LinkD: Element-based Data Interlinking of RDF datasets in Linked Data

Mohamed Salah Kettouch[1] and Cristina Luca[1*†]

[1*]School of Computing and Information Science, Anglia Ruskin University, East Road, Cambridge, CB1 1PT, United Kingdom.

*Corresponding author(s). E-mail(s): cristina.luca@aru.ac.uk;
Contributing authors: mohamedkettouch@gmail.com;
†These authors contributed equally to this work.

Abstract

One of the main obstacles in publishing in a Linked Data way is to connect the dataset being published externally with related data sources in the cloud, known as Data Interlinking. This paper proposes LinkD, a new element-based interlinking approach. LinkD interlinks an RDF dataset, resulted from transformed semi-structured data, with its counterparts in the web of Linked Data. To provide similarity links, the existence of published data in the Linked Data cloud is done in the first place. Different algorithms for similarity measurement are employed while the domain of the dataset being interlinked is taken into account. The techniques utilised allow the processing of a large number of Linked Data datasets. The evaluation of LinkD shows high precision, recall and performance.

Keywords: Data interlinking, Linked Data, Semi-structured Data, Link Discovery, Instance Matching, Semantic Web

1 Introduction

Linked Data is the paradigm that enables meaning that is both machine and human-readable; a longstanding aim of the Semantic Web community. The reduction of restrictions in publishing Linked Data led to dramatic growth in the Web of Data, and an extension to many areas and domains [1].

The value of the Web of Data rises and falls with the amount and the quality of links between different data sources [2]. Datasets residing on dispersed data sources without links resemble islands of data [3], where every island stores part of the data needed by the user. To gather all the necessary pieces of information, the user needs to manually find each island.

The ideal scenario in publishing Linked Data is to allocate a unique URI to every real-world entity. Having multiple identities for the same resource reduces its discoverability and, therefore, significantly reduces its value and the chances of it being reused. Considering the distributed nature of the Linked Data paradigm [4] and the massive number of real-word things that exist, this ideal scenario is practically unachievable. Hence, alternative solutions, such as data interlinking, have to be used.

Other data sources employing semi-structured data are also still growing and publishing in the Web. Figure 1 shows the steady increase of the number of Web APIs, which are access tools that use semi-structured data in exchanging information, and considered one of their primary sources. Other indications exhibit that this rapid growth of semi-structured data will be maintained by their usage in emerging technologies such as remote sensors, social media, smartphones and archives [5].

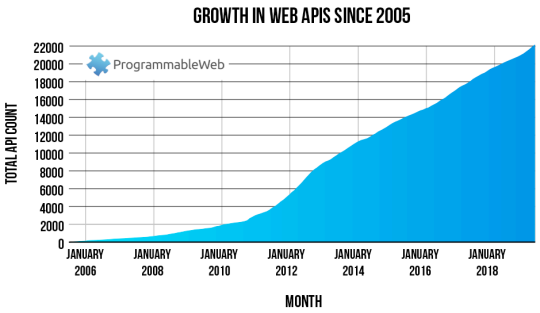

Fig. 1 The increasing growth of Web API. [6]

Although semi-structured data are linked implicitly and enable the machine readability side of the story, semantic links in Linked Data allow Web publishers to make these links explicit, giving therefore access to more data [7]. This leads to a Web where data is more discoverable and usable for both machine and human users. Generating semantic links between different datasets creates the Web of Data, a global database where data is connected to other relevant data.

Unlike publishing semi-structured data in the Web of Data, there are many tools and approaches proposed to interlink Linked Data. Most of these tools are proposed as part of the yearly event of OAEI [8]. Their aims are to link structured RDF datasets with the Web of Data. These approaches employ

frequently unavailable or incomplete information, such as the structure and resource types, in order to find identical instances between sets of source and target resources. Additionally, the ontology-based transformation process is time-consuming and requires a significant amount of input and manual settings to convert a considerable amount of semi-structured data, in order to generate some structural information, which can be incomplete, imprecise, or inaccurate [9] (see Section 2.3.3).

The scope of this paper is to provide a new approach, LinkD, and an implementation tool to externally link transformed semi-structured data with the Linked Data cloud. LinkD initially verifies the existence of the URI of the resource being published in the cloud to establish similarity links with the findings. LinkD takes in an RDF file resulted from transformed semi-structured data. We have introduced a domain detection phase to impose variable weights to the properties of the data being interlinked. LinkD utilises element-based data interlinking that takes into account solely the properties of the source and target datasets, without the need to process or align the overall structures or ontologies. We have used an asymmetric and unsupervised algorithm to compute the similarities. This approach is different to other existing interlinking solutions that take in already published data both as the source and the target. The overall aim of the research is to facilitate the best practices and recommendations [10] in publishing data into the Linked Open Data cloud.

The rest of the paper is structured as follows: Section 2 introduces an overview of concepts and the techniques employed to build our method. Section 3 gives a summary of existing works related to our research. Section 4 presents the proposed approach, LinkD including the extraction of the semantically distinct properties and the creation of the global schema. The implementation of LinkD is described in Section 5. In Section 6, the testing and evaluation are discussed. Finally, the conclusions are drawn in Section 7 stating the limitations of our approach and the future works envisaged.

2 Background

This section presents the background needed to understand the methods/algorithms used in the LinkD approach that we introduce in this paper.

2.1 Linked Data

Linked Data is a pragmatic approach for the transformation from a document-based Web to a Web of interlinked structured data. The idea was to create a Web where anything can be linked to anything. Linked Data aims to provide links between different data sources in order to create a single global data space, "the Web of Data" [11]. These links ought to be machine-readable and connect to related data whether from the same or from other external sources. This objective can be achieved by utilising RDF, URIs and HTTP to publish and interlink structured data on the Web. More formally, Linked Data refers to a set of best practices for publishing and interlinking structured data on the

Web, described by Tim Berners-Lee in his Web architecture note on Linked Data [10]:

- Using URIs as names for things.
- Using HTTP URIs so that people can look up those names.
- When someone looks up a URI, provide useful information.
- Include links to other URIs, so that they can discover more things.

2.2 Structuring Semi-structured Data

Semi-structured data are "schema-less" data [12], meaning they do not have any rigid and predetermined schema upfront, and "self-describing" [12], which refers to the fact that the structure and the values are embedded in the same file), being therefore the most suitable and natural data model to accommodate heterogeneity.

The problem of converting hierarchical or tree-based data models (such as JSON and XML) to graph-based data models has existed for more than a decade. Various solutions have been proposed [13–16] that can be classified into two categories: ontology-dependent RDF transformation and fixed RDF transformation.

The systems in the first class are based on ontologies when converting semi-structured data schema, frequently XML, to an RDF schema. It is a challenging task to project the representation of concepts and the relationships between them of a given ontology while converting from one data model to another. An example of this approach is [17] that proposes a system that takes as inputs an XML file, an OWL ontology and the mapping document describing the links between the XML file and the ontology. RDF instances conforming to the OWL ontology are the outcome of this tool.

The fixed RDF transformation consists of syntactical and generic conversions from one data model and format to another. The transition consists of mainly restructuring and reorganising different components of semi-structured data (namespace, root, tags, attributes and values) into a subject, predicate and object RDF structure. The RDF file generated is a set of triples describing resources according to the hierarchy of the transformed tree-based XML or JSON file. This operation is not considered challenging as an XSLT[1] script or the combination of JSON/XML parser with Jena framework can achieve an acceptable result. The disadvantage of this operation is the fact that no meaning will be associated with the resultant RDF file. Many examples of tools appertain to this class of systems can be stated including [18] or the Java library XmlToRdf[2].

The fixed transformation is selected to be used in this paper because it can be automated and requires less pre-transformation effort, which is necessary for the system we propose to be adapted to interlink large-scale datasets. Interlinking datasets with the Web of Data necessitates the utilisation of

[1] Extensible Stylesheet Language Transformations
[2] https://github.com/AcandoNorway/XmlToRdf

lightweight processes and avoidance of operations such as type identification or type-related comparisons.

2.3 Data Interlinking

Data interlinking provides `owl:sameAs` links, as illustrated in Figure 2, between items representing the same resources that may be situated in the same or in different data sources. `owl:sameAs` links, as provided by OWL semantics, allow the discoverability of references to identical resources residing in different machine readable data repositories. They are also used to materialise inferable knowledge and to potentially generate additional results [19].

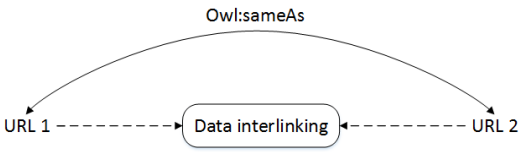

Fig. 2 The Data Interlinking Process

In this section we identify and explain four of the main phases and concepts related to the interlinking task; two of these are fairly indispensable, being blocking and instance matching. Figure 3 shows how these stages are positioned in the data interlinking process. The use of ontologies and similarity measures are two popular methods employed to determine whether two descriptions or labels, respectively, refer to the same real-world entity.

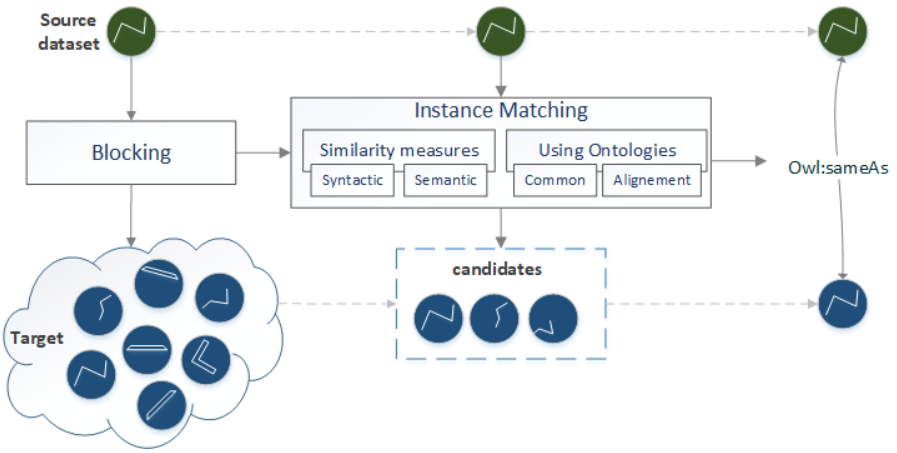

Fig. 3 General architecture of data interlinking approaches

2.3.1 Blocking

Blocking, in this context, means grouping similar objects using a blocking key. It is the initial stage in the interlinking process whereby the number of candidates is reduced. As a result, a block that consists of a set of potential identity pairs of instances is generated. This is an important step as it affects the performance of the system, considering that the inputs of the heavy processing operations in the instance matching stage will have resulted from the blocking. The blocking stage aims to optimise two evaluation metrics, Reduction Ratio and Pair Completeness.

1. **Reduction Ratio**
   This measure represents the efficiency of the blocking [20]. It quantifies the ability of a blocking algorithm to minimise the number of comparisons (in further stages) by removing obvious non-matches. More formally:

$$RR\,(Reduction\ Ratio) = 1 - \frac{N}{\mid S \mid \cdot \mid T \mid} \tag{1}$$

   where $\mid S \mid \cdot \mid T \mid$ is the number of all pairs between S (number of inputs of the source dataset) and T (number of inputs of the target dataset).
   N indicates the number of pairs produced by the blocking.
2. **Pair Completeness** This value measures the number of true matches (C_m) identified by the blocking algorithm versus the number M of those that exist in the entire dataset, as described in the equation 2:

$$PC\,(Pair\ Completeness) = \frac{C_m}{M} \tag{2}$$

   Therefore, theoretically: $C_m \leq M$

2.3.2 Instance Matching

Instance matching goes by a number of different names, these being: record linkage, data matching, the merge-purge problem and entity resolution [21, 22]. Instance matching is the problem of matching pairs of instances that refer to the same underlying entity [23]. Instance matching is a technique originating from knowledge discovery and data mining algorithms [22]. But recently, it has seen numerous applications in Web Semantics.

In data interlinking, this is the stage that immediately succeeds the blocking step. The matching status of the resulted pairs is verified in order to discover identity pairs [24].

Three measures are utilised to verify the effectiveness of an instance matching approach, and all of them have been used in the evaluation of LinkD:

1. **Recall** The recall measure represents the ability to retain the true matches, or true `owl:sameAs` links in the Linked Data terms. It is calculated using the equation below:

$$Recall = \frac{\text{The number of true } sameAs \text{ discovered links}}{\text{The number of actual links}} \tag{3}$$

2. **Precision** The precision measure represents the percentage of true matches that lie within the discovered links. The equation to calculate the precision is similar to that used to calculate recall, but instead of dividing the number of true matches by the number of actual links, for precision they are divided by all the discovered links.

$$Precision = \frac{\text{The number of true } sameAs \text{ discovered links}}{\text{The number of all discovered links}} \tag{4}$$

3. **F1 score**

   Neither precision nor recall separately will accurately reflect the match quality since their values can be maximised at the expense of each other (high recall can be easily achieved at the cost of poor precision by returning as many candidates as possible, and to maximise the precision at the expense of poor recall the matcher may return only a few correct correspondences) [25]. Therefore, it is necessary to take into account both measures or a combined measure. F1 is the combined measure and the harmonic mean of the recall and the precision.

$$F1 = \frac{2 * precision * recall}{precision + recall} \tag{5}$$

2.3.3 Using Ontologies in Data Interlinking Alignment

Ontology alignment is the process of finding correspondences [26] between concepts, properties, or instances in two or more ontologies, based on their similarities [27]. Ontologies in data interlinking are generally used to identify and compare instances that are part of the same classes, based on them having the same properties.

Using ontologies does not exclude the possibility of using other similarity techniques. Their utilisation can serve as a hint that materialises as a coefficient or as an element of a similarity algorithm, for example. Experiments have revealed also "that the use of ontology features increases accuracy of instance matching for data integration" [28, p. 1].

There are many methods by which ontologies can take part in an interlinking process. They can be summarised, however, under two broad headings. The first approach is to describe the two resources using a common ontology before the interlinking and matching process takes place, as Figure 4 illustrates. The second approach is to align the independent ontologies of the two resources to draw correspondence that will then be used in the interlinking, as Figure 5 shows.

2.3.4 Similarity Measures

Similarity algorithms are used to measure the distances between the properties of the elements of the source and target datasets. They can be sorted into one of two categories: Syntactic or Semantic.

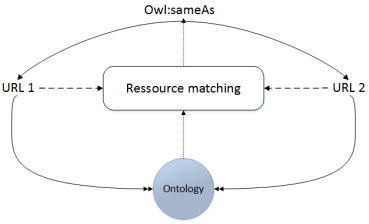

Fig. 4 Data interlinking via a Common Ontology

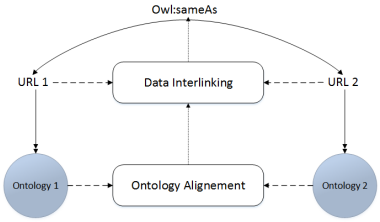

Fig. 5 Ontology Alignment in Data Interlinking

Syntactic similarity refers to the set and string similarity algorithms that are used in some instance matching approaches to calculate the syntactic distance between two predicates or entity labels. Jaro-Winkler [29] is a popular example of a string similarity algorithm. This algorithm uses a mixture of string and set similarity, meaning that the compared values may be tokenised before the standard Jaro-Winkler algorithm is applied and the maximal total score is selected.

In semantic similarity algorithms and tools, the distance used is based on the meaning of the word rather than on its label or lexical form. The UMBC tool is an example of one kind of tool which has been proposed for semantic similarity measurement. It is constructed by combining the use of LSA word similarity and WorldNet knowledge. UMBC focuses on the semantics of the word but not on its lexical category. This makes it a typical similarity measurement mean for data interlinking and integration approaches which take Linked Data as at least one of their inputs, since the available vocabularies for describing resources in this paradigm vary between nouns and verbs.

3 Related Work

This section presents a review of the most popular and more related solutions that differ significantly in terms of how they addressed the data interlinking issue.

SERIMI and SLINT are two approaches presented as part of the yearly OAEI event [30] [31]. Both tools do not require any ontology alignment upfront or prior knowledge of the data or the schema. SERIMI is based on existing traditional information retrieval and string matching algorithms, whilst SLINT uses coverage and discriminability to select important predicates.

The limitation of SERIMI and similar approaches is their restriction and focus on a single (or few) property(ies) considered in the matching phase [32]. Additionally, the similarity threshold and other parameters ought to be specified manually.

Risk Minimization based Ontology Mapping (RiMOM) [33], first developed in 2006 [34], is an instance of a multi-strategy ontology matching and property matching approach. It is based on the combination of three lexical strategies being: EditDistance, Vector-Distance and WordNet [35]. AgreementMaker is

another example of a powerful and extensible ontology matching system that has been in development since 2001 [36]. It was first proposed to map large-scale schemas and ontologies that are extracted from relational, XML, and RDF sources. Initially it was designed to focus on geospatial applications but has since expanded to cover other domains including biomedical applications [37].

Among the projects aiming at interlinking data in a specific domain, Event-Media [38] and the approach proposed by [39] are more close to the approach proposed in this paper. As part of their projects, the authors tried to find the most accurate weights to be given to the properties in the selected domain.

The Link Discovery Framework (SILK) [2] and LIMES [40] are link discovery systems that provide support to publishing data while setting explicit links between the source and target datasets. SILK utilises its own declarative language, Silk - Link Specification Language (Silk-LSL), that data publishers can use to choose which types of RDF links ought to be discovered between data sources and which conditions the data items must fulfil in order to be interlinked. LIMES, however, focuses on improving the processing time when mapping large knowledge bases. It views the problem of data interlinking from a metric space perspective. It uses mathematical characteristics, such as triangle inequality, to compute pessimistic approximations of distances and to estimate the similarity between instances [41]. Based on these approximations, LIMES finds and excludes a large number of computations without losing links. LIMES, however, is limited to utilise only frequently used properties [42] and does not perform as efficiently with uncommon properties.

Legato [43] and [44] are two good examples of approaches aiming at reconciling the heterogeneity of two graph-based datasets described in multiple ontologies, in order to provide similarity links. Legato does not require any prior user input, and as initially proposed, does not compare property values, as opposed to SILK or Limes. It utilises a concept called a bag-of-words that consists of all extractable literal values. Whereas in [44], the authors introduced a new concept called linkkey, which is a set of properties that are a key for two classes at the same time, suggesting similarities between resources that have identical values for the set of these properties. In contrast with the topic covered in this paper, Legato and [44] address heterogeneity at an ontological level; hence, accommodating other data structures is beyond its scope.

When the authors designed LinkD, one of their objectives was to automate the process and nullify the manual input to avoid the limitations of the existing systems such as the use of a dedicated syntax or a declarative language, and of a specific implementation of the approach. LinkD is based on instance matching and similarity measurement algorithms that allow processing a large number of Linked Data datasets. Moreover, this paper aims at breaking free from any dependency on a specific domain or topic, and using tools, such a string measurement, to generalise the mapping of properties. Balancing between performance and precision is another important aspect that was not the scope of any of the presented approaches specifically, and reviewed

approaches generally. The comparison carried out in the evaluation of these tools is mainly centred on the precision and recall but not the performance.

One common feature shared by the approaches discussed in this section, apart from SILK, is to discover identity and/or other links in existing published data. Taking the performance into account, it would be theoretically more efficient finding links in the publishing stage. Meaning that the tool should automatically interlink the data being published with its existing counterpart in the Web of Linked Data.

LinkD has similarities with SERIMI system [30], and is based on the assumptions of a variety of domain-dependent interlinking systems, such as EventMedia [38] and the system proposed by [39]. However, LinkD uses property weights of the instances that have not been taken into consideration in the SERIMI approach.

4 LinkD architecture

The scope of this section is to present LinkD, that aims at interlinking an RDF dataset with its counterpart in the Linked Data cloud using different algorithms for similarity measurement, and taking into account the domain of the dataset being interlinked. Following the best practices, publishing data into the Linked Data cloud is challenging because of the scale of the data. Therefore, LinkD also aims to facilitate and automate this process.

LinkD externally links an RDF file with no explicit meaning or structure associated with it. The RDF file is the result of the fixed RDF transformation from a semi-structured file, as described in Section 2.2. The labels of the semi-structured data are the only features that can be confirmed to be maintained after the fixed transformation (see Section 2.2).

One of the novel ideas presented in this paper is to add a domain detection phase, before matching the instances, in order to impose variable weights to the properties of the data being interlinked. The weights are extracted from the existing observations of the specialised systems (EventMedia [38] and [39]) and extended by the authors' observation in other domains. The variable weights are applied to the value of the matched resource properties according to the domain extracted. For instance, the similarity of the values of the resource properties longitude and latitude will be given more weights in the case where the domain detected is geospatial; whereas, if the domain is an event, the coefficient of the similarity index between the value of resource properties time and place will be higher than other properties.

LinkD is based on string and set similarity as it takes in account the characteristics of the RDF file, and aims at using lightweight processes and avoid operations such as type identification or type-related comparisons. An asymmetric and unsupervised approach is used in LinkD to compute the similarities. It would be impractical to gather all the prerequisites needed to process a large

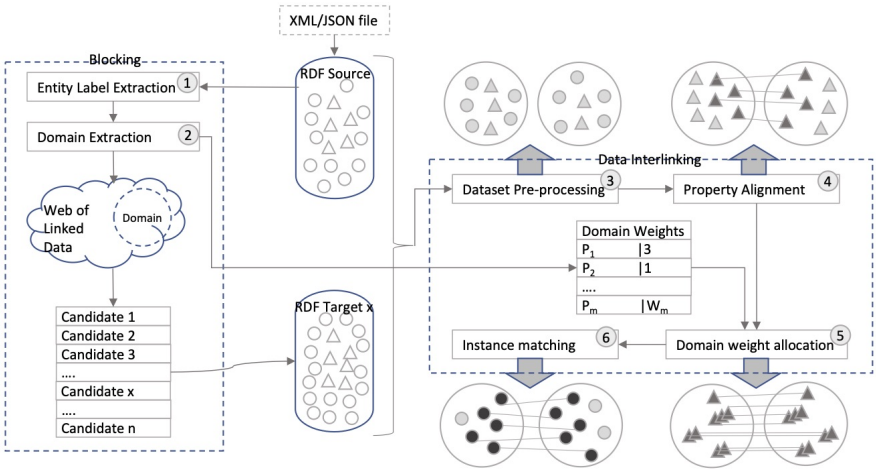

Fig. 6 General architecture of LinkD

source of Linked Data using manual or supervised techniques [30]. The common drawback of unsupervised algorithms, however, is the high computational cost required to implement them.

Figure 6 shows that the source dataset goes through many stages before matches in the cloud can be found. These stages can be organised and grouped in two main phases being Blocking (described in Section 2.3.1) and Data Interlinking (described in Section 2.3). Figure 7 shows the interactions between the LinkD's processes represented as rectangles, where the labels on the arrows are the output of one process that is the input of the next one. The labeled arrow has been replaced with a data block, the parallelogram, for the output of the "Extraction and pre-process properties" as it is the input of more than one process.

Contrary to the common instance matching approaches, LinkD starts with one dataset, which is the source dataset. The source dataset is an RDF file derived from XML/JSON files that might not be described by an ontology or associated with any meaningful structure. The target datasets are retrieved after running a keyword SPARQL search query on the Linked Data namespaces considered. The SPARQL query is composed by extracting the domain and the entity label (labels that represent the dataset) of the source dataset. The latter is generally the content of the property title, name or label that has a literal value shorter than 200 characters [30].

The domain extraction is an important phase in finding the counterparts of a dataset in Linked Data cloud in the proposed system. The domain is determined by extracting the content of the property rdf:type and classifying it with one of the pre-defined domain categories available in the system. In the case where rdf:type is not used, the domain is selected manually.

Having the domain extracted, the potential candidate for the interlinking will be significantly reduced and limited as a result of more specific SPARQL

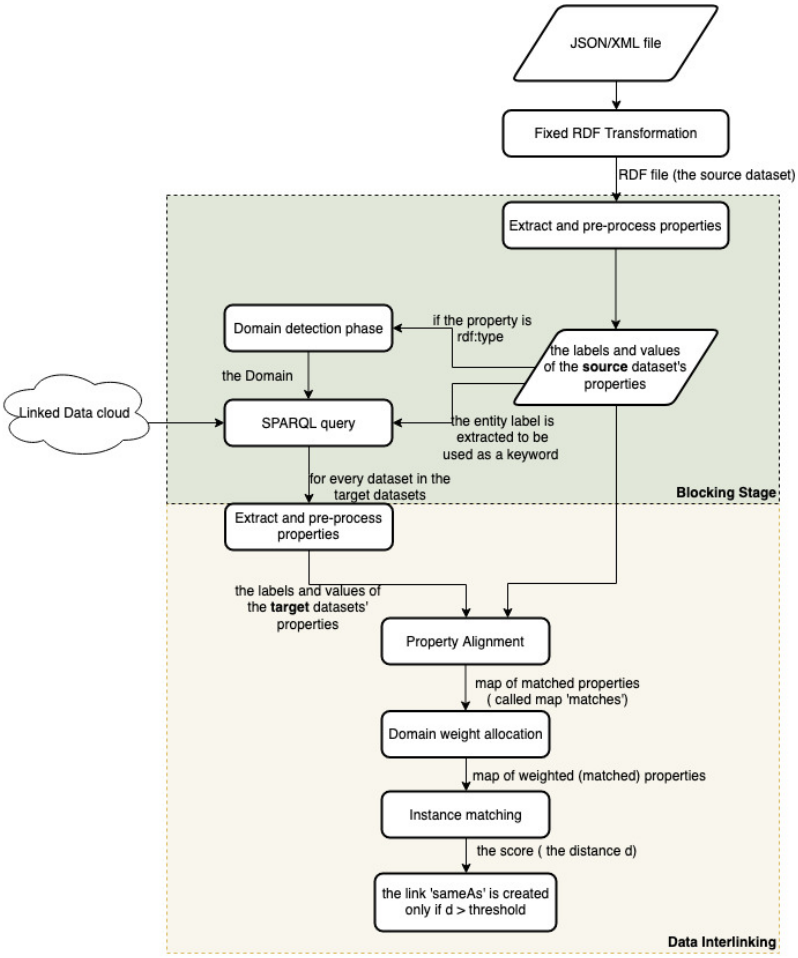

Fig. 7 Flowchart of LinkD's processes

keyword search. In the next stage (Data interlinking), different weights will be applied according to the domain detected. More weight will be given to the properties that define more the identity of the dataset (properties with unique values) and create less conflict with the other datasets. For example, longitude and latitude for location or ISBN number for books.

Listing 1 is an example of the SPAQRL template implementation. Keywords, such as "London", are the entity label properties extracted from the fixed RDF transformation of XML/JSON file, and are every literal value with less than 200 characters (detailed more in Section 4.1.3). The domain "schema:Movie" is extracted from the domain tags of the XML/JSON file, or from the pre-categorisation of their source Web APIs.

Listing 1 SPARQL query to search for target datasets from DBpedia using a keyword

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX dbp: <http://dbpedia.org/property/>
SELECT * WHERE {
  ?title rdf:type schema:Movie;
    dbp:title ?keyword.
  FILTER(REGEX(?keyword, "London", "i"))
}
LIMIT 4
```

4.1 Data Interlinking

After the blocking stage, the system compares the properties of each of the candidates (from the target datasets) with the properties of all sources of the datasets in order to determine which candidate can be linked with it. Three types of links exist in Linked Data: relationship, identity or vocabulary links. Identity links or `owl:sameAs` are the most common type of links addressed by the existing interlinking systems and this relationship is the focus of LinkD. Along with allowing the representation of semantic equivalence in an independent and reusable way, `owl:sameAs` can serve as hints to a reasoner system on how to unify data.

The LinkD data interlinking stage consists of four steps described in the following sub-sections.

4.1.1 Preparing the Datasets

A pre-processing step is performed to extract the value from the resources that are described using URIs. According to Linked Data principles, the value is the last part of an URI. Commas and underlines will be also replaced by spaces to improve the accuracy of the matching algorithms.

Example: the output of the http://dbpedia.org/page/London_River after pre-processing is `London River`.

4.1.2 Property Alignment (SimiMatch)

This stage is responsible for matching between the semantically similar properties of the source and the target (candidate for interlinking) datasets. It is based on a schema matching approach called SimiMatch, previously proposed by the authors [45, 46]. SimiMatch is an element-based schema matching approach that targets two data models, the semi-structured (hierarchical) model and Linked Data (graph) model. SimiMatch does not utilise any reference, such as a knowledge base or an ontology in generating the matching rules. As a result, it has the ability to process large-scale sources. It is used in LinkD to generate matching rules between the two schemas instead of creating a global schema.

Algorithm 1 is the adaptation of SimiMatch [47] to align properties of two property sets in LinkD. SimiMatch measures the semantic distance between

the label of the source's predicates and the candidate datasets, and compares it to a threshold. The matching rules are expressed using a map data structure. The map (called **matches**) stores data in the form of (key, value) pairs where every key is unique. As a result, Algorithm 1 iterates, for each key, through values in order to find the pair with the highest semantic similarity (or the lowest similarity distance) score that is above the threshold. If a new pair with a higher similarity score is discovered, the new value of the key in the **matches** map will replace the previous one. A candidate property is matched with one of the source properties.

An optimal threshold (0.75) used by SimiMatch was chosen after experimenting with different values on the same datasets [46].

SimiMatch utilises a semantic similarity tool called SemanticDistance which is based on a reimplementation of UMBC [48]. The local version helps to eliminate the time penalty of connecting to the API every time a semantic distance is calculated.

Algorithm 1 SimiMatch in LinkD

Require: **set1, set2:** PropertiesSets
threshold
Ensure: **matches:** Map

```
sizeSet1 = size (set1)
sizeSet2 = size (set2)

while i < sizeSet1 do
    temp_similarity=0;
    while j < sizeSet2 do
        similarity = SemanticSimilarity(set1[i], set2[j]);
        if similarity > threshold AND similarity > temp_distance then
            matches.add(set1[i], set2[j])
            temp_similarity = similarity
        end if
    end while
end while

return matches;
```

4.1.3 Domain Weight Allocation

It is observed and validated in [49] that RDF datasets referring to the same real world object or describing resources in the same domain share roughly the same properties even though the syntax may be expressed differently. More importantly, it is noticed in many systems for interlinking domain-dependent Linked Data, that some properties are more precise in defining the identity

of a dataset [49]. Having a prediction of a limited list of the properties that will be matched in a particular domain, it becomes feasible to set rules for allocating weights for the similarity index of the content of these properties.

Many factors come into play in defining the weight of the properties in Data interlinking with the Web of Linked Data. Three decisive criteria are identified in this paper:

- **Number of repetitions of the property and its content:** Similarly to the primary key in a database, the property or the instance with higher weight needs to be unique; thus, its value should not repeat in the candidate set or in a particular domain.
- **Content length:** the result of the string similarity (both semantic and syntactic) applied in LinkD can be negatively affected by long literals.
- **Time relatedness:** this criterion identifies unique properties that do not change over time, something that can mislead the interlinking process. To find whether a property is time related, at least two versions of the published resource need to be compared.

Several approaches attempted to improve data interlinking, and instance matching performance and precision using property weights, such as: RIMOM [50], CODI [51] and BOEMIE [52]. These approaches are not adapted to be utilised in LinkD for many reasons, including:

- They do not consider all three criteria listed above, or
- The properties with distinct values are not domain-dependent, or
- The processing of the weight is embedded in the matching process, or
- They are not element based and depend on the structure.

The first and second reasons can make the weight allocation process incomplete. The third and last reasons can make the interlinking system not suitable to process large scale of data as it can affect considerably the performance and computational time.

The weight allocation used by LinkD is based on the weight generator proposed by [53] as it is close to meeting the criteria identified above. The approach is based on penalising repeated properties by a negative probability factor. LinkD adds two other negative factors that represent "properties' content length" and "properties' time-relatedness". This allows LinkD to cover all the identified requirements including the calculation of the uniqueness of the properties and their abilities to define the datasets they describe. Equation 6 defines the function λ used by LinkD to calculate the weight of a property p.

$$\lambda(p) = (1.0 - np1(p))(1.0 - np2(p))(1.0 - np3(p))(1.0 - np4(p)) \quad (6)$$

The weight λ is penalised by three factors `np1`, `np2`, `np3` and `np4`.

`np1` is the ratio of the number of repetitions to the number of instances the property belongs to, as described in equation 7.

$$np1 = \frac{Rep(p)}{\mid p \in I \mid} \tag{7}$$

`np2` represents the repetition of the content of resource properties over the total number of instances (I) (which it does not necessarily belongs to), as defined in equation 8.

$$np2 = \frac{Rep(p)}{\mid I \mid} \tag{8}$$

The value `np3` penalises properties with literals that are longer than 200 characters. This value has been chosen as a result of various tests done by the authors. It is also considered by the SERIMI's authors as a representative entity label.

$$np3 = \begin{cases} 0 & length(obj(p)) \leqslant 200 \\ \dfrac{1}{length(obj(p))} & otherwise \end{cases} \tag{9}$$

The value of `np4` is 1 if the object of the property can change from version to version ($ver_{1..n}$) and/or depends on time. For example: the values of the predicates http://dbpedia.org/ontology/address and http://dbpedia.org/ontology/currentMember are examples of properties that may change over time.

$$np4 = \begin{cases} 1 & if\ obj(p_{ver1}) \neq obj(p_{ver2}) \\ 0 & otherwise \end{cases} \tag{10}$$

4.1.4 Instance Matching

Having the list of matched properties between the source dataset and each of the target datasets, LinkD extracts their content (instance). The similarity of the instances is then measured using Jaro-Winkler algorithm. The Jaro distance d between two strings ($s1$ and $s2$) is the result of the equation 11.

$$d(s1, s2) = \begin{cases} 0 & if\ m = 0 \\ \dfrac{1}{3}\left(\dfrac{m}{s1} + \dfrac{m}{s2} + \dfrac{m-t}{m}\right) & otherwise \end{cases} \tag{11}$$

Where:

- $s1$ and $s2$ are the labels of the instances of source and target datasets respectively.
- m is the number of matching characters.
- t is half the number of transpositions.

Finally, the similarities of the properties and their instances are combined in a linear combination of the measures described in Tversky's contrast model as shown in the equation below:

$$Tversky(A, B) = \lambda f(A \cap B) - \alpha f(A - B) - \beta f(B - A) \quad (12)$$

Where: α, β, and $\lambda \geq 0$. Three parts can be noticed in the Tversky model:

- (A ∩ B) represents the set of common properties between A and B
- (A - B) are the set of distinct properties found in A but not B
- (B - A) are the set of distinct properties found in B but not A

The coefficients α, β, and λ represent the weights of the commonalities and differences in the equation. Since the distinctness between the resources is not relevant in our case, α and β are set to 0. The authors decided to set the value of λ according to the domain of the source dataset.

f is a function that calculates the distances between the values of all the properties passed as parameters, as described in equation 11. The final value is the average of all the distances divided by the sum of considered weights. A large scale study carried out in [54] showed that LinkD performs best when the threshold is above 0.75, and more specifically around 0.82, which is the value used in the evaluation presented in this paper.

The Instance Matching is the last step of establishing `owl:sameAs` links between the source dataset and the target datasets.

5 Implementation

The IM@OAEI [55] benchmark is a popular measure of data connectedness that was chosen to evaluate LinkD due to its ability to cover datasets from different domains.

LinkD [3] [4] and the semantic similarity tool are implemented using Java and Jena libraries. As explained previously in this paper, Linked Data sources are changing quickly over time. For example, in DBpedia version 2016-04, triples are filtered from the Raw Infobox Extractor and some properties will not be loaded on the public endpoint. Thus, running the system at a different time can require a different method to prepare the input and may display different results but the trend will be maintained. The public services (SPARQL endpoints) of Linked Data sources frequently apply resource limits and they are occasionally unavailable [56]. Therefore, RDF dumps were downloaded locally in HDT[5] (Header, Dictionary, Triples) format to avoid this limitation. HDT is a compact data structure and binary serialisation format for RDF that keeps big datasets compressed to save space while maintaining search and browse operations without prior decompression.

In the implementation of LinkD, the blocking stage is run separately from the data interlinking. The output of the blocking is the input of the Data Interlinking stage. The implementation of the Data Interlinking system follows the description in Section 4.1.3. It consists of four modules:

[3] https://github.com/medke/LinkD-tool
[4] https://github.com/medke/SimiMatch-tool
[5] http://www.rdfhdt.org/

- A pre-processing module that is responsible for extracting and preparing the labels of the properties.
- SimiMatch project was linked to the LinkD project in order to create an instance and use some of its functionalities, particularly the semantic distinct and distance.
- the WeightAllocator module is called at different stages of the running of the program to prioritise properties over others according to their significance in the defining the resource being interlinked.
- InstanceMatcher utilises WeightAllocator, the semantic distance of SimiMatch (re-implementation of UMBC tool) and Jaro-Winkler set similarity in order to perform the last step of establishing `owl:sameAs` Links between resources.

6 Testing and Evaluation

Five datasets from IM@OAEI2011 were utilised containing four identified domains (movies, people, locations and organisations) and three Linked Data sources (DBpedia, LinkedMDB and NYTimes). In the other approaches utilising this benchmark, the number of target pairs is the same as the source pairs as their aims are to find identity links between two sets. In LinkD, however, the aim is to provide links with the Linked Data cloud; hence, the target pairs are the entire DBpedia repository (English DBpedia 3.9[6]). Although DBpedia is not the Linked Data cloud, it is the largest Linked Data repository that can be used as a target to evaluate LinkD against large-scale data. Having many Linked Data providers on the target side removes the possibility of approximate numerical comparison against other related systems. Table 1 gives the overview of the considered datasets (D1 to D5).

Table 1 Details of the considered datasets from IM@OAEI

ID	Source	Target	Domain	Source Pairs	Target Pairs	Target Domain Pairs
D1	LinkedMDB	DBpedia	movies	10108		77769
D2	LinkedMDB	DBpedia	people	3650		831558
D3	NYTimes	DBpedia	locations	2083	474M	639450
D4	NYTimes	DBpedia	people	4588		831558
D5	NYTimes	DBpedia	organisations	1274		209471

6.1 Evaluation of the Blocking stage

Table 2 shows the result of the evaluation of the blocking stage. To calculate the pair completeness (PC), the evaluation needs to have a gold standard upon which the true positive (correct) candidates can be counted. It is something

[6] https://downloads.dbpedia.org/3.9/

that related approaches do not clarify in their evaluation and the results are displayed without giving details about the values of the components of the equation.

It can be seen from the results that the blocking stage fulfills its role effectively, by reducing the amount of potential matches by more than 0.97 (column RR) - except in dataset D1 - while preserving the vast majority of true candidates. The results obtained at this stage have a significant impact on the comparable end results, which serve as an input for the next stage; hence, have a significant impact on the performance and precision of overall evaluation.

Table 2 Results of the blocking stage

ID	Instance Pairs	Target Pairs (after blocking)	Correct Candidates	Candidate Pairs	RR	PC
D1	786079023	16868	9829	170501744	0.78	0.97
D2	3035186700	24123	3447	88048950	0.97	0.94
D3	1331974 350	12545	2000	42044235	0.98	0.96
D4	3815188104	18740	4496	85155120	0.98	0.98
D5	266866054	6672	1269	8500128	0.97	0.97

The correct candidates in Table 2 are based on estimating the number of occurrences of the actual `sameAs` links of the datasets in the target pairs.

6.2 Evaluation of Instance Matching Stage

Three metrics are used to evaluate the Instance Matching Stage: recall, precision and F1 [57].

Table 3 reports the results of the instance matching stage. These values represent the lower band results as the benchmark utilised is created in 2011, which means new resources that may contain true positives that are not listed could have been published since then.

Table 3 Results of the instance matching stage

ID	Source Pairs	Links Discovered	Matched Instances	Unmatched Instances	Properties Aligned	Runtime (seconds)	Rec	Pr	F1
D1	10108	10989	9586	522	1044776	51007	0.95	0.87	0.91
D2	3650	3896	3388	262	706400	2241	0.93	0.87	0.9
D3	2083	1825	1811	272	327825	1774	0.87	0.99	0.93
D4	4588	4765	4476	112	338037	2247	0.98	0.94	0.96
D5	1274	1306	1198	76	135015	1519	0.94	0.92	0.93

The weight allocation stage is run before the interlinking. It is a separate process that is not repeated for every interlinking unless new datasets that belong to a domain that has not been previously processed are added. The

result is an array for every domain considered that contains properties labels and their weights.

The results of this stage are generally good, as shown in the F1 column of Table 3, and sufficiently high to enable comparison with other similar systems. Comparison is not carried out at this stage because instance matching efficacy represents only one aspect in interlinking approaches, and does not take into consideration other measures, such as the amount of targeted the data as well as the performance, which will be covered in the next section.

6.3 Comparison with Previous Interlinking Systems

Figure 8 provides a comparison between LinkD and the selected interlinking systems. It can be clearly noticed the extent of the improvements that LinkD introduced to SERIMI. Whilst D1 and D2 are joined together in the other approaches, in LinkD, however, they are separated into two domains being movies and people (actors, writers, directors, etc.).

Table 4 Comparison of LinkD with related systems

ID	Target Pairs			
	LinkD	**SLINT**	**SERIMI**	**Agree.Maker**
D1	474M		10108	
D2			3650	
D3			2083	
D4			4588	
D5			1274	

Although Figure 8 shows that SLINT is performing better in terms of F1 score in all the datasets considered, the scale of the targeted data is significantly larger in LinkD, as highlighted in table 4. This points out the distinction between the nature of the problem addressed by LinkD as opposed to the other approaches. It is the only way, however, to numerically evaluate LinkD and to show its performance, despite the difference in terms of the scale of the data targeted.

Table 5 reports the time it took LinkD to process the datasets D1-D5 comparing to SLINT. It is not a direct comparison given the discrepancy between the amount of target pairs considered by each system, as highlighted in Table 4. For instance, with the presumption that SLINT performance strongly and directly correlates with the amount of the target datasets, its performance for D1 would be 474 Millions divided by 10108, multiplied by 67, the results is approximately 3,141,867 seconds.

Table 6 reaffirms the features enabled by LinkD comparing to the state-of-the-art. LinkD and SERIMI are both unsupervised approaches that enable higher automation and less relatedness to a training data, thus to a specific domain. LinkD also does not rely on the ontology of the input data, something

Fig. 8 Comparison of LinkD with related systems

that could negatively impact the scalability and performance if otherwise. It employs, on the other hand, many string based tools to first filter the amount of candidates through blocking, along with other less resource-hungry tools, such as the property weight technique, to achieve compareable precision and recall. Overall, the system we proposed in this paper can produce, on average, 93% correct owl:sameAs links between an input data and its counterpart in Linked Data cloud that consists of over 474M pairs.

7 Conclusion

This paper presented a new data interlinking approach, LinkD, that takes only a source dataset as input and provides identity external links with many sources of Linked Data cloud. The characteristics of the RDF file are used

Table 5 Performance evaluation of LinkD against SLINT

ID	LinkD (seconds) for 474M	SLINT (seconds)
D1	51 007	67 (for 13758)
D2	2 241	
D3	1 774	3.55(for 2083)
D4	2 247	12.74 (for 4588)
D5	1 519	4.29 (for 1274)

Table 6 Comparison of LinkD with related systems

	LinkD	SLINT	SERIMI	AgreementMaker
Learning based	n	y	n	y
Ontology matching	n	n	n	y
Data Input	RDF	RDF	SPARQL	SPARQL
Supported Link type	owl:sameAs	owl:sameAs	owl:sameAs	owl:sameAs
Blocking	y	n	n	n
String similarity measure	y	y	y	y
Property weights	y	n	n	n
Domain detection phase	y	n	n	n
Blocking strategy	Filtering	Indexing	Indexing	Indexing

as requirements in designing LinkD. A variety of novel distance measurement tools and algorithms were used to calculate the similarity between the labels describing the resources. Neither the structure nor the ontology of the dataset were considered on the proposed system in order to maintain its feasibility to target large-scale datasets. The major challenges faced are the high computational cost and the incorporation of dynamic allocation of the weights according to the domain and the number of the matched properties.

Further work will explore ways to extend LinkD to a publishing approach of semi-structured data as Linked Data. LinkD would also benefit from disambiguation stage since it focuses on the label of the properties. This allows to include data sources that are not data mining and parsing friendly; therefore, expanding its use to other use cases.

References

[1] Abele, A., McCrae, J.: The Linking Open Data cloud diagram (2017). http://lod-cloud.net/ Accessed 25 June 2017

[2] Volz, J., Bizer, C., Gaedke, M., Kobilarov, G.: Silk-a link discovery framework for the web of data. In: Proceedings of the WWW2009 Workshop on Linked Data on the Web-Volume 538 (2009). CEUR-WS.org

[3] Groza, T., Grimnes, G.A., Handschuh, S., Decker, S.: From raw publications to linked data. Knowledge and Information Systems **34**(1), 1–21 (2013)

[4] Hu, W., Yang, R., Qu, Y.: Automatically generating data linkages using class-based discriminative properties. Data & Knowledge Engineering **91**, 34–51 (2014)

[5] Yuliana, O.Y., Chang, C.-H.: A novel alignment algorithm for effective web data extraction from singleton-item pages. Applied Intelligence **48**(11), 4355–4370 (2018)

[6] ProgrammableWeb: ProgrammableWeb Research Center (2019). https://www.programmableweb.com/api-research Accessed 20th January 2020

[7] Ristoski, P., Paulheim, H.: Semantic web in data mining and knowledge discovery: A comprehensive survey. Web semantics: science, services and agents on the World Wide Web **36**, 1–22 (2016)

[8] Jimenez-Ruiz, E.: Ontology Alignment Evaluation Initiative (2017). http://oaei.ontologymatching.org/ Accessed 08 May 2017

[9] Pomp, A., Lipp, J., Meisen, T.: Enabling the continuous evolution of ontologies for ontology-based data management. International Journal of Robotic Computing (2019)

[10] Berners-Lee, T.: Linked Data (2006). http://www.w3.org/DesignIssues/LinkedData.html Accessed 04 January 2017

[11] Hausenblas, M.: Utilising linked open data in applications. In: Proceedings of the International Conference on Web Intelligence, Mining and Semantics, p. 7 (2011). ACM

[12] Buneman, P., Fan, W., Siméon, J., Weinstein, S.: Constraints for semistructured data and xml. ACM Sigmod Record **30**(1), 47–54 (2001)

[13] Garcia-Gonzalez, H., Labra-Gayo, J.E.: Xmlschema2shex: Converting xml validation to rdf validation. Semantic Web **11**(2), 235–253 (2020)

[14] Johnson, T.: Indexing linked bibliographic data with json-ld, bibjson and elasticsearch. Code4lib Journal **19**, 1–11 (2013)

[15] Dubey, S., Patel, A., Jain, S.: Conversion between semantic data models: the story so far, and the road ahead. In: Web Semantics, pp. 23–30. Elsevier, Academic Press (2021)

[16] Hildebrand, M., Tourkogiorgis, I., Psarommatis, F., Arena, D., Kiritsis, D.: A method for converting current data to rdf in the era of industry 4.0. In: IFIP International Conference on Advances in Production Management Systems, pp. 307–314 (2019). Springer

[17] Deursen, D.V., Poppe, C., Martens, G., Mannens, E., d. Walle, R.V.: Xml

to rdf conversion: A generic approach. In: 2008 International Conference on Automated Solutions for Cross Media Content and Multi-Channel Distribution, pp. 138–144 (2008). https://doi.org/10.1109/AXMEDIS.2008.17

[18] Breitling, F.: A standard transformation from xml to rdf via xslt. Astronomische Nachrichten **330**(7), 755–760 (2009)

[19] Umbrich, J., Hogan, A., Polleres, A., Decker, S.: Improving the recall of live linked data querying through reasoning. In: Krötzsch M., S.U. (ed.) Web Reasoning and Rule Systems. RR 2012. Lecture Notes in Computer Science, pp. 188–204. Springer, Berlin, Heidelberg (2012)

[20] Guillet, H.J. Fabrice J.; Hamilton: Quality Measures in Data Mining vol. 43. Springer, Berlin (2007). https://doi.org/10.1007/978-3-540-44918-8

[21] Christen, P.: Data Matching: Concepts and Techniques for Record Linkage, Entity Resolution, and Duplicate Detection. Springer, Berlin, Heidelberg (2012)

[22] Aydar, M., Ayvaz, S.: An improved method of locality-sensitive hashing for scalable instance matching. Knowledge and Information Systems **58**(2), 275–294 (2019)

[23] Scharffe, F., Ferrara, A., Nikolov, A.: Data linking for the semantic web. Semantic Web: Ontology and Knowledge Base Enabled Tools, Services, and Applications **169**, 326 (2013)

[24] Nguyen, K., Ichise, R., Le, B.: Slint: a schema-independent linked data interlinking system. In: Proceedings of the 7th International Conference on Ontology Matching-Volume 946, pp. 1–12 (2012). CEUR-WS. org

[25] Do, H.-H., Melnik, S., Rahm, E.: Comparison of schema matching evaluations. In: Net. ObjectDays: International Conference on Object-Oriented and Internet-Based Technologies, Concepts, and Applications for a Networked World, pp. 221–237 (2002). Springer

[26] Euzenat, J., Shvaiko, P., *et al.*: Ontology Matching vol. 18. Springer, Berlin (2007)

[27] Gunaratna, K., Lalithsena, S., Sheth, A.: Alignment and dataset identification of linked data in semantic web. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery **4**(2), 139–151 (2014)

[28] Wang, C., Lu, J., Zhang, G.: Integration of ontology data through learning instance matching. In: IEEE/WIC/ACM International Conference on Web Intelligence, pp. 536–539 (2006). IEEE

[29] Jaro, M.A.: Advances in record-linkage methodology as applied to matching the 1985 census of tampa, florida. Journal of the American Statistical Association **84**(406), 414–420 (1989)

[30] Araujo, S., Hidders, J., de Vries, A.P., Schwabe, D.: Serimi: resource description similarity, rdf instance matching and interlinking. In: Proceedings of the 6th International Conference on Ontology Matching-Volume 814, pp. 246–247 (2011). CEUR-WS. org

[31] Nguyen, K., Ichise, R., Le, B.: Interlinking linked data sources using a domain-independent system. In: Joint International Semantic Technology Conference, pp. 113–128 (2012). Springer

[32] Nentwig, M., Hartung, M., Ngonga Ngomo, A.-C., Rahm, E.: A survey of current link discovery frameworks. Semantic Web **8**(3), 419–436 (2017)

[33] Zhang, Y., Jin, H., Pan, L., Li, J.: RiMOM results for OAEI 2016. In: Proceedings of the 11th International Workshop on Ontology Matching Co-located with the 15th International Semantic Web Conference (ISWC2016), pp. 210–216 (2016). CEUR-WS.org

[34] Li, Y., Li, J., Zhang, D., Tang, J.: Result of ontology alignment with rimom at oaei'06. In: Proceedings of the 1st International Conference on Ontology Matching-Volume 225, pp. 181–190 (2006). CEUR-WS. org

[35] Niu, X., Wang, H., Wu, G., Qi, G., Yu, Y.: Evaluating the stability and credibility of ontology matching methods. The Semantic Web: Research and Applications, 275–289 (2011)

[36] Cruz, I.F., Antonelli, F.P., Stroe, C.: Agreementmaker: efficient matching for large real-world schemas and ontologies. Proceedings of the VLDB Endowment **2**(2), 1586–1589 (2009)

[37] Dragisic, Z., Ivanova, V., Li, H., Lambrix, P.: Experiences from the anatomy track in the ontology alignment evaluation initiative. Journal of biomedical semantics **8**(1), 1–28 (2017)

[38] Khrouf, H., Troncy, R.: Eventmedia: A LOD dataset of events illustrated with media. Semantic Web **7**(2), 193–199 (2016)

[39] Zhang, M., Yuan, J., Gong, J., Yue, P.: An interlinking approach for linked geospatial data. ISPRS-International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences **1**(2), 283–287 (2013)

[40] Ngomo, A.-C.N., Auer, S.: Limes-a time-efficient approach for large-scale link discovery on the web of data. In: Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence, pp. 2312–2317

(2011)

[41] Aufaure, M.-A., Chiky, R., Curé, O., Khrouf, H., Kepeklian, G.: From business intelligence to semantic data stream management. Future Generation Computer Systems **63**, 100–107 (2016)

[42] Tempelmeier, N., Demidova, E.: Linking openstreetmap with knowledge graphs—link discovery for schema-agnostic volunteered geographic information. Future Generation Computer Systems **116**, 349–364 (2021)

[43] Achichi, M., Bellahsene, Z., Ellefi, M.B., Todorov, K.: Linking and disambiguating entities across heterogeneous rdf graphs. Journal of Web Semantics **55**, 108–121 (2019)

[44] Atencia, M., David, J., Euzenat, J.: Data interlinking through robust linkkey extraction. In: ECAI, pp. 15–20 (2014)

[45] Kettouch, M., Luca, C., Hobbs, M.: Schema matching for semi-structured and linked data. In: Semantic Computing (ICSC), 2017 IEEE 11th International Conference On, pp. 270–271 (2017)

[46] Kettouch, M.S., Luca, C., Hobbs, M., Dascalu, S.: Using semantic similarity for schema matching of semi-structured and linked data. In: Internet Technologies and Applications (ITA), 2017, pp. 128–133 (2017). IEEE

[47] Kettouch, M., Luca, C., Hobbs, M.: Semild: mediator-based framework for keyword search over semi-structured and linked data. Journal of Intelligent Information Systems, 1–25 (2018)

[48] Han, L., Kashyap, A.L., Finin, T., Mayfield, J., Weese, J.: Umbc_ebiquity-core: Semantic textual similarity systems. In: Proceedings of the Second Joint Conference on Lexical and Computational, pp. 44–52 (2013)

[49] Kettouch, M.S., Luca, C., Hobbs, M.: An interlinking approach based on domain recognition for linked data. In: Industrial Informatics (INDIN), 2015 IEEE 13th International Conference On, pp. 488–491 (2015). IEEE

[50] Zheng, Q., Shao, C., Li, J., Wang, Z., Hu, L.: Rimom2013 results for oaei 2013. In: Proceedings of the 8th International Conference on Ontology Matching-Volume 1111, pp. 161–168 (2013)

[51] Huber, J., Sztyler, T., Noessner, J., Meilicke, C.: Codi: combinatorial optimization for data integration-results for oaei 2011. In: Proceedings of the 6th International Conference on Ontology Matching, pp. 134–141 (2011)

[52] Castano, S., Ferrara, A., Montanelli, S., Lorusso, D.: Instance matching for ontology population. In: Proceedings of the Sixteenth Italian Symposium on Advanced Database Systems, pp. 121–132 (2008)

[53] Deb Nath, R.P., Seddiqui, H., Aono, M.: A novel automatic property weight generator for semantic data integration. In: 16th Int'l Conf. Computer and Information Technology, pp. 408–413 (2014)

[54] Kettouch, M.S.: A new approach for interlinking and integrating semi-structured and linked data. PhD thesis, Anglia Ruskin University (2017)

[55] Instance Matching at OAEI 2011 (IM@OAEI2011) (2011). http://oaei.ontologymatching.org/2011/instance/ Accessed 08 May 2020

[56] Verborgh, R., Hartig, O., De Meester, B., *et al.*: Querying datasets on the web with high availability. In: International Semantic Web Conference, pp. 180–196 (2014)

[57] Goutte, C., Gaussier, E.: A probabilistic interpretation of precision, recall and f-score, with implication for evaluation. In: European Conference on Information Retrieval, pp. 345–359 (2005)