Extracting core knowledge from Linked Data

Valentina Presutti[12], Lora Aroyo[3], Alessandro Adamou[12], Balthasar Schopman[3], Aldo Gangemi[2], and Guus Schreiber[3]

[1] Alma Mater Studiorum Università di Bologna, Italy
[2] ISTC, National Research Council, Italy
[3] Vrije Universiteit Amsterdam, The Netherlands

Abstract. Recent research has shown the Linked Data cloud to be a potentially ideal basis for improving user experience when interacting with Web content across different applications and domains. Using the explicit knowledge of datasets, however, is neither sufficient nor straightforward. Dataset knowledge is often not uniformly organized, thus it is generally unknown how to query for it. To deal with these issues, we propose a dataset analysis approach based on knowledge patterns, and show how the recognition of patterns can support querying datasets even if their vocabularies are previously unknown. Finally, we discuss results from experimenting on three multimedia-related datasets.

1 Introduction

The constant expansion trend of Linked Data (LD) is broadening the potential exploitation range of their datasets for improving search through related data. Current research [6, 1] and established Web search firms like Google and Powerset show the benefits of using explicit semantics and LD to refine search results. However, using efficiently the explicit knowledge of each dataset can be awkward and ineffective. Datasets typically cover diverse domains, do not follow a unified way of organizing the knowledge, differ in size, granularity and descriptiveness. To avoid burdensome, dataset-specific querying schemes, the following are required: (1) measures and indicators that provide a landscape view on a dataset; (2) a way to query a dataset even with no prior knowledge of its vocabulary.

We propose an approach to examine LD with these problems in mind. It employs a strategy for inspecting datasets and identifying emerging *knowlege patterns* (KPs). A key step of this method is the construction of a formal logical architecture, or *dataset knowledge architecture*, which summarizes the key features and figures of one or more datasets, thus addressing requirement (1). This, in turn, relies on the notions of KPs and type-property *paths*. We identify the *central properties and types*, i.e. those able to capture most of the knowledge in a dataset, and extract KPs based on the central types. In other words, we extract the dataset vocabulary and analyse the way the data are used in terms of patterns. We also associate general-purpose measures, such as *betweenness* and *centrality*, to the knowledge architecture components of a dataset for performing empirical analysis. These notions and measures will be defined throughout the paper. Using KPs and paths, we can provide prototypical ready-to-use queries for core and concealed knowledge to emerge, thus addressing requirement (2). Although the method applies to datasets whose logical structure is not known a priori, it is meant to analyse LD for serendipitous knowledge. Unlike a mere reverse-engineering exercise, our method discovers new knowledge about datasets, such as their central types and properties and emerging patterns.

The method was partly applied manually and our claims on it are observed empirically, yet it can be generalised and fully automated, as the construction of a dataset architecture and computation of its measures are all derived by directly querying the data using metalevel constructs from RDF and OWL.

The paper is organized as follows. In Section 2 we discuss the general approach for data retrieval and analysis, compounded with our leading hypotheses and basic definitions of recurring terms in our methodology. In Section 3 we describe the *dataset knowledge architecture*, the evaluation measures and an overview of the datasets specifically considered in this analysis. In Section 4 we present and discuss the results of our empirical study, including an example of query, knowledge pattern and dataset figures. After an overview on related work in Section 5, we present our conclusions and future work in Section 6.

2 Approach

Linked Data typically combine types and properties defined either in in-house ontologies, or in widespread existing ones e.g. FOAF[4], DC[5] or GeoNames[6]. It may occur, though, that in-house ontologies have not been formalized, or are not disclosed. Even when the ontologies are available, they do not tell which relevant part of them is actually used in a dataset, and what links are drawn (through the data) between entities across various ontologies. Knowing these details about a dataset is a pre-condition for evaluating its adequateness for being reused in a certain context, for inspecting its content, for integrating it with other (possibly legacy) knowledge; in other words, for using it. We hypothesize that employing KPs for analysing and possibly authoring LD addresses this problem. In this paper, we focus on querying a dataset when its vocabulary is previously unknown, by proceeding as follows:

– we define a method and an ontology for analyzing a dataset and producing a synthesis of it i.e. a modular abstraction named *dataset knowledge architecture*, that highlights how a dataset knowledge is organized, and what its core knowledge components (e.g. central types and associated KPs) are;
– we show how this general method and ontology can be exploited for identifying principal KPs extracted from a dataset for producing prototypical queries, through which we are able to retrieve a dataset core knowledge.

As another premise to our approach to be described in Section 2, we define a few terms that are used throughout the remainder of this paper.

A **knowledge pattern (KP)** for a type in an RDF graph includes: (i) the properties by which instances of this type relate to other individuals; (ii)

[4] Friend-Of-A-Friend, `http://xmlns.com/foaf/0.1/`
[5] Dublin Core, `http://purl.org/dc/elements/1.1/`
[6] GeoNames, `http://www.geonames.org/ontology/ontology_v2.2.1.rdf`

the types of such individuals for each property. A KP is an *invariance across observed data or objects* that allows a *formal or cognitive interpretation* [4]. A KP embeds the key relations that describe a relevant piece of knowledge in a certain domain of interest, similar to linguistic frames and cognitive schemata.

A **path** is an ordered type-property sequence that can be traversed in an RDF graph. Note the use of *types* in lieu of their instances, which instead denote multiple occurrences of the same path. The *length* of a path is the number of properties involved (possibly even with repetitions).

Our approach uses a strategy aimed at modelling, inspecting, and summarizing Linked Data sets, thereby drawing what we call their *knowledge architecture*, which relies on the notions of paths and KPs defined above. The application of this approach is sketched in Figure 1, and can be synthesized as follows:

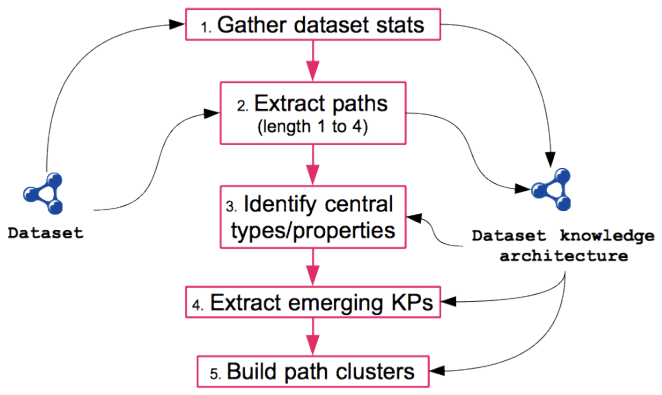

1. Gather dataset stats
2. Extract paths (length 1 to 4)
3. Identify central types/properties
4. Extract emerging KPs
5. Build path clusters
Dataset
Dataset knowledge architecture

Fig. 1. Linked Data analysis methodology Side arrows denote whether the dataset or the knowledge architecture is being accessed for reading or writing on each step.

1. Gather **property usage statistics** for a chosen dataset and store them as ABox of the *knowledge architecture ontology*;
2. Query the dataset for extracting **paths**. Store all paths with length up to 4, with their usage statistics, in the knowledge architecture dataset[7];
3. Identify *central types* and *central properties* based on their frequencies in a key position in paths, i.e. *betweenness*, and number of instantiations;
4. Extract **emerging KPs** based on the dataset's central types and properties;
5. Select clustering factors among central properties, i.e. those properties occupying the same position in a set of paths, and construct **path clusters**;

The following section illustrates how the components of a dataset knowledge architecture are constructed for performing the steps of our method.

3 Method and datasets

Our method described in Section 2 focuses on two main empirical results: (1) building a knowledge architecture of a dataset able to summarize its key

[7] The choice of maximum path length 4 was dictated by computational boundaries and the extreme redundancy empirically observed in longer paths (cf. Section 3).

features; and (2) extracting the central KPs of a dataset. In this Section we focus on what a dataset knowledge architecture is and how to construct it.

3.1 The knowledge architecture

A **dataset knowledge architecture** is an ontology that expresses a dataset vocabulary in a modular way. Its components are selected based on measures that indicate their importance in capturing the core knowledge in a dataset. In other words, it is an abstraction over an RDF graph, which offers a modular view as opposed to the usual "class-property" view provided by vocabularies and ontologies, since it is open to queries that are agnostic to specific types and properties used in the dataset. To populate it, we inspect a dataset for (i) the types and properties it uses, (ii) its typical paths i.e. type-property sequences, and (iii) quantitative statistics about their usage. The knowledge architecture schema is available online[8]. This formalism allows us to empirically analyse a dataset architecture through SPARQL queries. Figure 2 depicts the main entities defined by the dataset knowledge architecture ontology. With the help of

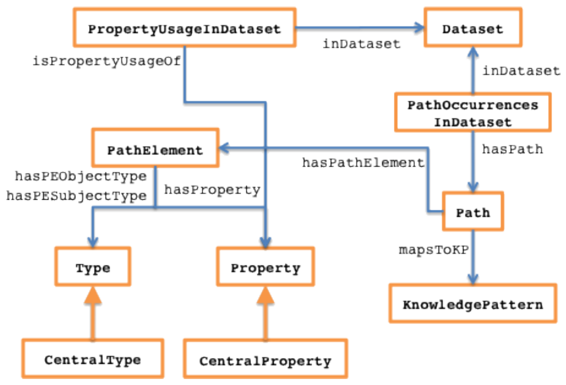


Fig. 2. Diagram of the knowledge architecture ontology. Property arcs denote either class restrictions or domain/range pairs across their nodes.

this ontology, we aim at deriving, in a bottom-up way, the ontology actually employed for representing the data in a LD dataset and produce additional useful knowledge about a dataset e.g., its central types. We identify (i) the properties used in a dataset triples, and model them through the class `Property`; (ii) the types (classes or literals) of the subject and object resources of such triples, and model them through the class `Type`; and (iii) the typical paths that connect triples in a dataset, and model them through the class `Path`.

A `Path` is an ordered set $\{T_1, p_1, ..., p_l, T_{l+1}\}$, where T_i is a `Type`, p_i is a property, and l is the path length. Each ordered subset $\{T_i, p_i, T_{i+1}\}$ of a `Path` of length l is called `PathElement`, and is associated with its position $i = 1, ..l$,

[8] `http://www.ontologydesignpatterns.org/ont/lod-analysis-properties.owl`, which imports the `paths` module as well.

in the path. For example, one DBTune Jamendo[9] instance of `Path` is:

{`mo:MusicArtist foaf:made mo:Record mo:available_as mo:Torrent`}

has length= 2, and is composed of the following `PathElements`:

{`mo:MusicArtist, foaf:made, mo:Record`} `(position 1)`
{`mo:Record, mo:available_as, mo:Torrent`} `(position 2)`

where `mo:` and `foaf:` are prefixes for the Music Ontology (i.e., `http://purl.org/ontology/mo/`) and FOAF namespaces (i.e., `http://xmlns.com/foaf/0.1/`). We then define four properties describing `PathElement`: `hasProperty`, `hasPosition`, `hasPathElementObjectType` and `hasPathElementSubjectType`. Each `Path` is associated to an instance of `PathOccurrencesInDataset`, which indicates the observed number of occurrences of that path in a `Dataset`.

We also define the concepts `CentralType` and `CentralProperty`, which identify the entities capturing most of the knowledge in a dataset. The type `KnowledgePattern` is used for storing the emerging knowledge patterns.

3.2 Measures

Table 1 illustrates the measures that we associate to the knowledge architecture of a dataset to empirically analyse and interpret them to support our conclusive statements in Section 4. Measures from `#Triples` to `#PathOcc` hold for a dataset as a whole, while the others are related to a type or property and can be computed either for one dataset or across multiple datasets. Most of the measures have been computed by combining SPARQL queries and software scripting[10].

The measures for identifying the central types and properties of a dataset are also shown. *Type betweenness* and *Property betweenness* are simplifications of centrality measures used in graph theory. Although we do not model the knowledge architecture of a dataset as a graph, its structure approximates it through the notion of directed paths and based on the empirical observation that all paths longer than 3 are composed of the observed shorter paths.

We can then compute betweenness of types by counting the participation of types as subjects in paths of length 2 at position 2, and betweenness of properties by counting the participation of properties in paths of length 3 in position 2.

The combination of betweenness values and number of instances indicates the *value of centrality* of a type or property for a dataset. Central types and properties are able to capture most of the knowledge expressed in a dataset.

3.3 Datasets

We initially examined several datasets including general-purpose, cross-domain and multimedia domain-specific datasets. As selection criteria for ensuring us a statistically relevant sample for our experiments, we selected datasets:

1. addressing a specific knowledge domain;

[9] DBTune Jamendo, `http://dbtune.org/jamendo/`
[10] Queries available at `http://stlab.istc.cnr.it/stlab/LOD-Analysis-Statistics`

Measure	**What it indicates**	**How it is computed**
# Triples	Number of triples that constitute a dataset.	Sum all `PropertyUsageInDataset` triples.
# Props	Number of used properties.	Count all `PropertyUsageInDataset`.
# Types	Number of used types.	Cardinality of the union set of types related to `PathElement` with either `hasPathElementSubjectType` or `hasPathElementObjectType`.
# Paths	The number of observed paths of length 2 to 4.	Count paths of length n for $n = 2...4$.
# PathOcc	Observed occurrences of paths up to length 4.	Sum the values of `hasNumberOfOccurrences` of paths of length n, $n = 2...4$.
Property usage in paths.	Sparseness indicator of a dataset knowledge architecture.	Divide the number of properties that participates in paths of length n by the total number of used properties, with $n = 2...4$.
Type betweenness	The capability of a type to catch meaningful knowledge.	Count the number of paths of length 2 in which a type participate in at position 1 with subject role.
Property betweenness	The capability of a type to catch meaningful knowledge.	Count the number of paths of length 3 in which a property participates at position 2.
# Triples for property	Number of triples instantiating a property.	Get the value of `hasNumberOfTriples` for `PropertyUsageInDataset` about a property.

Table 1. Measures of dataset dimensions and characteristics in terms of number of triples, number of types and properties used, number of paths of length 2 to 4, number of occurrences of paths of length 2 to 4, and property usage in paths of length 2 to 4.

2. addressing the same specific domain, or a conceptually related one - hence leading to possible cross-relations among them;
3. with different sizes
4. that use third-party as well as in-house ontologies.

We eventually chose three datasets related to the multimedia domain.

Jamendo[11] is an online distributor of independent musical artists. The represented data focus on record authorship, release and distribution over internet channels. Being part of the DBTune service, its data representation relies on the Music Ontology[12] and parts of FOAF, Dublin Core, Event[13], Timeline[14] and Tags[15] ontologies. The indie nature of its hosted artists, who are scarcely represented in other datasets, makes Jamendo a primary source for its content.

John Peel Sessions (JPeel)[16] includes data related to live musical performances for the John Peel Show aired on BBC Radio One, and the resulting record releases. It is also a DBTune dataset, but being more event-focused, it mostly reuses a different portion of the Music Ontology vocabulary than Jamendo does.

[11] Jamendo DBTune home, `http://dbtune.org/jamendo`
[12] The Music ontology, `http://purl.org/ontology/mo/`
[13] Event Ontology, `http://purl.org/NET/c4dm/event.owl#`
[14] Timeline ontology, `http://purl.org/NET/c4dm/timeline.owl#`
[15] Tag vocabulary, `http://www.holygoat.co.uk/owl/redwood/0.1/tags/`
[16] John Peel DBTune home, `http://dbtune.org/bbc/peel`

Dataset	Jamendo	JPeel	LMDB
nTriples	1,047,950	271,369	6,147,978
nProps	24	24	221
nTypes	11	9	53

Table 2. Dimension indicators for the three datasets we analysed.

Measure	Dataset	L=2	L=3	L=4
nPath	Jamendo	33	31	26
	JPeel	56	65	73
	LMDB	546	1,663	3,757
nPathOcc	Jamendo	999,052	1,452,645	2,259,097
	JPeel	1,948,999	14,447,400	1,240,815,607
	LMDB	25,765,513	184,950,315	1,402,705,472
Property usage in paths	Jamendo	1	0.917	0.834
	JPeel	0.917	0.834	0.792
	LMDB	0.847	0.747	0.747

Table 3. Path and property-related figures for the three datasets we analysed.

LinkedMDB (LMDB)[17] is a triplified database of the film industry domain. It encompasses the entities involved with film production and release, plus additional metadata concerning ratings and events such as film festivals. The LinkedMDB ontology is unpublished[18] and almost entirely in-house, with a few exceptions such as FOAF and Dublin Core terms.

These datasets address domain-specific knowledge, thus satisfying our criterion 1. They also address criterion 2 as Jamendo and JPeel share the music production domain (albeit with different data and perspectives), while LMDB addresses the movie production domain, which is related to music e.g. through soundtrack authorship. Table 2 shows how they differ in dimensions, thus satisfying criterion 3. Finally, as for criterion 4 Jamendo and JPeel heavily reuse external ontologies, while LMDB mainly uses a proprietary one. Additionally, their vocabulary usage has little to no overlap.

We excluded general-purpose and cross-domain datasets, e.g. DBPedia and GeoNames, based on the already existing research experience on applying patterns to them. Examples are [10], which addresses the application of patterns to general-purpose datasets such as WordNet[19], and [17] which applies patterns to the Thesaurus of Geographic Names[20] addressing geographical as well as art-related knowledge. In other words, we opted to experiment on a different type of resource in order to lay the basis - in our future work - for comparing our method and results with existing approaches.

For each dataset we computed the measures from Table 1. They provided us with a means to objectively describe datasets according to our selection criteria.

[17] LinkedMDB home, `http://www.linkedmdb.org/`

[18] The base namespace `http://data.linkedmdb.org/resource/movie/` would not resolve as of August 2011, thus forcing us to assume an implicit schema for the dataset.

[19] WordNet, `http://wordnet.princeton.edu/`

[20] Geographic Names, `http://www.getty.edu/research/tools/vocabularies/tgn/`

By the figures in Table 3, the three datasets are not very sparse, due to their high property usage values. This favours the identification of central types and properties. Additionally, datasets differ by several orders of magnitude in number of paths and their occurrences, thus confirming their variety in size. The full list of types and properties used in the three datasets is available online[21].

4 Results and examples

Based on the described entities and the associated measures (cf. Sections 3.2 and 3.3), we can compute each dataset's central types and properties and extract their most representative `Path`s, to the aim of building prototypical queries. These can be constructed by identifying a set of path elements that allow us to retrieve the relevant knowledge about central types.

We extract all emerging KPs through querying all distinct paths of length 1 and group them by their subject types. The KP of a type C (i.e., $KP(C)$) includes the type C, all the properties used for describing its instances, and the object types connected to them. The list of KPs extracted from the three datasets is published online[22].

A prototypical query has to provide an effective summarization of a dataset knowledge about a central type. Such effective summarization includes more than the properties used for describing such type's instances, that is, we need a way to identify an *interesting* neighborhood of the types' instances. To this aim, we exploit the notion of central properties and their clusters of paths of length 3 (where the property is at position 2). Examples of clusters, the queries used for retrieving them, and the full list of extracted KPs are available online[23].

We identify the types and properties through which most of the dataset knowledge transits (i.e., central types and properties, respectively), and show that they have a primary role for selecting KPs and paths for building prototypical queries. We report here the statistics computed for these types and properties in Jamendo[24]. Figures 4(a) and 4(b) show the metrics used for identifying central types; that is, as our task is to build prototypical queries, we look at types with both high betweenness and many individuals, thereby addressing centrality and recall. In the case of Jamendo, we select `mo:Playlist`, `Track`, and `Signal`. Figures 4(c) and 4(d) show the metric values for identifying central types, i.e. the number of triples that instantiate a property and the betweenness of properties. By adopting the same policy as for central types, we select `mo:available_as`, `mo:published_as`, and `foaf:made`.

A prototypical query, which provides a meaningful summarization of a dataset knowledge about a central type, is built by combining KPs and central properties' clusters of paths of length 3, by implementing the following algorithm:

[21] `http://stlab.istc.cnr.it/stlab/LOD-Analysis-TypesAndProperties`

[22] List of KPs extracted from Jamendo (10 KPs), JPeel (8 KPs), and LMDB (51 KPs), `http://stlab.istc.cnr.it/stlab/LOD-Analysis-EmergingKP`

[23] Example clusters at `http://stlab.istc.cnr.it/stlab/LOD-Analysis-Clusters`

[24] Webpage `http://stlab.istc.cnr.it/stlab/LOD-Analysis-Graphs` contains the complete charts; `http://stlab.istc.cnr.it/stlab/LOD-Analysis-Statistics` shows the same data as tables, along with the queries for computing them.

1. Create an empty list PE to store path elements;
2. Let C be a central type;
3. Take the knowledge pattern of C KP(C), and add its path elements to PE;
4. Identify the central properties $p_1, ..., p_n$ involved in $KP(C)$;
5. For all $p_i, i = 1, ..., n$, take paths of length 3 with clustering factor $= p_i$ (select paths with property p_i in position 2);
6. identify path elements in the paths that inlcude C and that are not in $KP(C)$ and add them to PE;
7. Build a CONSTRUCT query by using path elements in PE by using the OPTION construct.

Let us exemplify the generation of a prototypical query, following the algorithm above, for C=`mo:Track`, which is a central type in Jamendo. Based on $KP(mo : Track)$, PE will be initialized by 4 paths, characterized by the properties {`dc:title, mo:available_as, mo:license, mo:track_number`}[25]. Among them, only `mo:available_as` is a central property in Jamendo, hence we pick up its cluster of paths (of length 3) in order to enrich the set PE that will be used for building the prototypical query. From such cluster, we collect 3 additional paths as they are connected to `mo:Track`. Two of them identify incoming links to `mo:Track` (i.e., `mo:track` and `mo:pulished_as`), and one identifies one link in the neighborhood of `mo:Track` (i.e., `dc:format`), which reaches a 2-degree distance from it in its knowledge graph. This additional link in the neighborhood shows how this approach allows to build queries that capture more meaningful knowledge than a simple SPARQL DESCRIBE. An interesting investigation that we have planned in our future work is to study the cognitive soundness of these queries with respect to user interaction tasks.

The result of the described procedure is the following query. Figure 3 shows the resulting graph if such a query would be applied to a specific instance of `mo:Track`, http://dbtune.org/jamendo/track/7593 in this specific case.

```
construct {
 ?t a mo:Track . ?t dc:title ?t1 . ?t mo:available_as ?t2 . ?t2 dc:format ?f .
 ?t mo:license ?t3 . ?t mo:track_number ?t4 .
 ?s a mo:Signal . ?s mo:published_as ?t .
 ?r a mo:Record . ?r mo:track ?t .
 }
 from jamendo_dataset
 where {
 ?t a mo:Track .
 ?t dc:title ?t1 .
 {{OPTIONAL { ?t mo:available_as ?t2 .
  ?t2 dc:format ?f }} UNION
 {OPTIONAL { ?t mo:license ?t3 }} UNION
 {OPTIONAL { ?t mo:track_number ?t4 }} UNION
 {OPTIONAL { ?s a mo:Signal .
  ?s mo:published_as ?t }} UNION
 {OPTIONAL { ?r a mo:Record .
  ?r mo:track ?t }}}
 }
```

These steps allow us to build a summary of a dataset, which supports the retrieval of the most representative knowledge for its domain as (i) paths of length 3 are enough for capturing all knowledge structures, (ii) central types catch most representative knowledge of the dataset, (iii) KPs convey the description of types, and (iv) central properties link the most representative KPs of a dataset. This analytic approach showed that we can summarize a dataset through a relatively small knowledge architecture, thus limiting the impact of empirical analysis on computational complexity. In our three experiments of Section 3.3, we built a

[25] For the sake of readability we omit the object types of the path elements.

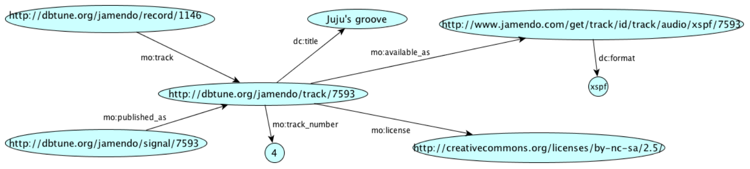
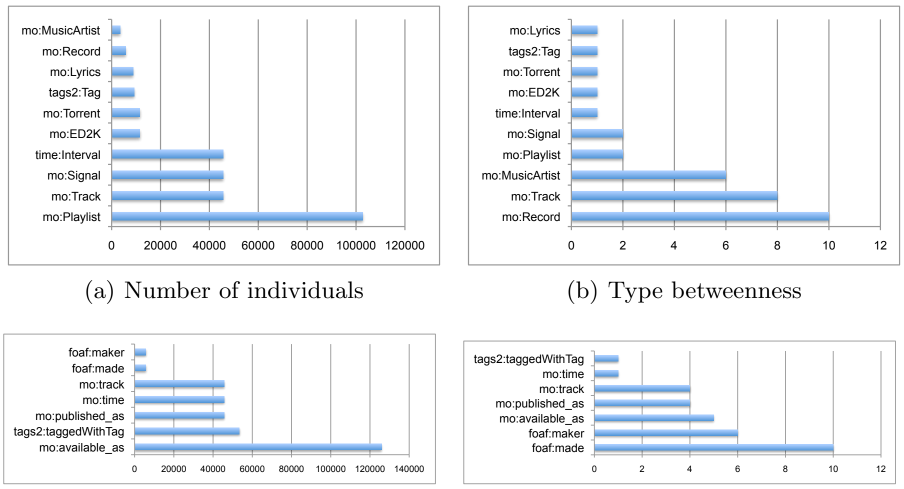

http://dbtune.org/jamendo/record/1146
mo:track
Juju's groove
dc:title
http://www.jamendo.com/get/track/id/track/audio/xspf/7593
mo:available_as
dc:format
xspf
http://dbtune.org/jamendo/track/7593
mo:published_as
http://dbtune.org/jamendo/signal/7593
mo:track_number
4
mo:license
http://creativecommons.org/licenses/by-nc-sa/2.5/

Fig. 3. Result of a the prototypical query for mo:Track filtered on the instance http://dbtune.org/jamendo/track/7593.

knowledge architecture dataset of 130, 373 triples representing three datasets whose combined size sums up to over $7 \cdot 10^6$ triples. The architecture is available online[26].



(a) Number of individuals

(b) Type betweenness

(c) Number of triples

(d) Property betweenness

Fig. 4. Central types and properties in Jamendo.

5 Related work

There has been valuable research work on understanding the LD cloud recently, which highlights different approaches, perspectives, and specific goals. Some work focus on providing macroscopic analysis on LD as a whole such as [5], which analyzes the typical usage of the owl:sameAs standard property; [11], which identifies and generates relations between ontologies used in LD; [7], which discusses how LD would benefit from vocabulary alignments with foundational ontologies; [13], which identifies the most important vocabularies and classes over large-scale distributed datasets.

Works such as [9], [3], [15] focus mainly on query optimization.

[26] `http://ontologydesignpatterns.org/ont/lod-analysis-properties-data.owl`

Other research efforts exploit LD for supporting user interaction with content-intensive applications, such as [12], [16], and [14] but do not try to provide a summarization of the used RDF datasets.

Finally, [2] and [8] focus on providing a compact representation of RDF datasets. In both cases, the main difference with our approach is the lack of design perspective and conceptual analysis. In particular, the taxonomy of classes and properties is not considered, they treat all classes and properties in the same way, while we use them for eliminating redundancies. Furthermore, [2] do not consider the notion of central types and properties (or analogous), while [8] does not exploit semantic web technologies for storing the RDF datasets summaries, which is instead a characteristic of our approach. Finally, our approach focuses on identifying prototypical queries that convey meaningful conceptual organization around a certain type based on the notions of centrality.

6 Conclusion and future work

We have shown how to summarize Linked Data sets by treating them as sets of connected knowledge patterns, in order to identify their core knowledge components. We have experimented on three datasets from the LD cloud, and showed how to build prototypical queries for them even when the ontologies that model them are unknown. We have planned, in our future work, to compare ontologies explicitly published and used for a dataset with the knowledge architecture that arises from our analysis.

Our ongoing and future work focuses on extending our strategy, in order to (i) demonstrate how by aligning emerging KPs of a dataset to general KPs improves interoperability across datasets, and detects incompatibility issues (ii) compare analysis data about different datasets, and (iii) improve user interaction in searches for relevant content. We have planned to improve the method by performing additional analysis on an extensive coverage of the multimedia domain, and subsequently evaluate the cross-domain portability of our approach.

Acknowledgements This work has been part-funded by the European Commission under grant agreement FP7-ICT-2007-3/ No. 231527 (IKS - Interactive Knowledge Stack)

References

1. Aroyo, L., Stash, N., Wang, Y., Gorgels, P., Rutledge, L.: CHIP demonstrator: Semantics-driven recommendations and museum tour generation. In: Semantic Web Challenge. CEUR Workshop Proceedings, vol. 295. CEUR-WS.org (2007)
2. Basse, A., Gandon, F., Mirbel, I., Lo, M.: Dfs-based frequent graph pattern extraction to characterize the content of rdf triple stores. In: Proceedings of the WebSci10: Extending the Frontiers of Society On-Line, Raleigh, NC: US (2010)
3. Fokoue, A., Kershenbaum, A., Ma, L., Schonberg, E., Srinivas, K.: The summary abox: Cutting ontologies down to size. In: Cruz, I.F., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L. (eds.) International Semantic Web Conference. Lecture Notes in Computer Science, vol. 4273, pp. 343–356. Springer (2006), `http://dblp.uni-trier.de/db/conf/semweb/iswc2006.html#FokoueKMSS06`
4. Gangemi, A., Presutti, V.: Towards a pattern science for the semantic web. Semantic Web 1(1-2), 61–68 (2010), `http://dblp.uni-trier.de/db/journals/semweb/`

semweb1.html#GangemiP10

5. Halpin, H., Hayes, P., McCusker, J.P., McGuinness, D., Thompson, H.S.: When owl:sameAs isn't the same: An analysis of identity in Linked Data. In: 9th International Semantic Web Conference (ISWC2010) (November 2010), http://data.semanticweb.org/conference/iswc/2010/paper/261
6. Heim, P., Hellmann, S., Lehmann, J., Lohmann, S., Stegemann, T.: RelFinder: Revealing relationships in RDF knowledge bases. In: Proceedings of the 3rd International Conference on Semantic and Media Technologies (SAMT). Lecture Notes in Computer Science, vol. 5887, pp. 182–187. Springer (2009)
7. Jain, P., Hitzler, P., Yeh, P.Z., Verma, K., Sheth, A.P.: A.p.: Linked Data is merely more data. In: In: AAAI Spring Symposium Linked Data Meets Artificial Intelligence, AAAI. pp. 82–86. Press (2010)
8. Khatchadourian, S., Consens, M.P.: Explod: Summary-based exploration of interlinking and rdf usage in the linked open data cloud. In: Aroyo, L., Antoniou, G., Hyvnen, E., ten Teije, A., Stuckenschmidt, H., Cabral, L., Tudorache, T. (eds.) ESWC (2). Lecture Notes in Computer Science, vol. 6089, pp. 272–287. Springer (2010), http://dblp.uni-trier.de/db/conf/esws/eswc2010-2.html#KhatchadourianC10
9. Maduko, A., Anyanwu, K., Sheth, A., Schliekelman, P.: Graph summaries for subgraph frequency estimation. In: Hauswirth, M., Koubarakis, M., Bechhofer, S. (eds.) Proceedings of the 5th European Semantic Web Conference. LNCS, Springer Verlag, Berlin, Heidelberg (June 2008), http://data.semanticweb.org/conference/eswc/2008/papers/330
10. Malaisé, V., Hollink, L., Gazendam, L.: The interaction between automatic annotation and query expansion: a retrieval experiment on a large cultural heritage archive. In: Bloehdorn, S., Grobelnik, M., Mika, P., Tran, D.T. (eds.) SemSearch. CEUR Workshop Proceedings, vol. 334, pp. 44–58. CEUR-WS.org (2008)
11. Nikolov, A., Motta, E.: Capturing emerging relations between schema ontologies on the Web of Data. In: First International Workshop on Consuming Linked Data (COLD2010) (2010), http://ceur-ws.org/Vol-665/NikolovEtAl_COLD2010.pdf
12. Passant, A., Raimond, Y.: Combining social music and Semantic Web for music-related recommender systems. In: Social Data on the Web (SDoW2008) (2008), http://data.semanticweb.org/workshop/sdow/2008/paper/3
13. Qu, Y., Ge, W., Cheng, G., Zhiqiang, G.: Class association structure derived from linked objects. In: Proceedings of WebSci'09: Society On-Line, Athens, Greece. (2009)
14. Stankovic, M.: Open Innovation and Semantic Web : Problem solver search on Linked Data. In: 9th International Semantic Web Conference (ISWC2010) (November 2010), http://data.semanticweb.org/conference/iswc/2010/paper/439
15. Stuckenschmidt, H., Vdovjak, R., Houben, G.J., Broekstra, J.: Index structures and algorithms for querying distributed rdf repositories. In: WWW. pp. 631–639 (2004)
16. Wang, Y., Stash, N., Aroyo, L., Gorgels, P., Rutledge, L., Schreiber, G.: Recommendations based on semantically enriched museum collections. Web Semantics: Science, Services and Agents on the World Wide Web 6(4), 283 – 290 (2008), http://www.sciencedirect.com/science/article/B758F-4TT7153-1/2/f1bd28cd4d79a0ff70d74439e3f5e3fc, semantic Web Challenge 2006/2007
17. Wang, Y., Stash, N., Aroyo, L., Hollink, L., Schreiber, G.: Semantic relations for content-based recommendations. In: Proceedings of the fifth international conference on Knowledge capture. pp. 209–210. K-CAP '09, ACM, New York, NY, USA (2009), http://doi.acm.org/10.1145/1597735.1597786