Received: 8 August 2019 | Revised: 11 February 2020 | Accepted: 12 February 2020
DOI: 10.1002/cpe.5746

WILEY

RESEARCH ARTICLE

Data linking over RDF knowledge graphs: A survey

Ali Assi[1] | **Hamid Mcheick**[2] | **Wajdi Dhifli**[3]

[1]University of Quebec at Montreal, Montreal, Quebec, Canada

[2]University of Quebec at Chicoutimi, Chicoutimi, Quebec, Canada

[3]Univ. Lille, CHU Lille, ULR 2694 - METRICS: Évaluation des Technologies de Santé et des Pratiques Médicales, Lille, France

Correspondence
Wajdi Dhifli, Univ. Lille, CHU Lille, ULR 2694 - METRICS: Évaluation des Technologies de Santé et des Pratiques Médicales, F-59000 Lille, France.
Email: wajdi.dhifli@univ-lille.fr

Summary

Instance matching (IM) is the process of matching instances across Knowledge Bases (KBs) that refer to the same real-world object (eg, the same person in two different KBs). Several approaches in the literature were developed to perform this process using different algorithmic techniques and search strategies. In this article, we aim to provide the rationale for IM and to survey the existing algorithms for performing this task. We begin by identifying the importance of such a process and define it formally. We also provide a new classification of these approaches depending on the "source of evidence," which can be considered as the context information integrated explicitly or implicitly in the IM process. We survey and discuss the state-of-the-art IM methods regarding the context information. We, furthermore, describe and compare different state-of-the-art IM approaches in relation to several criteria. Such a comprehensive comparative study constitutes an asset and a guide for future research in IM.

KEYWORDS

data linking, instance matching, knowledge graph, record linkage, semantic web, web of data

1 | INTRODUCTION

The linked open data (LOD) includes several resource description framework (RDF) knowledge bases (KBs) expressed by ontologies from various domains such as geography, biology, and so on.[1] These KBs are interlinked by typed links between the resources (entities or instances), denoted by uniform resource identifiers (URIs), that compose them. They permit to describe the relationships among these resources, allowing intelligent agents to navigate between KBs as if they operated a local integrated database. As a result, a richer and more enriched information is provided in response.

The majority of typed links are *identity links*.[2] By convention, an *identity link* is defined between two resources by the property `owl:sameAs`. It expresses that both resources refer to the same real-world object (co-referent).

KBs emerging in LOD are often created independently of each other. They may contain resources (with overlap descriptions) that are co-referring, but not explicitly defined. Each KB then becomes an island of isolated information. According to Reference 3, only 56% of the LOD KBs are linked. In this context, LOD can be seen as a (partial) alignment over many different KBs. Thus, smart agents would not be able to use them to take advantage of the wealth of information in the linked data. Linking co-referencing instances with `owl:sameAs` allows KBs to complement each other. This problem is the so-called *Instance Matching (IM)*.

Example 1. Figure 1 shows two distinct descriptions (given in two RDF graphs) for the famous soccer player "Lionel Messi." The aim of the IM process is to detect that the two descriptions are *identical or closely related (ie, strong similarity degree)* and eventually link them through an identity link `owl:sameAs`.

In general, any IM method can be seen as a *two-pass approach*. The first pass generates a set of potential co-referent pairs (ie, candidates) based on some criteria called blocking keys.[4] The methods implementing the first pass are called indexing methods. Indexing is also referred to as blocking in the literature. They are out of the scope and they are not covered in this article (for more details, we refer the reader to References 5-7). The

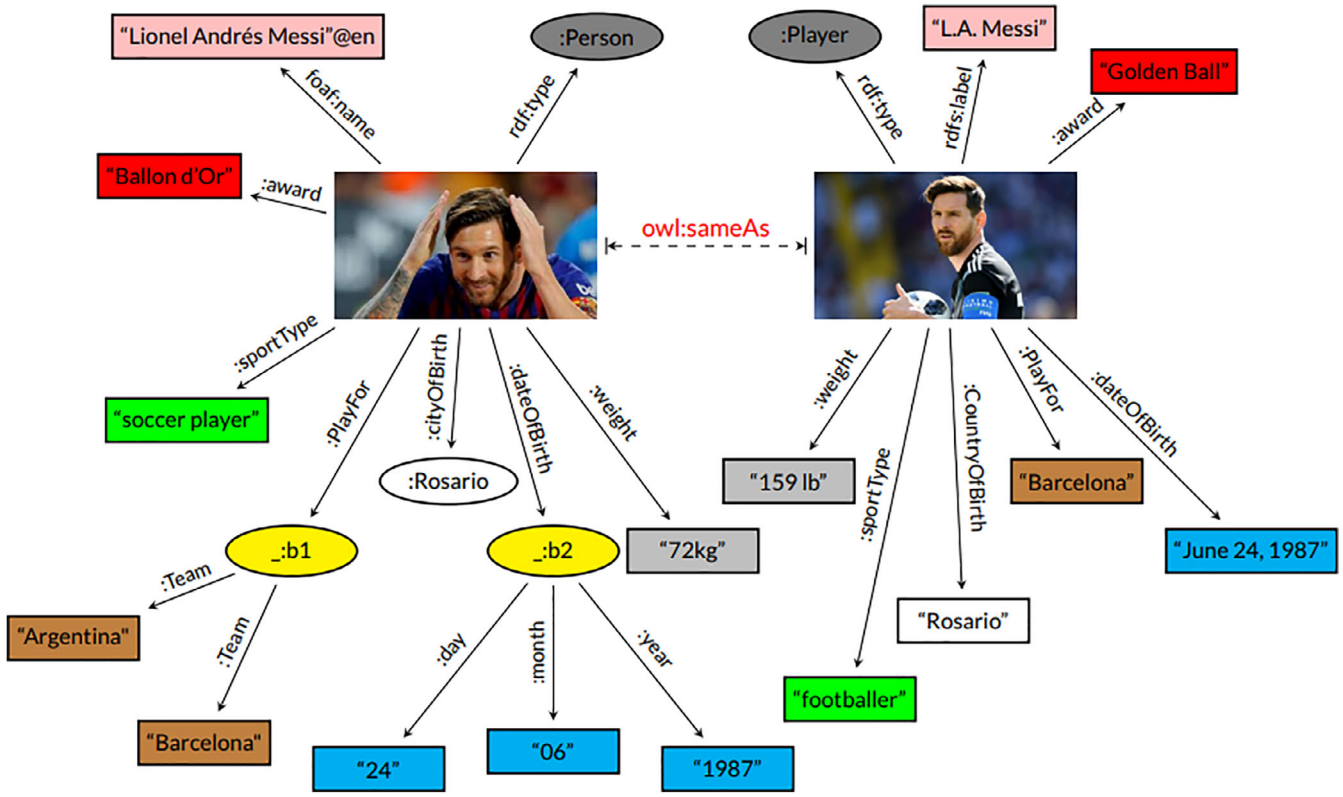


FIGURE 1 An illustrative example of the instance matching process. Two graphs in two different KBs are given to describe the soccer player "Lionel Messi." In these graphs, URIs are preceded by ":," blank nodes by "_:" and literals are enclosed in quotation marks. The images are given for simplicity and clarity of the example. In reality, each is given through a URI

second pass is to apply the IM algorithm on these generated candidates. Indexing methods serve two purposes. First, the search space is reduced by eliminating the unnecessary pairwise comparisons between instances that are unlikely to be matches. Thus, the number of pairwise comparisons to be performed by the IM algorithm will be minimized. Second, the candidate pair set should cover all the possible instance pairs that can logically be matched. These two purposes constitute a trade-off for any pruning techniques (ie, indexing methods).

Several categorizations of the existing IM approaches have already been defined.[8] Some studies categorize them into supervised and unsupervised approaches.[9,10] Supervised approaches need training data and sometimes require the intervention of a domain expert (ie, oracle). Unsupervised approaches use knowledge declared in an ontology or provided by an oracle. Depending on the degree of the expert's intervention in the IM task, some studies classify IM approaches into automated and semiautomated.[11-13] Other studies categorize them into domain-dependent and domain-independent approaches.[11-15] An IM approach is called domain-dependent when it deals with KBs related to a specific domain (eg, music domain). Otherwise, it is called domain-independent. Another categorization is schema-dependent and schema-independent[16] approaches. When the IM relies on the existence of a fixed schema is required, the approach is then called schema-dependent. Whereas, when the approach does not rely on that information, it is called schema-independent. Other studies categorize existing IM approaches into local and global approaches.[17-19] Local approaches are property matching dependent; they align the properties from two KBs which have similar semantics. These properties are then used to compare two instances from these two KBs. So, each pair of instances is explored independently. In global approaches, the properties and relations are exploited to propagate the discovered identity links to help in discovering other co-referents.

Contributions: In this article, we present several IM approaches. Our main contribution is to categorize these approaches based on their use of context. We consider as *Context* any information that can be used to characterize the situation of an instance.[20] This information plays a prominent role in the performance of an IM approach. It provides an additional evidence to match two instances. Thus, we define the context to be a synonym to the environment, where the latter can be external, neighborhood, or domain. To the best of our knowledge, the literature lacks any study that views the IM approaches through the angle of the different kinds of context like the one we are presenting. Indeed, in the literature, the context is only limited to neighbor instances, which is equivalent in this case to our definition of the neighborhood environmental context (detailed later).

Outline. The article is structured as follows. In Section 2, we define the RDF data model. Section 3 describes the IM problem, its challenges, and its general pipeline. In Section 4, we present the methodology of our survey. In Sections 5 and 6, we describe the context-free and context-dependent approaches, respectively. In Section 7, we compare these approaches according to several features. We also consider the published performance evaluation of several approaches on benchmark datasets from the IM track of the Ontology Evaluation Alignment Initiative (OAEI). In Section 8, we identify the existing challenges for the IM task, we propose some solutions to tackle them, and we discuss the remaining open challenges for future research on IM. A concluding section is given in Section 9.

2 | RDF DATA MODEL

The RDF data model[21] represents the descriptions of the resources (ie, instances and concepts) by RDF expressions, called statements,* in the form `<subject, predicate, object>`. A `subject` can be a URI or a blank node. The latter represents an anonymous resource. An `object` can be a URI, a blank node, or a basic value (eg, a string, a date, an integer, etc.). A `predicate` allows to model a relationship between the `subject` and the `object`. When the `object` is a URI, the `predicate` is called an object-type property, and when it is a basic value it is called a data-type property.[1] Formally:

Definition 1 (Statement). A statement t[1] is the smallest irreducible representation for linking one object s to another object o or a literal l via a predicate p. Formally: $t =< s, p, o >$ where $s \in \mathcal{E} \cup \mathcal{B}, p \in \mathcal{P}$ and $o \in \mathcal{E} \cup \mathcal{B} \cup \mathcal{L}$.

Definition 2 (RDF knowledge graph). An RDF knowledge graph is a set of facts in the form `<subject, predicate, object>` $\in (\mathcal{E} \cup \mathcal{B}) \times \mathcal{P} \times (\mathcal{E} \cup \mathcal{L} \cup \mathcal{B})$, where $\mathcal{E}$ is the set of instances, $\mathcal{B}$ is the set of blank nodes, $\mathcal{P}$ is the set of predicates, and $\mathcal{L}$ is the set of literals (basic values).

Alternatively, an RDF knowledge graph is a labeled multi-digraph $G = (V, E)$, where V is the set of nodes (including URIs and literals), and E is the set of edges which are URIs corresponding to predicates. Figure 1 shows an example of two RDF knowledge graphs.

Definition 3 (RDF instance). The RDF term *rdf:type*[1] is used for stating that a resource is an *instance* of a given class. Formally, the triple <*s*, *rdf:type*, *C*> declares that the URI *s* (which can appear as a subject or as an object in other triples) is an instance of the class *C*.

An RDF knowledge graph can adhere or not to a specific ontology. In both cases, it can be expressed by an RDF syntax. We recall here that knowledge graph is a specific type of KBs where the contained data is expressed in a graph format of resources (eg, instances) and semantic relationships. In the rest of the article, we write KB or KG, interchangeably (respectively, property, relation) to shortly refer to RDF knowledge graph (respectively, data-type property, object-type property).[1]

Example 2. In Figure1, the property:award in the triple <:Lionel_Messi, :award, "Ballon d'Or"> is a data-type property since it defines the literal value of the subject ":Lionel_Messi." Indeed, the property :cityOfBirth, which links the subject ":Lionel_Messi" to another resource (ie, URI) ":Rosario," is called object-type property. Finally, the property ":PlayFor" connects ":Lionel_Messi" to the unnamed resource (ie, blank node) "_:b1."

3 | PROBLEM STATEMENT: IM

IM is the problem of identifying instances that co-refer to the same object of the real world. It can be seen as a process of building identity links between the co-referent instances residing in two KBs. Formally:

Definition 4 (Instance matching). Given two sets of instances $\mathcal{S}$ and $\mathcal{T}$ belonging to two KBs (KB_1 and KB_2), the aim of IM is to discover the set $\mathcal{M}$ of identity links *owl:sameAs*, according to a chosen identity criterion, which are not already defined in the KBs. Formally, $\mathcal{M} = \{(i_1, i_2)|(i_1, i_2) \in \mathcal{S} \times \mathcal{T}, < i_1, owl{:}sameAs, i_2 >\}$ where $\mathcal{M} \not\subset KB_1 \cup KB_2$.

It is important to note that the IM task is different from the ontology alignment (OA) problem. The latter seeks to find a mapping between the concepts and properties of two ontologies, while the mission of the former is to determine the mapping between the instances of these ontologies. The majority of IM approaches exploit either the correspondences between the elements of the two ontologies or use their common elements. These correspondences are determined either manually by an expert or automatically via an alignment tool. Even if the two KBs conform to the same ontology, the identity link detection problem remains an important issue to be solved. IM and OA are considered as two

*Statement is also known as triple. In this article, they are interchangeably used.

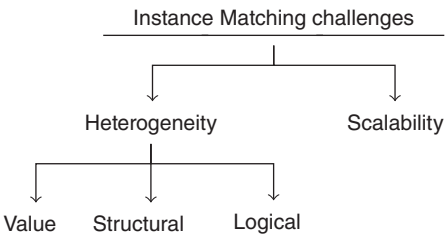


FIGURE 2 Different challenges in an IM process

interrelated problems, they are neither necessary nor sufficient to solve each other. In Section 3.1, we describe the challenges related to the IM task.

3.1 | IM challenges

IM is not a trivial task. The core challenge of IM is related to the existing heterogeneity between the compared KBs. The authors in Reference 22 categorize this heterogeneity according to three aspects: value, structural, and logical. Besides the heterogeneity aspect challenge, we add (see Figure 2) another aspect: scalability. In the following, we outline all the mentioned aspects.

3.1.1 | Heterogeneity challenges

Value heterogeneity

This aspect represents the values of the predicates. Several types of heterogeneity may exist at this level:

- *Multilingual*. The value of similar predicates can be expressed in different natural languages. For example, in Figure 1, the values of the property `:award` in both given graphs have the same meaning. Comparing these values using any string similarity metric results in a low similarity value. To solve such heterogeneity, few approaches for automatic cross-lingual data linking[23-25] are proposed. Google† Translator API and Bing‡ Translator API are, respectively, used by References 25 and 23 to translate the instance descriptions into the same language. BabelNet[26] multilingual lexicon is integrated in Reference 24 to bridge the gap between the islands of monolingual KBs. The quality of IM task in such approaches depends on the quality of translations between the multiple languages of the input KBs.[27] Recently, embedding-based models, focusing only on structural information, encode instances and relations of the KBs into the same (or different) vector space.[28-32] As a result, no machine translation is needed between cross-lingual KBs.
- *Data format*. The data format denotes the textual variation (ie, acronyms, abbreviations, etc.) of the predicates' values. For example, in Figure 1, the name of the player is given in a full version ("Lionel Andrés Messi") in the first KB and in abbreviation ("L.A.Messi") in the second KB. Another example of abbreviation can be observed in the values of the `:weight` property (ie, kg and lb). To maximize the similarity value between the two descriptions, approaches like in References 33-36 can be applied as a preprocessing step to generate the full form of an abbreviation or an acronym. In addition, textual variations can include the semantic deviation. In Figure 1, the sport type in both descriptions is described using synonym values (ie, footballer, soccer player). Thus, to solve such heterogeneity, external resources (eg, WordNet,[37] Wikipedia, word2vec,[38] etc.) can be applied to assess the semantic similarity between the predicates' values.
- *Data quality*. The Web of data is by nature an open and unrestricted information environment.[39] Everyone can publish data without any control on the data quality, which may raises several concerns:
  - *Ignoring LOD principles*. The LOD principles suggest the reuse of existing instance identifiers (ie, URIs) rather than the creating of new ones. In Figure 1, when defining the descriptions for the real-world instance (ie, Messi) in both graphs, the same URI is recommended to be reused.
  - *Inconsistency*. Values of the same properties in two similar descriptions can hold conflicting data. For example, DBpedia identifies the population of Montreal as 1 649 519, while it is 1 600 000 according to Geonames.
  - *Incompleteness*. This results from the fact that similar descriptions carry partial data for a similar property. For example, in Figure 1, the property `:team` specifies different subset values in the both descriptions.

†http://code.google.com/apis/ajaxlanguage/

‡http://datamarket.azure.com/dataset/bing/microsofttranslator

FIGURE 3 Example of outdated facts (Source: Reference 40)

```
<http://ksl.stanford.edu/people/ding/foaf.rdf#dingli>
  foaf:schoolHomepage <http://www.stanford.edu> .

<http://www.cs.rpi.edu/~dingl/foaf.rdf#me>
  cv:job_position "Research Scientist" .
```

- *Incorrectness*. The incorrectness simply refers to the data typographical errors.
- *Outdated data*. The compared KBs can include "correct" data taken at different time periods. As a result, the dissimilarity between similar descriptions may increase. For example, in Figure 3, Li Ding has two descriptions: one at Stanford University, where he was working several years ago, and another more recent description indicating his new appointment at RPI. So even if we succeed to link these two descriptions, wrong information can be generated when we integrate them. This information reflects that Li Ding is a "Research Scientist" at Stanford University, which has never been the case.

Structural heterogeneity

This aspect represents the heterogeneity at the schema (ie, ontology) level. For example:

- *Vocabulary heterogeneity*. A large number of vocabularies are designed to be used by the data publishers. Indeed, the KBs are generally created independently. Thus, the same information can be expressed using different vocabularies in different KBs. In Figure 1, the name of the player is given by `foaf:name` in the first graph and by `rds:label` in the second graph. In order to solve this type of heterogeneity, ontology matching tools can be used before the IM task or by coupling the ontology and the IM tasks together.
- *Predicate level*. This kind of heterogeneity is related to the different ways the predicate is used to model an information. In Figure 1, the date of birthday is given by a chain of predicates in the left hand RDF graph, while it is represented by a one data-type property in the right hand RDF graph. In addition, this subcategory includes the different kind of predicates used to represent an information. Again, in Figure 1, the country of birth is given by relation (ie, object-type predicate) in the left hand RDF graph, meanwhile it is given by an attribute predicate (ie, data-type predicate) in the other given RDF graph.
- *Predicate granularity*. This type of heterogeneity results from the use of different aggregation criteria for predicates representation. In Figure 1, three properties (ie, `:day`, `:month`, `:year`) are used to represent the date of birthday, while the latter is merged and given as a one piece of information in the right hand RDF graph.

Logical heterogeneity

This aspect carries out the concept hierarchical variation to which instances belong to. Co-referent pairs can be instantiated in different ways within the RDF graphs to be compared. For example, they can belong to subclasses of the same super-class without changing their interpretation. On the contrary, instances can be instantiated from disjoint classes (ie, different interpretations). Thus, even if two instances possess co-related (ie, high similarity value) descriptions, they should not be reported as co-referents. Indeed, properties of heterogeneity belong to this level of heterogeneity. A reasoning step on properties is required to infer the equivalence between two values. Two instances `:player1` and `:player2` referring to the same real world entity can have two properties that are semantically reversed (ie, the properties `leadOf` and `hasLeader`). In this case, these two properties from two different datasets convey the same information as depicted in the following example: `<:player1, leadOf, ~BarceloneFc~>` in KB1; `<:club2, hasLeader, :player2>` and `<:club2, hasClubName, ~BarceloneFc~>` in KB2. Here, the instance comparison process has to go beyond the value and property levels by comparing an explicitly specified value and an implicitly specified one between the two entities.

3.1.2 | Scalability challenge

In recent years, the amounts and availability of openly accessed data have witnessed an impressive growth. Combined with the advances in algorithmic techniques for information extraction, this has facilitated the design and structuring of information, giving rise to large-scale KBs in a "Big Data" context. In the IM process, the large amount of instances to be processed in these large-scale KBs can be very costly. Several hours or even days can be required to accomplish this process.[41] As a result, besides the qualitative challenges that the classic IM approaches should face, additional quantitative and scalability challenges are introduced. To solve these challenges, two opportunities can be pursued. (i) Indexing (ie, blocking) techniques can be used to reduce the search space. Such techniques split instances into blocks (that may overlap) and execute the matching process

for the instances only within the same block they lie in. (ii) Parallel and distributed programming models (such as Spark,[42] and Map-Reduce[43]) constitute another (complementary) technique that can be used to improve the scalability of the matching process. Indeed, their distributed architectures permit to partition the IM process into several matching tasks that could be executed in parallel across several servers (also called nodes or workers).

3.2 | IM pipeline

IM pipeline consists of three main steps: pre-processing, matching algorithm, and post-processing. In the following, we describe briefly each step (for more detailed, we refer the reader to References 8,44,45).

3.2.1 | Pre-processing

This step improves the quality of the data in the input KBs to be comparable and more usable. Possible transformations on the properties and their contents can be applied so that they have a uniform format and structure. For example, removing unwanted characters (eg, comma, semicolon, etc.) and stop-words, correcting spelling errors, and the segmentation of predicates containing several pieces of information into several other predicates or vice versa (eg, the address predicate can be transformed into several predicates: street, postal code, etc.). In addition, external linguistic resources (eg, BabelNet, WordNet,[37] etc.) can be integrated to resolve the multilingual problem.

This phase also includes the orthogonal step to the IM process which is the indexing. Remember that the indexing step allows to speed up the IM process by reducing the brute force pairwise alignment between the input KBs' instances, which is equal to $|\mathcal{S}| \times |\mathcal{T}|$. Therefore, this step is very helpful, especially for large-scale IM task because it is illogical to check all pairs of instances. To evaluate an indexing approach and estimate its effectiveness, three metrics are considered: reduction ratio, pairs completeness, and pairs quality.[46-49] Pairs completeness and pairs quality require that the set of the true match instance pairs (ie, identity information or ground truth), for instance, pairs must be available in the test KBs.

Let $\mathcal{S}, \mathcal{T}, \mathcal{C}$, and $\mathcal{A}$ be the set of source KB, target KB, instance pairs (ie, candidates) produced by the indexing approach, and the true co-referents between the source and target KBs, respectively.

Definition 5 (Reduction ratio). The reduction ratio (RR) metric measures the relative reduction in the number of instance pairs to be compared. RR is defined as:

$$\mathrm{RR} = 1 - \frac{|\mathcal{C}|}{|\mathcal{S}| \times |\mathcal{T}|}. \tag{1}$$

RR takes values in the interval $[0, 1]$, where a higher value means a higher reduction in the number of comparisons.

Definition 6 (Pairs completeness). The pairs completeness (PC) metric measures how many of the true matches are in the candidate set versus those in the ground truth (ie, true matches between $\mathcal{S}$ and $\mathcal{T}$). *PC* is defined as:

$$\mathrm{PC} = \frac{|\mathcal{C} \cap \mathcal{A}|}{|\mathcal{A}|}. \tag{2}$$

where $|\mathcal{C} \cap \mathcal{A}|$ is the number of true co-referents in the set of the candidates produced by the indexing approach. PC takes values in $[0, 1]$, where a higher value means that the considered indexing approach have placed the duplicate instance pairs in the same block. Optimizing both PC and RR can achieve the objectives behind the blocking methods. First, it can minimize the number of detailed comparisons in the matching step (ie, second pass). Second, the set of generated candidates will covers as much as possible the true co-referents.

Definition 7 (Pairs quality). The pairs quality (PQ) metric measures the proportion of true matches from the selected candidates. PQ is defined as:

$$\mathrm{PQ} = \frac{|\mathcal{C} \cap \mathcal{A}|}{|\mathcal{C}|}. \tag{3}$$

PQ takes values in $[0, 1]$, where a higher value means that the considered blocking method does not lose a lot of the true positive matches (i.e.,co-referents).

Two kinds of benchmark datasets are usually used to evaluate the performance of the IM process: real and synthetic. The latter provides a rigorous gold standard since it is automatically constructed from small size datasets. As a result, this kind of benchmarks permits a precise evaluation of the IM process. The real benchmarks are based on large-scale KBs. Identifying the complete and the correct gold standard in these benchmarks is very hard and the corresponding gold standard is generally prone to errors.

3.2.2 | Matching

The purpose of this step is to determine the status ("match" or "nonmatch") of the candidate pairs generated in the previous step according to an identity criterion. In Sections 5 and 6, we will describe the state-of-the-art approaches for IM. In fact, each generated link $l = (e_1, e_2) \in \mathcal{S} \times \mathcal{T}$ by an IM approach is:

- *True positive (TP)*: The instances e_1 and e_2 are already true matches (ie, $l \in \mathcal{A}$).
- *False positive (FP)*: The instances e_1 and e_2 are false matches (ie, $l \notin \mathcal{A}$).

Indeed, a true match that is not discovered (ie, inferred) by an IM approach (ie, missed) is the so-called *false negative (FN)*. Three known metrics are widely used to assess the effectiveness of an IM algorithm, namely, precision, recall, and F-measure.

Definition 8 (Precision). Precision measures the performance of an IM algorithm by filtering the incorrect links. Precision is defined as:

$$\text{Precision} = \frac{|\text{TP}|}{|\text{TP}| + |\text{FP}|}. \tag{4}$$

In fact, the more the precision is high, the more the applied IM algorithm is accurate in inferring the co-referents.

Definition 9 (Recall). Recall measures the performance of an IM algorithm by generating the correct links. Recall is defined as:

$$\text{Recall} = \frac{|\text{TP}|}{|\text{TP}| + |\text{FN}|}. \tag{5}$$

The more the recall is high, the more the IM algorithm will captures all the identity links in the gold standard.

Definition 10 (F-measure). F-measure is the harmonic mean of the precision and the recall. F-measure is defined as:

$$F_\beta = (1 + \beta^2) \times \frac{\text{Precision} \times \text{Recall}}{\beta^2 \times \text{Precision} + \text{Recall}}. \tag{6}$$

F-measure captures the trade-off between precision and recall. When $\beta = 1$, F-measure is known by F1-score or simply F-score.

3.2.3 | Post-processing

Sometimes, the generated set of identity links can contain erroneous `owl:sameAs`. Thus, the post-processing step refines this set to avoid inconsistencies. For that purpose, data patterns can be inferred or applied to detect the erroneous `owl:sameAs`. Functionality and inverse functionality property restrictions represent two examples of such patterns.[50] For example, when linking two movies in related KBs, most aligned movies have the same release date and an overlapping set of actors. Thus, if two aligned movies violate these patterns, then they are more likely to be wrong. Existing approaches to detect invalid `owl:sameAs` can be used in this context.[40,51-56]

4 | SURVEY METHODOLOGY

In this section, we follow the guidelines and methodologies used in References 57 and 58 for conducting our systematic review on IM approaches.

4.1 | Related surveys

The problem of IM has well been studied in the literature. This problem is referred to, in the databases community, by several keywords such as "entity resolution," "co-reference resolution," "merge-purge," "data deduplication," "instance identification," "entity matching," and so on. Indeed, several surveys were published on this problem. Reference 59 surveyed the entity resolution approaches and introduced a new term, "efficacy," for discussing the power of these approaches. Reference 60 presented a detailed analysis of the duplicate record detection approaches, including techniques to match record fields, techniques for improving the efficiency and scalability of duplicate detection algorithms, industrial

tools, and a brief discussion of open problems for this task. Reference 61 surveyed only works that rely on a similarity function when executing the entity resolution process. Reference 62 provided an overview of 11 frameworks for entity matching and their evaluations. It also studied how the corresponding approaches can be combined within a unified matching strategy. The authors stated that their comparison of criteria enables to categorize and compare more frameworks. Reference 63 provided a comprehensive empirical study that evaluates the quality of the clustering algorithms used in entity resolution approaches. This study proposed and showed that the Markov clustering can be applied for duplicate detection and it is among the most accurate and efficient algorithms for such task. Reference 64 focused on sources of uncertainty in the entity resolution process. Reference 65 discussed both the practical aspects and theoretical bases of the entity resolution problem as well as the existing tools developed in academia and industry, current challenges, and open research directions for this problem. Reference 66 surveyed entity resolution approaches mainly based on parallel solutions along with challenges and potential solutions and further research directions for parallel entity resolution. All the above surveys focused mainly on cases of structured data in data warehouse settings (relational tuples, records, XML). In the community of semantic web, "reference reconciliation," "entity linkage," "data linking," "object co-reference resolution," "instance matching," and so on, are all the terms used to refer to the same problem mentioned above. Reference 44 surveyed the existing approaches from a global perspective according to three main steps: pre-processing, IM, and post-processing. Reference 67 derived a generic architecture of ten "Link Discovery" frameworks. The authors compared their features and evaluated their performances. In contrast, the aim of our survey is to provide a comprehensive overview on works for IM specifically developed for the semantic web field, their advantages, and drawbacks.

4.2 | Search strategy

We start our systematic review by selecting a set of articles mainly focusing on the IM problem in the field of semantic web. Thus, to cover as much as possible of the existing articles, we formulate a search query based on the different terms used to refer to the IM problem (extracted mainly from surveys) in this field. The used query is the following:

- (*data reconciliation* OR *entity linkage* OR *data linking* OR *object co-reference resolution* OR *instance matching* OR *interlinking* OR *owl:sameAs* OR *identity link*) AND (*Semantic Web* OR *Web of Data* OR *Linked Data*)

This query is executed against the following five scientific digital libraries:

- ACM Digital Library.
- IEEE Xplore Digital Library.
- SpringerLink.
- ScienceDirect.
- Scopus.

4.3 | Paper selection criteria

A pertinent article to our survey must verify the following inclusion and exclusion criteria:

- *Paper Inclusion Criteria*
  - Including only articles written in English.
  - Including only articles published between 2008 and 2019.
  - Including articles focusing on the IM problem.
- *Paper Exclusion Criteria*
  - Counting each article belonging to more than one digital library only once.
  - Excluding articles that are extended abstracts and conference summarizes.
  - Excluding survey papers.

- Excluding articles focusing on indexing techniques.
- Excluding articles that not focus on IM in the semantic web field.

4.4 | Paper selection results

From the list of selected IM articles retrieved after applying our inclusion/exclusion criteria, we check their related work sections in order to identify more articles related to our study and not yet included in our set of articles. After all, we selected 53 articles to be included in our survey and that are mainly focusing on the semantic web field. These articles are published in the following journals, conferences, and workshops:

- *Journals.*
  - Journal of Computer Science and Technology, Springer - 1 article
  - Transactions on Knowledge and Data Engineering, IEEE - 1 article
  - Knowledge-Based Systems, Elsevier - 2 articles
  - Journal of Web Semantics, Elsevier - 2 articles
  - Journal on Data Semantics XII, Springer - 1 article
  - Journal of Data and Information Quality, ACM - 1 article
  - Transactions on Information and Systems, IEICE (Japan) - 1 article
- *Conferences.*
  - International Semantic Web Conference (ISWC) - 7 articles
  - International Joint Conference on Artificial Intelligence (IJCAI) - 1 article
  - International Conference on World Wide Web (WWW) - 2 articles
  - Proceedings of the VLDB Endowment (PVLDB) - 2 articles
  - Joint International Semantic Technology Conference (JIST) - 2 articles
  - International Conference on Extending Database Technology (EDBT) - 1 article
  - International Conference on Web Information Systems and Technologies (WEBIST) - 1 article
  - Conference on Artificial Intelligence (AAAI) - 2 articles
  - European Conference on Artificial Intelligence (ECAI) - 2 articles
  - Extended Semantic Web Conference (was European Semantic Web Conference) (ESWC) - 1 article
  - Proceedings of the Knowledge Capture Conference, (K-CAP) - 1 article
  - International Conference on Knowledge Engineering and Knowledge Management (still maintaining the original *EKAW* acronym and brand) - 2 articles
  - International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems (IEA/AIE) - 1 article
  - Information and Communication Technology and Accessibility (ICTA) - 1 article
  - Confederated International Conferences (CoopIS) - 1 article
  - IEEE Big Data conference (BigData) - 2 articles
- *Workshops.*
  - Identity, Reference, and Knowledge Representation (IR-KR) - 1 article
  - Linked Data on the Web (LDOW) - 3 articles
  - International Workshop on Ontology Matching (OM) - 5 articles
  - Workshop on Knowledge discovery and data mining meets linked open data (Know@ LOD) - 1 article

- ACM Symposium on Document Engineering (DocEng) - 1 article
- International Workshop on Consuming Linked Data (COLD) - 1 article
- New Forms of Reasoning for the Semantic Web: Scalable & Dynamic (NeFoRS) - 1 article
- AAAI Spring Symposium: Linked Data Meets Artificial Intelligence - 1 article
- Workshop on Web Data (WOD) - 1 article

4.5 | Categorization

The investigation and expansion of a broad set of IM approaches led us to categorize the selected approaches based on generic criteria. We utilize the context used as a complementary information in the matching step. However, an IM approach can either be context-dependent or context-free. Indeed, in each category, we also re-categorize the assigned approaches according to the main algorithmic strategy behind the matching step, applied to generate co-referents. For each (sub-)category, we identify and discuss the advantages and drawbacks of its methodology. We also compare these approaches based on a set of features usually used in the literature for the same purpose as well as four more features we added based on the rational provided in this article. In the rest of our survey, we will describe in details our new categorization for the IM problem.

5 | CONTEXT-FREE APPROACHES

Context-free techniques compare two instances by relying only on their internal local features (their specified properties). The compared instances should have some corresponding properties to be able to decide if they match or not. The structure of the instance (ie, predicates and their values) in RDF KBs can be seen as being analogous to the structure of the record in relational database. In the database community, "Record Linkage"[68] catches the same records across two tables that refer to the same real world entity. Thus, IM process is similar to the "Record Linkage" problem.

In the following, we will describe state-of-the-art approaches which inherit their "concepts" from the database community.

5.1 | Key-based approaches

One of the main approaches developed for record linkage is the use of keys. In relational databases, a key is formed by a minimal set of discriminative attributes to *uniquely* identify each record. If a correspondence between the keys' attributes across two tables can be established, then the same records are uniquely identified in both databases. Two records that share the same values for a key are both considered identical.

Definition 11 (Key). A key is a set of properties that can *distinguish* every instance in the KB.

Definition 12 (Minimal key). A key is *minimal* if none of its subsets is also a key.

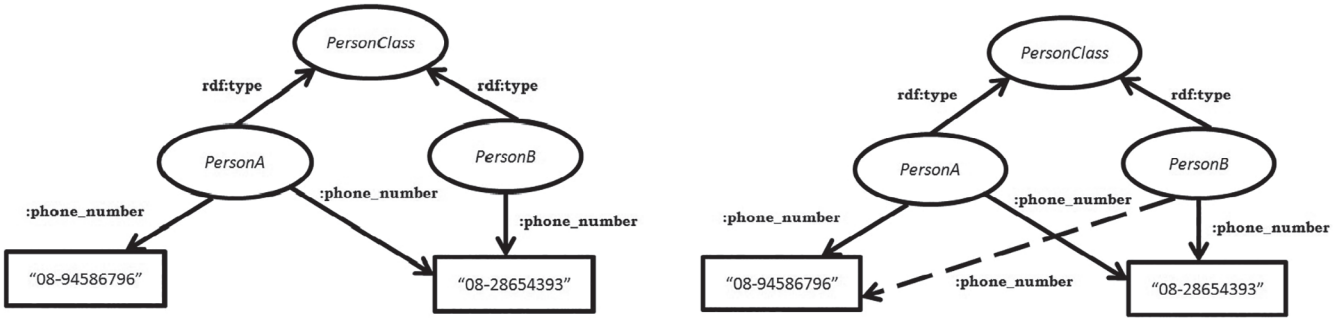
Note that, when a set of properties forms a key, all of its super-sets are also keys, that is, adding a property to the key yields another key. This key's property is known as key monotonicity.

IM can rely on key techniques developed for record linkage by taking into consideration the difference between the two domains (see Table 1). In the SW, a predicate can have multiple values, while in a relational database, an attribute can have at most one value. This crucial difference makes key discovery approaches in the database field inapplicable in SW. Some more differences between the two domains can be resumed as in Table 1. Thus, due to these differences, record linkage techniques used in databases are not suited for RDF KBs.

Several approaches have been recently proposed to automatically engender keys from RDF KBs and then exploit them for several purposes. They are used as *logical interlinking specification rule* in the IM process to infer identity links.[71-73] They are also introduced as an indexing scheme to identify the candidates.[74,75] In addition, they are used to guide the expert in selecting the more discriminant properties in order to build more complex linking rules such as in LIMES,[14] L2R[73] and SILK[76] approaches (see later in Section 5.2.1).

In the following, we decompose key discovery methods into two groups. The first one aims at finding a composite key that can contain multi-valued properties, while in the second group, the key contains only one quasi mono-valued property, that is, a noncomposite key (see Figure 4).

For the first group (ie, the composite key approaches), we categorize the existing key discoveries approaches into two groups based on the considered world assumption where the keys are discovered. The word *distinguish* in Definition 11 depends on that assumption: Closed World Assumption (CWA) or Open World Assumption (OWA)).



(A) Graph1: ":*phone_number* " is defined as Some-key.

(B) Graph2: ":*phone_number* " is defined as Forall-key.

FIGURE 4 State-of-the-art key discovery approaches

TABLE 1 Fundamental differences between database and SW domains

	Database	Semantic Web
Multi-valued	False	True
Data Volume	In order of millions	In order of billions
Semantic types	Single relation	several classes
Open World Assumption	False/True[a]	True
Unique Name Assumption	True	False
KB structure	Same schema	Multiple ontologies in the same KB

Note: *Unique Name Assumption (UNA)* states that two instances in the same KB having distinct URIs refer to two distinct real-world entities. Therefore, they cannot be co-referents. *Open World Assumption (OWA)* states that any unproven knowledge is not necessarily false.
[a]Recent works consider the incompleteness of data in the relational databases.[69,70]

5.1.1 | OWA-based approaches

In the Open World, two instances are supposed to be distinguishable if they do not share any value for all predicates of the key. OWA states that any unproven knowledge is not necessarily false. For example, in Table 2, the student_2 has no friendship with student_3. This does not mean that the fact: `<student_2 HasFriend student_3>` is false. As the incompleteness of data is a common issue in the context of SW, OWA will be the relevant hypothesis to consider in order to overcome this issue.

Example 3. In Table 2, the property FirstName is a key under OWA assumption since there does not exist student that shares its first name with another student. On the contrary, LastName cannot be considered as a key since student_1 shares the value "Monet" with student_2.

The key discovered under OWA is called Some-key. Two instances violate the key constraint if they share at least one value for each predicate in the key. Therefore, they are considered as co-referent.

Example 4. Consider the two instances PersonA and PersonB given in Figure 5A. Assume that the property : *phone_number* is a some-key. As PersonA and PersonB share a common value "08-28654393" for this some-key, then they will be considered as similar.

This key semantics coincide with the OWL2 `owl:HasKey` constructor principle,[77] which declares a set of predicates as a key for a given class. This constructor is still rarely used. In 2012, a study found that only one KB uses the `owl:HasKey` constructor.[78] In the following, we describe some approaches fitting in this category.[68,75,79-81]

TABLE 2 Toy Students Dataset

Instances	FirstName	LastName	HasFriend
student_1	Claude, Robert	Monet	student_2
student_2	Lucie	Tremblay, Monet	—
student_3	Roger	Tremblay	student_2

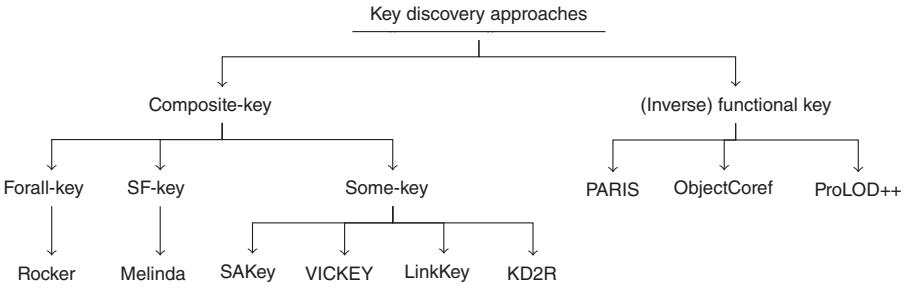


FIGURE 5 Illustrative example: some-key versus Forall key. A, Graph1: ": *phone_number*" is defined as Some-key. B, Graph2: ": *phone_number*" is defined as Forall-key

Inspired from the GORDIAN[82] method that is based on depth-first strategy, KD2R[83] determines first a set of maximal nonkeys and then derive from them the minimal keys. In this way, the number of computation of the keys is minimized.

The keys are generated independently from the IM process. Thus, to match the instances of two KBs, the keys in each KB are discovered, and then a merge step between these keys is applied. As a result, the mutual valid keys on both KBs are discovered. It assumes that the alignment across the two ontologies is available. This alignment is exploited to find the mutual keys.

KD2R[83] assumes that the UNA is satisfied. KD2R first discovers the sets of maximal nonkeys and undetermined keys are determined. Each maximal nonkey is a combination of predicates that share the same values for at least two distinct instances. An undetermined key is a combination of predicates that cannot be considered neither as a key nor as a nonkey. The case of an undetermined key happens due to the incomplete descriptions of instances in a KB. To deal with this case, pessimistic and optimistic heuristics are defined. The former supposes that the information is complete in the case of an instantiated predicate, while in the opposite case, all the existing values in the KB for that predicate can be taken. Under this consideration, undetermined keys are nonkeys. If there exist from the latter other values that do not appear in the KB, then they are different from all the already existing ones. In this case, undetermined keys are considered as keys. Therefore, a key is neither included nor equal to any maximal nonkey or maximal undetermined key. To derive keys, the union of the maximal nonkeys and undetermined keys sets is computed. In this union, each element is a set and only the maximal sets of properties are kept. For each one of them, a complement set is computed. By applying the Cartesian product on all the complement sets, all keys are generated from which the minimal subsets are returned as minimal keys. In general, KD2R performs well in the KB including classes with small set of properties. In terms of IM results, KD2R-optimistic has better results than KD2R-pessimistic.

SAKey[80] is an extension to KD2R to make it scalable. It discovers keys called *almost-keys* for a given KB even if it is huge and including errors and duplicates in the property values. This relaxed notion of a key allows *n* instances (called *exceptions*) to have duplicate values for that key. Like KD2R, SAKey finds first the maximal (n+1) nonkeys and then derive the minimal n-almost-keys. A maximal (n+1) nonkey is a nonkey having at least $(n+1)$ exceptions. For that purpose, SAKey applies filtering rules, pruning strategies (eg, elimination of irrelevant sets of properties) and ordering heuristics to prune the space of nonkeys (ie, minimize the number of candidate n-non keys). Almost-keys with 0 exceptions discovered by SAKey are the same as the keys found by KD2R with the optimist constraint.

Example 5. In Table 3, $\{F1, F2, F3\}$ shared the value "B. Pitt" and $\{F2, F3, F4\}$ shared the value "G. Clooney." Thus, the set of exception *EP* is $\{F1, F2, F3\} \cup \{F2, F3, F4\}$. As n-almost key is a set of properties where $|EP| \leq n$, therefore, the property HasActor is a 4-almost key. Similarly, {HasActor,HasDirector} is a 3-almost-key since there are three exceptions, $\{F1, F2, F3\}$.

The generated minimal mutual keys can be of large number. Thus in order to minimize the intervention of the oracle in the IM task, RANKey[84] defines a ranking metric to identify the more appropriate mutual keys to be used in the IM mission. This metric is defined as the multiplication of the support scores for a mutual key in both KBs. Defined as in Melinda,[85] support metric measures the frequency of the co-occurrence of the properties

TABLE 3 Toy Movies Dataset

Films	HasName	HasActor	HasDirector	ReleaseDate	HasWebSite
F1	Ocean's 11	B. Pitt, J. Roberts	S.Soderbergh	3/4/01	www.oceans11.com
F2	Ocean's 12	B. Pitt,G. Clooney,J. Roberts	S.Soderbergh, R. Howard	2/5/04	www.oceans12.com
F3	Ocean's 13	B. Pitt, G.Clooney	S.Soderbergh, R. Howard	30/6/07	www.oceans13.com
F4	The descendants	N. Krause, G.Clooney	A. Payne	15/9/11	www.descendants.com
F5	Bourne Identity	D. Liman	—	12/6/12	www.bourneIdentity.com
F6	Splash	Tom Hanks	R. Howard	9/3/84	—

that appear in a given key (ie, completeness of a key). The authors assume that a frequent key can lead to more identity links. Therefore, they apply a merging strategy of the identity links inferred by the top-n ranked keys. Unfortunately, the obtained results come opposite to such intuition. In fact, the top ranked keys improve the recall but they do not guarantee the precision improvement.

Unlike RANKey,[84] KeyRanker[86] applies a combining strategy of the complementary keys to improve the results. It discards the keys that generate identity links covered by other keys. In that way, the final combined keys could consist of nonconsecutive ranked key. In addition, it defines the ranking metric as the number of instances in source KB that instantiate the key and share its values with instances in a target KB. The more the instances in source KB sharing their values' key increase, the more the considered mutual key is suitable for the IM task. For every property in a key, all the pairs of instances in the source and target KBs are compared. Only the ones that have a cosine similarity greater than a predefined threshold are selected. The aggregation of the similarity scores of all the properties appearing in a mutual key should exceed a predefined threshold to consider an instance pair as similar.

VICKEY[81] discovers the conditional keys in KB where no keys or a small number of keys can be generated for a given KB. A conditional key is a valid key for instances belonging to a given class but under a specific condition. In other words, it can be seen as a key for the set of instances satisfying the considered condition in the KB. Therefore, a key is a conditional key no matter what the condition is. The instances identified uniquely by a conditional key can build a nonatomic class (ie, more complex class expression). OWL2 allows declaring such a key for a nonatomic class by using the two axioms `owl:DataHasValue` or `owl:ObjectHasValue`. Such axioms express conditions on the properties and relations values.[87] In VICKEY, to derive a relevant conditional key that permits to identify a reasonable number of instances, two measures are set-up: support and coverage. The former is the number of instances satisfying the condition imposed and instantiating the properties of the key. The latter measures the ratio of subjects in the KB identified by the conditional key. Using SAKey algorithm, VICKEY starts by discovering the maximal nonkeys from which the conditional keys can be derived. For a given maximal nonkey, several partitions of properties can be generated. Each partition consists of two subsets: condition subset with only one property and key subset with the remaining properties. Therefore, each partition yields a conditional key graph (CKG) where the nodes are the combination of the properties in the considered key subset. This graph is built only if the condition subset has a support greater than a predefined threshold. Once, all the CKGs are constructed, VICKEY discovers all minimal (conditional) keys of size one. All the nodes of a given graph containing the discovered minimal key are pruned. The resulting graph is used in the next iteration to discover the minimal keys for size two. This process is repeated on all the CKG until all the sizes are explored.

Linkkey[79] generates keys for two KBs simultaneously. The key is a maximal set of corresponding properties defined across two not disjoint classes from both KBs. It is a generalization of two aligned keys identifying the same instances between the two KBs. First, candidate keys are generated in order to reduce the number of compared properties to define the keys. A set of properties is called a candidate key if it permits to lead at least to one co-referent, and it is maximal for at least one link or it is the intersection of several candidate keys. To extract the candidate keys, the subject-property pairs are indexed according to their values. This step unfolds in each KB. Then, another index is generated by iterating over these indexes and computing for each pair of subjects the maximal set of pair of properties on which they agree. In order to select the most promising candidate keys that can generate a large number of links (co-referents) or covering all the instances, two metrics are set up to estimate their quality. The first one approximates the precision and recall on a set of co-referent samples (supervised case). The second one measures how close the extracted links would be from one-to-one and total by an assessment of the discriminability and key coverage (unsupervised case).

Relative to *some-key* semantics, in Reference 75, the authors define a key in the purpose to select the potential co-referent instances. Only the pairs of instances having partial coincidence in their key literal values are selected as potential co-referents. The key is determined based on three measures: discriminability, coverage, and their harmonic average (F_L). The discriminability of a property is computed as the ratio of its range size on the number of triples that instantiated it. It measures its `objects` diversity. The coverage of a property is determined as the ratio of the number of instances that instantiated it, on the total number of instances in the KB. It measures the instancewise frequency of a property. Initially, all the properties are selected as a candidate keys set. A property should have a minimum discriminability degree (greater than a predefined threshold) to be not discarded from this set. If F_L for a given property is greater than a predefined threshold, then this property is returned as a final key and the discovering key process ends. Otherwise, it combines the property with largest F_L with each property in the candidate keys set to create new properties. The algorithm starts again with this new set as being the candidate keys set and so on. The number of iterations depends on the predefined thresholds.

5.1.2 | CWA-based approaches

CWA is the assumption that what is not in the KB, does not hold in the real world. It is opposite to OWA. Differently from the key semantics defined previously, two instances are distinguishable if they do not share all the values for each property of the key. Otherwise, they are co-referent.

Example 6. In Table 2, the property LastName is a key under CWA assumption, since the sets of values for this property are distinct, that is, $\{Monet\} \neq \{Tremblay, Monet\} \neq \{Tremblay\}$. Indeed, LastName is not a key according to OWL2 semantics since there exist two different students that have the same last name (ie, "Monet").

Example 7. Consider the two instances PersonA and PersonB given in Figure 5B. Assume that the property : *phone_number* is a forall-key. PersonA and PersonB can be discovered as similar because they share all the values "08-28654393" and "08-94586796" for the forall-key : *phone_number*. In addition, if : *phone_number* is defined as forall-key in Figure 5A, PersonA and PersonB cannot be detected as same.

In the following, we describe two approaches in this category.[85,88] The key definition of these approaches is valid when the properties are locally complete. In ROCKER,[88] the key is called For-All key. In Melinda,[85] it is called SF-key. This latter is a trade-off between some-key and for-all key. An important thing to note is that References 85 and 88 cannot be applied directly in the IM process as no mutual keys are generated. In contrast to some-keys that are more resistant to incompleteness issue, for-all keys and SF-keys would better work in a more complete KB.

ROCKER[88] is a refinement-operator-based approach for minimal keys and almost-keys discoveries. This operator is defined over the space of properties for a given class of instances. The elements of that space are ordered in a directed tree in an ascending way based on their score function. These elements are the nodes of the tree and they are considered as candidate keys. A directed edge between two nodes means that the score of the source node is less than the score of its target. The score function is defined as the ratio of the number of instances distinguishable by the given set of properties (node) on the total number of instances in a given class. This function induces a quasi-ordering over the set of all key candidates. However, a set of properties is reported as a key if its score is equal to one (ie, it covers and distinguishes all the instances). Note that by following the order of the sets of properties with the highest scores, the number of visited nodes in the tree will be reduced. In other words, nodes that are explored first are the ones with the highest power to distinguish the instances. Whenever a key is discovered, all branches containing parts of that key are pruned from the tree based of the key monotonicity constraint. By applying this "fast search" discovery strategy, the running time is improved.

Example 8. In Table 3, according to ROCKER, there are six distinguished instances using HasActor. In contrast to SAKey, HasActor is a key ($score(HasActor) = \frac{6}{6} = 1$). HasWebSite also is a key since all the films are distinguished w.r.t HasWebSite. In fact, F6 is distinguished from other films since no other instances (ie, films) has 0 web site. Indeed, if film F5 has no web site, then HasWebSite cannot be considered as key. In this case, F5 is not distinguished using this property since another instance F6 has no web site value. Thus, HasWebSite is a 2-almost-key.

To detect the minimal keys, Melinda[85] proposes an approach where all the properties of the key are assumed to be instantiated in all the instances (we will refer to this approach by Melinda according to the project where it was developed). To verify that a set of properties forms a key, Melinda constructs a partition of the subjects (instances) induced by this set and according to their shared values. Therefore, this partition should be made-up of sub-sets of size 1. Otherwise, this set is not a key. This partition representation is in line with the partition and breadth-first search strategy applied in TANE.[89] Two measures are defined to measure the quality of a key: support and discriminability. The former is defined as the ratio of the instances having all the properties of the key instantiated on the number of all the instances. This definition is similar to the coverage metric in Reference 75. The latter is defined as the ratio of instances verifying the key constraint. In another words, it is the number of singletons induced by a set of properties on the total number of partitions. These two metrics should be greater than a predefined threshold. To deal with exceptions, a set of predicates is supposed to be a pseudo key (ie, almost-key) if the discriminability metric is bigger than a predefined threshold.

Example 9. In Table 3, the partitions generated according to attribute HasName consist of six singletons: $\{F1\}$, $\{F2\}$, $\{F3\}$, $\{F4\}$, $\{F5\}$ and $\{F6\}$. Therefore, HasName is returned as key with $support = \frac{6}{6} = 1$ and $Discriminability = \frac{6}{6} = 1$.

If F2 has only two actors B. Pitt and J.Roberts, then HasActor cannot be a key. In this case, the generated partitions according the shared values of HasActor are $\{F1, F2\}$, $\{F3\}$, $\{F4\}$, $\{F5\}$ and $\{F6\}$. Indeed, it is pseudo-key with $Support = \frac{6}{6} = 1$ and $Discriminability = \frac{4}{5} = 0.8$. In contrast to ROCKER, if *F5* and *F6* have no actors, then they will still be distinguished from each other. In this case, HasActor is a pseudo-key that hold in a part of the data (ie, F1, F2, F3, and F4).

5.1.3 | (Inverse) Functional keys

In this category, a key is defined by approximating the semantic of the `owl:InverseFunctionalProperty` (IFP) axiom defined in OWL 1. This key consists of only one representative property. It looks like the primary key (ie, noncomposite key) in a relational database. Indeed, IFP characteristic means that we cannot find two instances with the same value for this property. Otherwise, those instances are co-referent. Furthermore, the functional characteristic can lead to the same objective. A functional property is a property that each subject has with a unique object as the property's value. Thus, if $p(x, y)$ is functional in a KB and $p'(x, y')$ is functional in another KB and p matches p', then y and y' are considered as matches.

In Reference 90, the authors propose several metrics (eg, average inverse cardinality-AIC) to determine if a property can be a quasi-key. AIC is similar to the discriminability metric defined in Reference 75. The lower the AIC value of a property is, the more this property can identify a unique instance.

PARIS[91] (discussed later in Section 6.2) identifies the pseudo (inverse) functional properties during the IM process. We cited it there just to mention this identification phase. It is a context-dependent approach.

ObjectCoref[92] adopts a self-training approach to learn the discriminability of *property-value pairs* from the co-referent instances used in the training step. The definition of *property-value pairs* is similar to the definition of the inverse-functional property. The discriminability measures the ability of each pair of properties to determine in the nonlabeled data whether two instances are co-referent or not. Indeed, for a given instance, an initial set of its co-referent instances is built by exploiting the semantics of declared properties in the ontology such as `owl:InverseFunctionalProperty`, `owl:sameAs`, `owl:FunctionalProperty`, `owl:cardinality`, and `owl:maxCardinality`. From this set, the most discriminating *property-value pair* is learned in an iterative way. The property-value means that the value already exists in every instances' description in the set. Consequently, it should be infrequently seen when it is applied to the nonlabeled data to identify a new co-referent instance, which will has this value in its description. The algorithm finishes after a predefined number of iteration.

ProLOD++[93] proposes three metrics to determine if a property can be a key or not for a given cluster. This latter is a class with its subclasses in the ontology's hierarchy. *Uniqueness* measures how much the values of a property are unique. It looks like the degree of a property to be functional. The *Density* is defined as the coverage of a given property in the considered cluster. In the case where a property has multiple values, the density is defined as the sum of the separate value densities divided by the number of property values, that is, the average of all densities. When these two metrics are both equal to 1, the considered property is called a *full-key candidate*. Otherwise, the third threshold-based metric *Keyness* is defined as the harmonic mean of uniqueness and density. A property should have a keyness more than a given threshold to be a key candidate. In their study, the authors showed the different behavior in property uniqueness and density along the class hierarchy. They found three types of property keyness. When this latter decreases per class hierarchy, it is called less specific. In the opposite case, it is called more specific. Otherwise, it is generic when it stays approximately equal throughout this hierarchy. Having this KB profiling, the end-user can decide which properties could serve as keys.

Discussion

In general, keys permit to find a suitable representation (ie, selecting predicates) of instances. By using only such a representation in the matching step, the number of the selected properties to be compared between the instance descriptions will be minimized.

One of the main requirements in detecting the keys is to apply pruning strategies to avoid the scalability problem. This problem is exponential in the number of KBs' properties. Finding keys from n properties requires to exploit $2^n - 1$ combinations of these properties. Indeed, in the worst case, the computational complexity of the key-based approaches is of the order $O(2^n)$. Another requirement that can also be a limitation for applying the key-based approaches in the IM task is the existence of alignment between the properties of the compared KBs. Each predicate in a key's source KB should have a corresponding predicate in the target KB. This relationship can be seen as an "onto" function between the keys' properties. If this relationship does not exist, which is the case in a lot of cases, the found "mutual keys" cannot be used for identifying the same instances across the two compared KBs, and thus the comparison between the instances cannot be performed. Indeed, the majority of key-based approaches (eg, KD2R, SaKey, Linkkey, Vickey) assume that the ontologies (schema) of the KBs are equal. In real case scenario, KBs can include different vocabularies and such assumption is not guaranteed to be verified. Even if the alignment can be provided by applying an ontology aligner, which means that the required time for performing the IM task will be increased.

Except KeyRanker,[86] the existing key-based approaches for the IM task treat the values of a relation (object-type property) in the same way as literals. This treatment is considered as a limitation of these approaches. In fact, each KB uses a specific pattern to generate the URIs of its instances. The same real world object can be referred in two KBs by two different URIs, therefore, seeing these URIs as literal is unfair and lead to wrong results. To overcome this limitation, Reference 86 propose a rewriting scheme using the CBD (Concise Bounded Description) to render the instance descriptions more compatible. They also propose property chain transformations that explore the neighbor CBDs of a given instance to retrieve its complete information. This latter can be located multiple-steps away from this instance. As a result, the chains of properties are replaced by artificial properties. This data transformation step increases the IM running time.

Another way to treat the URIs as literals that we suggest is by applying the approach presented in Reference 94. This approach detects a pattern from the characters of the URIs. This pattern has the form (prefix-infix- (suffix)). Prefix is the URI domain. Suffix (optional) contains details about the format of the data. Infix represents the local identifier of an instance which can be used as the value of the relation involved in the key.

Linkkey, KD2R, and SAKEY provide mutual keys for the IM task without any strategy to rank them. The number of such keys is usually large. A key ranker as in References 84 and 86 could be used on top of these key-based approaches to provide such strategy. Indeed, the key ranker in Reference 86 could be affected by the heterogeneity of the key property values. For example, consider the property country as a mutual key. Suppose there is an instance source with "FR" as value for that key, and an instance target with "FRANCE" as value. So, depending on the similarity metric used, the rank of country as key will variate.

Data change periodically, and no key-based approaches take this update into account. As a result, these approaches will recompute the keys from scratch. The same reasoning applies to the approaches that calculate IFPs. A mapping between an IFP in source KB and an IFP in target KB should exist to be used in detecting the similar instances. If such mapping does not exist the same instances cannot be identified by IFPs. Note that IFPs properties are rarely defined in KB in an explicit way.

Several approaches (eg, ObjectCoref, ProLOD++, Hogan et al, etc.) include statistical measures in their approaches. The limitation of such measures is that they cannot generalize cases of nonfrequent data. Ignoring this kind of data leads to ignoring the possible `owl:sameAs` links, including such data. For example, in ObjectCoref, the $pair_1$ = (property,value) with the highest discriminative value is used to discover new co-referents. The $pair_2$ = (property,value) with less discriminative value is not considered. Thus, the instances including only $pair_2$ will not be detected as only $pair_1$ is considered.

5.2 | Interlinking rules

The idea behind Interlinking Rules (IRs) is that it is possible to identify a set of predicates that together are able to distinguish each instance. So, IRs specify the necessary conditions to consider when comparing two instances to determine if they are co-referents. The syntax of an IR can be defined as: *IF* `condition(s)` *Then* `action`. For example, the following IR $SSN(person1, number1) \wedge SSN(person2, number1) \Rightarrow sameAs(person1, person2)$ asserts that if two persons (ie, instances) share the same social security number (SSN), then they are considered as identical in the real world. In the following, we classify the rule-based approaches in two groups based on how they are generated: manually or using machine learning techniques.

5.2.1 | Manual rule-based approaches

IRs can be defined manually by a domain expert, as in the semi-automatic tools: SILK,[76] LIMES,[14] and ScSLINT.[95] In these tools, the IRs are expressed by a script file that is composed of two components: a string-based similarity function that allows comparing the values for the given aligned properties of the instances and aggregation operators that can be used to combine these similarities to get a more complex composite similarity measure. The latter represents the final similarity score between two instances.

SILK[76] provides a declarative language (Silk-LSL) to describe the script file. The `condition(s)` uses different aggregate-based operators such as min, max, average (weighted), and so on. To reduce the search space of comparison between instances, SILK includes a pre-match step to select instances that could be co-referents. The expert defines the properties that should be used in the pre-match step. To accomplish this goal, a multidimensional index, called MultiBlock,[74] is applied.

Example 10. An example of a script file[§] defined in SILK is given bellow:

```
<Silk>
 <Prefixes> ... </Prefixes>
 <DataSources>
    <DataSource id="dbpedia"> ... </DataSource>
    <DataSource id="geonames"> ... </DataSource>
 </DataSources>
   <Interlinks>
    <Interlink id="cities">
      <LinkType>owl:sameAs</LinkType>
      <SourceDataset dataSource="dbpedia" var="a">...</SourceDataset>
      <TargetDataset dataSource="geonames" var="b">...</TargetDataset>
      <LinkageRule>
        <Aggregate type="average">
            <Compare metric="levenshteinDistance" threshold="1">
                <Input path="?a/rdfs:label"/>
                <Input path="?b/gn:name"/>
            </Compare>
            <Compare metric="num" threshold="1000" >
                <Input path="?a/dbpedia:populationTotal"/>
```

[§] https://app.assembla.com/wiki/show/silk/Link_Specification_Language

```
            <Input path="?b/gn:population"/>
          </Compare>
        </Aggregate>
      </LinkageRule>
      <Outputs>
        <Output type="file" minConfidence="0.95">
          <Param name="file" value="accepted_links.nt"/>
          <Param name="format" value="ntriples"/>
        </Output>
        <Output type="file" maxConfidence="0.95">
          <Param name="file" value="verify_links.nt"/>
          <Param name="format" value="alignment"/>
        </Output>
      </Outputs>
    </Interlink>
  </Interlinks>
</Silk>
```

In this example, the aim is to identify across the source KB (namely Dbpedia) and the target KB (namely, Genonames), the instances that denote the same cities in the real-world. For this purpose, the expert states in this script file that pairs of cities are compared based on their name and their population properties. To compare the names, the Levenshtein string similarity metric is used, while for the population ,a similarity measure proposed by Silk for numerical values is applied. Two cities are considered as co-referents if the average similarity of these two properties is greater than 0.95.

LIMES[14] uses the triangle inequality in order to reduce the number of comparisons between the KBs. This inequality splits the metric space of comparisons into subspaces represented each by an example (ie, an instance y in the target KB T). Therefore, knowing the distance from an instance x in the source KB S to an example permits to approximate the distance from x to any other instance belonging to the subspace of that exemplar. This approximated distance is inferred by computing a lower and upper bounds of the distance, that is, $m(x, z)$ in the following equation:

$$m(x, y) - m(y, z) \leq m(x, z) \leq m(x, y) + m(y, z)$$

where $x \in S$, $(y, z) \in T \times T$, $m(x, y)$, and $m(y, z)$ are known. Therefore, the set M in Definition 4 is approximated by computing the set $M' = \{(s, t) \in S \times T, m(s, t) \leq \sigma\}$, where m is a distance measure and $\sigma \in [0, \infty]$ is the distance threshold. The number of comparisons is then reduced by computing only the exact distance $m(x, z)$ when the lower bound (ie, $m(x, y) - m(y, z)$) is smaller than σ. It is worth noting that aggregation functions are available such as *min* and *max*, which are also combined with distance measures verifying the triangular inequality. The measures that do not satisfy this mathematical inequality like the *Jaro-Winkler* measure cannot be applied. To reduce the cost of executing the script file an execution optimizer called HELIOS[96] was implemented in LIMES. This optimizer combines techniques such as PPJoin+[97] and HR3[98] with theoretical operators to generate possible plans, approximate their running time and select the one that is most time-efficient. Moreover, ROCKER was integrated as part of LIMES.

ScSLINT[95] is a scalable interlinking framework. It consists of six modules. (i) Property alignment generator selects the important predicates using discriminability and coverage metrics. The former is similar to the one defined in Reference 25, while the latter computes the frequency of a given predicate over all the instances in the KB. These important predicates are then aligned using an overlap measure. This threshold based metric computes the overlap between the objects of two given predicates to decide if they can be aligned or not. (ii) Similarity function generator module assigns similarity measures for each predicate alignment. This module uses the aligned predicates from the previous module as input and initializes a list of initial similarity functions. This module permits the users to configure new similarities metrics in this framework. (iii) Candidate generator module detects the candidates' pairs where two instances are considered as candidates if they share at least one first token in their objects for any string property alignment between them. An annotation step is applied to this candidates set by using a set S of co-referents as labeled input data. All the candidates pairs belonging to S are labeled positive (ie, $(s, t) \in S$). The other candidates pairs that include the first element (s) of an existing positive labeled pair are labeled negative (ie, (s, z) where $z \neq t$ is labeled negative).[99] Finally, the candidates pairs are divided into training and validation sets and they are provided to the fourth configuration module. (iv) This latter generates a default configuration by the intervention of the expert. A configuration specifies the similarity functions, the similarity aggregator, and the co-reference filter with all their corresponding parameters. (v) Similarity aggregator module executes the similarity functions and the aggregation function (eg, linear, quadratic, and

Boolean) specified in the previous module to compute the final matching score for the unlabeled candidates. (vi) The final module, co-reference filter, applies an adaptive filter based on stable matching problem[100] to generate the final co-referents from the matching scores generated in the previous module. In another words, two instances will be considered as co-referent if their final matching score is larger than any score where these instances appear.

5.2.2 | Machine learning rule-based approaches

Another way to define the IRs is by using (semi) automatic learning techniques that generate *optimal* combinations of properties, similarity measures, and thresholds used to compute instances similarity. A large spectrum of approaches exists based on this strategy.

Genetic programming (GP) is a supervised learning method largely used in the IM domain.[101,102] GP starts with an initial set of candidate solutions. In IM, this set consists of potential IR rules. In each iteration, the fitness of all candidate IRs in the current population is computed and a new evolved population is generated using three genetic operators:[103] reproduction, crossover, and mutation. By applying one of these operators, an individual from a current population evolves to the new population. GP ends either when a predefined number of iterations have been reached or an optimal solution has been detected (solution with fitness closer to the desired solution).

GenLink[101] models the IRs as trees that can be easily understood and improved by an expert. These rules combine the properties selected for the comparison. The transformation operators that can be applied to the values of these properties in order to normalize them are the similarity operators (the comparison operator and the comparison aggregation operator) and the relative thresholds. The initial population used by GenLink is not completely random. It is composed of IRs built from pairs of properties having at least one lexeme (value) in common. GenLink stops in the case where a predefined number of iterations is reached or when the F-Measure of an IR tends to 100%. The best IR in the final population is returned and then applied to determine the co-referencing instances.

Example 11. An example of an IR learned using the GenLink approach is given in Figure 6. The aim is to interlink cities. Thus, the learned rule compares the labels and the coordinates of the instances in order to generate identity links. The labels are converted first to lower case prior to comparing them using the Levenshtein string similarity metric while allowing for a maximum distance of 1. The coordinates are compared using a geographic similarity score. Both labels and coordinates similarity scores are then aggregated into a single score by using the minimum aggregation. As a result, two cities are considered as co-referents if both scores are greater than 0.5.

To overcome the problem of the availability of the training data, ActiveGenLink[104] combines GP and active learning for IR generation in an interactive way. It applies GenLink for IM task but after the generation of the training set. This set is populated by exploiting one of the two active learning strategies: query-by committee (QBC) and query by-divergence (QBD). They select the most informative candidate instances pairs to be judged by an oracle. The former chooses the pair of instances whose committee vote, composed of the candidate IRs (current population), gives the most disagreement. The latter selects the candidate that has the maximum divergence (different) from any existing identity link already generated. However, QBD allows only the IRs that cover a specific link in the training set to vote.

EAGLE[102] approach is similar to ActiveGenLink. The learned IRs are represented as a tree. However, EAGLE does not support the transformation operators applied to the property values (eg, normalization, tokenization, and concatenation of values) as defined in GenLink.

Other approaches model the IM process as a classification problem as in References 105-107. Therefore, the aim of the IM process is to find a classifier that projects the co-referent and non-co-referent instances into two separate classes.

RAVEN[105] uses active learning for IM. It finds a pair of classes whose instances are to be linked. Then, a set of corresponding properties that represents the instances belonging to those classes is determined. Two properties are aligned if there exists an overlap between their object values.

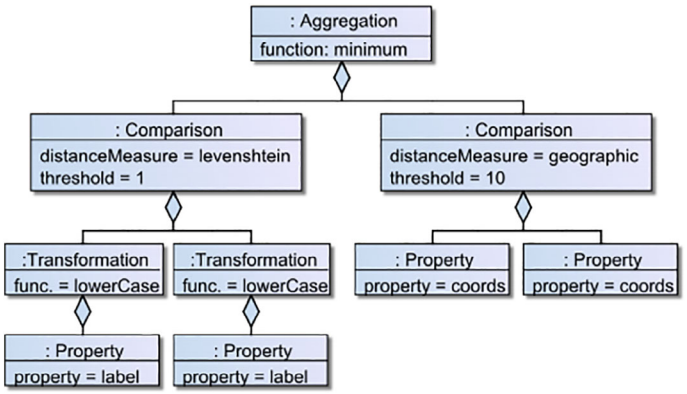


FIGURE 6 Example of an expressive interlinking rule (Source Reference 101)

The previous steps are done by providing some links to an oracle to criticize by using the uncertainty sampling strategy. This query strategy selects the pairs for which the current model is the least certain. In fact, they design two classifiers: linear and Boolean. The former is composed of two corresponding properties where each one is from a distinct KB. The latter is a conjunction (logical and operator) of several linear classifiers and it represents the IR used later to generate the co-referents. In another words, the IRs are represented as a conjunction of the corresponding properties of the two compared set of instances.

As the matching of properties is not a trivial task, Adaboost[106] proposes a schema-free approach by extracting literal information from the instances. The similarity of a pair of instances is computed based on the similarity of their literal information. This similarity is represented by a vector computed by three string-based metrics. The vectors are then fed to a binary Random Forest classifier to perform the IM task. It is important to note that this approach uses *Transfer Learning*[108] to reduce the manual labeling work for the training examples.

In Reference 107, the authors use machine learning techniques to train an SVMLight[109] classifier for detecting co-referent `FOAF` instances. The pairs of instances are assessed by using a number of property-specific features such as IFP, string-based metrics for values, properties comparison, and partial analysis of the graphs centered on the instances by following the `foaf:knows` properties. The set of the properties relevant to the classification problem is assumed to be given. Indeed, this approach cannot scale for KBs with many properties.

SLINT+[110] is a training-free older release of ScSLINT.[95] It consists of four modules. The predicate selection and predicate alignment modules play together the same role as the property alignment generator module in ScSLINT. The main difference is in the overlap metric which follows the same idea of Jaccard metric. The aligned predicates are then used in candidates selection and IM. The fourth module is the IM. It computes the similarity between two instances as the weighted average of the similarities of the objects over each aligned predicates between them. The weight of two predicates is equal to their overlap measure as previously computed. The similarity of two sets of objects is computed using the TF-IDF metric. This module applies the same co-referent filter, adaptive filter principle, as in ScSLINT.

cLink[99] is a supervised approach, which is implemented as a part of ScSLINT. It uses a heuristic search method permitting to learn an optimal matching configuration (ie, combination of similarity functions as well as other settings of the IM process) defined in ScSLINT.

NjuLink[111] builds a small training set of 20 positive and 20 negative examples. Then, it extracts a set of discriminative property pairs. In fact, each pair of properties is discriminative if its value in terms of information gain on the training set is greater than a predefined threshold (ie, the authors used 0.2). Finally, a combined set of instance pairs from the source and target datasets is defined where the similarity for each pair is mainly computed based on the available pairs of discriminate properties between them.

Discussion

The manual configuration of IRs by a domain expert requires remarkable efforts. Without these efforts, IRs lack precision. The expert must have a good knowledge on the compared KBs. This knowledge enables him to choose the discriminant predicates for comparison by taking into account the incompleteness of data. Indeed, they must specify the suitable modification to be applied on the contents of the predicates to normalize them before the comparison step. The measures of similarities appropriate to the contents of the predicates must be chosen in order to minimize the influence of noise (eg, spelling mistakes). They are also asked to determine the good thresholds and the suitable aggregation functions (IRs) corresponding to the KBs in consideration. All these requirements become so hard to define when the KBs contain a huge number of triples and predicates. To alleviate the discriminant predicates requirement, the keys can be used to form IRs in cases of approaches that require rules from the users (eg, SILK, LIMES, etc.).

Note that the manual rule-based approaches apply the rules in one-shot manner and they cannot be chained (ie, no reasoning step will be applied). Thus, the recall will be low due to incomplete data in which the defined rules cannot by triggered to infer the identity links. Another major drawback of the machine learning-based rules approaches is that their effectiveness heavily depends on the availability of training sets and the high quality of links in these sets. The quality of these links plays a very important role for the precision of the obtained IRs. These links should cover all the particular cases in the data that include ambiguities. GP-based approaches provide high F-measure but require significant running time as they will check every combination of properties to generate the optimal IRs. The generation of IR is not deterministic and depends on the initial population. By changing the initial population, another IR can be provided that is completely different as the genetic algorithm can get stuck in a local optimum. Also, determining the IRs via the corresponding properties can miss some correct co-referents. The existing case, where the two co-referent instances have a set of corresponding properties not completely or partially included in the IR, cannot be returned as co-referent. Note that NjuLink was designed for a specific task in OAEI and it is not applicable to all the tasks especially when the positive and negative examples belong to the same semantic type (ie, class). This limits its applicability to real world datasets which usually includes multiple classes.

6 | CONTEXT-DEPENDENT APPROACHES

Context-dependent techniques are based on the idea of performing IM by considering not only their local features (ie, data-type properties), but also their relationships with their "contexts." We see the context to be similar to the environment. As stated in Reference 20, context is any information that can be used to characterize the situation of an entity. We adapt this definition to the use we desire. Enumerating the "information" used to match two instances permits us to define the "context" as follows:

- *External Environment (EE)*: represents the external resources connected to the compared KBs. By resources we mean either the expert which confirms the matching result of the IM approach or the external data (eg, KBs, dictionaries, etc.) where the intended instances take advantage to enrich their descriptions (Figure 7).
- *Neighborhood Environment (NE)*: represents the surrounding objects (ie, neighbors instances), which often present important information about intended instances. The interaction between an instance and its neighbors can be viewed in two ways. An instance can include the similarity of its neighbors when it looks for its co-referent or the similarity between two instances can be used to guide the IM process by just considering in their neighbors as a good source for the IM to process.
- *Domain Environment (DE)*: represented by the information projected from the semantic of the domain of the KBs used in the IM process. More precisely, using specific ontology to describe KB's instances permits adding constraints from the ontology domain to infer new knowledge. This latter adds more source of evidence in the IM process. We note that in some cases the context is very related to the semantic of the domain of interest of the KBs. Detecting that two authors of a given scientific paper are identical helps in identifying more co-referents (eg, the co-authors of the considered paper). This means that the knowledge representing the semantic of the domain of interest should be translated and introduced in the IM task to serve as a context evidence.

In the following, we categorize the IM methods into static and dynamic collective approaches. By "collective" we mean the approaches including the context in the matching process. The context used in each of the collective approaches are cited in Table 4.

6.1 | Static collective approaches

In static collective approaches, all the sources of the similarity evidence are estimated only once per candidate pair. In other words, a noniterative framework will be applied to resolve the IM problem. In the following, we describe the state-of-the-art approaches for this type of methods.

6.1.1 | Schema-free approaches

MinoanER[112] relies on a blocking scheme defined as a disjunction of different evidences coming from the values, names, and neighbors of the candidate instances. The names and the neighbors are, respectively, selected by recognizing the most important predicates (ie, properties and relations) in the input KBs. Indeed, the support and discriminability for each predicate are computed. The entire instance names and the tokens existing in the instance descriptions are considered as blocking keys. Thus, name and token blocks are inferred. Four threshold-free heuristics are applied (in order) on these blocks to filter them and discover co-referent pairs. These heuristics rely on a value-based similarity metric, which is based on the intuition that if two instances share many infrequent token in their descriptions, then they will have a high similarity value. Thus, the number and frequency of the shared tokens can be derived from the number and the size of common blocks shared by two instances, respectively. Block statistics also permit to determine the instance similarity based on neighbors. This similarity aggregates the value-based similarity of all instance pairs that are *important* neighbors for the candidate pair. MinoanER shows a significant result when it is applied on KBs exhibiting high heterogeneity.

VDLS[1,113] generates for each instance a virtual document (as in VMI[114] and I-Match[115]) from its local description (ie, data-type properties) and instances related to it through object-type properties (ie, k-hop neighbors). Then, it transforms the IM problem into a document matching problem and solves it by a vector space embedding technique. Indeed, it considers a pre-trained word embeddings to assess words similarities at both the lexical and semantic levels.

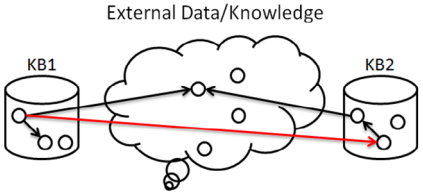


FIGURE 7 Utilizing external data/knowledge

TABLE 4 Summary table of the main instance matching approaches

Approaches	Domain	Input	Output	PP	Context	PD	DD	M	A	OA	H	Auto
RDF-AI[25]	✗	script file	owl:sameAs	CC	EE	✓	✓	✓	✓	✓	✗	-
SILK[76]	✗	script file	owl:sameAs	✗	✗	✓	✓	✓	✓	✗	✗	-
LIMES[14]	✗	sript file	owl:sameAs	SM, HR	✗	✓	✓	✗	✓	✗	✗	-
RiMOM-IM[131]	✗	script file	owl:sameAs	validation	NE	✓	✓	✗	✓	✗	✗	-
ObjectCoref[92]	✗	RDF	owl:sameAs	✗	✗	✓	✓	✗	✗	✗	✗	+
SERIMI[134]	✗	RDF	owl:sameAs	✗	DE	✓	✓	✗	✓	✗	✗	+
GNAT[127]	music	RDF	owl:sameAs	✗	DE	✓	✓	✗	✓	✗	✗	+
Rowe[128]	Social Network	RDF	owl:sameAs	✗	DE	✓	✓	✗	✗	✗	✗	+
PARIS[91]	✗	RDF	owl:sameAs	✗	NE	✓	✓	✗	✓	✓	✗	+
VMI[114]	✗	script file	owl:sameAs	✗	NE	✓	✓	✗	✗	✗	✗	-
SBUEI[15]	✗	RDF	owl:sameAs	✗	NE	✓	✓	✗	✗	✓	✗	+
RinsMatch[130]	✗	RDF	owl:sameAs	✗	EE	✓	✓	✗	✗	PA	✗	-
MinoanER[112]	✗	RDF	owl:sameAs	Reciprocity	NE	✗	✓	✗	✓	✗	✗	+
ViewSameAs[116]	✗	script file	owl:sameAs, viewSameAs	✗	NE	✓	✓	✗	✗	✗	✓	-
Adaboost[106]	✗	RDF	owl:sameAs	✗	✗	✗	✓	✗	✗	✗	✗	+
Import-by-query[120]	✗	script file	owl:sameAs, owl:differentFrom	✗	EE	✓	✓	✗	✗	✗	✗	-
ProbFR[119]	✗	script file	owl:sameAs, owl:differentFrom	✗	NE	✓	✓	✗	✗	✗	✗	-
LN2R[73]	✗	script file	owl:sameAs, owl:differentFrom	✗	NE	✓	✓	✗	✓	✗	✗	-
SLINT+[110]	✗	RDF	owl:sameAs	SM	✗	✓	✓	✗	✓	PA	✗	+
cLink[99]	✗	RDF	owl:sameAs	SM	✗	✓	✓	✗	✓	PA	✗	+
ScSLINT[95]	✗	RDF	owl:sameAs	SM	✗	✓	✓	✗	✓	PA	✗	+
GenLink[101]	✗	RDF	owl:sameAs	✗	✗	✓	✓	✗	✓	✗	✗	-
ActiveGenLink[104]	✗	RDF	owl:sameAs	✗	✗	✓	✓	✗	✓	✗	✗	-
EAGLE[102]	✗	RDF	owl:sameAs	✗	✗	✓	✗	✗	✓	✗	✗	-
RAVEN[105]	✗	RDF	owl:sameAs	✗	✗	✓	✗	✗	✗	✓	✗	-
Nikolov-2010[133]	✗	RDF	owl:sameAs	(I)FPP	EE	✓	✓	✗	✗	✓	✗	+
Lesnikova-2014[23]	✗	RDF	owl:sameAs	✗	EE	✗	✓	✓	✗	✗	✗	+
Lesnikova-2015[24]	✗	RDF	owl:sameAs	✗	EE	✗	✓	✓	✗	✗	✗	+

TABLE 4 (Continued)

Approaches	Domain	Input	Output	PP	Context	PD	DD	M	A	OA	H	Auto
RANKey[84]	✗	RDF	owl:sameAs	✗	✗	✓	✓	✗	✗	✗	✗	+
KeyRankey[86]	✗	RDF	owl:sameAs	✗	✗	✓	✓	✗	✓	✗	✗	+
NjuLink[111]	DOREMUS	RDF	owl:sameAs	✗	✗	✓	✓	✗	✓	✗	✗	-
VDLS[1]	✗	RDF	owl:sameAs	✗	NE	✗	✓	✗	✗	✗	✗	+
Legato[118]	✗	script file	owl:sameAs	HC	NE	✗	✓	✗	✓	✗	✗	-
LogMap[121]	✗	RDF	owl:sameAs	OM	EE	✓	✓	✗	✓	✓	✗	+
AML[135]	✗	RDF	owl:sameAs	OM	EE	✓	✓	✗	✓	✓	✗	-
Lily[137]	✗	RDF	owl:sameAs	OMD	NE	✓	✓	✗	✓	✓	✗	+
I-Match[115]	✗	RDF	owl:sameAs	✗	✗	✗	✓	✗	✗	✗	✗	+
BIGMAT[138]	✗	RDF	owl:sameAs	SBM	NE	✗	✓	✗	✗	✗	✗	+
HolisticEM[140]	✗	RDF	owl:sameAs	✗	NE	✓	✓	✗	✗	✗	✗	-

Abbreviations: CC, consistency checking; (I)FPP, (inverse) functional property pattern; HC, hierarchical clustering; HR, hospital-resident; OM, Ontology modularization; OMD, Ontology mapping debugging; PA, predicate alignment; SBM, symmetric best match; SM, stable marriage.
Note: -: semi-automatic (needs a human input file), +: automatic.

I-Match[115] collects first from the local description of each instance a bag of words (BoW). Then, the similarity between two instances is computed based on the similarity of their normalized strings (in their Bows) using the NLP techniques. Two instances are considered as co-referents if their similarity is above a predefined minimum threshold.

Discussion

The main advantage of MinoanER and VDLS is that are both based on statistical metrics to define the importance of words to distinguishing the instances. In fact, the more a word is importance, the more it encompasses a discriminative information. Indeed, MinoanER defines the harmonic mean of the predicates to select the *N* most important ones, which will be used in the matching step. Thus, this cut-off of the predicates based on their statistical importance may only capture frequent predicates and ignore other important yet rare predicates. Therefore, the identity links that MinoanER generates could be incomplete. VDLS gives promising results due to 1) the incorporation of a large set of information, 2) the use of a pre-trained word embeddings during the instance comparison making the approach take into account words semantics. However, this overwhelmingly large set of information makes this approach suffer a high computational cost. As a result, its usage in large KBs matching scenario becomes difficult. Similarly, I-Match cannot be used in a large scale scenario since it applies a pairwise similarity between instances of a source and a target KBs (ie, no indexing method is applied).

6.1.2 | Translation-based approaches

Approaches in this category suppose the compared KBs are expressed by different languages. The IM process is translated to be a document matching problem. Thus, it can be solved by using a traditional vector space technique. We cite two works in this category.[23,24]

In Reference 23, the authors introduce an approach to process the IM described by different natural languages. In each KB, each instance is represented by a virtual document consisting of textual information extracted from its local features (eg, rdfs:label, rdfs:comment, etc.) and from the local features of its neighbors (ie, n-hope). Once the extraction step for each instance in both KB is done, a matching translation step is executed to translate both virtual documents created in a pivot natural language. A statistical translation engine, Bing Translator API, is used for that purpose. The similarity between two instances is then computed by the cosine similarity of their (translated) virtual documents by using the standard term weighting scheme TF-IDF.

Instead of statistical translation engine, the authors in Reference 24 use a multilingual lexicon (ie, BabelNet) to project words onto the same semantic space. Once the virtual documents are constructed, each term in them is replaced by its identifier (ID) from the multilingual lexicon. Word sense disambiguation step is applied in case more than one sense exists for a given term in order to select the pertinent sense. The similarity between two instances is computed similarly as in the previous approach.

Example 12. In Figure 8, the resource (ie, instance) on the left is in English, while the resource on the right is in Chinese. For each resource, a corresponding document is constructed. Then, the two documents are mapped to a common identifier using the babel lexicon. Finally, the similarity between the two resources is computed based on the similarity between their common identifiers.

Discussion

The quality of IM task in this category depends on the common "medium" used to perform the IM. In the case of the statistical translation engine, three limitations can be cited: (i) it could not support translation from one language to another language. For example, Bing API does not support Asian languages. Thus, translating language from English to Chinese cannot be handled. (ii) During translation, some words could not be translated.

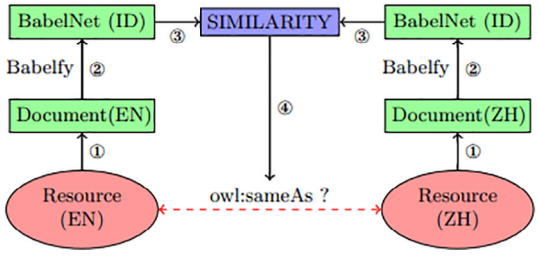


FIGURE 8 Example of an instance matching method based on a multilingual lexicon (Source: Reference 24)

In Reference 23, these words are eliminated from the virtual documents and then are not included in the IM step. (iii) The "lack of context" in the virtual documents is generated from the RDF data due to the use of unnatural language. In a natural language, a text document consists of sentences where each has a well-formed structure. The meaning of a given word, especially which has an ambiguous or has multiple meanings (ie, polysemous or homonyms), can be resolved by its accompanied words (ie, context). Therefore, the lack of context returns the translation step to be word to word translation without respecting the meaning.

In the case of the multilingual lexicon, the sense disambiguation step used can also be affected by the lack of context. Indeed, the incompleteness of the multilingual lexicon can lead to discarding the terms not found in it as it is designed in Reference 24. Discarding words from a virtual document is in itself a limitation due to the fact that these words may be useful and specific to represent the corresponding instance.

6.1.3 | Property classification-based approaches

Approaches in this category are based on classifying instance properties. For example, a property can be classified as a *Discriminative* or a *Descriptive* property. The former (eg, `rdf:type`) can be used directly to distinguish the instances, while the latter (eg, `rdfs:comment`) can rather be used to describe the instances.[114,116-118]

In Reference 116, the authors use the *discriminative* properties to generate candidates. A light version of identity link called `ViewSameAs` is defined to track the instances sharing discriminative property values. Therefore, this new link is established between those instances (candidates). The *descriptive* properties are then used to refine the candidates set. If two instances are linked by `ViewSameAs` and sharing similar descriptive properties values greater than a predefined threshold, then this link is replaced by `owl:SameAs`.

In order to increase the number of identity links, the authors in Reference 117 refine the generated set of `ViewSameAs`. For each instance, in `ViewSameAs` relation, a cluster is created saving the information of the number of similar descriptive property values shared between itself and the second part of the mentioned relation. After the clusters are generated, bags of instances are created. From a set of instances related by `ViewSameAs`, a master instance is selected including all the descriptive properties values of the other instances in this set. Those instances form a bag. The `ViewSameAs` is replaced by an identity link between the master instance and each instance in that bag. References 116 and 117 did not provide any evaluation showing the effectiveness of their approaches.

VMI[114] classifies the information extracted from the instances' properties into six categories: URI, name, schema information of the instance (Meta), descriptive property values, discriminative property values, and neighbors. For each instance, two vectors are created: name vector and virtual document vector. The former contains the name information, while the latter includes the descriptive property values and the sum of the weighted neighbor information of the given instance. Neighbor information consists of the vector name and the descriptive property values of that neighbor. Two inverted indices are used to index each group of vectors. Candidate instance pairs are selected using some simple heuristic rules applied on the inverted indexes. By comparing the values of the corresponding discriminating properties of a candidate instance pair, the initial generated candidates pairs are refined. The similarity between the component of a resulting candidate instance pair is computed as the root mean square of the cosine similarities between their vector names and their virtual document vectors.

Legato[118] is a five-stages IM system. It first filters the mono-property keys that are valid on both KBs. Then, it creates profiles (ie, vectors) from the instances descriptions and their neighbors by using NLP techniques (tokenization and stop-words removal). After that, it computes the correlation between the created vectors by applying the cosine similarity and fixing the similarity threshold. The filtering step can be considered as a property classification into key and nonkey properties. Finally, to reduce the number of false positives, Legato clusters instances, within each KB, which have highly similar descriptions. Then, it identifies the similar cluster pairs. For each group of pairs, co-referents are generated based on the similarities using a best-key property.

Discussion

Approaches in this category depend mainly on a configuration file introduced by an oracle. This file contains the classification of the properties, their alignments, and the meta information of the two KBs. The lack of information concerning the properties used in the candidate selection pass reduces the efficiency of these approaches. In addition, considering a large range of information for matching instances could increase the precision of results, but also hinders the running time that increases drastically with the number of properties. We would like to note here that Legato requires the type of instances to be compared. However, in cases where this information is not available, Legato cannot be used. Indeed, some instances may not include their types due to data incompleteness. Consequently, these instances will be omitted in the matching step and, thus, the recall will be decreased.

6.2 | Dynamic collective approaches

In contrast to the approaches presented in Section 6.1, dynamic collective approaches apply an iterative process to assess the different source of evidence, specifically the impact of neighbor similarity in the IM process. At each iteration, the aligned instances are used to match the remaining instances. State-of-the-art approaches in this group are decomposed in three subcategories and discussed in the following paragraphs.

6.2.1 | Reasoning-based approaches

Approaches in this category handle logical rules with reasoning strategy to infer identity links. Constraints such as inverse functional properties, transitivity, and domain knowledge are defined as logical rules. Approaches as in References 71-73,119-121 fit in this category.

L2R[72] adopts a purely logical deduction-based approach by using a set of Horn rules. These rules are expressed using a newly designed language called RDFS+. It combines a set of ontology axioms (FPs, IFPs, `owl:disjointWith`, etc.) as well as the expressive SWRL rules. By using this language, a set of rules is generated from the axioms defined in the KB (eg, UNA) and the ontology axioms. A forward chaining algorithm (SLD) is then used to propagate similarities and to infer new facts: identity and nonidentity (`owl:differentFrom`) links. The two compared KBs are assumed to conform to the same ontology (RDFS+).

N2R[73] implements a numerical approach to solve the IM task. It models the dependencies between instances similarities by a graph. This graph is expressed by a system of nonlinear equations, which is resolved iteratively (until reaching a fixed point) by a method inspired from Reference 122. In this system of equations, the similarity between two instances of the same type is represented by variables while the similarities between basic values are expressed by constants. The latter are computed by using a string-based similarity metric. Each variable strongly depends on the similarity value obtained from the properties that are involved in the *common* keys (IFPs) and weakly on the similarity of other predicates that do not belong to the keys. When the two compared KBs are described by two different ontologies, the same procedure in KD2R[83] is used to generate the common declared keys. It is worth noting that L2R and N2R can be applied separately or in combination. This combined approach is LN2R.[123]

To take into account the similarities of the comparable but nonaligned properties, N2R-part[71] extends N2R for that objective. The similarity between two instances becomes dependent on the similarities of their common identified keys, their mapped properties and the others comparable properties.

Import-by-query approach[120] is a backward chaining algorithm, which combines local reasoning with external querying to overcome the local data incompleteness. It alternates steps of query rewriting and of distant query evaluation. The query rewriting step is made by adapting the Query-SubQuery algorithm[124,125] originally developed for answering queries in Datalog. Thus, by enriching the RDF KBs with Datalog rules, Import-by-query builds "on demand" for each targeted `owl:sameAs` (ie, initial query) a sequence of sub-queries to some entry points of linked data. This imports the specific missing triples in order to infer identity links. In average, import-by-query requires three iterations of rewritings.

In contrast, ProbFR[119] introduces a forward-chaining algorithm for reasoning with uncertain RDF facts and rules in order to generate the identity links. Each input fact (certain or uncertain) is modeled as a probabilistic fact. That permits to unify modeling of any kind of uncertainty as probabilistic Datalog rules. Rules and facts are associated with different symbolic events denoting when the corresponding facts or rules are true. For each inferred fact, an event expression is computed. This expression encapsulates its provenance. It is defined as a Boolean combination of all symbolic (or expression) events associated with the rules and facts implicated in its generation. The probabilities of facts are computed from the event expressions. They can be seen as probabilistic weights associated to rules.

LogMap[121] is an iterative IM system that relies on lexical and structural inverted indexes to improve its scalability. The authors used an external lexicon (eg, WordNet or UMLS Lexicon) to enrich the inverted indexes in order to boost the matching between the resources. Indeed, to reduce the number of logical inconsistencies between candidates, LogMap integrates a reasoning and repairing (ie, ontology modularization[126]) techniques. On the one hand, the reasoning technique allows to generate the basic hierarchies of resources in the intended ontology. On the other hand, the repairing technique is executed on the initially generated candidates (from the intersection of the inverted indexes). These candidates rely on a string matching metric which takes into consideration the similarity between their neighbors.

Discussion

Approaches in this category assume that the logical rules are given by the expert. They also suppose that the correspondence between the schema of the compared KBs is solved before the IM task will be executed as in L2R and N2R, or given as rules by the expert as in import-by-query and ProbFR. Even if the difficulty to define the rules by an expert can be resolved by integrating key-based approaches, these approaches depend heavily on the data. In fact, except ProbFR,[119] all the mentioned approaches in this category assume the applied logical rules as certain. Theoretically, this observation over clean data guarantees the intended approach to reach a high precision. But in practice, this assumption is not always verified due

to the data itself such as its incompleteness and its possible errors (eg, misspellings). A small error in the decision of similarity of the two instances will affect the other decisions for other instances.

6.2.2 | Domain-dependent approaches

Approaches in this category assume that the data comes from similar data sources. They adhere to a specific domain. GNAT[127] and Rowe[128] fit in this category. In both approaches, the instances are represented as graphs, in which the nodes are linked instances and their properties. Then, the matching phase becomes a graph matching technique.

GNAT[127] is an approach to match music-related KBs expressed according to similar ontologies. The similarity between two instances is computed according to the similarity between the corresponding triples in their graphs. The similarity between two corresponding triples is the similarity between their two subjects and their two objects. The similarity between two resources (subjects or objects) is determined by comparing literals directly attached to these resources using string-based metrics. The graph similarity value between two instance graphs is then equal to the summed similarity values of each possible pair of resources (potential co-referents) from these graphs normalized by the number of those pairs. The component of each pair of resources in the graph with the highest similarity value presents the final co-referents.

Rowe[128] matches similar profiles (instances) of `FOAF:Person` in a social graph by computing the graph similarity. Three techniques for this purpose are introduced. The first one considers two graphs are similar if they share a large number of vertices and/or arcs. The second is the one defined in Reference 127. The third is the reasoning on the graph which is based on the comparison of the values of the `FOAF:name` property in the two graphs.

Discussion

Approaches belonging to this category are applicable on KBs described by domain specific ontologies. These KBs involve special type of data (eg, audio fingerprinting in GNAT or user biographical information in Rowe), which require specific similarity functions. In other words, domain-specific approaches rely on specific-domain similarity functions, depending on the specificity of the involved data type. As a result, this prevents them to be applied on generic KBs.

6.2.3 | Ontology evidence-based approaches

Approaches in this subcategory integrate more evidence from the ontology level in order to solve the IM process. Taking advantage of the OA can be considered as an example of such evidence. According to Reference 129, ontology matching is the process of defining corresponding classes, properties, and relations between two ontologies. In the following, we describe the state-of-the-art approaches fitting in this subcategory. Some of these approaches[130-132] assume that the equality between the schema (ie, ontology) of the compared KB is given. They only concentrate on comparing the data at the instance level by taking advantage of the schema equality. Other pieces of related works use the result of the IM step in the schema matching and vice-versa. For example, PARIS[91] and SBUEI[15] interleave the instance and the schema matching steps to find the co-referent pairs. Indeed, some related works as the approaches proposed in Reference 133 and RDF-AI[25] also exploit the result of each step in a sequential way. Another way to take advantage of the ontology level is to rely on the premise that the co-referent must belong to two approximately equivalent classes between the two compared KBs as in SERIMI.[134]

RiMOM-IM[131] computes the similarity between two instances by using a threshold-based weighted similarity metric, which aggregates the similarities of all their aligned predicates. The similarity between two aligned relations is computed as the Jaccard similarity between their objects (ie, instances). Indeed, RiMOM-IM uses the predicates with their values (ie, tokens and URIs) available in the instance descriptions as blocking keys to generate the candidates. More precisely, it applies two heuristics to generate some initial co-referents and then uses them to generate iteratively more new co-referents. The first Unique Subject Matching heuristic exploits each block of size two where each instance come from a different KB and considers them as co-referent. The blocking keys (predicate,object) for such that blocks means that (predicate,object) occurs only once in each input KB. The instances in every KB verifying such blocking keys constraint are denoted by a unique instance set (UI). Thus, based on such constraint, more co-referents will be generated. For example, suppose that (a, b) are discovered as matched, then the unique subject neighbors $(c, d) \in (UI_{\text{Source}} \times UI_{\text{Target}})$ of (a, b) such that $< c, p, a >$ and $< d, p, b >$ will also be considered as co-referent. The second heuristic is called one object left. It generates more co-referents based on the following assumption: if the descriptions of two matched subjects (s, t) have an aligned predicate pair (p, q) with k objects of which $k - 1$ are matched, then the k-th remaining object pair (u, c) will be resolved as co-referent. For a candidate pair not belonging to URIs, the defined similarity metric is used to identify them as a match or not.

RinsMatch[130] uses graph locality, neighborhood similarity, and the Jaccard measure to compute the co-referents. It is based on the assumption that elements of two graphs are similar when their adjacent elements are similar. If the similarity between two instances is higher than a predefined threshold, then they will be returned to the user to confirm if they can be merged or not. RinsMatch retains all their predicates. If two predicates are connected to a common object (ie, same neighbor), then they will be returned to the user for confirmation if they are two similar predicates or not. Indeed, RinsMatch pairs the object nodes that are connected with a common predicate to both subjects in the already merged node. The paired objects will be also returned to the user for confirmation. The algorithm stops when no more new matching candidate pairs are suggested.

The authors in Reference 132 build the graph of an instance in the same way as in Reference 15. They consider the graph as a bag of paths. The similarity between two instances is equal to the weighted average of their most similar comparable paths. Two paths are comparable if they have the same length and their properties in the sequence are also comparable (for example, two properties related by a subsumption relation). The weight of a pair of the most similar paths is equal to the average of their weights. In fact, each path has a weight defined as the product of the discriminability of each property in the path by its factor. The discriminability of a property is calculated as in Reference 75. The factor of a property in the path is the value indicating the importance of the `object` of that property relative to its parent (`subject`) in the graph. It is an inverse to the number of the paths starting from that `subject`. Note that the similarity between two comparable paths is equal to the similarity between their last two nodes. If these nodes are literals, then a string-based similarity metric is used. If they are URIs and their strings are not identical, then the similarity is calculated recursively.

SBUEI[15] adopts an interlacing approach that uses the result of the IM task to solve the schema matching task and vice-versa. Indeed, this approach takes two similar concepts from two schemas and then detects the co-referent instances between them. To do that, for each instance, a graph of a given depth is built using its properties and relations. This latter permits to add to the graph other neighbor instances belonging to another concept. In other words, each graph is treated as a bag of subgraphs where their number is equal to that of the neighbor-instances. Therefore, once a co-referent is detected, SBUEI reapply the same IM strategy but between their neighbor-instances. In this way, the IM is guided to find co-references around the two instances which are certainly identical. Once more two neighbors-instances co-referent are detected, the concepts to which they belong are considered to be potentially similar. Similar concepts are those having similar instances. By using some OA algorithms, SBUEI detects new similar concepts and it reuses them to find their co-referent instances by applying the previous IM method. This procedure is performed until no new concepts are found. The similarity between two graphs of instances is based on the similarity of their properties values using a string-based similarity metric. The final similarity score between those instances is the weighted average sum of the most similar properties in both graphs. The weight of a given property is an inverse to the depth of the `subject` of that property in the graph to which it belongs.

PARIS[91] identifies the pseudo (inverse) functional properties. For this purpose, the functionality degree of a property is computed as the harmonic mean of the local functionality degrees across all the instances. It represents the probability of a property to be a (inverse) functional. Therefore, only properties with prominent degrees of (inverse) functionality are favorable. This means there exist scarce co-referent instances. The semantic of this metric is similar to the discriminability defined in Reference 75 with the difference that it is defined for all instances in a KB and not for instances belonging to a given class as in Reference 75. It approximates the some-key notion. The local degree functionality of a property for a given instance (ie, a subject of a triple) is the number of its distinct values. By analogy, an inverse functional property degree is defined. The probabilities between the equivalent instances (of the two KBs) are calculated by taking into account the probabilities of the equivalent properties and vice versa. PARIS stops when the similarity values between the instances do not change anymore from an iteration to another, or the predefined number of iterations is reached. The spread between the IM and the alignment of properties give PARIS a specific originality.

RDF-AI[25] proposes a five sequential independent steps IM approach: preprocessing, matching, fusion, interlink, and post-processing. In this approach, the instances and their properties build graphs. The similarity between two instances is calculated as the average of the similarities between the neighboring nodes. A neighboring node similarity value is calculated using a string-based similarity metric between the objects of two comparable properties. More precisely, the algorithm generates all pairs of values, called candidate-pairs, from each two comparable properties. At each iteration, the highest similarity value of a candidate-pair is selected, and all the other candidate-pairs including one of the values of the selected candidate-pair are discarded. This process continues until all the possible candidate-pairs are fixed (reconciled). For a given instance, the value of a predefined property is used to select all of its candidates. RDF-AI selects the highest similarity value between an instance and its candidates.

In Reference 133, the authors present an approach performing the ontology schema matching in order to improve the performance of the IM process. It starts by producing class correspondences and property correspondences from a provided set of identity links. These links are defined through intermediate repositories which are exploited as a background knowledge for ontology matching techniques. Then, the discovered ontology correspondences will be used to identify the subsets of the compared KBs, which probably contain co-referring instances. The identified subsets can be exploited by any IM approach.

SERIMI[134] implements an unsupervised learning approach that combines the direct matching and the class-based matching to resolve the IM task across heterogeneous KBs. A KB is supposed to be partitioned into disjoint components representing classes. A component consists of a set of instances where each two of the latter must share at least one feature (eg, a predicate). Therefore, for a given component, SERIMI starts by selecting for each instance, in this component, its set of candidates is called a *pseudo-homonym* set. This selection is done by applying a direct matching between the value of the most discriminating property in that component and the instances in the target KB sharing that value. The pseudo-homonym set of

an instance may contain several candidates that may belong to different classes in the target KB, thus requiring a disambiguation phase. This phase aims to find for each instance in the input component, a co-referent from its pseudo-homonym. The principle of disambiguation is based on the fact that for the given instances of the same component, their pseudo-homonym sets would reflect similarities that distinguish a co-referent for each instance of the starting component. In another words, similar components are those which have similar instances.

AgreementMakerLight (AML)[135] is originally an ontology matching tool derived from AgreementMaker.[136] In this approach, the output of multiple matchers is combined by simply joining the alignments and keeping the highest similarity in case of repeated mappings. This output can be exploited in an interactive selection way (based on the similarity scores) to pick candidates for re-checking, also to take into account specious mapping pairs. In addition, AML enables the user to review and reject mapping candidates. It also permits to displays information about the conflicting mappings.

Lily[137] is an ontology mapping that combines several matching techniques to facilitate alignments (eg, instance ontology matching, generic ontology matching, etc.). It is based on the extraction of semantic subgraph (ie, neighbor graph) and exploiting both linguistic and structural information (in this subgraph) to generate one-to-one alignments between the compared ontologies. Then, a similarity propagation strategy is applied to produce more alignments. After that, an ontology mapping debugging technique is used to improve the alignment results.

Discussion

It is clear that the ontology matching adds more evidences permitting to overcome the obstacle of the ontological heterogeneity in the IM process. Indeed, it could be damaging if it is done badly. In the following, we discussed the main inconvenience of each of the above mentioned approaches.

In PARIS experiments, no guarantee of convergence is proved. Most calculations are ended after two or three iterations. In addition, in some cases an entity is described by relations in a source KB while these relations could be modeled by other entities in a target KB. This kind of structural heterogeneity cannot be discovered by PARIS. In SBUEI, for each two similar concepts, each instance belonging to one of them (source concept) will be compared with all the other instances of the other concept (target concept). Without applying any heuristic to reduce the number of the candidate instances, this approach cannot scale to large KBs. In Nikolov et al, the OA produced is partial. Only equivalent relations between two ontologies are undiscovered. In addition, the third-party repositories, used as background knowledge for schema matching techniques, are not always available for different ontologies. SERIMI gets competitive results with other approaches where the two compared KBs include a high structure heterogeneity. Yet, it gets bad results compared to other approaches in the case where all the pseudo-homonym sets contain instances belonging already to the same class. In such a case, the refining step (ie, class-based disambiguation step) will not have any effect to reduce the candidates. RinsMatch also does not provide any details on how the candidate pairs are selected. Integrating statistical metrics (ie, $\breve{U}$ functionality, discriminating property-value, etc.) can be used to solve this requirement. In addition, RinsMatch does not apply any heuristic to prune the pairing predicates when merging a pair of co-referents. Such requirements become mandatory in the large scale KBs scenario. They can reduce the intervention of the expert in the IM process and, therefore, render the approach automatic. In Song et al, the approach ignores the modeling cases where the cycles exist in the instances' graphs. Such cases are very likely to be faced in the SW. Indeed, sometimes the instance for which we want to find the co-referent will be the object in all the triples where it appears (sink node). Consequently, building its graph is not available. It is not clear how the authors in this approach solve this case.

6.2.4 | Random walk-based approaches

Approaches in this category are based on the random walk technique. The latter designs the input data as a stochastic graph where the vertices usually correspond to the data objects. Then, a random walk is executed on all the paths in this graph to deduce the importance (ie, rank) of the vertices. These ranks are then used to reconstruct the mapping between the nodes of the source and target KGs.

BIGMAT[138] is an IM approach implemented in Spark[42] and based on an affinity-preserving random walk technique. It represents the IM problem as a graph-based node ranking[139] and selection problem in a constructed candidates association graph. It first builds an expanded candidates association graph consisting of pairs (each as a graph node) of IM candidates. The edges between nodes reflect their pairwise structural harmony. Then, it ranks each candidate pair through the stationary distribution vector computed from a Markov random walk strategy on that graph. Candidate pairs with higher rank scores in this vector are more likely to be co-referents.

HolisticEM,[140] a Map-Reduce[43] based approach, constructs a similarity graph from candidate pairs, generated based on the overlap across their attributes and their neighbors. The local and global information from the similarity graph are propagated using personalized page rank[141] to update and compute the final similarity scores for the candidate pairs.

Discussion

Random walk-based approaches can be considered as nodes ranking methods. They lie mainly on a "good" constructed stochastic graph. The nodes of such graph are the candidates between the source and target KGs. These candidates are usually of large number and thus relevant heuristics are required to reduce their numbers. Indeed, some source instances may have some target instances with the same scores. Thus, selecting the relevant target ones is made arbitrary. Thus, an adequate strategy becomes mandatory to prevent some probably generated false positives instance pairs.

6.2.5 | Discussion of dynamic collective approaches

Instances are often described by a large number of predicates. Therefore, including all of them in the IM process can increase the number of required comparisons between the instances descriptions. All the approaches in the dynamic collective based approaches (except SERIMI) suffer from that problem. In fact, some neighbors can include more information about an intended instance. They have more impact of that instance. Thus, favoring those neighbors can reduce the number of comparisons. MinoanER[112] solves this problem by estimating which relations in the neighborhood relationship are more beneficial to consider in the matching step.

7 | COMPARISON OF IM APPROACHES

In this section, we summarize several IM approaches based on several features widely used in the literature about the IM process.[67,142] We add four new features to the already identified ones: context, property-dependent, data-dependent, and history. As a result, the features are the following:

1. *Domain*. This feature specifies if the approach is related to a specific domain. Otherwise, it is domain-independent.
2. *Input*. This feature specifies the required input by the approach and its format.
3. *Output*. This feature specifies the kind of output produced by the approach (ie, *owl:sameAs* triples).
4. *Post-processing (PP)*. This feature specifies if the approach provides any treatment (ie, consistency checking and inconsistency resolution) after the IM step.
5. *Context*. This specifies the type of context category in the IM approach.
6. *Property-dependent (PD)*. This feature specifies if the IM approach depends on the result of property matching (ie, properties from different data sources, which have similar semantics).
7. *Data-dependent (DD)*. This feature specifies if an approach is compatible to various "formats" of data.
8. *Multilingualism (M)*. This feature specifies if the system can match RDF KBs where the values are described by a different natural language.
9. *Availability (A)*. This determines if the approaches' codes are publicly available for usage (Please see Table 5).
10. *Ontology alignment (OA)*. This feature specifies if the IM approach can support OA.
11. *History (H)*. This feature specifies if the IM approach saves the instances sharing several values of important properties (ie, local features).
12. *Automation (Auto)*. This criterion specifies if an IM approach needs human input as a script file (ie, property alignment, property classification, similarity threshold, types of instances, etc.) or not.

Table 4 provides a comprehensive overview of the considered approaches according to the aforementioned features. We do not include the "technique used" as a feature in each approach. Such feature is already presented in the detailed description of each of them. If the intended approach requires a configuration file, then we mention that by a script file in the input column. Otherwise, we note RDF to describe the two KBs RDF files. Two things worth remembering are: first, the majority of key-based approaches which compute only the keys (ie, KD2R, SAKey, ROCKER, Linkkey, VICKEY, and Melinda) can be integrated in the pre-processing step, they did not determine the *owl:sameAs*. For this reason, they did not appear in our comparison table (ie, Table 4). A summary of these approaches is given in Table 6. Second, the empirical studies to illustrate the comparison between the different surveyed approaches is unfeasible due the fact that the codes (of the almost approaches) are not available.

7.1 | Comparison of evaluation results

In this section, we analyze the empirical results for the surveyed approaches on benchmark datasets from the instance track of the OAEI. Due to the fact that the implementations of most approaches are unavailable, we only report the results of the approaches that either participated in this track or used the benchmark datasets in their papers evaluations. We compare the considered approaches in terms of precision, recall, and F-measure, w.r.t., to a gold standard including the existing mapping between source and target KBs.

TABLE 5 An overview of available code approaches

Approaches	Codes
RDF-AI[25]	https://code.google.com/p/rdfai/
SILK[76]	https://github.com/silk-framework/silk
GenLink[101]	included in SILK[76] implementation
ActiveGenLink[104]	included in SILK[76] implementation
LIMES[14]	http://aksw.org/Projects/LIMES.html
EAGLE[102]	included in LIMES[14] implementation
RiMOM-IM[131]	http://keg.cs.tsinghua.edu.cn/project/RiMOM/
SERIMI[134]	https://github.com/samuraraujo/SERIMI-RDF-Interlinking
GNAT[127]	http://sourceforge.net/p/motools/code/HEAD/tree/
PARIS[91]	http://webdam.inria.fr/paris/
MinoanER[112]	https://github.com/vefthym/MinoanER
LN2R[73]	https://gforge.inria.fr/projects/ln2r/
SLINT+[110]	http://ri-www.nii.ac.jp/SLINT/index.html
ScSLINT[95]	http://ri-www.nii.ac.jp/ScSLINT/index.html
cLink[99]	included in ScSLINT[95] implementation
KeyRankey[86]	https://github.com/HFarah/KEYRANKER-2017
NjuLink[111]	https://github.com/nju-websoft/njuLink
Legato[118]	https://github.com/DOREMUS-ANR/legato
LogMap[121]	https://github.com/ernestojimenezruiz/logmap-matcher
AML[135]	https://github.com/AgreementMakerLight
Lily[137]	https://drive.google.com/file/d/1irGjC4tZdofpG57kHXpblBJcf75ZwUWf/view

7.1.1 | OAEI benchmark datasets

Synthetic data

- *PR (Persons/Restaurant)*. This benchmark, provided in OAEI 2010, includes data about persons (Person1 and Person2) and restaurants (Restaurant) which are artificially modified by adding duplicates and variations of property values. The datasets in this benchmark are relatively of small size.
- *SPIMBENCH*. This benchmark consists of two sets of datasets with different sizes including SPIMBECNH Sandbox and SPIMBENCH Mainbox. They describe data about creative works (NewsItem, BlogPost, and Programme) and they are generated and transformed using the Semantic Publishing IM Benchmark (SPIMBENCH) by altering the original source instances based on their values, structure, and semantics in order to create the target instances. This task is provided in the instance track of 2017 and 2018.

Real-world data

- *DI*. This benchmark, provided in OAEI 2010, includes five datasets related to healthcare: (i) Sider that includes data about drugs and their effects, (ii) DrugBank that is about drugs and their chemical information, (iii) Diseasome that is about disorders and genes, (iv) Dailymed that is about marketed drugs and chemical structure, usage, and adverse reactions for the drugs, and (v) Dbpedia that is a large scale dataset.
- *DOREMUS*. Following a common ontology given by the DOREMUS project, this task includes data about musical works and events. They are coming from the catalogs of two French cultural institutions named the National Library of France (BnF) and the Philharmonie de Paris (PP). This task includes two sub-tasks. This first one, denoted HT, includes different types of heterogeneities such as multilingualism, differences in catalogs, differences in spelling, different degrees of description, different property names for the same instance types, missing property values, missing titles, and use of synonym, and so on. The goal is to check how the different IM approaches deal with these heterogeneities. The second subtask, denoted FPT, includes instances with highly similar descriptions across the two compared

TABLE 6 Summary of the key discoveries tools

Approach	Optimization	Class distinction	World assumption	Input	Output
KD2R	Gordian	Yes	OWA	NA	NA
	Pruning strategies				
SAKey[a]	Gordian	Yes	OWA	N-Triples	(n-almost) Keys
	Pruning strategies				
VICKEY[b]	Gordian	Complex Class	OWA	N-Triples	Unique Conditional Keys
	Pruning strategies			Nonkeys file	
				Support threshold	
ROCKER[c]	Fast Search	Axiom Class	CWA	N-Triples	(n-almost) Keys
Linkkey[d]	Inverted-index	Axiom Class	OWA	Source KB (RDF/XML/TTL)	Keys
				Target KB (RDF/XML/TTL)	#instances
				owl:sameAs file	Discriminability
					Coverage
					H-mean
Melinda[e]	TANE	Axiom Class	CWA	Turtle	(pseudo-)Keys
				RDF/XML	XML/N3 file
				N-Triples	Support
				RDF/JSON	Discriminability
				TriG	
				N-Quads	

[a]https://www.lri.fr/sakey
[b]https://github.com/lgalarra/vickey
[c]http://github.com/AKSW/rocker/
[d]http://melinda.inrialpes.fr/linkkey/
[e]https://scm.gforge.inria.fr/anonscm/git/melinda/melinda.git

datasets. The aim is to show the ability of IM approaches to correctly disambiguate these instances and avoid the generation of false positives.

7.1.2 | Results on OAEI tasks

Table 7 reports the results of several IM approaches on the PR benchmark provided in the IM track of 2010. The results show that this benchmark is easy to solve for Person1 and Person2. The poor performance of SERIMI especially on Person2 is because all candidates belong to the same class and there is not enough information for SERIMI to distinguish and disambiguate instances. For Restaurant dataset, the provided (ie, gold standard) mappings contain some errors which makes the comparison between the approaches unfair or at least difficult.

The F-measure results for DI benchmark are reported in Table 8. The results show that this benchmark was hard for the majority of the approaches. Noting that, Sider, DrugBank, and Dailymed contain for each instance multiple aliases and several nonmatching instances have several aliases. However, ObjectCoref uses the names and aliases as discriminative properties. These properties do not match well, which leads to a poor result for Sider-Diseasome comparing to the other approaches. In contrast, the class-disambiguation applied in SERIMI seems to be effective in refining candidates especially for the dataset-pairs, Sider-Diseasome, and Sider-Drugbank, which were problematic for the alternative approaches. We note that the results for Paris are reported from cLink paper.[99]

In 2017, four approaches have participated to the SPIMBENCH task: AML, I-Match, Legato, and LogMap. The results of these approaches are reported in Table 9. The results show that AML and I-Match have a high recall but a low precision on this benchmark. In contrast, Legato and LogMap achieved a very good precision but do not succeed to get a large number of the expected correct links, leading to a low recall. The good precision results of Legato and LogMap could be due to the fact that they take into account several issues related to the structure heterogeneity. Differently,

TABLE 7 Results in terms of F-measure of different instance matching approaches on the PR benchmark datasets

Approach	Restaurant (2010)	Person1 (2010)	Person2 (2010)
LN2R	0.75	**1.0**	0.94
ObjectCoref	0.73	**1.0**	0.95
RIMOM	0.81	**1.0**	0.97
PARIS* (fixed)	0.91	**1.0**	**1.0**
SERIMI*	0.75	**1.0**	0.46
MinoanER* (fixed)	**1.0**	—	—
VDLS* (fixed)	0.98	**1.0**	**1.0**
BIGMAT* (fixed)	**1.0**	**1.0**	**1.0**

Note: The superscript "*" means that the results of the approach were obtained outside of the OAEI competition. The term *fixed* means that the results of the approach were obtained on the corrected version of the Restaurant dataset.

TABLE 8 Results in terms of F-measure of different instance matching approaches on the DI benchmark datasets

Approach	Sider-DrugBank	Sider-Diseasome	Sider-Dailymed	Sider-Dbpedia	Dailymed-DBpedia	Dailymed-Sider	Diseasome-Sider
cLink*	0.911	0.824	0.777	**0.6414**	0.424	—	—
Adaboost*	0.903	0.794	0.733	0.641	0.375	—	—
ObjectCoref	—	0.45	—	—	—	0.70	0.74
RIMOM	0.504	0.458	0.629	0.576	0.267	0.62	—
PARIS*	0.649	0.108	0.149	0.502	0.219	—	—
SERIMI*	**0.97**	0.87	0.66	0.55	**0.43**	0.67	0.87
VDLS*	—	0.81	0.76	—	—	0.75	0.81
BIGMAT*	—	**0.88**	**0.78**	—	—	**0.78**	**0.88**

Note: The superscript "*" means that the results of the approach were obtained outside of the OAEI competition.

AML and I-Match present similar performances on both Sandbox and Mainbox datasets, while Legato and LogMap have better performances on Sandbox than on Mainbox datasets. In 2018, only three approaches, namely, AML, LogMap, and Lily, participated to the SPIMBENCH task. Overall, Lily has the best results compared to both AML and LogMap approaches. It achieves a perfect recall, while LogMap has a good precision. Indeed, it is not clear from the literature why the results of AML are different in 2018 than those obtained in 2017. This difference may be due to the difference in the instance descriptions. Concerning DOREMUS task, two approaches participated in this task in 2016, namely, AML and RIMOM. To show how these approaches deal with the heterogeneity, two experiments were conducted. The first one includes four types of heterogeneities, including (i) orthographic differences, (ii) multilingual titles, (iii) missing titles, and (iv) missing properties. We denote this experiment by 4HT. The second experiment has nine types of heterogeneities, among them, we cite multilingualism, differences in catalogues, differences in spelling, and different degrees of description (number of properties). We refer to this experiment by 9HT in Table 9. AML achieves the best performance in terms of F-measure by setting the similarity threshold to 0.2 on all tasks. In 2017, only five approaches return results on this track: AML, I-Match, Legato, LogMap, and NjuLink. Among these approaches, only Legato and NjuLink show better performances in terms of F-measure. Indeed, on HT test, NjuLink is ranked as the best approach, while on FPT, Legato has an almost full F-measure, making it the best approach on this test. As a result, both Legato and NjuLink seem to be well-suited to cope with real-world data. The repairing step in Legato gives it an advantage to deal with data across KBs with highly similar descriptions.

8 | MAIN LESSONS AND OPEN CHALLENGES

In general, the interlinking process between two KBs should be able to generate any type of links including the identity links. In the approaches summarized in this survey, we did not find any tool capable of performing this task perfectly and in a complete way for all the data representations.

TABLE 9 Results of different instance matching approaches on the SPIMBENCH and DOREMUS benchmark datasets

Benchmarks (year)	Approaches	Precision	Recall	F-measure
SPIMBENCH Sandbox (2017)	**AML**	0.849	1.000	0.918
	I-Match	0.854	0.997	**0.920**
	Legato	0.980	0.730	0.840
	LogMap	0.938	0.763	0.841
SPIMBENCH Mainbox (2017)	**AML**	0.855	1.000	**0.922**
	I-Match	0.856	0.997	0.921
	Legato	0.970	0.700	0.810
	LogMap	0.893	0.709	0.790
SPIMBENCH Sandbox (2018)	**AML**	0.849	0.896	0.865
	Lily	0.849	1.000	**0.919**
	LogMap	0.938	0.763	0.841
SPIMBENCH Mainbox (2018)	**AML**	0.839	0.884	0.860
	Lily	0.855	1.000	**0.922**
	LogMap	0.893	0.709	0.791
HT (2017)	**AML**	0.851	0.479	0.613
	I-Match	0.680	0.071	0.129
	Legato	0.930	0.920	0.930
	LogMap	0.406	0.882	0.556
	NjuLink	0.966	0.945	**0.955**
4HT (2016)	**AML**	0.966	0.875	**0.918**
	RIMOM	0.813	0.813	0.813
9HT (2016)	**AML**	0.934	0.776	**0.848**
	RIMOM	0.746	0.746	0.746
FPT (2016)	**AML**	0.921	0.854	**0.886**
	RIMOM	0.707	0.707	0.707
FPT (2017)	**AML**	0.914	0.427	0.582
	I-Match	1.000	0.053	0.101
	Legato	1.000	0.980	**0.990**
	LogMap	0.119	0.880	0.210
	NjuLink	0.959	0.933	0.946

Another important aspect is that with the rapid and continuous growth of KBs, IM approaches are expected to be capable of handling large and incremental updates of these KBs continuously. Yet, across the surveyed approaches in this article (except *viewSameAs*[117]), we are unaware of any existing method capable of saving the history of potential co-references such that the method could leverage it for next iterations on updates of the KBs. Existing IM approaches must restart the linking process from scratch at each run. In fact, by materializing the discovered links between potential candidates, an IM algorithm can be re-executed only on the candidates that have been updated. This option presents an essential requirement for the next generation of IM approaches, especially in this era of *big data*.

In addition, all the IM approaches in the semantic web rely on a traditional (offline) strategy to handle large scale KBs. This strategy applies first a blocking step then computes the similarities only among instances within the same block. In some situations, the execution of the IM process can be constrained by a limited amount of time. Thus, it will be interesting to design an IM process in a pay-as-you-go[143] strategy. The latter uses an execution plan at the instance and block levels in order to maximize the number of resolved co-referents under time constraints. Such a strategy

becomes more required in big data applications that require real-time data analyses. To the best of our knowledge, no work except[45] exploits this idea to the IM problem in the semantic web field.

Another important aspect here is that the IM process aims to provide a mean to complement and enrich information for a given entity-centric query through building a unified KB. To the best of our knowledge, none of the existing approaches evaluate the richness of the obtained joint KB obtained with regard to the IM results. To measure this aspect, new metrics can be defined in line with the idea of the diversity principle in Reference 144.

It is also important to note that a large number of IM approaches are relying on domain experts to provide the script file (ie, aligning predicates, predicates classification, etc.). In large-scale KBs, IM approaches should be more independent of such a requirement. Thus, either the development of an automation step for the creation of such a file or schema and oracle-independent approaches become a main demand.

In the last few years, much interest has been addressed to advancing research on multilingualism in natural language processing in general. This aspect has been omitted by most IM approaches. With the continuous growth of heterogeneous KBs and the multilingual nature of the web, handling this aspect should be taken into consideration by IM approaches. For this purpose, machine translation can be used to overcome the language obstacles across the compared KBs.[23,24] Yet, using such a solution makes a cross-lingual IM approach suffers from the uneven quality of translations between languages. Another potential solution here could be to study the possibility of leveraging the structure information of the KBs and network embedding models through the projection of information in the compared KBs in a joint representation space.

A similar problem to the multilingualism is the multimodal data aspect, which become more frequent especially in the big data era. This aspect is challenging because it requires detailed understanding of different modalities and the correspondence between them.[145] In fact, all the surveyed approaches are executed across RDF KBs. None of them can be executed on KBs expressed in different formats (ie, web tables, image-based KBs, etc.). Yet, the above described translation-based idea between different KBs in a joint latent space of representation could be applied to overcome this multi-modal data challenge. However, in real-world applications, obtaining labeled data points from multiple modalities is very expensive.[146]

Again, we remark that the majority of IM approaches do not apply a post-processing step to polish the obtained identity links. The automation of this step is a requirement especially for large-scale KBs. In fact, it is unfeasible to manually inspect the correctness of a large set of inferred identity links. For this purpose, a bipartite graph could first be built from the inferred identity links. Then, a post-processing, including stable marriage problem (SMP),[100] Hungarian algorithm,[147] and symmetric best match strategy (SBM)[148]) could be adopted to obtain the final set of co-referent pairs from this bipartite graph as in BIGMAT.[138]

We also note that the majority of IM approaches rely on property matching (ie, are schema dependent). They start by finding a suitable representation of an instance by selecting a subset of predicates from the instances' descriptions. Then, they use this subset for IM. Indeed, predicates with a similar semantic (eg, rdfs:label, foaf:name) in two instance representations must be matched. Yet, there are some cases where the descriptions of instances do not carry properties with similar semantics. However, they can contain information with some expressive relationships. As a result, such information will be ignored even if it is worthy to consider it for IM. Thus, integrating a semantic similarity strategy is paramount for solving such cases. To this end, it is possible to leverage the advances in deep learning for natural language processing by incorporating pre-trained word embeddings of, for instance, Wikipedia during the similarity computation between words of the compared instances (as in VDLS[1,113]). This makes the approach take into account the semantics of words during the IM.

Since data sources usually contain their "own" designed ontologies to describe their "own" instances, it becomes very natural to confront similar properties in different KBs, including values expressed in specific formats (eg, values expressed in meter in one KB and in centimeter in another KB). We did not find any approach (except VDLS[1,113]), which can handle such cases. To address this challenge, adding additional steps (such as normalization), or special heuristics/patterns to convert these different formats to a common one, or adopting a semantic similarity metric can be applied. These solutions become a necessity to make the IM more compatible to various kinds of data, more robust and more domain knowledge independent. We believe that the more an IM approach uses information in the matching step, the more it is expected to obtain precise mappings. One of such information can be extracted from the ontology itself. We note that only a few approaches integrate this information in their processes.

We also note that all the context-dependent approaches (except MinoaER) which take into consideration the neighbors during the IM process do not incorporate any preference between them. However, some neighbors have more impact than others on the IM for the considered instance. Thus, in large-scale KBs matching considering only those neighbors could lead to a decrease in the number of comparisons executed in the matching step and an increase in the accuracy of the results. To this purpose, statistical metrics like the harmonic mean of the support and discriminability of a predicate could be adopted. In fact, for each instance, only the neighbors connected via one of the top-N harmonic mean predicates will be selected as the most important ones for the matching process.

Although the OAEI is considered as an international competition permitting to evaluate the performance of IM approaches, we notice that the number of participating approaches decreased in recent years. Besides, the unavailability of implementations of most IM approaches prevents recent literature works to perform a comparison with these approaches as well as to evaluate them on recent benchmarks. A framework integrating several IM approaches from the literature is needed.

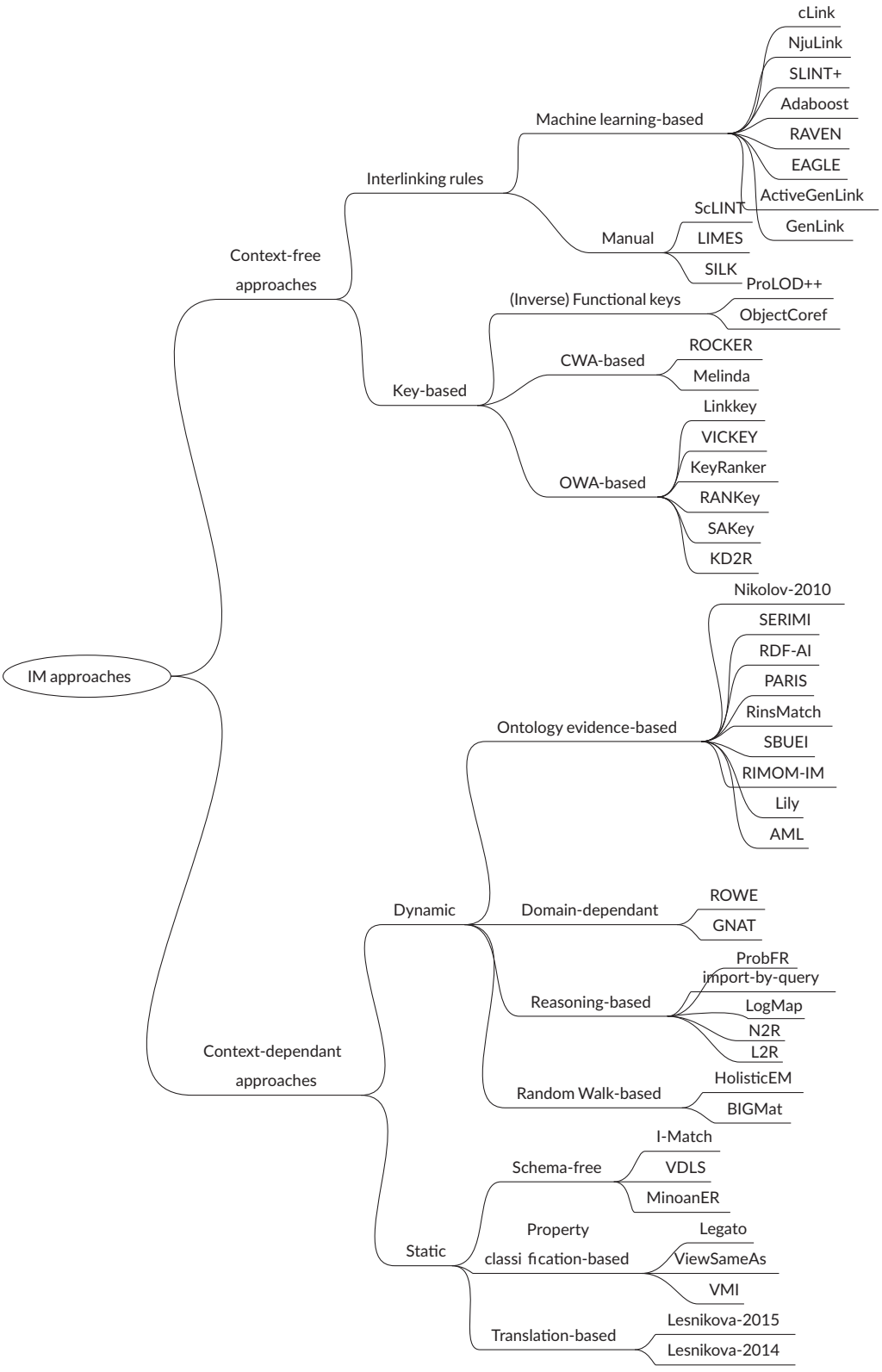


FIGURE 9 Taxonomy of IM approaches according to context as evidence source

Finally, to support the big data characteristics (ie, variety, volume, velocity, and veracity), we believe that a perfect IM approach must include the combinations of (i) schema-free to go beyond the value and property levels, (ii) semantic-based similarity to overcome the properties that cannot match across heterogeneous KBs but incorporate connotative information, (iii) context-dependent to include more evidence to disambiguate the instances, (iv) leverage parallel and distributed computation (ie, data and task) to handle the large scale KBs, and (v) incorporate a post-processing strategy to refine the inferred identity links and enhance the quality of the IM results. Considering these aspects in future approaches makes an asset for advancing research on IM in RDF knowledge graphs.

9 | CONCLUSION

IM is a critical problem in multiple domains of applications, including mainly the semantic web but also schema/ontology integration, data warehouses, and so on. Multiple IM approaches have been proposed in the literature. In this article, we have reviewed a wide range of the existing IM approaches applied for the RDF KBs. We categorized them in two categories based on their use of the context of the instances in the matching Figure 9. We also discussed and identified the advantages and limits of each of these categories. Context-free approaches are suitable in the cases where the structures of the compared KBs are (approximately) similar, and the local information (ie, data-type properties) describing the instances are rich. Thus, a set of properties is selected as an IR. Machine learning methods can be effective means to build and improve iteratively such IRs by assembling positive and negative references. Unfortunately, this is not the case in multiple real-world KBs. Moreover, the huge amount of data in these KBs hinders the efficiency and the running time of machine learning based approaches.

In conclusion, this study has pointed out multiple advantages and drawbacks in the existing state-of-the-art IM approaches. Such a comprehensive comparative study constitutes an asset and a guide for future research in IM. Indeed, one of the important future directions to enhance the efficiency of IM approaches is to leverage more evidential information about the matching of instances from their neighbors in their respective KBs. Incorporating the topological relations between instances and propagating the matching decisions through the neighbors could also allow IM approaches to increase their accuracy and efficiency for performing other matches. Besides, by aggregating multiple different sources of evidences from the instance descriptions, topological properties, and relations in the RDF graph, IM approaches could become able to cover several KBs representation cases and relax the effect of noisy and missing data. This information could be accounted in the computation of similarities between instances to get more accurate IM results.

ORCID

Ali Assi https://orcid.org/0000-0003-3709-1519

REFERENCES

1. Assi A, Mcheick H, Karawash A, Dhifli W. Context-aware instance matching through graph embedding in lexical semantic space. *Knowl-Based Syst*. 2019;186:104925.
2. Papaleo L, Pernelle N, Sais F, Dumont C. Logical detection of invalid sameAs statements in RDF data. In: *International Conference on Knowledge Engineering and Knowledge Management*, New York, NY: Springer; 2014. p. 373–384.
3. Schmachtenberg M, Bizer C, Paulheim H. Adoption of the linked data best practices in different topical domains. Paper presented at: Proceedings of the International Semantic Web Conference (ISWC 2014); 2014:245-260; Springer.
4. Christen P. A survey of indexing techniques for scalable record linkage and deduplication. *IEEE Trans Knowl Data Eng*. 2012;24(9):1537-1555.
5. Papadakis G, Alexiou G, Papastefanatos G, Koutrika G. Schema-agnostic vs schema-based configurations for blocking methods on homogeneous data. *Proc VLDB Endow*. 2015;9(4):312-323.
6. Papadakis G, Svirsky J, Gal A, Palpanas T. Comparative analysis of approximate blocking techniques for entity resolution. *Proc VLDB Endow*. 2016;9(9):684-695.
7. Papadakis G, Papastefanatos G, Koutrika G. Supervised Meta-blocking. *Proc VLDB Endow*. 2014;7(14):1929-1940.
8. Ferraram A, Nikolov A, Scharffe F. Data linking for the semantic web. *Semantic Web: Ontology and Knowledge Base Enabled Tools, Services, and Applications*. Vol 169; 2013:326. Idea Group Inc.
9. Kejriwal M, Miranker DP. An unsupervised instance matcher for schema-free RDF data. *Web Semant Sci Serv Agents World Wide Web*. 2015;35:102-123.
10. Soru T, Ngomo ACN. A comparison of supervised learning classifiers for link discovery. Paper presented at: Proceedings of the 10th International Conference on Semantic Systems (SEMANTiCS 2014); 2014:41-44; ACM.
11. Hassan MM, Lehmann J, Ngomo ACN. Interlinking: performance assessment of user evaluation vs. supervised learning approaches. Paper presented at: Proceedings of the Linked Data on the Web (LDOW 2015) at 24th International World Wide Web Conference (WWW 2015); 2015.
12. Scharffe F, Euzenat J. MeLinDa: an interlinking framework for the web of data. *CoRR*. 2011;abs/1107.4502.
13. Wölger S, Hofer C, Siorpaes K, Thaler S, Simperl E, Bürger T. Interlinking data-approaches and tools. Unpublished technical report, STI Innsbruck, Austria; 2011.
14. Ngomo ACN, Auer S. Limes-a time-efficient approach for large-scale link discovery on the web of data. Paper presented at: Proceedings of the 22nd International Joint Conference on Artificial Intelligence, IJCAI 2011, Barcelona, Catalonia, Spain, July 16-22, 2011:2312-2317.
15. Taheri A, Shamsfard M. Instance coreference resolution in multi-ontology linked data resources. Paper presented at: Proceedings of the Joint International Semantic Technology Conference JIST; 2012; 129-145; Springer.

16. Nguyen K, Ichise R, Le B. SLINT: a schema-independent linked data interlinking system. Paper presented at: Proceedings of the 7th International Conference on Ontology Matching OM 2012 - Volume 946; 2012:1-12.
17. Bhattacharya I, Getoor L. Collective entity resolution in relational data. *ACM Trans Knowl Discov Data (TKDD)*. 2007;1(1):5.
18. Isele R. Learning Expressive Linkage Rules for Entity Matching using Genetic Programming (PhD thesis). University of Mannheim; 2013.
19. sais F. Semantic Integation of Data Guided by Ontology (PhD thesis). University of Paris-Sud 11; 2007.
20. Dey AK. Providing Architectural Support for Building Context-Aware Applications (PhD thesis). Georgia Institute of Technology; 2000.
21. Klyne G, Carroll JJ, Resource Description Framework (RDF): Concepts and Abstract Syntax; 2004. http://www.w3.org/TR/2004/REC-rdf-concepts-20040210/.
22. Ferrara A, Lorusso D, Montanelli S, Varese G. Towards a benchmark for instance matching. Paper presented at: Proceedings of the 3rd International Conference on Ontology Matching-Volume 431; 2008:37-48.
23. Lesnikova T, David J, Euzenat J. Interlinking English and Chinese RDF data sets using machine translation. Paper presented at: Proceedings of the 3rd ESWC Workshop on Knowledge Discovery and Data Mining Meets Linked Open Data (Know@ LOD) No Commercial Editor; 2014.
24. Lesnikova T, David J, Euzenat J. Interlinking English and Chinese RDF data using Babelnet. Paper presented at: Proceedings of the 2015 ACM Symposium on Document Engineering ACM; 2015:39-42.
25. Scharffe F, Liu Y, Zhou C. Rdf-ai: an architecture for rdf datasets matching, fusion and interlink. Paper presented at: Proceedings of IJCAI 2009 workshop on Identity, reference, and knowledge representation (IR-KR), Pasadena (CA US); 2009.
26. Navigli R, Ponzetto SP. BabelNet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network. *Artif Intell*. 2012;193:217-250.
27. Spohr D, Hollink L, Cimiano P. A machine learning approach to multilingual and cross-lingual ontology matching. Paper presented at: Proceedings of the International Semantic Web Conference ISWC 2011; 2011:665-680; Springer.
28. Chen M, Tian Y, Yang M, Zaniolo C. Multilingual knowledge graph embeddings for cross-lingual knowledge alignment. Paper presented at: Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence IJCAI 2017, Melbourne, Australia, August 19-25, 2017:1511-1517; AAAI Press.
29. Guan S, Jin X, Jia Y, Wang Y, Shen H, Cheng X. Self-learning and embedding based entity alignment. Paper presented at: Proceedings of the 2017 IEEE International Conference on Big Knowledge, ICBK 2017; 2017:33-40; IEEE.
30. Hao Y, Zhang Y, He S, Liu K, Zhao J. A joint embedding method for entity alignment of knowledge bases. Paper presented at: Proceedings of the China Conference on Knowledge Graph and Semantic Computing, CCKS 2016 2016:3-14; Springer.
31. Sun Z, Hu W, Li C. Cross-lingual entity alignment via joint attribute-preserving embedding. Paper presented at: Proceedings of the International Semantic Web Conference ISWC 2017; 2017:628-644; Springer.
32. Zhu H, Xie R, Liu Z, Sun M. Iterative entity alignment via joint knowledge embeddings. Paper presented at: Proceedings of the 26th International Joint Conference on Artificial Intelligence IJCAI'17; 2017:4258-4264; AAAI Press.
33. Li C, Ji L, Yan J. Acronym disambiguation using word embedding. Paper presented at: Proceedings of the 29th AAAI Conference on Artificial Intelligence; 2015:4178-4179; AAAI Press.
34. Yamamoto Y, Yamaguchi A, Bono H, Takagi T. Allie: a database and a search service of abbreviations and long forms. *Database J Biol Databases Curation*. 2011;2011:bar013.
35. Ciosici MR, Sommer T, Assent I. Unsupervised abbreviation disambiguation; 2019. arXiv preprint arXiv:190400929 2019.
36. Charbonnier J, Wartena C. Using word embeddings for unsupervised acronym disambiguation. Paper presented at: Proceedings of the 27th International Conference on Computational Linguistics Santa Fe, New Mexico: Association for Computational Linguistics; 2018:2610-2619.
37. Miller GA. *WordNet: A Lexical Database for English*. Vol 38. New York, NY: ACM Press; 199538(11):39–41.
38. Mikolov T, Sutskever I, Chen K, Corrado GS, Dean J. Distributed representations of words and phrases and their compositionality. Advances in Neural Information Processing Systems; 2013:3111-3119.
39. Meusel R, Petrovski P, Bizer C. The webdatacommons microdata, RDFA and microformat dataset series. Paper presented at: Proceedings of the International Semantic Web Conference ISWC 2014; 2014:277-292; Springer.
40. Li Ding TF Joshua Shinavier, McGuinness DL. Owl:sameAs and linked data: an empirical study. Paper presented at: Proceedings of the 2nd Web Science Conference Raleigh NC; 2010.
41. Kolb L, Rahm E. Parallel entity resolution with dedoop. *Datenbank-Spektrum*. 2012;13:23-32.
42. Zaharia M, Chowdhury M, Franklin MJ, Shenker S, Stoica I. Spark: cluster computing with working sets. Paper presented at: Proceedings of the 2nd USENIX Workshop on Hot Topics in Cloud Computing, HotCloud'10, June 22, 2010; Boston, MA.
43. Dean J, Ghemawat S. MapReduce: simplified data processing on large clusters. *Commun ACM*. 2008 Jan;51(1):107-113.
44. Achichi M, Bellahsene Z, Todorov K. A survey on web data linking. *Revue des Sciences et Technologies de l'Information*. 2016;21(5-6):11-29.
45. Simonini G, Papadakis G, Palpanas T, Bergamaschi S. Schema-agnostic progressive entity resolution. *IEEE Trans Knowl Data Eng*. 2019;31(6):1208-1221.
46. Baxter R, Christen P, Churches T. A comparison of fast blocking methods for record linkage. Paper presented at: Proceedings of the ACM SIGKDD '03 Workshop on Data Cleaning, Record Linkage, and Object Consolidation; 2003:25-27.
47. Christen P, Goiser K. Quality and complexity measures for data linkage and deduplication. *Quality Measures in Data Mining*. Berlin, Heidelberg / Germany: Springer; 2007:127-151.
48. De Vries T, Ke H, Chawla S, Christen P. Robust record linkage blocking using suffix arrays and Bloom filters. *ACM Trans Knowl Discov Data (TKDD)*. 2011;5(2):9.
49. Elfeky MG, Verykios VS, Elmagarmid AK. TAILOR: a record linkage toolbox. Paper presented at: Proceedings of the 18th International Conference on Data Engineering ICDE 2002; 2002:17-28; IEEE.
50. Nikolov A, Uren VS, Motta E. Data linking: capturing and utilizing implicit schema-level relations. In: Bizer C, Heath T, Berners-Lee T, Hausenblas M, eds. Linked Data on the Web (LDOW 2010) at 19th International World Wide Web Conference (WWW 2010), vol. 628; 2010.
51. Ding L, Shinavier J, Shangguan Z, McGuinness DL. SameAs networks and beyond: analyzing deployment status and implications of owl: sameAs in linked data. Paper presented at: Proceedings of the International Semantic Web Conference ISWC 2010; 2010:145-160; Springer.
52. Guéret C, Groth P, Stadler C, Lehmann J. Assessing linked data mappings using network measures. Paper presented at: Proceedings of the Extended Semantic Web Conference, ESWC 2012; 2012:87-102; Springer.

53. Halpin H, Hayes PJ, Thompson HS. When owl: sameAs isn't the same redux: a preliminary theory of identity and inference on the semantic web. Paper presented at: Proceedings of the 2011 International Conference on Discovering Meaning on the go in Large Heterogeneous Data, LHD 2011 Morgan Kaufmann Publishers Inc; 2011:25-30.
54. de Melo G. Not Quite the same: identity constraints for the web of linked data. Paper presented at: Proceedings of the 27th AAAI Conference on Artificial Intelligence; July 14-18, 2013; Bellevue, Washington, DC.
55. Papaleo L, Pernelle N, Sais F, Dumont C. Logical detection of invalid SameAs statements in RDF data. Paper presented at: Proceedings of the International Conference on Knowledge Engineering and Knowledge Management; 2014:373-384; Springer.
56. Raad J, Beek W, Van Harmelen F, Pernelle N, Sais F. Detecting erroneous identity links on the web using network metrics. Paper presented at: Proceedings of the International Semantic Web Conference, ISWC 2018; 2018:391-407; Springer.
57. Kitchenham B. Procedures for performing systematic reviews. *Keele, UK, Keele Univ*. 2004;33(2004):1-26.
58. Moher D, Liberati A, Tetzlaff J, Altman DG. Preferred reporting items for systematic reviews and meta-analyses: the PRISMA statement. *Ann Int Med*. 2009;151(4):264-269.
59. Brizan DG, Tansel AU. A survey of entity resolution and record linkage methodologies. *Commun IIMA*. 2006;6(3):5.
60. Elmagarmid AK, Ipeirotis PG, Verykios VS. duplicate record detection: a survey. *IEEE Trans Knowl Data Eng*. 2007 Jan;19(1):1-16.
61. Dorneles CF, Gonçalves R, dos Santos Mello R. Approximate data instance matching: a survey. *Knowl Inf Syst*. 2011;27(1):1-21.
62. Köpcke H, Rahm E. Frameworks for entity matching: a comparison. *Data Knowl Eng*. 2010;69(2):197-210.
63. Hassanzadeh O, Chiang F, Lee HC, Miller RJ. Framework for evaluating clustering algorithms in duplicate detection. *Proc VLDB Endow*. 2009 Aug;2(1):1282-1293.
64. Gal A. Uncertain entity resolution: re-evaluating entity resolution in the big data era: tutorial. *Proc VLDB Endow*. 2014;7(13):1711-1712.
65. Getoor L, Machanavajjhala A. Entity resolution: theory, practice & open challenges. *Proc VLDB Endow*. 2012;5(12):2018-2019.
66. Chen X, Schallehn E, Saake G. Cloud-scale entity resolution: current state and open challenges. *Open J Big Data (OJBD)*. 2018;4(1):30-51.
67. Nentwig M, Hartung M, Ngonga Ngomo AC, Rahm E. A survey of current link discovery frameworks. *Semantic Web*. 2017;8(3):419-436.
68. Michelson M, Knoblock CA. Learning blocking schemes for record linkage. Paper presented at: Proceedings of the 21st National Conference on Artificial Intelligence - Volume 1 AAAI'06; 2006:440-445; AAAI Press.
69. Köhler H, Link S, Zhou X. Possible and certain SQL keys. *Proc VLDB Endow*. 2015;8(11):1118-1129.
70. Berti-Équille L, Harmouch H, Naumann F, Novelli N, Thirumuruganathan S. Discovery of genuine functional dependencies from relational data with missing values. *Proc VLDB Endow*. 2018 Apr;11(8):880-892.
71. Pernelle N, Sais F, Safar B, Koutraki M, Ghosh T. N2r-part: identity link discovery using partially aligned ontologies. Paper presented at: Proceedings of the 2nd International Workshop on Open Data ACM; 2013:6.
72. Saıs F, Pernelle N, Rousset MC. L2r: a logical method for reference reconciliation. Paper presented at: Proceedings of the 22nd AAAI Conference on Artificial Intelligence; 2007:329-334.
73. Saıs F, Pernelle N, Rousset MC. Combining a logical and a numerical method for data reconciliation. *J Data Semantics XII*. 2009;12:66-94.
74. Isele R, Jentzsch A, Bizer C. Efficient multidimensional blocking for link discovery without losing recall. Paper presented at: Proceedings of the 14th International Workshop on the Web and Databases, WebDB, Vol. 2011; 2011, Athens.
75. Song D, Heflin J. Automatically generating data linkages using a domain-independent candidate selection approach. Paper presented at: Proceedings of the International Semantic Web Conference ISWC'2011; 2011:649-664; Springer.
76. Volz J, Bizer C, Gaedke M, Kobilarov G. Discovering and maintaining links on the web of data. Paper presented at: Proceedings of the International Semantic Web Conference ISWC 2009; 2009:650-665; Springer.
77. Hitzler P, Krötzsch M, Parsia B, Patel-Schneider PF, Rudolph S. OWL 2 web ontology language: Primer. W3C recommendation; 2017.
78. Glimm B, Hogan A, Krötzsch M, Polleres A. OWL: yet to arrive on the web of data? Paper presented at: Proceedings of the Linked Data on the Web (LDOW 2012) at 21st International World Wide Web Conference (WWW 2012) CEUR Workshop Proceedings; 2012.
79. Atencia M, David J, Euzenat J. Data interlinking through robust linkkey extraction. Paper presented at: Proceedings of the 21st European Conference on Artificial Intelligence ECAI'14; 2014:15-20; IOS Press.
80. Symeonidou D, Armant V, Pernelle N, Saıs F. SAKey: scalable almost key discovery in RDF data. Paper presented at: Proceedings of the 13th International Semantic Web Conference ISWC 2014, vol. 8796; 2014:33-49; Springer.
81. Symeonidou D, Galárraga L, Pernelle N, Saıs F, Suchanek F. VICKEY: mining conditional keys on knowledge bases. Paper presented at: Proceedings of the International Semantic Web Conference ISWC 2017; 2017:661-677; Springer.
82. Sismanis Y, Brown P, Haas PJ, Reinwald B. GORDIAN: efficient and scalable discovery of composite keys. Paper presented at: Proceedings of the 32nd International Conference on Very Large Data Bases VLDB Endowment; 2006:691-702.
83. Pernelle N, Saıs F, Symeonidou D. An automatic key discovery approach for data linking. *Web Semant Sci Serv Agents WWW*. 2013;23:16-30.
84. Achichi M, Ellefi MB, Symeonidou D, Todorov K. Automatic key selection for data linking. Paper presented at: Proceedings of the European Knowledge Acquisition Workshop; 2016:3-18; Springer.
85. Atencia M, David J, Scharffe F. Keys and pseudo-keys detection for web datasets cleansing and interlinking. Paper presented at: Proceedings of the International Conference on Knowledge Engineering and Knowledge Management; 2012:144-153; Springer.
86. Farah H, Symeonidou D, Todorov K. KeyRanker: automatic RDF key ranking for data linking. Paper presented at: Proceedings of the Knowledge Capture Conference, KCAP 2017; 2017:7; ACM.
87. Hitzler P, Krötzsch M, Parsia B, Patel-Schneider PF, Rudolph S. OWL 2 web ontology language primer. *W3C recommendation*. 2009;27(1):123.
88. Soru T, Marx E, Ngonga Ngomo AC. ROCKER: A refinement operator for key discovery. Paper presented at: Proceedings of the 24th International Conference on World Wide Web (WWW 2015) International World Wide Web Conferences Steering Committee; 2015:1025-1033.
89. Huhtala Y, Kärkkäinen J, Porkka P, Toivonen H. TANE: an efficient algorithm for discovering functional and approximate dependencies. *Comput J*. 1999;42(2):100-111.
90. Aidan H, Polleres A, Jürgen U, Zimmermann A. Some entities are more equal than others: statistical methods to consolidate linked data. Paper presented at: Proceedings of the Workshop on New Forms of Reasoning for the Semantic Web: Scalable & Dynamic, Heraklion, Greece; 2010.
91. Suchanek FM, Abiteboul S, Senellart P. Paris: probabilistic alignment of relations, instances, and schema. *Proc VLDB Endow*. 2011;5(3):157-168.

92. Hu W, Chen J, Qu Y. A self-training approach for resolving object coreference on the semantic web. Paper presented at: Proceedings of the 20th International Conference on World Wide Web (WWW 2011); 2011:87-96; ACM.
93. Jentzsch A, Mühleisen H, Naumann F. Uniqueness, density, and keyness: exploring class hierarchies. Paper presented at: Proceedings of the 6th International Workshop on Consuming Linked Data co-located with 14th International Semantic Web Conference ISWC 2105, Bethlehem, Pennsylvania, USA, October 12th, 2015, vol. 1426; 2015.
94. Papadakis G, Demartini G, Fankhauser P, Kärger P. The missing links: Discovering hidden same-as links among a billion of triples. Paper presented at: Proceedings of the 12th International Conference on Information Integration and Web-based Applications & Services; 2010:453-460; ACM.
95. Nguyen K, Ichise R. ScSLINT: time and memory efficient interlinking framework for linked data. Paper presented at: Proceeding of the International Semantic Web Conference (Posters & Demos); 2015.
96. Ngomo ACN. Helios–execution optimization for link discovery. Paper presented at: Proceedings of the International Semantic Web Conference, ISWC 2014; 2014:17-32; Springer.
97. Xiao C, Wang W, Lin X, Yu JX, Wang G. Efficient similarity joins for near-duplicate detection. *ACM Transactions on Database Systems (TODS)*. 2011;36(3):15.
98. Ngomo ACN. Link discovery with guaranteed reduction ratio in affine spaces with minkowski measures. Paper presented at: Proceedings of the International Semantic Web Conference, ISWC 2012 ; 2012:378-393; Springer.
99. Nguyen K, Ichise R. Linked data entity resolution system enhanced by configuration learning algorithm. *IEICE Trans Inf Syst*. 2016;99(6):1521-1530.
100. Gale D, Shapley LS. College admissions and the stability of marriage. *Am Math Mon*. 1962;69(1):9-15.
101. Isele R, Bizer C. Learning expressive linkage rules using genetic programming. *Proc VLDB Endow*. 2012;5(11):1638-1649.
102. Ngomo ACN, Lyko K. Eagle: Efficient active learning of link specifications using genetic programming. Paper presented at: Proceedings of the Extended Semantic Web Conference, ESWC 2012; 2012:149-163; Springer.
103. Koza JR. Genetic programming as a means for programming computers by natural selection. *Stat Comput*. 1994;4(2):87-112.
104. Isele R, Bizer C. Active learning of expressive linkage rules using genetic programming. *Web Semant Sci Serv Agents WWW*. 2013;23:2-15.
105. Ngomo ACN, Lehmann J, Auer S, Höffner K. RAVEN-active learning of link specifications. Paper presented at: Proceedings of the 6th International Conference on Ontology Matching Volume 814 OM'11; 2011:25-36.
106. Rong S, Niu X, Xiang EW, Wang H, Yang Q, Yu Y. A machine learning approach for instance matching based on similarity metrics. Paper presented at: Proceedings of the International Semantic Web Conference ISWC 2012; 2012:460-475; Springer.
107. Sleeman J, Finin T. A machine learning approach to linking FOAF instances. Paper presented at: Proceedings of the AAAI Spring Symposium Linked Data Meets Artificial Intelligence; 2010.
108. Pan SJ, Yang Q. A survey on transfer learning. *IEEE Trans Knowl Data Eng*. 2010;22(10):1345-1359.
109. Joachims T. Svmlight: support vector machine. SVM-light support vector machine. University of Dortmund; 1999;19(4). http://svmlight joachims org/.
110. Nguyen K, Ichise R, Le B. Interlinking linked data sources using a domain-independent system. Paper presented at: Proceedings of the Semantic Technology, Second Joint International Conference, JIST 2012, Nara, Japan, December 2-4, 2012; 2012:113-128; Springer.
111. Lyu X, Zhang Q, Hu W, Sun Z, Qu Y. njuLink: results for instance matching at OAEI 2017. Paper presented at: Proceedings of the 12th International Workshop on Ontology Matching, OM 2017; 2017:158-165.
112. Efthymiou V, Papadakis G, Stefanidis K, Christophides V. MinoanER: schema-agnostic, non-iterative, massively parallel resolution of web entities. Paper presented at: Proceedings of the Advances in Database Technology - 22nd International Conference on Extending Database Technology, EDBT; 2019, 2019:373-384; Lisbon, Portugal.
113. Assi A, Mcheick H, Dhifli W. Context-aware instance matching through graph embedding in lexical semantic space. Paper presented at: Proceedings of the Advances and Trends in Artificial Intelligence. From Theory to Practice - 32nd International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems, IEA/AIE 2019, Graz, Austria, July 9-11, 2019; 2019:422-433.
114. Li J, Wang Z, Zhang X, Tang J. Large scale instance matching via multiple indexes and candidate selection. *Knowl-Based Syst*. 2013;50:112-120.
115. Khiat A, Mackeprang M. I-Match and ontoidea results for OAEI 2017. Paper presented at: Proceedings of the 12th International Workshop on Ontology Matching, OM 2017; 2017:135-137.
116. Ghemmaz W, Benchikha F. Instance matching based on discriminative property values. Paper presented at: Proceedings of the 2015 5th International Conference on Information Communication Technology and Accessibility, ICTA 2015; 2015:1-6; IEEE.
117. Ghemmaz W, Benchikha F. ViewSameAs: a novel link in instance matching process. Paper presented at: Proceedings of the International Conference on Web Information Systems and Technologies WEBIST 2016; 2016:274-279; SciTePress.
118. Achichi M, Bellahsene Z, Ellefi MB, Todorov K. Linking and disambiguating entities across heterogeneous RDF graphs. *J Web Semant*. 2019;55:108-121.
119. Al-Bakri M, Atencia M, David J, Lalande S, Rousset MC. Uncertainty-sensitive reasoning for inferring sameas facts in linked data. Paper presented at: Proceedings of the 22nd European Conference on Artificial Intelligence ECAI 2016; 2016:698-706; IOS Press.
120. Al-Bakri M, Atencia M, Lalande S, Rousset M. Inferring same-as facts from linked data: an iterative import-by-query approach. Paper presented at: Proceedings of the 29th AAAI Conference on Artificial Intelligence, January 25-30, 2015:9-15; Austin, TX.
121. Jiménez-Ruiz E, Grau BC. Logmap: logic-based and scalable ontology matching. Paper presented at: Proceedings of the International Semantic Web Conference, ISWC 2011; 2011:273-288.
122. Golub GH, Van Loan CF. *Matrix Computations*. 3rd ed. Baltimore, MD: Johns Hopkins University Press; 1996.
123. Saıs F, Niraula N, Pernelle N, Rousset MC. LN2R–a knowledge based reference reconciliation system: OAEI 2010 Results. Proceedings of the 5th International Workshop on Ontology Matching, OM 2010 2010:172.
124. Abiteboul S, Hull R, Vianu V. *Foundations of Databases: The Logical Level*. Boston, MA: Addison-Wesley Longman Publishing Co.Inc; 1995.
125. Vieille L. Recursive axioms in deductive databases: the query/subquery approach. Proceedings of the Expert Database Conference; 1986.
126. Doran P, Tamma V, Iannone L. Ontology module extraction for ontology reuse: an ontology engineering perspective. Paper presented at: Proceedings of the 16th ACM Conference on Conference on Information and Knowledge Management; 2007:61-70; ACM.
127. Raimond Y, Sutton C, Sandler MB. Automatic interlinking of music datasets on the semantic web. Paper presented at: Proceedings of the Linked Data on the Web (LDOW 2008) at 17th International World Wide Web Conference (WWW 2008), vol. 369; 2008.
128. Rowe M. Interlinking distributed social graphs. Paper presented at: Proceedings of the Linked Data on the Web (LDOW 2009) at 18th International World Wide Web Conference (WWW 2009); 2009.
129. Euzenat J, Shvaiko P. *Ontology Matching*. Vol 18. New York, NY: Springer; 2013.

130. Aydar M, Melton A. RinsMatch: a suggestion-based instance matching system in RDF Graphs. Paper presented at: Proceedings of the 10th International Workshop on Ontology Matching Collocated with the 14th International Semantic Web Conference (ISWC 2015); October 12, 2015:224-225; Bethlehem, PA.
131. Shao C, Hu LM, Li JZ, Wang ZC, Chung T, Xia JB. RiMOM-IM: a novel iterative framework for instance matching. *J Comput Sci Tech*. 2016;31(1):185-197.
132. Song D, Heflin J. Domain-independent entity coreference for linking ontology instances. *J Data Inf Quality (JDIQ)*. 2013;4(2):7.
133. Nikolov A, Uren VS, Motta E. Data linking: capturing and utilising implicit schema-level relations. Paper presented at: Proceedings of the Linked Data on the Web (LDOW 2010) at 19th International World Wide Web Conference (WWW 2010), vol. 628; 2010.
134. Araújo S, Tran DT, de Vries AP, Schwabe D. SERIMI: class-based matching for instance matching across heterogeneous datasets. *IEEE Trans Knowl Data Eng (TKDE)*. 2015;27(5):1397-1410.
135. Faria D, Pesquita C, Santos E, Palmonari M, Cruz IF, Couto FM. The agreementmakerlight ontology matching system. Paper presented at: Proceedings of the OTM Confederated International Conferences" on the Move to Meaningful Internet Systems"; 2013:527-541; Springer.
136. Cruz IF, Antonelli FP, Stroe C. AgreementMaker: efficient matching for large real-world schemas and ontologies. *Proc VLDB Endow*. 2009;2(2):1586-1589.
137. Tang Y, Wang P, Pan Z, Liu H. Lily results for OAEI 2018. Paper presented at: Proceedings of the 13th International Workshop on Ontology Matching, OM 2018; 2018:179-186.
138. Assi A, Mcheick H, Dhifli W. BIGMat: a distributed affinity-preserving random walk strategy for instance matching on knowledge graphs. Paper presented at: Proceedings of the IEEE International Conference on Big Data, BigData; 2019:1028-1033; Los Angeles, CA.
139. Page L, Brin S, Motwani R, Winograd T. The pagerank citation ranking: bringing order to the web. Stanford InfoLab; 1999.
140. Maria P, Yakout M, Chakrabarti K. Holistic entity matching across knowledge graphs. IEEE Big Data; 2015:1585-1590.
141. Haveliwala TH. Topic-sensitive pagerank. Paper presented at: Proceedings of the 11th international conference on World Wide Web (WWW 2002); 2002:517-526; ACM.
142. Wölger S, Siorpaes K, Bürger T, Simperl E, Thaler S, Hofer C. A survey on data interlinking methods; 2011.
143. Whang SE, Marmaros D, Garcia-Molina H. Pay-as-you-go entity resolution. *IEEE Trans Knowl Data Eng*. 2012;25(5):1111-1124.
144. Siangliulue P, Arnold KC, Gajos KZ, Dow SP. Toward collaborative ideation at scale: Leveraging ideas from others to generate more creative and diverse ideas. Paper presented at: Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work & Social Computing; 2015:937-945; ACM.
145. Karpathy A, Joulin A, Fei-Fei L. Deep fragment embeddings for bidirectional image sentence mapping. Paper presented at: Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2 NIPS'14; 2014:1889-1897.
146. Gao N, Huang SJ, Yan Y, Chen S. Cross modal similarity learning with active queries. *Pattern Recogn*. 2018;75:214-222.
147. Munkres J. Algorithms for the assignment and transportation problems. *J Soc Ind Appl Math*. 1957;5(1):32-38.
148. Melnik S, Garcia-Molina H, Rahm E. Similarity flooding: a versatile graph matching algorithm and its application to schema matching. Paper presented at: Proceedings 18th International Conference on Data Engineering, ICDE 2002; 2002:117-128; IEEE.

How to cite this article: Assi A, Mcheick H, Dhifli W. Data linking over RDF knowledge graphs: A survey. *Concurrency Computat Pract Exper*. 2020;32:e5746. https://doi.org/10.1002/cpe.5746