Loupe - An Online Tool for Inspecting Datasets in the Linked Data Cloud

Nandana Mihindukulasooriya, María Poveda-Villalón, Raúl García-Castro, and Asunción Gómez-Pérez *

Ontology Engineering Group, Universidad Politécnica de Madrid, Spain
{nmihindu,mpoveda,rgarcia,asun}@fi.upm.es

Abstract. The Linked Data initiative continues to grow making more datasets available; however, discovering the type of data contained in a dataset, its structure, and the vocabularies used still remains a challenge hindering the querying and reuse. VoID descriptions provide a starting point but a more detailed analysis is required to unveil the implicit vocabulary usage such as common data patterns. Such analysis helps the selection of datasets, the formulation of effective queries, or the identification of quality issues. Loupe is an online tool for inspecting datasets by looking at both implicit data patterns as well as explicit vocabulary definitions in data. This demo paper presents the dataset inspection capabilities of Loupe.

Keywords: linked data, vocabulary, inspection, exploration, ontology

1 Introduction

In recent years, we have seen a big enthusiasm in publishing data in a way that can be easily understood by machines following the Linked Data principles leading to a large Web of Data. According to the State of the LOD Cloud 2014[1], there were more than 1,000 publicly available datasets describing more than 8 million resources. However, in order to know which of these datasets can be reused for a given task or to perform queries to find some specific information, people still have to perform multiple data discovery tasks beforehand to find out which type of data each of the datasets contains, their structure, the vocabularies[2] used, and other characteristics.

One way of providing such information about a dataset is to provide a VoID description. However, VoID descriptions are not present in every dataset, and even when they are, the information provided is not enough to understand the structure of data and formulate proper queries. For instance, if we consider a popular dataset, the English DBpedia, it has instances of 321,506 distinct classes

* This research is partially supported by the 4V Spanish national project (TIN2013-46238-C4-2-R) and the FPI grant (BES-2014-068449).

[1] See http://linkeddatacatalog.dws.informatik.uni-mannheim.de/state/

[2] Along this work we will use the terms "vocabulary" and "ontology" interchangeably.

and 58,059 distinct properties associated with those instances. Thus, in order to use such dataset and to understand its content, one needs to know which classes and properties are most commonly used, which properties are generally associated with an instance of a given class, the potential ranges of a given property, etc.

In this paper we present Loupe, an online tool[3] for inspecting the structure of datasets by analyzing both the implicit data patterns (*i.e.*, how the different vocabularies are being used in the data themselves) and the explicit vocabulary definitions (*i.e.*, any RDFS and OWL ontological axioms found in data).

2 Exploring a dataset with Loupe

There are several ways in which datasets can be linked to the ontologies used to annotate their content. Loupe makes use of a series of parametrized queries in order to unveil such links between datasets and ontologies[4].

The set of data annotations themselves represent the vocabularies used in a dataset. In this case, they represent an **implicit** reference to the ontologies appearing in such underlying vocabulary. For inspecting these implicit vocabularies Loupe provides the following capabilities:

- **Class inspector**: reports the classes used in the dataset together with the number of instances for each class and groups those classes by their namespaces. In addition, the properties used by individuals belonging to a given class are also provided. Finally, information about multiclassification is provided, that is, to which other classes, the individuals of a given class belongs.
- **Property inspector**: shows the properties used together with the number of triples in which the property appears as predicate. For each property, a subject and an object analysis are performed. For the subject analysis the system distinguishes between whether the elements found are URI resources or blank nodes. For the objects analysis, the system shows whether they are URI resources, blank nodes, or literals (data types).
- **Triples inspector**: provides the common patterns that appear in the dataset by inspecting all the triples. For each triple, one or more patterns of the form <subjectType, predicate, objectType> are extracted. The subject field represents the different classes to which the individual being the subject of a triple belongs to. The predicate field represents the predicate appearing in the given triple. The object field contains the different classes to which an individual appearing in the object position of a triple belongs to or a literal (or the defined xsd:datatypes).
- **Namegraph inspector**: lists the graphs appearing in the dataset together with the number of triples that each graph contains.

[3] A demo is available at `http://loupe.linkeddata.es/loupe/demo.html`

[4] Detailed information about the methods and SPARQL queries used for inspecting the datasets is provided in `http://loupe.linkeddata.es/loupe/methods.html`

In other cases vocabularies might have been included as part of the dataset itself; in this case, the vocabularies could be defined as instances of the class `owl:Ontology`. For this **explicit** ontological knowledge present in the datasets, Loupe's **ontology inspector** provides information about the ontologies, classes, object properties and datatype properties declared in the dataset. The graphs where this information is stored are also listed. Finally, some provenance information for the inspectors created for each dataset is provided.

3 High-level architecture

Fig. 1 illustrates the high-level architecture of Loupe, which consists of two main components: (a) Loupe Core and (b) Loupe UI (User Interface). In addition, Loupe makes use of an ElasticSearch[5] server for indexing dataset information and a Virtuoso server[6] for providing local SPARQL endpoints when public SPARQL endpoints are not reliable to run the indexing queries.

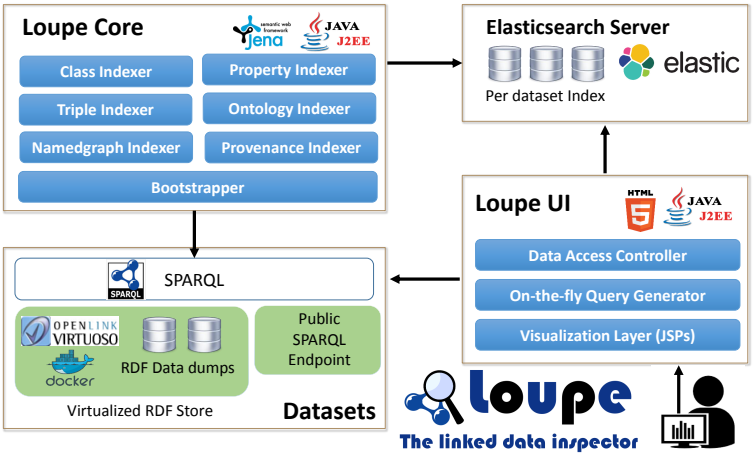

Fig. 1. Loupe high-level architecture

Given a new dataset, Loupe Core creates an index in the ElasticSearch server and indexes the information about the dataset (see Fig. 1) using the methods discussed in Section 2. Once the indexes are created, they are picked up by the Loupe UI, which allows users to browse through the indexed data. In addition most important data is diplayed in tabular formats and search and autocomplete features are provided to make it easier for users to discover the information about used classes and properties.

[5] https://www.elastic.co/products/elasticsearch
[6] https://github.com/openlink/virtuoso-opensource

4 Related Work

make-void[7] is a tool for generating VoID statistics for RDF files while RDF-Stats [1] generates statistics of RDF sources including histograms. Loupe goes beyond the properties defined in VoiD descriptions and extracts more detailed characteristics of classes and properties used in a dataset as well as common triple patterns.

LODStats [2] is a framework for dataset analytics and its portal[8] provides statistics for approximately 10K datasets. However, LODStats does not include information such as the properties associated with instances of a given class or a detailed analysis of subject and object values of a property, as those require complex graph pattern queries that do not fit within its statement-stream-based approach. ABSTAT [3] provides a summary of the most commonly used abstract knowledge patterns similar to the triple-patterns shown by Loupe.

5 Conclusion and future work

This paper presents Loupe, an online tool for inspecting Linked Datasets. It provides a summary of implicit vocabulary information about the dataset such as the classes and properties used in the dataset and a detailed analysis of them. In addition, it inspects the dataset for explicit ontological axiom definitions.

One of the challenges for Loupe is the generation of indexes from the public SPARQL endpoints due to endpoint unreliability, query limitations, and interoperability issues. To address this challenge, (a) queries are decomposed into simpler ones while doing some aggregations and computations in the Loupe-Core and (b) local highly-available endpoints are created if RDF dumps are available. One limitation of Loupe is that it is most suitable when the datasets are updated in a batch mode periodically. If the dataset is frequently updated, the index will have to be recreated to synchronize with the dataset. As future work, we will look into ways of detecting and keeping track of changes to the dataset and updating only the relevant parts of the index.

References

1. Langegger, A., Wöß, W.: Rdfstats-an extensible rdf statistics generator and library. In: 20th International Workshop on Database and Expert Systems Application, 2009. DEXA'09, IEEE (2009) 79–83
2. Auer, S., Demter, J., Martin, M., Lehmann, J.: Lodstats–an extensible framework for high-performance dataset analytics. In: Knowledge Engineering and Knowledge Management. Springer (2012) 353–362
3. Palmonari, M., Rula, A., Porrini, R., Maurino, A., Spahiu, B., Ferme, V.: ASBTAT: Linked Data Summaries with ABstraction and STATistics. In: Demo at the 12th Extended Semantic Web Conference (ESWC2015), Portoroz, Slovenia (June 2015)

[7] `https://github.com/cygri/make-void`
[8] `http://stats.lod2.eu/`