Multidimensional Integration of RDF Datasets

Jam Jahanzeb Khan Behan[1,2], Oscar Romero[1], and Esteban Zimányi[2]

[1] Universitat Politècnica de Catalunya, Calle Jordi Girona, 1-3, 08034 Barcelona
{behan,oromero}@essi.upc.edu

[2] Université libre de Bruxelles, Avenue Franklin Roosevelt 50, 1050 Bruxelles
{jbehan,ezimanyi}@ulb.ac.be

Abstract. Data providers have been uploading RDF datasets on the web to aid researchers and analysts in finding insights. These datasets, made available by different data providers, contain common characteristics that enable their integration. However, since each provider has their own data dictionary, identifying common concepts is not trivial and we require costly and complex entity resolution and transformation rules to perform such integration. In this paper, we propose a novel method, that given a set of independent RDF datasets, provides a multidimensional interpretation of these datasets and integrates them based on a common multidimensional space (if any) identified. To do so, our method first identifies potential dimensional and factual data on the input datasets and performs entity resolution to merge common dimensional and factual concepts. As a result, we generate a common multidimensional space and identify each input dataset as a cuboid of the resulting lattice. With such output, we are able to exploit open data with OLAP operators in a richer fashion than dealing with them separately.

Keywords: Entity Resolution · Resource Description Framework (RDF) · Data Integration · On-Line Analytical Processing (OLAP) · Multidimensional Modeling.

1 Introduction

Data availability on the Web is ensured as users constantly upload data. Since multiple users can share the same entity, data duplication and unconnected related data grew on the Web. As a consequence, integration of web sources became a necessity and the Web of Linked Data was obtained. Linked Open Data (LOD) enables the sharing of information, structured querying formats, and facilitates access to data by means of Uniform Resource Identifiers (URIs). Yet, due to the heterogeneity of the Web of Linked Data, it is still problematic to develop Linked Data (LD) applications. Nowadays, we cannot assume that all URI aliases have been explicitly stated as links and therefore data integration is still an open issue. Nevertheless, the size of LOD has been increasing exponentially. A study released in April 2014 highlights that the LD cloud has grown to more than 1000 datasets from just 12 datasets cataloged in 2007 [15] having more than 500 million explicit links between them.

The Resource Description Framework (RDF) models information that describes LD in the form of RDF triples. Each triple contains a subject, a predicate, and an object defined by a URI. Data providers publish RDF datasets using a personalized dictionary, hence a single entity has multiple definitions. Yet, most RDF datasets have common characteristics indicating that they share common schemata information, that may enable a tighter integration. The aim of this project is to facilitate the user in querying similar RDF datasets on an integrated, multidimensional fashion (which we refer to as **Integrated Multidimensional Dataset** (IMD)). IMD is annotated as a cube with QB4OLAP vocabulary [5], that enables OLAP functionalities and resolves compatibility issues such as different resource names or granularity details. We claim that performing OLAP analysis on top of disparate RDF datasets allows a richer analysis than regular analysis on each independent or–manually glued together–dataset(s).

To showcase our approach, we use RDF datasets available at the US, the UK and the Eurostat (Linked) data portals as running examples. We focus on carbon emission data, as it is a major factor for air quality degradation. Additionally, we use QBOAirbase[3], a QB4OLAP-compliant dataset, as the basis of our model as it contains several factual data with varying dimensional granularity. We refer to the carbon datasets obtained from the US[4], the UK[5] and the Eurostat[6] data portals as C_{US}, C_{UK}, and C_{EU} respectively. The comparison of these datasets is not straightforward, however, taking a close look, one may identify them as cuboids of the same multidimensional space. For example, C_{US} and C_{UK} contain *location* details on a finer granularity (property/state level), whereas C_{EU} contains higher level data (country level). We perform certain operations to generate a single integrated dimension to enable data compatibility.

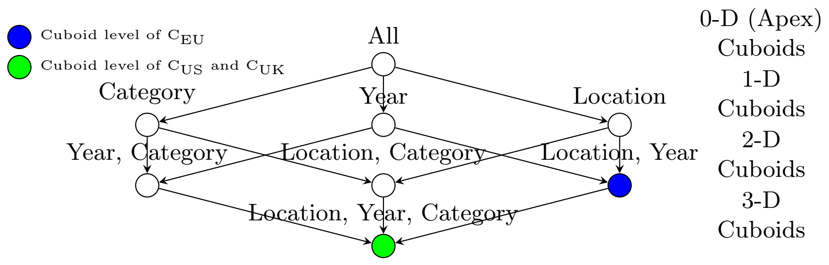
In this paper, we propose a framework to integrate and cross external RDF datasets into a unified multidimensional view. Our framework consists of four modules to: (i) identify potential dimensional and factual resources, (ii) independently perform entity resolution (ER) on dimensional and factual resources, (iii) and identify new dimensional and (iv) factual resources to be added in the current IMD (from new links identified in ER module (ii)). Our proposal is an incremental approach that builds the IMD schema by unifying RDF datasets at hand. As a result, based on the granularity level of the measure in the input RDF datasets, each dataset represents a cuboid of the IMD lattice. We thus enable the capability, using OLAP processing, to answer queries at a certain granularity, by means of rolling-up appropriate RDF datasets that are at a lower granularity level. Thus, the input RDF datasets play the role of materialized views of the IMD lattice and, combining these views, we provide a "bigger picture" of the data. However, since our approach is purely data-driven, the resulting lattice might be highly sparse and there might be no input RDF datasets to compute

[3] The European Air Quality RDF Database: http://qweb.cs.aau.dk/qboairbase/
[4] https://catalog.data.gov/dataset/2015-greenhouse-gas-report-data
[5] https://opendata.camden.gov.uk/resource/4txj-pb2i
[6] http://estatwrap.ontologycentral.com/page/t2020_rd300

the granularity for certain cuboids. Fig. 1 shows the structure of the IMD lattice obtained after integrating the running example datasets.

The rest of the paper is as follows: Sec. 2 contains the description of our framework and the steps we take to integrate datasets. Sec. 3 provides the technical details for implementing our framework. Sec. 4 illustrates the experiments we performed and the results obtained. Sec. 5 explains other ER frameworks for LD and we finally present our conclusion in Sec. 6.

Fig. 1: Lattice of the IMD created for C_{EU}, C_{US}, and C_{UK}



2 Framework

In this section, we present an overview of our framework and how we obtain the IMD by following a purely data-driven approach. Our process is conducted pair-wise, where one input is the new RDF dataset we want to integrate with IMD and the other input is the IMD itself. Fig. 2 depicts the data flow and the components of the framework, note that in the first iteration the flow is reduced to integrating two datasets and generating IMD for the first time.

In Sec. 2.1 we propose a supervised learning approach to analyze the RDF dataset D_I and identify resources as dimensions or measures. We refer to the resulting RDF annotated with MD concepts as D_A. We perform ER between same label resources in D_A, thus, obtaining the graph D_R (further explained in Sec. 2.2). In Sec. 2.3, and after performing ER, we enrich IMD and its schema with potentially new dimensions, hierarchies, and dimensional values. Finally, as elaborated in Sec. 2.4, we align factual data by considering potential unit misalignments.

We define the resulting IMD dataset using the QB[7] and QB4OLAP [5] vocabularies. An excerpt defining in QB4OLAP the basic structure of IMD for our running example follows:

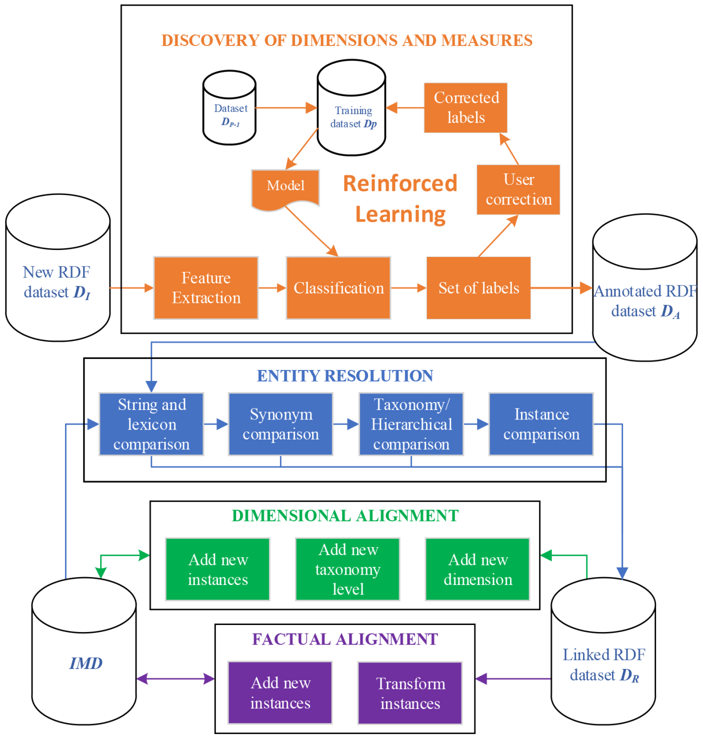
```
1  schema:IMD rdf:type qb:DataStructureDefinition;
2   dct:conformsTo <http://purl.org/qb4olap/cubes_v1.3>;
3  data:IMD rdf:type qb:DataSet;
4   qb:structure schema:IMD;
5   dct:title "Integrated Multidimensional Dataset for carbon emissions"@en.
6  qb:component [qb:measure schema:C ; qb4o:aggregateFunction qb4o:avg];
7  schema:C rdf:type qb:MeasureProperty;
8   rdfs:label "Carbon emission quantity"@en;
9   rdfs:range xsd:float.
```

At line 1, we define the schema of the IMD and state that it follows a Data Cube structure. Line 2 states that this schema has an established standard to

[7] https://www.w3.org/TR/vocab-data-cube/

which the described resources can conform to. At line 3, we state that the data in the IMD schema is represented as a collection of observations, that can be organized into various slices, and thereby conforming to some common dimensional structure. Line 4 indicates that the structure of the IMD schema conforms to the dataset defined at line 3. Finally, in line 5, we give a name to our schema and state that the name is in English. At line 6 we define a new measure for "carbon emissions" and the schema of this new measure is defined at lines 7–9. We explain how this measure is added in Sec. 2.4.

Fig. 2: Framework and modules



2.1 Discovery of Dimensions and Measures

Performing ER at a large scale is rather costly [17]. Thus, we propose a first step to automatically identify potential multidimensional concepts out of the input RDF datasets. Using this technique, the subsequent ER steps will be performed between elements labeled as of the same MD class (i.e., between dimensions or between measures). RDF metadata information, which represents a fair amount of triples, is removed from the process as they do not have a MD meaning.

We automatically label resources in the input RDF dataset as dimensions, measures or metadata using a Decision Tree (DT). This DT is trained on a user labeled dataset, where each resource contains data features and a data label. We use a DT over other ML techniques since a DT provides "rules" used for *explanatory analysis* of the resource labeling. Since the size of the search space is well-limited, as discussed in Sec. 4, using a DT shows good classification results. Then, we give the output to the user to validate the annotations and correct them if needed. Using a reinforcement method, the DT is continuously retrained by means of the user provided corrections. This approach follows a pay-as-you-go model that learns as new datasets come. Our tests show that few iterations are needed to obtain good recall and accuracy. We perform the following steps each time a new dataset is to be integrated in IMD:

Feature Extraction For each resource in D_I, the following features are generated by analyzing the datasets schema, and dictionaries:

Unique Values: The ratio of unique values based on the total occurrences.

Data Types: Such as float, integer, string, boolean, categorical, date, geolocation, a resource (i.e., a URI) or description (containing metadata information).

URI Prefix and Resource Name: The URI is parsed to obtain these features. In our running example, <ds:location> (in C_{UK}), is parsed as <ds> and <location>.

URI Resource Name Length: The total number of characters in a URI.

Additive Property: Identifies numerical type resources as additive or non-additive.

Table 1: Sample of labelled URIs

Label	Dataset	URI
Measure	C_{EU}	<sdmx-measure:obsValue>
Measure	C_{UK}	<ds:carbon_emissions_kgco2e>
Measure	C_{US}	<ds:total_emissions_mt_co2e>
Dimension	C_{EU}	<geo rdf:resource="">
Dimension	C_{UK}	<ds:location>
Dimension	C_{US}	<ds:location_latitude_longitude>

Data Classification Each resource in D_I is enriched with the above-mentioned features and passed to the Classification module. This module contains the model that is trained with the union of all previous datasets (D_P), with user corrected labels. Enriched resources of D_I are given as input to the trained model that labels its new resources based on the rules identified in previous iterations. Table 1 shows some labeled resources obtained in our running example. The next steps will only be performed between resources with the same label. Note that resources labeled as metadata are excluded from the next steps.

Label Validation The Classification module provides the new annotated resources from D_I as a input to the User Validation module. The user corrects the labels of D_I, which is then merged with dataset D_{P-1} obtaining dataset D_P. Using reinforcement learning, the dataset D_P is re-fed to the Classification module to retrain the DT for the next iteration (i.e., when starting the process for a new RDF dataset to be integrated with IMD). In our running example, the DT is initially trained with the (manually labeled) dataset QBOAirbase. We then used this DT to label the dataset C_{UK}. If so, it would not assign a label to <ds:period> and <ds:location>, as these resources do not fit any of the current rules. We corrected these labels, as these resources contain dimensional data, and the combination of the QBOAirbase dataset and the corrected C_{UK} dataset will conform the training dataset for the next iteration (i.e., D_P). We retrain the DT with the new D_P training dataset and then use it to label the dataset C_{US}. Table 2 shows the accuracy and recall of the model achieved to label a newcoming dataset in each iteration as new (correctly labeled) resources of the previous iteration are added to the training dataset (D_P). In our running example, our experiments show that the model accuracy and recall improves drastically with few new input resources per iteration.

As one may infer, the order in which the datasets are ingested in our system may impact the accuracy and recall of each iteration. We discuss more about our approach for automatic dimension and measure elicitation in Section 4.1.

Table 2: Model accuracy per iteration

Iteration	Model Accuracy	Recall	Dataset Labeled	New Resources Added
First	22.2%	15.8%	C_{UK}	-
Second	70%	83.3%	C_{US}	30
Third	81.3%	100%	C_{EU}	27

2.2 Entity Resolution

Entity resolution plays a significant role in our research. Once the resources are labeled we obtain an annotated dataset D_A. We perform ER on the dimensional and factual resources of D_A separately to identify links, with IMD, between the same resources. This step is required due to the usage of different vocabularies in the input RDF datasets and IMD.

ER is done by obtaining lexicons from resource names of D_A and indexing them for linkage purposes (referred to as Rule 1). Additionally, we enrich the resources of D_A with their synonym-map and hierarchy-map (that are obtained using external dictionaries) and index them to perform ER (referred to as Rule 2 and Rule 3). We introduce a final rule to consider equivalence of *two dimensional resources* by means of instances. We apply Rules 1–3 on instance level to identify if two instances of separate concepts are the same (referred to as Rule 4). A resource in D_A can be linked with a resource in IMD either through equivalence (by having the same name lexicons or synonyms) or subsumption (through taxonomies). The formal definition of rules are as follows:

Rule 1 Given two resources, d_1 and d_2, d_1 is the same resource as d_2 if there is an equivalence when considering the lemmas in the names of both resources:

$\{d_1,d_2\} \mapsto \{d_2\}$ *iff* $d_1 \equiv d_2$ *where* $d_1 \in D_A \wedge d_2 \in IMD$

Rule 2 Given two resources, d_1 and d_2, d_1 is the same resource as d_2 if there is an equivalence (i.e., same lemma) when considering the synonym map (S_d) of both resources:

$\{d_1,d_2\} \mapsto \{d_2\}$ *iff* $d_3 \equiv d_4$ *where* $d_1 \in D_A \wedge d_2 \in IMD \wedge d_3 \in S_{d1} \wedge d_4 \in S_{d2}$

Rule 3 Given two resources, d_1 and d_2, d_2 subsumes or is subsumed by d_1 if d_1 and d_2 (or their synonyms) participate in the same hierarchical-map (H_d) either directly or through their synonym map:

$d_3 \equiv d_4 \wedge d_4 \sqsubseteq d_5$ *where* $(d_3 \in (d_1 \cup S_{d1}) \wedge d_4 \in (d_2 \cup S_{d2}) \wedge d_5 \in H_{d2})$ $\vee$ $(d_3 \in (d_2 \cup S_{d2}) \wedge d_4 \in (d_1 \cup S_{d1}) \wedge d_5 \in H_{d1})$

Rule 4 Given two resources, d_1 and d_2, there is an equivalence if I_{d1}, the instance space of d_1, has an intersection with I_{d2}, the instance space of d_2, that is greater than an input parameter θ, which is the required level of equal instances in both resources:

$d_1 \equiv d_2$ iff $I_{d1} \cap I_{d2} > \theta$ where I_{d1} = instances of d_1 $\wedge$ I_{d2} = instances of d_2 $\wedge$ $\theta \in \mathbb{R}$

If d_1 maps to d_2 by any of the rules given above, the other rules are not checked. As result, we link both concepts either by an equivalence relationships (Rules 1, 2 or 4) or by subsumption (Rule 3).

We now present some examples of each rule based on our running example. For example, there is an equivalence between dimensional URIs <ds:location> (in C_{UK}) and <ds:location_latitude_longitude> (in C_{US}), as they provide information for the same lemma **location** (Rule 1). These URIs also contain instances on the same dimensional level. However, <geo rdf:resource=""> (in C_{EU}) is linked with the location URIs via subsumption (Rule 3) at the instance level. When we apply Rule 4, it shows that the instances of <geo rdf:resource=""> subsumes the instances of <ds:location_latitude_longitude> and <ds:location>, as C_{UK} and C_{US} provide location information at a finer granulariy level than C_{EU} (at country level). Additionally, <ds:period> (in C_{UK}) is linked with <dcterms:date> (in C_{EU}), as **period** and **date** share the same synonym-map (Rule 2). The factual resources in the datasets, <ds:carbon_emissions_kgco2e> (in C_{UK}) and <ds:total_emissions_mt_co2e> (in C_{US}), are linked based on the same lemmas **emission** and **co2** (Rule 1).

Before finalizing this step, and provided that ER methods can hardly achieve a 100% recall, we let the user to manually fix any missing link we could not detect.

2.3 Dimensional Alignment

The resultant dataset D_R is linked to IMD by means of the relationships identified in the previous step: either through equivalence or through subsumption relations. This component identifies updates to be performed in the IMD dimensional data (both schema and instances) according to the links identified. As illustrated in Fig. 2, when the dataset D_R is given as input to the Dimensional Alignment component, the dimensional space of IMD can be updated by (i) adding new dimensions, (ii) adding new hierarchical levels to existing dimensions (schema level) or (iii) adding new instances to an existing dimension level. We formalize IMD with QB4OLAP notation and to add a new dimension, or dimensional level to IMD, we add the corresponding QB4OLAP triple.

Add Dimension If a dimension in D_R is not linked to any dimensional resource from IMD (i.e., Rules 1–4 from Sec. 2.2 are not satisfied) it means that there is no correspondence in IMD for this concept. Thus, we need to create a new dimension in the IMD multidimensional schema. In our running example, we add a new dimension for location in IMD and define it using QB4OLAP annotation:

```
10  schema:locationDim rdf:type qb:DimensionProperty;
11   rdfs:label "Location class dimension"@en;
12   qb4o:hasHierarchy schema:locationHier.
13  schema:locationHier rdf:type qb4o:Hierarchy;
14   rdfs:label "Location Hierarchy"@en;
15   qb4o:inDimension schema:locationDim;
16   qb4o:hasLevel schema:propertyLevel, schema:stateLevel.
17  _:station_hl1 rdf:type qb4o:HierarchyStep;
18   qb4o:inHierarchy schema:locationHier;
19   qb4o:childLevel schema:propertyLevel;
20   qb4o:parentLevel schema:stateLevel;
21   qb4o:pcCardinality qb4o:ManyToOne;
22   qb4o:rollup schema:inState.
23  schema:inState rdf:type qb4o:RollupProperty.
```

At lines 10–12 we define the schema for the location dimension and add hierarchies at lines 13–16. The hierarchical level for “State” is defined at lines 17–22, and the roll-up property from “Property” to “State” is defined at line 23.

With regard to our running example, this excerpt of QB4OLAP would be added when the location dimension is introduced in IMD for first time (i.e., with C_{UK} or C_{US})

Add Dimensional Level If Rule 3 identifies a subsumption relationship between dimensional concepts in D_R and IMD, we add a new dimensional level to the proper dimension in IMD. In our running example, as stated in Sec. 2.2, the URI <geo rdf:resource=""> (in C_{EU}) provides details on a higher taxonomy (country level). Therefore, we add a new lattice level to the schema of IMD. The following QB4OLAP-compliant definition states how we add the dimensional level for “Country” and update the schema of IMD:

```
24  schema:locationHier rdf:type qb4o:Hierarchy;
25   qb4o:hasLevel schema:propertyLevel, schema:stateLevel,
26    schema:countryLevel.
27  _:station_hl2 rdf:type qb4o:HierarchyStep;
28   qb4o:inHierarchy schema:locationHier;
29   qb4o:childLevel schema:stateLevel;
30   qb4o:parentLevel schema:countryLevel;
31   qb4o:pcCardinality qb4o:ManyToOne;
32   qb4o:rollup schema:inCountry.
33  schema:inCountry rdf:type qb4o:LevelProperty;
34   rdfs:label "type Level"@en;
35   qb4o:hasAttribute property:country.
36  property:country rdf:type qb4o:LevelAttribute;
37   rdfs:label "country Name"@en;
38   rdfs:range xsd:string.
39  schema:inCountry rdf:type qb4o:RollupProperty.
```

Notice that we update line 16 with lines 25–26 to add a new dimensional level. We define the new hierarchy in lines 27–38 and finally add the roll-up property from “State” to “Country” in line 39.

Add Instances We add instances to IMD in two cases, either by identifying equivalence relationship between dimensional resources (defined by Rules 1, 2 and 4 in Sec. 2.2) or when a new dimensional level is added (defined by Rule 3 in Sec. 2.2). In the first case, we need to apply ER again, but this time between the instances of D_R and IMD. This step is required for identifying equivalences (Rules 1 and 2) between the instances of D_R and IMD, to avoid duplication of instances in IMD. In the second case, however, we simply add all the instance resources related to that dimensional resource on the newly created level.

For this second step we rely on previous work [6]. For example, in our running example, when a new dimensional level from C_{EU} is identified (country level), we add instances from URI <geo rdf:resource="">. The instances are added to populate "Country Name" property as specified in lines 36–38 of the QB4OLAP-compliant definition when a new dimensional level is identified. Each cuboid in the lattice is required to have the unit of measure associated with it (further explained in Sec. 2.4). Hence, when adding the instances for "Country" dimension, we also add the unit of measure for the dataset C_{EU}.

2.4 Factual Alignment

The last phase of our framework is to populate the IMD with additional factual instances. Using Rules 1, 2 and 4 in Sec. 2.2 we obtain resources that contain factual data and we transform instances of dataset D_R to guarantee uniformity. Additionally, for Factual Alignment component, we require the unit of measure. The unit can be identified by either being explicitly provided in the dataset or we ask the user to provide this information. We require these units so that we can transform the factual data in D_R to that of IMD, using conversion functions. The conversion functions are stored in our framework for each unit of measure.

The factual instances of D_I can either have an equivalent measure from IMD, or do not have any counterpart in IMD yet. As IMD is QB4OLAP-compliant, the factual instances are instantiated as cuboids at a certain granularity level. As can be seen in Fig. 1, the datasets sit at different cuboid levels based on the granularity detail of the factual instances provided in those datasets. Therefore, if there is an equivalent measure from IMD in D_I then we compare the units, apply the transformation on instances of D_I to make them compatible, and add the transformed factual instances to IMD. If there is no equivalent measure from IMD in D_I, we import the factual instances of D_I as they are and annotate the unit of measure at that cuboid level. Note that we annotate the unit and transformation between units as this information is elicited to automate these tasks in future steps.

In our running example, the URIs that contain data for carbon emission are <ds:carbon_emissions_kgco2e> (in C_{UK}), <ds:total_emissions_mt_co2e> (in C_{US}), and <sdmx-measure:obsValue> (in C_{EU}). Each dataset contains emission data in a different unit of measure: kg for C_{UK}, mt for C_{US}, and tonnes for C_{EU}. C_{US} and C_{UK} contain factual instances on the same cuboid (3-D) level. Yet, they have different units of factual measures. We select the unit of measure from C_{UK} (kg) as the base unit at that cuboid level and transform factual instances of C_{US} from mt to kg before combining them. C_{EU} contains factual instances on a higher cuboid (2-D) level and therefore we can import the new factual instances to that cuboid level but we should also transform their values from tonnes to kg (to guarantee correct Roll-up between both cuboids). Additionally, the unit of measure for carbon emission are available in C_{UK} and C_{US}, but C_{EU} does not implicitly contain the unit of measure. Therefore, we extract this information from the Eurostat website and annotate it. We add the new unit of measure in our schema:C with use of rdfs:comment since each measure can have it's own unit. For future work, we plan to incorporate more advanced state-of-the-art techniques to integrate measures into the IMD through either formal representation of the measures dependencies by means of specific predicates [13, 4] or through the semantic representation of the calculation formulae of such measures [3].

3 Implementation

In this section, we first discuss the prototype created to implement our method and then discuss a set of experiments conducted to show its feasibility. We build a Java project to extract entities from input graphs using Apache Jena[8] and add features that are computed using the OpenLink Virtuoso[9] platform. Then, for building the DT model to label resources as dimensions, metadata or measures we opt for the KNIME Analytical Platform[10] and perform ER operations on the annotated datasets using LogMap [8] and Instance-based Unified Taxonomy [18] to finally merge the datasets. We use LogMap over other ontology alignments tools as it has low time complexity and high (successful) linkage discovery [1].

Next subsections elaborate on the implementation of the framework: Sec. 3.1 describes how we built our classification model. Sec. 3.2 describes the techniques used for performing ER operations and finally, Sec. 3.3 elaborates on how we extend the IMD schema. We use real datasets ($\mathrm{C_{US}}$, $\mathrm{C_{UK}}$, and $\mathrm{C_{EU}}$) to showcase the feasibility of our method.

3.1 Model Building in KNIME

In KNIME, we use the *Excel Reader nodes* that read the features specified in Sec. 2.1 (Feature Extraction). We train the *Decision Tree Learning node* using the training input data and we test the *Decision Tree Predictor node* using the testing data. In Sec. 2.1 we specify that in each iteration we increment the training datasets with new (user validated) label resources. This incremented data is then used to retrain the DT model. In Sec. 4.1 we perform Leave One Out (LOO) cross-validation to show the accuracy of our approach. That is, how likely is the trained model able to identify measures and dimensions from previous examples.

3.2 Entity Resolution for Dimensions and Measures

Our focus is to integrate RDF datasets in a common multidimensional space and build a lattice, therefore, we use state-of-the-art techniques to perform ER. We use LogMap for ER operations defined using Rules 1–3 (see Sec. 2.2) as LogMap uses WordNet[11] to obtain lexicons from resource names, indexes resources for linkage purposes with their synonym-map, and builds hierarchies for each class.

[8] https://jena.apache.org/
[9] https://virtuoso.openlinksw.com/
[10] https://www.knime.com/
[11] https://wordnet.princeton.edu/

Additionally, for ER Rule 3, we use the mapping technique (on instance level) proposed in [6] along with matching of resources using rdfs:subClassOf property.

Furthermore, we use Instance-based Unified Taxonomy [18] for ER Rule 4 stated in Sec. 2.2, as it builds a unified taxonomy of classes and links instances through owl:sameAs property and exact matching. Sec. 4.2 states the performance results obtained when applying ER operations on our datasets.

3.3 Schema Alignment

To add dimensions, dimensional hierarchies, and dimensional instances: we use Apache Jena to extract the annotated data from the input dataset, add QB4OLAP-compliant triples at new granularity level (if required), remove duplicate instances, and add it in the IMD dataset. To add factual instances, we extract the units of measure in the input dataset and compare it to the unit of measure of IMD, if the unit of measures differ, conduct transformations on the resource values. We also ask the user to provide conversion function, using the Java interface, where the unit of measure is not automatically convertible. We extract the dimensional hierarchy of measures from the input graph and compare them to the IMD dataset to either remove duplicate instances or to add a QB4OLAP-compliant triple at newly identified granularity level. Lastly, we add new instances to the IMD dataset.

4 Experiments and Results

This section discusses about the feasibility of our framework defined in Sec. 2 and implemented using the techniques provided in Sec. 3.

4.1 Model Testing in KNIME

Once we have (correctly) labeled all the resources in our running example, we use it as ground truth to perform LOO cross-validation to show the accuracy of our approach by picking a resource from the dataset, training the model on rest of the resources and check how well the model labels the one resource not used in training. This exercise is repeated for each resource, and the goal is to test the model's ability to see how our approach behaves with a resource that it has not seen before. We have 230 resources in total and therefore the LOO cross-validation is performed 230 times (once for each resource). After all 230 iterations are conducted, the model yielded an average error rate of 7.39% with a 6.84% variance between these error rates. Most of the errors produced were due to the resources not satisfying any of the rules obtained and hence the model labeled them as "?". Additionally, to check the validity of the model on a dataset from a different domain, we tested the DT (trained on *carbon* datasets) on a dataset from the *crime* domain[12]. This dataset contained 40 resources, of which the model correctly labeled 29 resources (i.e. obtained an accuracy of 74.36%). These experiments show that the domain is irrelevant when deciding

[12] https://data.cityofchicago.org/Public-Safety/Crimes-2001-to-present-Dashboard/5cd6-ry5g

the multidimensional role of a resource. In the future, we plan to validate this hypothesis with more thorough experiments.

Also, as one may expect, the order in which we take the resources has a big impact on the training. We will investigate how to make our method more stable in the future. Nevertheless, the big advantage of building a DT is that we obtain rules to label each resource. Table 3 contains the most relevant rules obtained from our tests. Relevantly, after training our model with different orders, we eventually get very similar rules. Thus, the order has an impact during the reinforced learning stage.

Indeed, after identifying the rules in this experiment, we notice that these rules are aligned with the main rules provided by previous research works in the literature (e.g., [14, 2] present thorough surveys). However, most of these approaches manually identify dimensions and measures, while our approach embeds a semi-automatic approach.

Table 3: Interesting rules identified for each label

Label	Rule
Measure	Resources that have *additive* feature marked as "Yes"
Metadata	Resources that contain W3C defined prefix in *URI Prefix*, such as foaf, owl, rdf, and rdfs
Dimension	Resources that have *Data Type* as "GeoLocation" or have ">99%" *Unique Values* and *Data Type* as "String"

4.2 Entity Resolution

We label the resources either as dimensions or measures to facilitate ER in our framework. The labeling of resources indicates that instead of performing an NxM comparison, only compare the resources that are equally labeled either as "dimension" or "measure". For example, when we compare $\mathrm{C_{UK}}$ with $\mathrm{C_{US}}$ with a generic ER framework, not considering dimensions or measures (i.e., labels), there would be 810 comparisons to be done. But, when we compare them with labels, the comparison drops to 139. Table 4 states the total number of resources in each dataset, and how many of them are dimensional and factual resources. Table 5 provides the number of comparisons made for matching these datasets, with and without labeling the resources. We observe that using labels aids in comparison as most of the resources in RDF datasets are metadata that only provide additional information. Also, when there are links identified in the previous ER iteration, the number of resources to be compared decrease in the next iteration. After labeling and comparing all three datasets ($\mathrm{C_{UK}}$, $\mathrm{C_{US}}$, and $\mathrm{C_{EU}}$), as stated in Table 5, we reduce the number of comparisons by 88% and the runtime by 81%.

As defined in Sec. 2.2, whenever a resource d_1 in D_A maps onto a resource d_2 in IMD, we always select the URI of d_2 to define the newly identified resource. To avoid re-computation of synonym-maps and hierarchical-maps every time a new resource needs to be matched, we store them in IMD.

Missing links The framework missed some links between resources that had to be added manually. For example, <sdmx-measure:obsValue> in $\mathrm{C_{EU}}$ contains carbon emission, yet it was not linked with either <ds:carbon_emissions_kgco2e> in C_{UK} or with <ds:total_emissions_mt_co2e> in C_{US}. Similarly, <ds:sub_sector> from C_{US} and <ds:category> from C_{UK} contain the type of areas (e.g. primary schools area, chemical production area, glass production area, etc.) where the carbon emission value is collected, yet these two resources were not linked. Based on this the recall of our approach, for identifying linkage, is 71.4%.

Table 4: Total & labeled resources

Dataset	Total	Dimension	Measures
C_{UK}	30	16	3
C_{US}	27	7	9
C_{EU}	16	2	1

Table 5: ER with & without labels

Using labels	Comparisons	Run-time (s)
Yes	201	31
No	1658	165

5 Related Work

In this section we mention some frameworks that provide entity resolution component for integration of RDF datasets.

The user can obtain a clean and consistent local target vocabulary using [16] while tracking data provenance. It includes an identity resolution component for discoverying URI aliases in data and maps them to single target URI based on user-provided matching heuristics using SILK [7].

The authors of [12] provide a mapping technique that achieves integration automatically using SPARQL CONSTRUCT mappings. The mappings are translated to the new consolidated, well specified and homogenous target ontology model, thereby, facilitating data integration from different academic RDF datasets by providing a bridge between heterogeneities of RDF datasets.

In [11], the authors define a multidimensional model based on the QB vocabulary and present OLAP algebra to SPARQL mappings. They use graph-based RDF data model along with QB vocabulary for querying and storing of Data cubes. They also define common OLAP operations on the Data cubes by re-using QB. The drawback in their approach is that it does not allow optimized OLAP queries to RDF and the entire ETL process needs to be repeated if new statistics are defined.

OLAP4LD [10] provides a platform for developers of applications using Linked Data sources reusing the QB vocabulary and explore the Eurostat data via Linked Data Cube Explorers. The authors focus on translating analytical operations to queries over LD sources and pre-processing to integrate heterogeneously modeled data.

A Federated Datawarehouse (F-DW) [9] is the integration of heterogeneous business intelligence systems set to provide analytical capabilities across the different function of an organization. Reference data is made common across various data warehouses for data consistency and integrity, and identical data value for a confirmed fact will be ensured. The prime design objective of a F-DW is to achieve a "single version of truth" and the authors stress that there should be defined, documented, and integrated business rules used across the component data warehouses in the whole architecture.

Our approach is different in the sense that our inputs are datasets and not multidimensional (MD) schemas. We aim towards a generic approach that generates a MD interpretation of RDF datasets. RDF datasets can embed a MD interpretation by using QB4OLAP but, unfortunately, publishers (such as Eurostat) are not yet publishing native MD datasets. Therefore, we first identify and assign a MD interpretation to online RDF datasets. Then, we perform ER but over the identified MD concepts of the same class. This way, we drastically reduce the problem complexity (typical ER tools are quadratic on the dataset size). In our case, many triples are disregarded (RDF metadata without MD meaning and not useful for data analysis) and create two big groups of concepts: facts and dimensions. In our tests, 48% of the resources were disregarded for data analysis, 34% were identified as dimensions and 18% as facts. As a consequence, we simplified the complexity of the entity resolution process considerably.

6 Conclusion

In this paper, we present a technique to semi-automatically combine RDF datasets into the same multidimensional space and identify each dataset as a cuboid of its lattice. Unlike other frameworks, we perform integration based on a multidimensional interpretation of arbitrary datasets. We showcase our method choosing RDF data as a use case, but the proposed methods could be generalized for other data models. Specifically, we have defined a set of steps to generate a single data cube out of independent, yet overlapping, RDF datasets. We use QB4OLAP to annotate the integrated data and define, for each input dataset, a cuboid at the right aggregation level of the IMD. As a consequence, the schema of the input RDF datasets is integrated, and we can use state-of-the-art tools to analyze QB4OLAP data to perform OLAP operations on the defined lattice. This novel method is a significative step towards automating effective multidimensional integration of related RDF datasets that were created independently. Indeed, our approach could be generalized to other data formats. Importantly, providing a MD interpretation of the datasets considerably reduce the complexity of ER when integrating the dataset schemas.

For our next work, we will propose a mechanism to query the source RDF datasets (example C_{UK}, C_{US}, and C_{EU} datasets used in this paper) via federated SPARQL queries and therefore avoiding to materialize the IMD.

7 Acknowledgements

This research has been funded by the European Commission through the Erasmus Mundus Joint Doctorate Information Technologies for Business Intelligence-Doctoral College (IT4BI-DC) program.

References

1. Achichi, M., Cheatham, M., Dragisic, Z., Euzenat, J., Faria, D., Ferrara, A., Flouris, G., Fundulaki, I., Harrow, I., Ivanova, V., et al.: Results of the Ontology

Alignment Evaluation Initiative 2017. In: Proceesings of the 12th International Workshop on Ontology Matching co-located with the 16th International Semantic Web Conference. vol. 2032, pp. 61–113. CEUR-WS (Oct 2017)
2. Cravero, A., Sepúlveda, S.: Multidimensional Design Paradigms for Data Warehouses: A Systematic Mapping Study. Journal of Software Engineering and Applications **7**(1), 53–61 (2014)
3. Diamantini, C., Potena, D., Storti, E.: Multidimensional query reformulation with measure decomposition. Information Systems **78**, 23–39 (2018)
4. Estrada-Torres, B., Richetti, P.H.P., del-Río-Ortega, A., Baião, F.A., Resinas, M., Santoro, F.M., Ruiz-Cortés, A.: Measuring Performance in Knowledge-intensive Processes. ACM Transactions on Internet Technology **19**(1), 15:1–15:26 (2019)
5. Etcheverry, L., Vaisman, A.A.: QB4OLAP: A New Vocabulary for OLAP Cubes on the Semantic Web. In: Proceedings of the 3rd International Conference on Consuming Linked Data. vol. 905, pp. 27–38. CEUR-WS.org (Nov 2012)
6. Gallinucci, E., Golfarelli, M., Rizzi, S., Abelló, A., Romero, O.: Interactive Multidimensional Modeling of Linked Data for Exploratory OLAP. Information Systems **77**, 86–104 (2018)
7. Isele, R., Jentzsch, A., Bizer, C.: Silk Server - Adding Missing Links While Consuming Linked Data. In: Proceedings of the 1st International Conference on Consuming Linked Data. vol. 665, pp. 85–96. CEUR-WS.org (Nov 2010)
8. Jiménez-Ruiz, E., Grau, B.C.: LogMap: Logic-Based and Scalable Ontology Matching. In: Proceedings of the 10th International Semantic Web Conference. pp. 273–288. Springer (Oct 2011)
9. Jindal, R., Acharya, A.: Federated Data Warehouse Architecture. Wipro Technologies White Paper (2004)
10. Kämpgen, B., Harth, A.: OLAP4LD–A Framework for Building Analysis Applications over Governmental Statistics. In: Proceedings of the 11th European Semantic Web Conference. pp. 389–394. Springer (May 2014)
11. Kämpgen, B., O'Riain, S., Harth, A.: Interacting with Statistical Linked Data via OLAP Operations. In: Proceedings of the 9th Extended Semantic Web Conference. pp. 87–101. Springer (May 2012)
12. Moaawad, M.R., Mokhtar, H.M.O., Al Feel, H.T.: On-The-Fly Academic Linked Data Integration. In: Proceedings of the International Conference on Compute and Data Analysis. pp. 114–122. ACM (May 2017)
13. Popova, V., Sharpanskykh, A.: Formal modelling of organisational goals based on performance indicators. Data & Knowledge Engineering **70**(4), 335–364 (2011)
14. Romero, O., Abelló, A.: A Survey of Multidimensional Modeling Methodologies. International Journal of Data Warehousing and Mining **5**(2), 1–23 (2009)
15. Schmachtenberg, M., Bizer, C., Paulheim, H.: Adoption of the Linked Data Best Practices in Different Topical Domains. In: Proceedings of the 13th International Conference on Semantic Web Conference, Part I. pp. 245–260. Springer (Oct 2014)
16. Schultz, A., Matteini, A., Isele, R., Bizer, C., Becker, C.: LDIF - Linked Data Integration Framework. In: Proceedings of the 2nd International Conference on Consuming Linked Data. vol. 782, pp. 125–130. CEUR-WS.org (Oct 2011)
17. Suchanek, F.M., Abiteboul, S., Senellart, P.: Paris: Probabilistic alignment of relations, instances, and schema. Proceedings of the VLDB Endowment **5**(3), 157–168 (2011)
18. Zong, N.: Instance-based Hierarchical Schema Alignment in Linked Data. Ph.D. thesis, Seoul National University Graduate School, Seoul, South Korea (2015)