to constitute a match is more restrictive than a matcher that requires only one triple equivalence to constitute a match. Among the different categories, there is no defined order of restrictiveness as different categories create matches based on different features with no natural ordering.

Initially, we add all pairs of entities that share an identical IRI to the alignment A, thereby ensuring that common properties such as `rdf:type` are matched to themselves and that literals with identical lexical representations are aligned accordingly. Then, the alignment process is conducted using an iterative procedure with two alternating stages: the *match proposal* phase and the *match selection* phase. During the *match proposal* phase we query the matchers of each category ordered by restrictiveness for matches they would create until at least one possible new match is found (i.e., there is at least one pair that does not violate the current alignment and type coherency, and is not ambiguous within the proposed matches by that matcher; the details will be explained in Section 4.1). In the *match selection* phase SECEA decides which categories' proposed matches are the most coherent given the current alignment A. The selected matches get added to A and all other matches are rejected for the moment. If no matcher proposes any valid matches, the alignment process ends. A visualization is shown in Figure 1.

4.1 Key Components Of Our Approach

In this section, we provide details about how we discover relations that indicate the name of entities, restrictions on valid matches, and how to track triple equivalences.

Finding Label Relations. A key task of our approach is finding relations that function as labels (we will denote them as *label relations* in the following) specifying the name of elements, e.g., `rdfs:label`, `skos:altLabel`, and `foaf:name`. Keeping a fixed list of relations that define a name for entities is possible to some extend for popular relations, however, as KGs may also use custom relations for defining names, this would limit the general applicability. Therefore, we compute a set of relations that may be label relations based on the properties of relations within the KG.

Label relations must be datatype properties, i.e., relations that exist between two entities are not label relations. Furthermore, label relations do not map several elements to the same value and should therefore be close to injective. Consequently, we filter out all relations whose triples do not have at least 95% unique tail values. Formally, we exclude all relations rel with $\frac{|\{t|\exists_h : \langle h, rel, t\rangle \in G\}|}{|\{\langle h, rel, t\rangle|\langle h, rel, t\rangle \in G\}|} < 0.95$. In order to also exclude relations that achieve this criterion only due to being very sparse, we additionally require label relations to be present for at least 10% of entities. All other relations are considered as label relations, denoted as $G_R^L \subseteq G_R$.

Triple Equivalence Counter. Throughout the alignment process, we keep track of all triple equivalences as defined in Definition 4. Whenever a pair is added to the alignment, we inspect all triples containing the source and target element from the newly matched

pair to determine whether the new correspondence leads to a new triple equivalence. We denote the number of equivalent triples for pair (s,t) as $\left| \underset{\text{(s,t)}}{\cong} \right|$. Formally,

$$\left| \underset{\text{(s,t)}}{\cong} \right| = \sum_{\substack{\langle s_h,s_r,s\rangle \in S \\ \langle t_h,t_r,t\rangle \in T}} \mathbf{1}\left[\langle s_h,s_r,s\rangle \overset{\text{HR}}{\cong} \langle t_h,t_r,t\rangle\right] +$$

$$\sum_{\substack{\langle s_h,s,s_t\rangle \in S \\ \langle t_h,t,t_t\rangle \in T}} \mathbf{1}\left[\langle s_h,s,s_t\rangle \overset{\text{HT}}{\cong} \langle t_h,t,t_t\rangle\right] + \sum_{\substack{\langle s,s_r,s_t\rangle \in S \\ \langle t,t_r,t_t\rangle \in T}} \mathbf{1}\left[\langle s,s_r,s_t\rangle \overset{\text{RT}}{\cong} \langle t,t_r,t_t\rangle\right],$$

where $\mathbf{1}(x)$ is 1 if x is true and 0 of x is false. Unique equivalent triples for a pair (s,t) are counted analogously and denoted $\left| \underset{\text{(s,t)}}{\cong^*} \right|$.

Restrictions on Match Creation. In alignment scenarios matches between elements are often created in a 1-to-1 manner. We also follow this constraint and make sure it is met by filtering the matches proposed by a matcher $M \subseteq (S_C \times T_C) \cup (S_P \times T_P) \cup (S_I \times T_I) \cup (S_L \times T_L)$ using the following methodology. First, we filter out all matches $(s,t) \in M$ that are of a different element type (e.g., we do not match any instance to a property). Additionally, all matches that violate existing matches present in the current alignment A are excluded. Formally, we do not consider matches $(s,t) \in M$, if $(s,t') \in A$ or $(s',t) \in A$. After this filtering, we still cannot directly add the remaining matches to the alignment due to the possibilities of duplicate source or target entries within the proposed matches. For example, a matcher may consider the pair (s,t') and (s,t^*) with $t' \neq t^*$ a match, which would lead to a non 1-to-1 mapping if applied. In this case, we do not create any match for s and ignore all pairs that would lead to an ambiguous alignment. We specifically do it this way instead of randomly choosing one match out of multiple possible because other matchers that are more decisive for s are still able to create a valid match this way and therefore create a more precise alignment.

4.2 Matcher Categories

In this section, we describe the different matcher categories and what conditions need to be met for them to propose a match. Note that the matchers explicitly only propose matches when they are run but the decision which matches are approved and added to the alignment is done using a matcher selection strategy, which we will explain in Section 4.3. The intuition of rules used by the matchers is similar to the one used in [4], however, we do not define any order and generalize the rules. First, we will state the individual rules with (hyper-)parameters n and f controlling how restrictive the matchers are. The parameters we use for each matcher are defined at the end of this section.

1) Unique Identical Label Matchers. The first matcher category proposes pairs that possess at least n unique identical labels. We check every combination of source and target label relations computed in Section 4.1 to find pairs of elements that have an

identical attribute from their respective label relation, which is also unique within the domain space of this relation in their own KG. Formally, the n-th matcher proposes to match two elements (s,t), iff $n \geq \left|\{(s_l,t_l) \mid s_l,t_l \in S_R^L \times T_R^L : \langle s,s_l,s_t\rangle \in S \wedge \langle t,t_l,t_t\rangle \in T \wedge s_t\ t_t \wedge \nexists_{s\neq s'}\langle s',s_l,s_t\rangle \in S \wedge \nexists_{t\neq t'}\langle t',t_l,t_t\rangle \in S\}\right|$, with $s_t = t_t$ indicating that the literal values s_t and t_t need to be identical after lower-casing them. This specific category is divided into three sub-categories based on the element type, i.e. this category exists for classes, properties, and instances separately.

2) Equivalent Triple Matchers. The second matcher category should match pairs that have at least n equivalent facts known about them. Formally, this matcher proposes a match for all pairs (s,t) with $n \geq \left|\underset{(s,t)}{\cong}\right|$.

3) Unique Equivalent Triple Matchers. The third matcher category is similar to the second one but should only match pairs that share at least n equivalent facts unique to that pair. Formally, this matcher proposes a match for all pairs (s,t) with $n \geq \left|\underset{(s,t)}{\cong^*}\right|$.

4) Unique Identical Label and Equivalent Triple Matchers. The fourth matcher category combines the first two categories. Here, n identical unique labels and n equivalent triples are needed for a match proposal. Formally, the n-th matcher proposes to match two elements (s,t), iff $n \geq \left|\{(s_l,t_l) \mid s_l,t_l \in S_R^L \times T_R^L : \langle s,s_l,s_t\rangle \in S \wedge \langle t,t_l,t_t\rangle \in T \wedge s_t = t_t \wedge \nexists_{s\neq s'}\langle s',s_l,s_t\rangle \in S \wedge \nexists_{t\neq t'}\langle t',t_l,t_t\rangle \in S\}\right| \wedge n \geq \left|\underset{(s,t)}{\cong}\right|$, with $s_t = t_t$ indicating that the literals s_t and t_t need to be identical after lower-casing them.

5) Literal TF-IDF Based Matchers. The fifth matcher category matches pairs (s,t) based on the TF-IDF similarity [10] of literal attributes. In particular, we concat all literal attributes of an element to function as the document for that element. Hence, the number of documents is given by the number of elements that posses at least one literal attribute. We filter out all words that appear in more than 1% of documents as they most likely do not carry as much information. We then compute the pairs (s,t) that are within the top $f \cdot \min(|\text{source documents}|, |\text{target documents}|)$ pairs with the highest TF-IDF similarity, which can be done efficiently using a heap, and propose them as matches.

6) Instance Based Class Matchers. The sixth matcher category matches class pairs (s_c,t_c) based on already matched instances similar to [4] when the ratio of matched instance pairs belonging to s_c and t_c compared to the total number of instances belonging to s_c and t_c is greater than f. Formally, this matcher category proposes matches between two classes (s_c,t_c), iff

$$f \geq \frac{2 \cdot |\{(s_i,t_i) \in \mathtt{I}(s_c) \times \mathtt{I}(t_c) \,|(s_i,t_i) \in A\}|}{|\{s_i \in \mathtt{I}(s_c) \,|\exists t_i \in T_I \wedge (s_i,t_i) \in A\}| + |\{t_i \in \mathtt{I}(t_c) \,|\exists s_i \in S_I \wedge (s_i,t_i) \in A\}|},$$

with $\mathtt{I}(x)$ being the instances of class x.

7) Unique Identical Labels and Equivalent Class Based Instance Matcher. The seventh matcher category works very similar to the third matcher type but comes with the additional requirement that the classes of an instance pair also have to match. Formally, the n-th matcher proposes to match two elements (s, t), iff $n \geq \left|\{(s_l, t_l) \mid (s_l, t_l) \in S_R^L \times T_R^L : \langle s, s_l, s_t\rangle \in S \wedge \langle t, t_l, t_t\rangle \in T \wedge s_t = t_t \wedge \nexists_{s \neq s'} \langle s', s_l, s_t\rangle \in S \wedge \nexists_{t \neq t'} \langle t', t_l, t_t\rangle \in S\}\right| \wedge (\texttt{class}(s), \texttt{class}(t) \in A)$, with $\texttt{class}(x)$ being the class of x defined by a $(x, \texttt{rdf:type}, \texttt{class}(x))$ triple.

Hyperparameters Used Within category 1, 4, and 7, we employ matchers with $\{n \in \mathbb{N} \mid n \leq |S_R^L| \cdot |T_R^L|\}$. This results in one matcher for each possible number of unique identical labels. Within category 2 and 3, we employ matchers with $\{n \in \mathbb{N} \mid n < 20\}$. Having 20 equivalent facts required to match an element pair would not noticeably reduce the risk of false positives compared to only requiring 19 equivalent facts but the more matchers exists, the higher the runtime gets. For the TF-IDF based category 5 we use $f \in \{0.2x \mid x \in \mathbb{N} \wedge x < 50\}$. Increasing the 50 further would lead to more matches, however, at the cost of an increasing runtime, which is not worth the performance gain after a certain point. For the class matchers from category 6, we use $f \in \{1.0, 0.9, \ldots, 0.4\}$, with 0.4 as a minimum value as lowering this further would result in false positive matches.

4.3 Matcher Selection

A crucial step for our framework to be accurate is to select the highest quality proposed matches, for which we will present multiple strategies in this section and analyze their strengths and weaknesses. Let M_c be the set of matches proposed by each category c.

One straightforward approach for this selection problem would be to compute performance measures similar to precision, recall, and F-measure of all proposed match sets M_c with respect to the current alignment (with proposed matches already in the alignment being true positives, proposed matches violating the current alignment as false positives, and other matches ignored). However, this has the strong drawback that since within a single category the matches of a certain matcher are also proposed by all less restrictive matchers. Consequently, after the first iteration of SECEA, this strategy would be biased towards the matcher category selected in the first iteration.

Another approach could be to select all matches that appear in multiple sets of proposed matches instead of selecting the matches of a specific matcher only. This approach also has a bias problem towards certain matcher categories as some categories' rules overlap with other categories' rules. For example, there are multiple matcher categories based on equivalent triples, which naturally tend to have a higher overlap with similarly based categories than with literal based matchers.

Consequently, we need to find a way to select the best set M_c looking at each proposed match set individually. For this, we propose to compute the ratio of pairs $(s, t) \in M_c$ that violate no assumptions about a high quality alignment regarding A as well as M_c itself, and choose the set M_c with the highest ratio of non-violating pairs. The possible violations are similar to the ones defined in Section 4.1. Specifically, a pair $(s, t) \in M_c$ is treated as a violation in three cases:

Table 1. Statistics of the entity alignment datasets from OpenEA [43].

Name	$\lvert S_{\mathcal{E}}\rvert$	$\lvert T_{\mathcal{E}}\rvert$	$\lvert S_{\mathcal{R}}\rvert$	$\lvert T_{\mathcal{R}}\rvert$
$15K_{EN\_FR}$	15000	15000	267	210
$15K_{EN\_DE}$	15000	15000	215	131
$15K_{D\_W}$	15000	15000	248	169
$100K_{EN\_FR}$	100000	100000	400	300
$100K_{EN\_DE}$	100000	100000	381	196
$100K_{D\_W}$	100000	100000	413	261

Table 2. Statistics of the OAEI KG track datasets [15].

Name	$\lvert G_I\rvert$	$\lvert G_P\rvert$	$\lvert G_C\rvert$
Star Wars (**sw**)	145033	700	269
The Old Republic (**swtor**)	4180	368	101
Star Wars Galaxies (**swg**)	9634	148	67
Marvel Database (**mv**)	210996	139	186
Marvel Cinematic Universe (**mcu**)	17187	147	55
Memory Alpha (**mma**)	45828	325	181
Star Trek Expanded Universe (**stx**)	13426	202	283
Memory Beta (**mmb**)	51323	423	240

- Either s or t is already part of another pair in the alignment. Formally, $\exists_{t \neq t'}(s, t') \in A \vee \exists_{s \neq s'}(s', t) \in A$.
- Either s or t is also part of another pair in M_c. Formally, $\exists_{t \neq t'}(s, t') \in M_c \vee \exists_{s \neq s'}(s', t) \in M_c$.
- The element type (class, property, instance, literal) of s and t is different. Formally, $(s \in S_C \wedge s \notin T_C) \wedge (s \in S_P \wedge s \notin T_P) \wedge (s \in S_I \wedge s \notin T_I) \wedge (s \in S_L \wedge s \notin T_L)$.

The overall best set M_c is accepted and its matches are added to the alignment as long as they do not violate any restrictions (see Section 4.1); all other sets of proposed matches get discarded.

5 Experiments

Herein, we evaluate SECEA on real-world datasets for KG alignment and entity alignment in Section 5.2 and Section 5.3, respectively. Furthermore, we conduct an ablation study on the impact of different matcher selection strategies in Section 5.4.

5.1 Experiment Setup

Datasets and Evaluation Metrics. As our approach supports both entity alignment and KG alignment, we the most popular benchmark datasets for each task.

For the *entity alignment evaluation*, we conduct our experiments on the OpenEA datasets [43] extracted from the English, German, and French DBpedia as well as Wikidata with $15,000$ and $100,000$ entities per KG. The dataset statistics are shown in Table 1. We only use the V1-version of the datasets as they resemble real-world conditions more closely than the V2-version. We do not use the *D-Y* datasets as they can be solved using only the entity name. The data for supervised approaches is divided into 20% for training, 10% for validation, and 70% for testing [43]. As our approach is unsupervised, we use the full dataset for testing with no need for training or validation data. The evaluation of accuracy on the OpenEA datasets is commonly done using Hits@k and MRR. However, our approach (and TIM [4]) only computes 1-to-1 matches and does not keep

any ranking of pairs. Consequently, for these approaches we use Hits@k = Hits@1 $\forall k \in \mathbb{N}$ and we excluded *not-found* matches from the MRR computation (which also implies MRR = Hits@1).

For the *KG alignment evaluation*, we use the KG track of the Ontology Alignment Evaluation Initiative (OAEI) [15] that are created using data extracted from real-world wikis [14]. The dataset statistics are shown in Table 2. Each year, several unsupervised KG alignment systems get evaluated by OAEI using precision, recall, and F-measure on the following KG pairs: `sw-swtor`, `sw-swg`, `mcu-mv`, `mma-mmb`, `mma-stx`. For the instance level evaluation, there are some issues with the reference alignment [2,23] due to them being automatically generated from cross wiki links. However, the class and property alignments were developed by ontology experts and are considered to be of high quality. Due to the lack of higher-quality KG alignment datasets, we still chose to use this dataset, even though the aforementioned issue may affect the instance-level results. Nonetheless, since we also evaluate entity alignment performance on the OpenEA datasets, we still obtain a rigorous evaluation of all aspects of SECEA.

Hardware, Software, and Implementation Details. We run the experiments on a Red Hat Enterprise Linux 8.10 virtual machine with a AMD Turin 9655 CPU with 48 cores, 192GB of main memory, PyTorch 2.11.0 and Python 3.10.4. As our algorithm is deterministic and no randomness is involved, we only report the results of a single run.

While KG alignment algorithms often rely on the IRI of entities, most entity alignment algorithms ignore them. Hence, to align with other works, we ignore IRIs on the OpenEA datasets. For the OAEI datasets, we add an additional relation `iri` that we add to all elements with the last part of their IRI (after the last "/" or "#") as the attribute.

To prevent the unnecessary explosion of pairs with equivalent triples, we exclude triples with the relations `rdf:type` and objects `owl:Class` from the triple equivalence counter as they are already processed separately to divide elements into instances, relations, and classes. Furthermore, for the KG alignment task, we also ignore the added `iri` relation and the `dbkwik:wikiPageWikiLink` relation for the equivalent triple counters. Additionally, as properties and classes are not always named in the singular form but sometimes in the plural form depending on the KG, the label based matchers ignore the last character of the labels in case it is an *s* if no exact match can be found.

Competing Approaches. We compare our approach to several popular state-of-the-art approaches. For the *entity alignment task*, we choose approaches based on: 1) translational embeddings: MTransE [8], IPTransE [58], BootEA [42], SEA [33], SHAF [49], 2) GNNs: RDGCN [51], RANM [6], GAEA [54], RHGN [28], GIA [56], MIKG [12], 3) transformers: IMEA [55], RoadEA [41], MixTea [53], and 4) structural rules: TIM [4]. For the *KG alignment task*, we choose all approaches competing in the most recent OAEI campaign [21] and further relevant systems based on: 1) string similarities: LogMap [20], LSMatch [36], Matcha [11], 2) language models OLaLa [16], DogMa [23], Flora [34], and 3) structural rules: TIM [4].

Table 3. Class F-measure results.

System	mcu-mv	mma-mmb	mma-stx	sw-swg	sw-swtor	Average
LogMap [20]	**1.00**	0.64	0.64	0.89	0.85	0.80
LSMatch [36]	**1.00**	0.44	0.70	0.75	0.83	0.74
Matcha [11]	0.00	0.80	0.82	0.89	0.86	0.67
OLaLa [16]	0.67	0.53	0.70	0.75	0.77	0.68
DogMa [23]	**1.00**	0.53	0.73	**1.00**	0.97	0.84
Flora [34]	**1.00**	0.73	0.92	0.57	**1.00**	0.84
TIM [4]	**1.00**	**0.96**	**0.96**	0.89	0.93	0.95
SECEA (ours)	**1.00**	**0.96**	**0.96**	**1.00**	0.97	**0.98**

Table 4. Property F-measure results.

System	mcu-mv	mma-mmb	mma-stx	sw-swg	sw-swtor	Average
LogMap [20]	0.00	0.00	0.00	0.00	0.00	0.00
LSMatch [36]	0.82	0.60	0.62	0.68	0.83	0.71
Matcha [11]	0.00	0.00	0.00	0.00	0.00	0.00
OLaLa [16]	0.76	0.90	0.90	0.82	0.77	0.79
DogMa [23]	**0.95**	0.93	**0.98**	**1.00**	0.96	0.96
Flora [34]	**0.95**	0.96	**0.98**	**1.00**	0.98	0.97
TIM [4]	**0.95**	0.97	**0.98**	**1.00**	0.95	0.97
SECEA (ours)	**0.95**	**1.00**	**0.98**	**1.00**	**1.00**	**0.99**

5.2 Comparison to Other Knowledge Graph Alignment Approaches

First, we compare SECEA to other KG alignment approaches on the OAEI KG track datasets with respect to the performance of class, property, and instance alignment.

Class Matching Performance. The class matching results are shown in Table 3. SECEA achieves the best overall class matching performance, attaining an average class F-measure of 0.98. This represents an improvement of 0.03 over the result achieved by TIM (the runner-up) and a substantial margin over all remaining approaches, which achieve an average F-measure of at most 0.84. SECEA reaches the best result on four of five datasets and reaches the second best result for the fifth dataset. These results clearly show that the dynamic matching strategy of SECEA surpasses the static order of TIM due to being able to adapt the order matchers based on the existing alignment.

Property Matching Performance. The results for the property matching performance are shown in Table 4. Similar to the class matching evaluation, SECEA outperforms all approaches on average with an F-measure of 0.99 while competing approaches achieve at most 0.97. SECEA reaches a perfect F-measure of 1.00 for two of the datasets and never achieves a score below 0.95. Additionally, SECEA is not outperformed on any

Table 5. Instance F-measure results.

System	mcu-mv	mma-mmb	mma-stx	sw-swg	sw-swtor	Average
LOGMAP [20]	0.60	0.82	0.82	0.80	0.86	0.77
LSMATCH [36]	0.50	0.66	0.63	0.36	0.82	0.60
MATCHA [11]	0.00	0.71	0.79	0.76	0.82	0.61
OLALA [16]	0.00	0.00	0.00	0.00	0.00	0.00
DOGMA [23]	0.80	0.93	**0.95**	0.89	0.93	0.90
FLORA [34]	**0.89**	**0.97**	**0.95**	**0.94**	**0.97**	**0.94**
TIM [4]	0.82	0.92	**0.95**	0.81	0.91	0.88
SECEA (ours)	0.81	0.92	**0.95**	0.83	0.92	0.89

dataset by any system. Even LLM- or BERT-based methods like OLALA, DOGMA, and FLORA cannot produce an equally effective alignment as our approach. Meanwhile LOGMAP and MATCHA do not output any meaningful property alignment, which could be caused by properties in the input KGs being typed as `rdf:Property` without a distinction into object- or datatype-properties [21]. Consequently, SECEA is the best choice when high quality property matches are crucial.

Instance Matching Performance. The results for the instance matching performance are shown in Table 5. This is the only category where SECEA gets outperformed by other approaches. FLORA is the best ranking system with an average F-measure of 0.94, followed by DOGMA with a score of 0.90, while SECEA only reaches the third place with an average F-measure of 0.89. However, our approach is able to outperform TIM, which only reaches a score of 0.88 on average.

After analyzing the results, we discovered that most of SECEA's deviations to the reference mapping are not actually false classifications but stem from inaccuracies in the gold standard, leading to underestimation of both precision and recall as also found out by other competitors [23,2]. For example, some KGs contain the same real-world entity twice but only one of them is linked to the other KG and disambiguation pages are linked to non-disambiguation pages in the reference mapping. Hence, we also compare the sole instance alignment performance on the OpenEA dataset to get a result that is not impacted by inaccurate reference mappings.

5.3 Comparison to Other Entity Alignment Approaches

In this section, we compare the performance of SECEA to other entity alignment approaches on the OpenEA [43] datasets using Hits@1, Hits@5, and MRR metrics. As explained in Section 5.1, the Hits@K evaluation that is common for entity alignment does not perfectly suit our approach and therefore underestimates its true performance. This limitation arises because our approach exclusively produces 1-to-1 alignments and thus does not benefit from multiple opportunities to rank the correct entity highly. In addition, our method allows entities to remain unmatched when no suitable counterpart exists in the other KG, whereas embedding-based approaches compute a complete

Table 6. Performance comparison on OpenEA-15K datasets.

Models	OpenEA-15K$_{EN_FR}$			OpenEA-15K$_{EN_DE}$			OpenEA-15K$_{D_W}$		
	H@1	H@5	MRR	H@1	H@5	MRR	H@1	H@5	MRR
MTRANSE [8]	0.247	0.467	0.351	0.307	0.518	0.407	0.259	0.461	0.354
IPTRANSE [58]	0.169	0.320	0.243	0.350	0.515	0.430	0.406	0.627	0.303
BOOTEA [42]	0.507	0.718	0.603	0.675	0.820	0.740	0.572	0.744	0.649
SEA [33]	0.280	0.530	0.397	0.530	0.718	0.617	0.360	0.572	0.458
SHAF [49]	0.524	0.771	0.634	0.716	0.891	0.793	0.598	0.804	0.690
RDGCN [51]	0.755	0.854	0.800	0.830	0.895	0.859	0.515	0.669	0.584
RANM [6]	0.925	**0.960**	**0.941**	0.950	**0.977**	**0.962**	-	-	-
GAEA [54]	0.486	0.746	0.602	0.684	0.854	0.760	0.562	0.768	0.654
RHGN [28]	0.500	0.739	0.603	0.704	0.859	0.771	0.560	0.753	0.644
GIA [56]	0.507	0.758	0.619	0.684	0.859	0.760	0.572	0.774	0.662
MIKG [12]	0.845	0.880	0.862	0.903	0.928	0.915	**0.724**	**0.834**	**0.774**
IMEA [55]	0.458	0.720	0.574	0.639	0.827	0.724	0.527	0.753	0.626
ROADEA [41]	0.810	0.854	0.831	0.868	0.905	0.886	0.653	0.789	0.714
MIXTEA [53]	0.528	0.807	0.680	0.724	0.877	0.797	0.647	0.832	0.731
TIM [4]	0.853	0.853	0.853	0.922	0.922	0.922	0.502	0.502	0.502
SECEA (ours)	**0.929**	0.929	0.929	**0.957**	0.957	0.957	0.699	0.699	0.699

ranking over all possible entity pairs. Even though we also report Hits@5 and MRR for the other approaches for completeness, we consider Hits@1 to be the most relevant measure of quality. In the context of creating `owl:sameAs` links between two knowledge graphs, linking a single entity to multiple candidates in the other KG is generally not meaningful. Therefore, our evaluation primarily focuses on Hits@1 performance. Moreover, Hits@K with $K > 1$ provides increasingly permissive evaluations, since any correct match counted in Hits@1 is necessarily also counted in Hits@K, making higher-K metrics less informative for assessing precise alignment quality.

Performance on the OpenEA-15K Datasets First, we anaylze the results on the datasets with 15000 entities each. The results are shown in Table 6. SECEA outperforms the Hits@1 performance of all approaches for the *EN-FR* and *EN-DE* dataset with Hits@1 results of 0.929 and 0.957, respectively. Furthermore, our approach also outperforms all but one approaches on Hits@5 and MRR even though we only count Hits@1 for our approach. For the *D-W* dataset, our approach only achieves the second best result with Hits@1 of 0.699, slightly lower than the result of MIKG with a Hits@1 value of 0.724. Note that our approach achieves these high quality results without any training data, which is required by the two other best performing systems RANM and MIKG. The direct comparison to TIM shows that dynamically selecting the best fitting matcher based on the current alignment is superior to a predefined order of matchers.

Performance on the OpenEA-100K Datasets The results for the OpenEA datasets with $100,000$ entities are shown in Table 7. Overall, all systems achieve lower per-

Table 7. Performance comparison on OpenEA-100K datasets.

Models	OpenEA-100K$_{EN_FR}$			OpenEA-100K$_{EN_DE}$			OpenEA-100K$_{D_W}$		
	H@1	H@5	MRR	H@1	H@5	MRR	H@1	H@5	MRR
MTRANSE [8]	0.138	0.261	0.219	0.140	0.264	0.204	0.210	0.358	0.282
IPTRANSE [58]	0.158	0.277	0.219	0.226	0.357	0.292	0.221	0.352	0.285
BOOTEA [42]	0.389	0.561	0.474	0.518	0.673	0.592	0.516	0.685	0.594
SEA [33]	0.225	0.399	0.314	0.341	0.502	0.421	0.291	0.470	0.378
RDGCN [51]	0.640	0.732	0.683	0.722	0.794	0.756	0.362	0.485	0.420
MIKG [12]	0.797	0.844	0.819	0.851	0.891	0.870	**0.629**	**0.742**	**0.681**
IMEA [55]	0.329	0.526	0.424	0.467	0.641	0.549	0.419	0.614	0.508
ROADEA [41]	0.743	0.799	0.769	0.831	0.868	0.847	0.572	0.694	0.629
TIM [4]	0.818	0.818	0.818	0.878	0.878	0.878	0.307	0.307	0.307
SECEA (ours)	**0.867**	**0.867**	**0.867**	**0.906**	**0.906**	**0.906**	0.600	0.600	0.600

Table 8. Evaluation of Matcher Selection Strategies. For the KG alignment datasets we report the average F-measure for classes, properties, and instances and for the entity alignment datasets, we report the average Hits@1 for the 15K and 100K datasets, respectively.

Strategy	**OAEI**			**OpenEA**	
	F_{Class}	$F_{Property}$	$F_{Instance}$	$H@1_{15K}$	$H@1_{100K}$
1) Highest ratio of possible matches	0.94	0.98	0.88	0.860	0.757
2) Lowest ratio of alignment violations	0.98	0.99	0.89	0.864	0.789
3) Lowest ratio of type violations	0.98	0.99	0.88	0.864	0.792
4) Lowest ratio of 1-to-1 violations	0.98	0.99	0.89	0.858	0.756
5) Lowest ratio of any violation (default)	0.98	0.99	0.89	0.862	0.791
6) Highest number of matches	0.97	0.89	0.87	0.852	0.717
7) Lowest number of matches	0.98	0.98	0.87	0.867	0.792

formance compared to the smaller datasets. This degradation can be attributed to the larger search space, where a higher number of entities increases the likelihood of incorrect candidates being identified as potential matches. SECEA places first for the *EN-FR* and *EN-DE* datasets and second for *D-W* with respect to Hits@1 values (0.867, 0.906, 0.600, respectively). However for the *D-W* dataset, SECEA only matches 60.8% of entities, i.e., 98.8%, of our matches are correct. If our approach would assign each entity a match (e.g., by similarity of their labels), the Hits@1 result would likely be higher.

5.4 Evaluation of Matcher Selection Strategies

In the last experiment, we evaluate multiple strategies for selecting the matcher whose matches should be applied to the alignment in each iteration of our approach. Specifically, test the following strategies: 1) Choosing the matcher with the highest ratio of matches that will be added to the alignment after filtering out invalid matches as described in Section 4.1; 2) choosing the matcher with the lowest ratio of matches that

violate the existing alignment; 3) choosing the matcher with the lowest ratio of matches that are incoherent regarding their type (i.e. the source element is a literal and the target element is a class); 4) choosing the matcher with the lowest ratio of matches that are not 1-to-1 matches within the proposed match set; 5) a combination of 2, 3, and 4 (this is the default option we use and is described and formalized in Section 4.3); 6) choosing the matcher with the highest number of proposed matches; 7) choosing the matcher with the lowest number of proposed matches. The results are shown in Table 8.

Our chosen strategy (5) performs best in all categories for the OAEI datasets. For the OpenEA datasets, it only achieves on average 0.005 and 0.001 H@1 less than the best performing strategy for the 15K and 100K OpenEA datasets, respectively. The worst strategy is choosing the matcher with the highest number of matches, which leads to a 0.084 H@1 drop for the 100K OpenEA datasets on average.

In general, all strategies perform similarly well due to all of them relying on the same matcher categories. Consequently, we arrive at the conclusion that the exact choice of strategy does not have a large impact on the result as long as the selection criterion reflects the quality of the proposed match set, no matter if it is external coherency (strategy 2), type coherency (strategy 3), internal coherency (strategy 4), or a combination of them (strategy 5). We refrain from doing an ablation study on the individual matcher categories as we use similar matching rules as TIM [4], which already has extensive ablation study about matcher categories showing that a diverse set of matchers is important and overlap between categories does not harm performance but improves reliability.

6 Discussion

Explainability. SECEA is entirely explainable. The matchers itself are rule based and therefore inherently explainable. The matcher selection process also relies on criteria that can be locally explained at each matcher selection step based on the coherence of the proposed matches. Additionally, no randomness is involved in SECEA and when there are conflicting matches within a selected matchers' matches, we discard all conflicting matches in order to not prefer one pair over another without a reason.

Scalability. Being scalable is essential for algorithms in today's world, where web-scale datasets need to be linked. Our approach runs fully on the CPU and does not require a GPU for embeddings, machine learning, or word embeddings. As far as runtime goes, to compute all alignments SECEA needs less than 15 minutes for the OAEI experiments and less than 8 minutes for the OpenEA experiments, making it not only very effective due to its high-quality alignment but also very efficient. This high efficiency indicates that SECEA is usable even for larger datasets with hundreds of thousands or even millions of entities. Furthermore, while other approaches require training data, which is a huge scalability barrier, our approach does not need any training data.

Possibilities for Improvement and Limitations. There are still several possibilities for improvement that can be integrated into our framework. Any other knowledge graph or entity alignment approach can be integrated as a new matcher category similarly to

our TF-IDF matcher category to match the highest confidence pairs first and less confident pairs later. Furthermore, LLMs could be introduced as matchers to compute the semantic equivalence of different classes, properties, and entities between the two input KGs. Another option for improvement could be to remove pairs from the alignment A when conflicting pairs appear more likely to be correct than the a pair already in the alignment. However, this would be difficult to implement as several hurdles exist like circles of matches depending on other matches or prohibited by them, leading to endless loops of adding and subsequently removing matches. Nonetheless, this self correction mechanism is a promising direction of research to improve performance further.

Further improvement possibilities lie in the current limitations to equivalence relations, i.e., we do not compute any subclass or subproperty relations; and instance matches are required for some of the class and property matchers, which could reduce performance when no instances are available.

7 Conclusion and Future Work

In this paper, we propose SECEA, an iterative unsupervised entity and knowledge graph alignment frameworks that is based on multiple categories of matchers each proposing matches between elements, with a matcher selecting strategy based on the matches coherence dynamically deciding which matches should be applied in the current iteration. We evaluate our approach on both entity and knowledge graph alignment with the result that SECEA achieves state-of-the-art performance, achieving the best performance for class and property matching in the KG alignment task and the best or second best Hits@1 for the entity alignment task.

In future work we aim to introduce LLM-based matchers to identify semantically similar elements across KGs. Furthermore, we plan to investigate the possibilities of replacing matches that are later found to be low-confidence pairs with conflicting high-confidence pairs to improve the alignment accuracy.

Supplemental Material Statement: The source code as well as instructions to run our approach for reproducing the results are available in our GitHub[1].

Declaration on Generative AI: The authors did not use GenAI tools in this work.

Acknowledgements: This work has been supported by the Ministry of Culture and Science of North Rhine-Westphalia (MKW NRW) through the projects SAIL (NW21-059D) and WHALE (LFN 1-04, Lamarr Fellow Network), and the German Federal Ministry of Research, Technology and Space (BMFTR) within the project KI-Akademie OWL under the grant no 16IS24057B, and European Union's Horizon Europe research and innovation programme under grant agreement No 101070305, and the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation): TRR 318/3 2026 – 438445824.

[1] `https://github.com/dice-group/SECEA`

References

1. Becker, A., Ngonga Ngomo, A.C., Sherif, M.A.: Glide: Knowledge graph linking using distance-aware embeddings. In: The Semantic Web – ISWC 2025. Nara, Japan (2025), https://papers.dice-research.org/2025/ISWC_GLIDE/public.pdf
2. Becker, A., Ngonga Ngomo, A.C., Sherif, M.A.: TIM results for OAEI 2025. In: OM 2025: The 20th International Workshop on Ontology Matching collocated with the 24th International Semantic Web Conference(ISWC 2025), November 2nd, 2025, Nara, Japan. CEUR Workshop Proceedings, vol. 4144, pp. 166–171. CEUR-WS (2025), https://papers.dice-research.org/2025/OM_TIM/public.pdf
3. Becker, A., Ngonga Ngomo, A.C., Sherif, M.A.: Conel: Contrastive neural link discovery leveraging literal similarities. In: The Semantic Web – 23rd European Semantic Web Conference, ESWC 2026, Dubrovnik , Croatia, May 10-14, 2026, Proceedings. Springer Nature Switzerland (2026), https://papers.dice-research.org/2026/ESWC_CONEL/public.pdf
4. Becker, A., Ngonga Ngomo, A.C., Sherif, M.A.: Tim: Tiered iterative knowledge graph matching. In: The Semantic Web – 23rd European Semantic Web Conference, ESWC 2026, Dubrovnik , Croatia, May 10-14, 2026, Proceedings. Springer Nature Switzerland (2026), https://papers.dice-research.org/2026/ESWC_TIM/public.pdf
5. Becker, A., Sherif, M.A., Ngonga Ngomo, A.C.: Blink: Blank node matching using embeddings. In: International Semantic Web Conference. pp. 218–236. Springer (2024)
6. Cai, W., Ma, W., Wei, L., Jiang, Y.: Semi-supervised entity alignment via relation-based adaptive neighborhood matching. IEEE Transactions on Knowledge and Data Engineering **35**(8), 8545–8558 (2022)
7. Chen, M., Tian, Y., Chang, K.W., Skiena, S., Zaniolo, C.: Co-training embeddings of knowledge graphs and entity descriptions for cross-lingual entity alignment. arXiv preprint arXiv:1806.06478 (2018)
8. Chen, M., Tian, Y., Yang, M., Zaniolo, C.: Multilingual knowledge graph embeddings for cross-lingual knowledge alignment. arXiv preprint arXiv:1611.03954 (2016)
9. Chen, S., Zhang, Q., Dong, J., Hua, W., Li, Q., Huang, X.: Entity alignment with noisy annotations from large language models. Advances in Neural Information Processing Systems **37**, 15097–15120 (2024)
10. Das, M., Alphonse, P., et al.: A comparative study on tf-idf feature weighting method and its analysis using unstructured dataset. arXiv preprint arXiv:2308.04037 (2023)
11. Faria, D., Silva, M.C., Cotovio, P., Ferraz, L., Balbi, L., Pesquita, C.: Results in oaei 2024 for matcha. In: OM@ ISWC. pp. 110–117 (2024)
12. Gao, A., Li, M., Liu, W., Sun, Z., Wan, N.: Maximizing mutual information across knowledge graphs for robust entity alignment. Expert Systems with Applications p. 129031 (2025)
13. Gomez-Perez, J.M., Pan, J.Z., Vetere, G., Wu, H.: Enterprise knowledge graph: An introduction. In: Exploiting linked data and knowledge graphs in large organisations, pp. 1–14. Springer (2017)
14. Hertling, S., Paulheim, H.: Dbkwik: A consolidated knowledge graph from thousands of wikis. In: 2018 IEEE International Conference on Big Knowledge (ICBK). pp. 17–24. IEEE (2018)
15. Hertling, S., Paulheim, H.: The knowledge graph track at oaei: Gold standards, baselines, and the golden hammer bias. In: European Semantic Web Conference. pp. 343–359. Springer (2020)
16. Hertling, S., Paulheim, H.: Olala: Ontology matching with large language models. In: Proceedings of the 12th knowledge capture conference 2023. pp. 131–139 (2023)

17. Hitzler, P., Krötzsch, M., Parsia, B., Patel-Schneider, P.F., Rudolph, S., et al.: Owl 2 web ontology language primer. W3C recommendation **27**(1), 123 (2009)
18. Hogan, A., Blomqvist, E., Cochez, M., d'Amato, C., Melo, G.D., Gutierrez, C., Kirrane, S., Gayo, J.E.L., Navigli, R., Neumaier, S., et al.: Knowledge graphs. ACM Computing Surveys (Csur) **54**(4), 1–37 (2021)
19. Jiang, X., Shen, Y., Shi, Z., Xu, C., Li, W., Li, Z., Guo, J., Shen, H., Wang, Y.: 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). pp. 7566–7583 (2024)
20. Jiménez-Ruiz, E., Cuenca Grau, B.: Logmap: Logic-based and scalable ontology matching. In: International Semantic Web Conference. pp. 273–288. Springer (2011)
21. Jiménez-Ruiz, E., Hassanzadeh, O., Trojahn, C., Hertling, S., Li, H., Shvaiko, P., Euzenat, J.: Results of the ontology alignment evaluation initiative 2025. In: Proceedings of the 20th International Workshop on Ontology Matching co-located with the 24th International Semantic Web Conference (ISWC 2025). CEUR Workshop Proceedings, vol. 4144 (2025)
22. Jin, X., Wang, Z., Chen, J., Yang, L., Oh, B., Hwang, S.w., Li, J.: Hlmea: Unsupervised entity alignment based on hybrid language models. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 39, pp. 11888–11896 (2025)
23. Kardos, P., Vass, M., Krész, M., Farkas, R.: Dogma results for oaei 2025. In: OM 2025: The 20th International Workshop on Ontology Matching collocated with the 24th International Semantic Web Conference (ISWC 2025), November 2nd, 2025, Nara, Japan (2025)
24. Kellogg, G., Seaborne, A., Champin, P.A., Hartig, O.: RDF 1.2 concepts and abstract syntax. W3C working draft, W3C (Feb 2025), https://www.w3.org/TR/2025/WD-rdf12-concepts-20250225/
25. Kim, J., Park, S., Kwon, Y., Jo, Y., Thorne, J., Choi, E.: Factkg: Fact verification via reasoning on knowledge graphs. In: Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). pp. 16190–16206 (2023)
26. Lehmann, J., Isele, R., Jakob, M., Jentzsch, A., Kontokostas, D., Mendes, P.N., Hellmann, S., Morsey, M., Van Kleef, P., Auer, S., et al.: Dbpedia–a large-scale, multilingual knowledge base extracted from wikipedia. Semantic web **6**(2), 167–195 (2015)
27. Li, S., Li, X., Ye, R., Wang, M., Su, H., Ou, Y.: Non-translational alignment for multi-relational networks. In: IJCAI. pp. 4180–4186 (2018)
28. Liu, X., Zhang, K., Liu, Y., Chen, E., Huang, Z., Yue, L., Yan, J.: Rhgn: Relation-gated heterogeneous graph network for entity alignment in knowledge graphs. In: Findings of the Association for Computational Linguistics: ACL 2023. pp. 8683–8696 (2023)
29. Mavromatis, C., Karypis, G.: GNN-RAG: Graph neural retrieval for efficient large language model reasoning on knowledge graphs. In: Findings of the Association for Computational Linguistics: ACL 2025. pp. 16682–16699 (2025)
30. Ngomo, A.C.N., Auer, S.: Limes-a time-efficient approach for large-scale link discovery on the web of data. integration **15**(3) (2011)
31. Ngomo, A.C.N., Lehmann, J., Auer, S., Höffner, K.: Raven-active learning of link specifications. Ontology Matching **2011** (2011)
32. Ngomo, A.C.N., Lyko, K.: Unsupervised learning of link specifications: deterministic vs. non-deterministic. In: Proceedings of the Ontology Matching Workshop. vol. 12 (2013)
33. Pei, S., Yu, L., Hoehndorf, R., Zhang, X.: Semi-supervised entity alignment via knowledge graph embedding with awareness of degree difference. In: The world wide web conference. pp. 3130–3136 (2019)
34. Peng, Y., Bonald, T., Suchanek, F.: Flora: Unsupervised knowledge graph alignment by fuzzy logic. In: International Semantic Web Conference (ISWC) (2025)

35. Rossi, A., Barbosa, D., Firmani, D., Matinata, A., Merialdo, P.: Knowledge graph embedding for link prediction: A comparative analysis. ACM Transactions on Knowledge Discovery from Data (TKDD) **15**(2), 1–49 (2021)
36. Sharma, A., Patel, A., Jain, S.: Lsmatch and lsmatch-multilingual results for OAEI. OM@ ISWC (2022)
37. Sherif, M.A., Ngonga Ngomo, A.C., Lehmann, J.: Wombat–a generalization approach for automatic link discovery. In: European Semantic Web Conference. pp. 103–119. Springer (2017)
38. Suchanek, F.M., Abiteboul, S., Senellart, P.: Paris: Probabilistic alignment of relations, instances, and schema. arXiv preprint arXiv:1111.7164 (2011)
39. Suchanek, F.M., Kasneci, G., Weikum, G.: Yago: a core of semantic knowledge. In: Proceedings of the 16th international conference on World Wide Web. pp. 697–706 (2007)
40. Sun, Z., Hu, W., Li, C.: Cross-lingual entity alignment via joint attribute-preserving embedding. In: International semantic web conference. pp. 628–644. Springer (2017)
41. Sun, Z., Hu, W., Wang, C., Wang, Y., Qu, Y.: Revisiting embedding-based entity alignment: A robust and adaptive method. IEEE Transactions on Knowledge and Data Engineering **35**(8), 8461–8475 (2022)
42. Sun, Z., Hu, W., Zhang, Q., Qu, Y.: Bootstrapping entity alignment with knowledge graph embedding. In: Ijcai. vol. 18 (2018)
43. Sun, Z., Zhang, Q., Hu, W., Wang, C., Chen, M., Akrami, F., Li, C.: A benchmarking study of embedding-based entity alignment for knowledge graphs. arXiv preprint arXiv:2003.07743 (2020)
44. Tang, J., Zhao, K., Li, J.: A fused gromov-wasserstein framework for unsupervised knowledge graph entity alignment. In: Findings of the association for computational linguistics: ACL 2023. pp. 3320–3334 (2023)
45. Tang, X., Zhang, J., Chen, B., Yang, Y., Chen, H., Li, C.: Bert-int: A bert-based interaction model for knowledge graph alignment. interactions **100**, e1 (2020)
46. Trisedya, B.D., Qi, J., Zhang, R.: Entity alignment between knowledge graphs using attribute embeddings. In: Proceedings of the AAAI conference on artificial intelligence. vol. 33, pp. 297–304 (2019)
47. Volz, J., Bizer, C., Gaedke, M., Kobilarov, G.: Discovering and maintaining links on the web of data. In: International semantic web conference. pp. 650–665. Springer (2009)
48. Vrandečić, D., Krötzsch, M.: Wikidata: a free collaborative knowledgebase. Commun. ACM **57**(10), 78–85 (2014). https://doi.org/10.1145/2629489
49. Wang, Y., Zhang, C., He, W.: A structure-aware and heterogeneous adaptive fusion approach for entity alignment. In: 2025 8th International Conference on Advanced Electronic Materials, Computers and Software Engineering (AEMCSE). pp. 533–542. IEEE (2025)
50. Wang, Z., Lv, Q., Lan, X., Zhang, Y.: Cross-lingual knowledge graph alignment via graph convolutional networks. In: Proceedings of the 2018 conference on empirical methods in natural language processing. pp. 349–357 (2018)
51. Wu, Y., Liu, X., Feng, Y., Wang, Z., Yan, R., Zhao, D.: Relation-aware entity alignment for heterogeneous knowledge graphs. arXiv preprint arXiv:1908.08210 (2019)
52. Wu, Y., Liu, X., Feng, Y., Wang, Z., Zhao, D.: Jointly learning entity and relation representations for entity alignment. In: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP). pp. 240–249 (2019)
53. Xie, F., Song, X., Zeng, X., Zhao, X., Tian, L., Zhou, B., Tan, Y.: Mixtea: Semi-supervised entity alignment with mixture teaching. In: Findings of the Association for Computational Linguistics: EMNLP 2023. pp. 886–896 (2023)

54. Xie, F., Zeng, X., Zhou, B., Tan, Y.: Improving knowledge graph entity alignment with graph augmentation. In: Pacific-Asia Conference on Knowledge Discovery and Data Mining. pp. 3–14. Springer (2023)
55. Xin, K., Sun, Z., Hua, W., Hu, W., Zhou, X.: Informed multi-context entity alignment. In: Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining. pp. 1197–1205 (2022)
56. Zhang, L., Li, Z., Li, Y., Wu, S., Chen, T.: Entity alignment with global information aggregation. Electronics **13**(12), 2331 (2024)
57. Zhang, Q., Sun, Z., Hu, W., Chen, M., Guo, L., Qu, Y.: Multi-view knowledge graph embedding for entity alignment. In: Proceedings of the 28th International Joint Conference on Artificial Intelligence (IJCAI) (2019)
58. Zhu, H., Xie, R., Liu, Z., Sun, M.: Iterative entity alignment via knowledge embeddings. In: Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI). pp. 4258–64 (2017)
59. Zhu, Q., Zhou, X., Wu, J., Tan, J., Guo, L.: Neighborhood-aware attentional representation for multilingual knowledge graphs. In: ijcai. pp. 1943–1949 (2019)