Towards a Linked-Data based Visualization Wizard

Ghislain Auguste Atemezing[1], Raphaël Troncy[1]

EURECOM, Campus SophiaTech, France

Abstract. Datasets published in the LOD cloud are recommended to follow some best practice in order to be 4-5 stars Linked Data compliant. They can often be consumed and accessed by different means such as API access, bulk download or as linked data fragments, but most of the time, a SPARQL endpoint is also provided. While the LOD cloud keeps growing, having a quick glimpse of those datasets is getting harder and there is a need to develop new methods enabling to detect automatically what an arbitrary dataset is about and to recommend visualizations for data samples. We consider that "a visualization is worth a million triples", and in this paper, we propose a novel approach that mines the content of datasets and automatically generates visualizations. Our approach is directly based on the usage of SPARQL queries that will detect the important categories of a dataset and that will specifically consider the properties used by the objects which have been interlinked via `owl:sameAs` links. We then propose to associate type of visualization for those categories. We have implemented this approach into a so-called Linked Data Vizualization Wizard (LDVizWiz).

Keywords: Visualization, human-machine Interaction, LOD, Consuming Linked Data

1 Introduction

With the growing adoption of the Linked Data principles, there is a real need to support data consumers in quickly getting visualizations that enable to explore a dataset. In order to involve more general Web users into the Semantic Web and Linked Data world, there is a need to build tools that reuse existing visualization libraries showing the key information about RDF datasets. Many datasets are published using SPARQL Endpoints and are not "visually" accessible. Thus, understanding the underlying graphs and consuming them require lay users to have some knowledge in writing queries.

The object of visualization is to develop insights from collected data. Moreover, according to Information Theory, vision is the sense that has the largest bandwidth (100 Mbits/s), which makes it the best suited channel to convey information to the brain [12]. Based on the Visual Information Seeking Mantra: *"overview first, zoom and filter, then details on demand"* [7], we advocate for more

visual interactive representations of RDF graphs using SPARQL Endpoints. At the same time, we use the term "Linked Data Visualization", to refer to a *combination of charts, graphics, and other visual elements built on top of 4-5 stars datasets accessible via a SPARQL endpoint.* Despite the presence of more and more datasets published as Linked Data, there is still a need to help end users to discover what (unknown) datasets describe by hiding the complexity of SPARQL queries from such users. Moreover, the task of identifying the key categories of datasets can help in selecting and matching the most suitable visualization types.

In this paper, we propose a first step towards making available a semi-automatic way for the production of possible visualization of linked data sets of high-level categories grouping objects that are worth viewing and we associate them with some very well known vocabularies. Then, we describe the implementation of a Linked Data Visualization Wizard and its main components. This wizard can be used to easily visualize slices of datasets based on generic types detected.

The remainder of this paper is structured as follows. We first provide some related work that further motivates this study (Section 2). In Section 3, we propose some important categories that are worth visualizing and we present a set of mapping views associated with vocabularies (Section 4). In Section 5, we describe the implementation of a wizard that can work on top of any RDF dataset. We detail the results of an experiment where high level categories and associated visualizations have been performed on numerous SPARQL endpoints (Section 6). Finally, we conclude and present some future work in Section 7.

2 Related Work

There are currently many projects aiming at visualizing (RDF) Linked Data. A survey by Dadzie and Rowe [2] concluded with the fact that many visualization tools are not easy to use by lay users. In [4], there is a recent review of some visualizations tools that can be summarized as follows:

- *Vocabulary based visualization tools:* these tools are built for specific vocabularies and that help in visualizing data modelled according to those vocabularies, such as CubeViz [6], FOAF explorer[1] and Map4rdf [3]. They aim at visualizing data modelled respectively with `dq,foaf` and `geo+scovo`.
- *Mashup tools:* they are used to create mashup visualizations with different widgets and some data analysis, such as DERI Pipes [5]. Mashup tools can be integrated into the LD wizard to combine different visual views.
- *Generic RDF visualization tools:* they typically support data browsing and entity rendering. They can also be used to build applications. In this category, we can mention Graphity[2], lodlive[3] and Balloon Synopsis[4].

[1] `http://foaf-visualizer.gnu.org.ua/`
[2] `https://github.com/Graphity/graphity-browser`
[3] `http://en.lodlive.it/`
[4] `https://github.com/schlegel/balloon-synopsis`

While these tools are often extensible and support specific domain datasets, they suffer from the following drawbacks:

- *They are not easy to set up and use by lay users.* Sometimes, users just need to have a visual summary of a dataset in order to start exploring the data. Our approach to this challenge is to provide such a lightweight javascript-based tool that supports a quick exploration task.
- *They do not make recommendation based on categories.* A tool similar to our approach is Facete[5][9] which shows a tree-based structure of a dataset based on some properties of an endpoint more target at geodata. A tabular view enables to visualize slices of data and a map view can be activated when there is geo data. Our approach aims to be more generic, offering more views (tabular, map, graph, charts, etc.) according to a systematic analysis of what are the high level categories present in a dataset.

Regarding wizard-based tools for visualizing data, similar approaches are available for tools consuming datasets in CSV/TSV. ManyEyes[6] is an IBM online tool that suggest charts according to the columns of a given CSV file. Similarly, Google Charts[7] help to achieve the same goals for creating embeddable charts by using DSP language in the framework. Datawrapper[8] is an open source tool to enable the creation of basic charts originally target at journalists inspired by ManyEyes and Google Charts. All the visualizations are based on the type of the columns/fields of the data. In Linked Data, vocabularies are used for modeling datasets in RDF, thus making it difficult to reuse directly those tools. The benefit of our approach is that it constructs specific SPARQL queries to detect the presence or not of predefined specific types of information, yielding to information type-specific visualisations to enable end users to quickly start to explore dataset in a generic manner.

3 Dataset Analysis

When developing an application, there are some "important" classes/categories, objects or datatypes that can be detected first to help to guide in the progress of creating a set of visualizations tied with those categories. We distinguish seven categories while acknowledging that this is not necessary an exhaustive list:

- § [**Geographic information**]: This category is for data modeled using `geo:SpatialThing`, `dbpedia-owl:Place`, `schema:Place` or `gml:_Feature` classes.
- § [**Temporal information**]: This category also includes dataset containing date, time (e.g: `xsd:dateTime`) and period or interval of time, using the OWL Time ontology.

[5] `http://cstadler.aksw.org/facete/`
[6] `http://www-958.ibm.com/software/data/cognos/manyeyes/`
[7] `https://developers.google.com/chart/`
[8] `http://datawrapper.de/`

- § [**Event information**]: This category is for any action of activity occurring at some place at some time.
- § [**Agent/Person information**]: This category is heavily influenced by the use of `foaf:Person` or `foaf:Agent`.
- § [**Organization information**]: This category is related to organizations or companies data, with the use of the `org` vocabulary[9] or the `foaf:Organization` class.
- § [**Statistics information**]: This category refers to statistical data generally modeled using the `data cube` vocabulary[10] or the SDMX model[11].
- § [**Knowledge Classification**]: This category refers to dataset describing schemas, classifications or taxonomies using the `SKOS` vocabulary.

4 Mapping Datatype, Views and Vocabularies

The On-line Library of Information Visualization Environments (OLIVE)[12] is a web site describing eight categories of information visualization environments differentiated by data type and collected by students, following a visualization course given at Maryland College Park, mostly inspired from the work of Ben Shneiderman [7]. Based on the classification provided by OLIVE, we propose a set of mappings between those categories (excluding the workspace dimension), views that can be applied to this category and a suitable list of vocabularies from the Linked Open Vocabularies catalogue [11][13] that correspond to those categories. Those vocabularies are easy to be found as there is a manual classification of vocabularies by the curators of the catalogue based on the content and scope of the terms and properties. According to the seven categories defined in Section 3, we have identified some of their corresponding one to one mapping with the set of vocabularies:

- **Geography** space, consisting of 21 vocabularies for features: `geo`, `gn`, `gf`, `om`, `geop`, `md`, `lgdo`, `loc`, `igeo`, `osadm`, `geod`, `ostop`, `place`, `geos`, `locn`, `coun`, `postcode`, `osr`, `geof`, `g50k` and `ad`.
- **Geometry** space, for vocabularies dealing with the geometries, mostly combined with the features, such as:
- **Time** space, consisting of 14 vocabularies, such as `cal`, `date`, `gts`, `interval`, `ncal`, `oh`, `te`, `thors`, `ti`, `time`, `tl`, `tm`, `tvc` and `tzont`.
- **Event** space, containing vocabularies such as `event`, `lode`, `music`, `sem`, `situ`, `sport`, `stories`, `theatre`, `tis` and `tisc`.
- **Government** space, with 9 vocabularies (`cgov`, `ctorg`, `elec`, `few`, `gc`, `gd`, `oan`, `odd`, `parl`) and the `org` vocabulary belonging to the W3C recommendation vocabularies at `http://lov.okfn.org/dataset/lov/lov#W3C`.

[9] `http://www.w3.org/TR/vocab-org/`
[10] `http://www.w3.org/TR/vocab-data-cube/`
[11] `http://sdmx.org/`
[12] `http://lte-projects.umd.edu/Olive/`
[13] `lov.okfn.org/dataset/lov/`

Metadata vocabularies, such as `rdfs`, `dcterms` or `dce` can be used in association with any of the visual element to give basic description of the resource of a giving dimension. For example, a popup information can be fired on a map view to display the relevant information of a geodata resource such as the label, the abstract or description. Another application can be to detect which visualization is best suited for geodata. Geodata belongs to a two-Dimension visual representation. Geodata is usually displayed using geographical-based visualizations (map, geo charts, etc.) and it is often modeled by vocabularies in the space named `Geometry` and `Geography`[14] vocabularies in RDF datasets. Hence those vocabularies can be combined to detect the presence or not of geographic information in a dataset, and thus yield to recommend a map view. Table 1 gives an overview of those mappings. For the tabular representation, it is the "default" visual representation of RDF data and can be used by any vocabulary without restriction.

Dimension	**Vocabulary Space**	**Visual element**
Temporal	Time space	TimeLine
one-Dimension	any	Tabular, text
two-Dimension	Geography space	Map view
	Geometry space	Maps view
three-Dimension	Event space	Map + TimeLine
Multi-Dimension	qb, sdmx-model, scovo	Charts, graphs
Tree	skos, Government space	Treemap, Org view
Network	any vocab.	Graph, network map

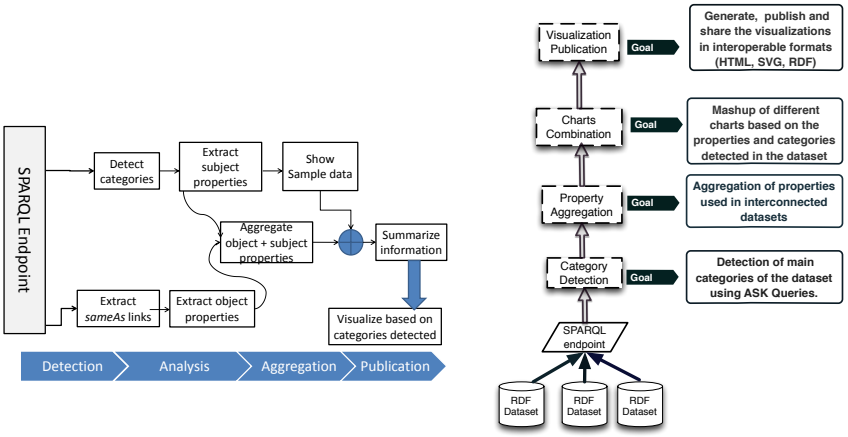
Table 1. A taxonomy of information visualization consuming Linked Datasets with associated views and suitable vocabulary space.

5 LDVizWiz: a Linked Data Visualization Wizard

We propose a workflow composed of four principal steps for a Linked Data Visualization Wizard, as depicted in Figure 1. Our requirement is to provide a tool that hide the complexity of SPARQL to lay users and at the same time, can be embedded in existing Linked Data infrastructure and workflow. First, we proposed to detect the presence of data belonging to one of the seven categories (Table 1) using generic SPARQL queries. More precisely, we perform `ASK` queries to test whether or not a particular query pattern has a solution. Second, we look at entities in a dataset that have `owl:sameAs` links with external objects and we retrieve the properties associated to those objects. We argue that the objects that are interlinked with other datasets are of primary importance in a visualization. We show the results of this mining process to the user (the categories that

[14] All the prefixes used for the vocabularies are the same used in LOV catalogue.

have been detected, the properties going with the categories and the external domain). Based on this information, the user can make a personalized "mashup" by aggregating the suitable visualization widgets. Some default visualizations are available according to the top categories detected. The last step is to publish the visualization and a metadata report in RDF/XML TURTLE or N3.



(a) The workflow of the different modules interacting in the Linked Data visualization wizard

(b) High level functionalities of the Linked Data visualization wizard

Fig. 1. Big picture and architecture of the Linked Data visualization wizard.

Let consider a graph $< G, c >$ to be $G = \{(s, p, o)|p \in URI, s \in URI, o \in (URI \cup LIT)\}$ where URI is the set of URIs, LIT is the set of literals, and c the context. We define $L = \{V_1, V_2, ..., V_n|V_i = P_i \cup T_i\}$ the list of vocabularies in LOV, with P_i and T_i respectively the properties and terms of a vocabulary V_i. Let also $D = \{D_1, D_2, .., D_m\}$ be the domains of vocabularies. We assume $\forall V \in L, \exists D_k \in \Phi(L, D)$. We define a generic function $\Sigma : (G, c) \mapsto B$ to detect categories in a dataset as follows: $\Sigma((G, c)) = \{B|(\exists(s, p, o) \in G : p \in V) \cup (\exists(s, rdf : type, o) \in G : o \in V)\}$ where $B = \{True, False\}$.

In the following sections, we describe each of the steps involved in the Linked Data Visualization Wizard in more details.

5.1 Category Detection

The goal of the category detection task is to use SPARQL queries to detect the presence of some high level categories in the dataset. We perform `ASK` queries as implementation of the Σ function using standard vocabularies as defined in the Table 1. We start with six categories, namely: geographic information, person,

organization, event, time and knowledge organization systems. We select popular vocabularies based on two existing catalogues: LOV [10] and prefix.cc[15].

```
ASK WHERE {
 {
  ?x a ?o.
  filter (?o= dbpedia-owl:Place ||
    ?o=gml:_Feature ||
    ?o=geo:SpatialFeature || ?o=gn:Feature ||
    ?o=admingeo:CivilAdministrativeArea ||
    ?o=spatial:Feature ||
    ?o=vcard:Location)
 }
 UNION {
  ?x ?p ?o. filter(?p=geo:lat || ?p=geo:long ||
  ?p=georss:point || ?p=geo:geometry ||
  geom:geometry)
 }
}
```

Listing 1.1. Generic query to detect geo data from a SPARQL endpoint

Listing 1.1 shows seven classes of different vocabularies are used, respectively for the namespaces `dbpedia-owl, geo, gn, admingeo, spatial` and `vcard`, with relevant classes to check the presence of geographic data.

```
        ASK WHERE {{?x a ?o. filter(?o=time:TemporalEntity ||
        ?o=time:Instant ||
                ?o=time:Interval || ?o=dbpedia-owl:TimePeriod ||
                ?o=time:DateTimeInterval || ?o=intervals:CalendarInterval)
        }
        UNION{ ?x ?p ?o. filter(?p=time:duration ||
        ?p=time:hasBeginning ||
        ?p=time:inDateTime || ?p=time:hasDateTimeDescription
        || ?p=time:hasEnd)}
```

Listing 1.2. Generic query to detect time data from a SPARQL endpoint, using `time, dbpedia-owl, intervals` vocabularies.

Listing 1.2 detects the presence of time information, while Listing 1.3, 1.4 and 1.5 detect persons, organizations and events respectively.

```
        ASK WHERE {?x a ?o. filter(?o = foaf:Person ||
         ?o=dbpedia-owl:Person ||
                ?o=vcard:Individual) }
```

Listing 1.3. Generic query to detect person categories from a SPARQL endpoint, using `foaf, dbpedia-owl, vcard` vocabularies.

```
PREFIX org:<http://www.w3.org/ns/org#>
        PREFIX foaf: <http://xmlns.com/foaf/0.1/>
        PREFIX dbpedia-owl: <http://dbpedia.org/ontology/>
        ASK WHERE {?x a ?o . filter (?o=org:Organization ||
        ?o=org:OrganizationalUnit ||
        ?o=foaf:Organization ||
        ?o=dbpedia-owl:Organisation)}
```

Listing 1.4. Generic query to detect ORG data from a SPARQL endpoint.

[15] `http://prefix.cc`

```
ASK WHERE{?x a ?o. filter (?o= lode:Event || ?o=event:Event ||
?o=dbpedia-owl:Event)}
```

Listing 1.5. Generic query to detect event data from a SPARQL endpoint, using `lode, event, dbpedia-owl` vocabularies.

For detecting data organized as taxonomy, `skos` vocabulary is used along with the most used classes and properties as showed in Listing 1.6.

```
ASK WHERE {{?x a ?o. filter(?o=skos:Concept ||
 ?o=skos:ConceptScheme || ?o=skos:Collection )}
     UNION{ ?x ?p ?o. filter(?p=skos:featureCode ||
    ?p=skos:altLabel || ?p=skos:prefLabel  || ?p=skos:relatedMatch)}}
```

Listing 1.6. Generic query to detect SKOS data from a SPARQL endpoint, using `skos` vocabulary.

5.2 Property Aggregation

We take the benefits of the `owl:sameAs` links between entities to have access to the properties of the entities in the external namespaces different from the origin dataset. This module also aggregates the properties found in the dataset with the ones found in the interlinked sets. This is based on the assumption that during the linkage process, external datasets not only help in not breaking the *follow-your-nose* principle, but also add more information to be viewed in visualization applications. As shown in the code below, at this stage, we have collected and aggregated external properties gathered from the enrichment process of the workflow.

```
1-LET Namespace(?s) = S and LET Namespace(?t) =T
2-SELECT  owl:sameAs links
LET SEMTERM = list of ?s owl:sameAs ?t
WITH T != S
3-IN T, SELECT distinct properties used in dataset
4-AGGREGATE (3) with properties FROM S.
```

5.3 Visualization Generator

This module aims at recommending the appropriate visualizations based on the categories detected by the wizard. It might also help the user to make a report summarizing the result of the mining process, and then use different visualization libraries to view the data. This module can be viewed as a recommender system because it derives visualizations based on the categories. The input to build each visualization is the corresponding SELECT query of each ASK queries used to detect the categories. Moreover, some adjustment are made to avoid blank nodes and to get the labels of the resources. The generator can be coupled with a mashup widget generator for some categories. For example, users could expect for event data, a combination of map view (where), a timeline (when) and facets based on the agents (who).

5.4 Visualization Publisher

The publisher module aims at exporting the combined visualizations, along with the report of all the process of mining the dataset, in a format easy to share, either as HTML, SVG or in the different RDF syntax flavor. For the latter, apart from using metadata information (creator, issued date, license), we model the categories we have detected using the `dcterms:subject` property of a `dcat:Dataset`, the queries used (using the `prov:wasDerivedFrom` property), the sample resources for each category (using the `void:exampleResource` property) and the visualization generated (using the `dvia` and `chart`[16] vocabularies).

6 Experiment and Implementation

In this section, we describe the experiments and report the evaluation on detecting categories on 444 endpoints. We then describe a prototype as a "`proof-of-concept`" of the proposal.

6.1 Experiment set up

We have evaluated our approach on the list of 444 endpoints referenced at `http://sparqles.okfn.org/` monitoring the availability, performance, interoperability and discoverability of SPARQL Endpoints registered in Datahub [1] . We have implemented a script in python to speed up the process and obtain the results. Every ASK query for the different category is implemented in a separate function requesting a JSON response.

6.2 Evaluation

From the 444 endpoints used on the detection category module, 278 endpoints (62.61%) were able to give satisfactory (yes/no on one of the seven categories) results based on the queries. However, almost 37.38% of the endpoints were either down at the time of our experiments or the response header was in XML instead of JSON (as set up in the script). This result shows that our proposal with the current implementation (not covering all the vocabularies in LOV) make use of most popular vocabularies reused in the Linked Data.

This also implies a good coverage of the method that uses standard queries and yet can be extended. The full result of the detection module on the queried services is available at `http://cf.datawrapper.de/3FuiV/2/`, where for each column, the value 0 stands for *no presence* and 1 for the *presence* of the categories. As provided in Table 2, 21.84% of geo data was detected, 13.288% of person data, 10.81% of org data and 3.6% of SKOS data.

Table 3 summarizes some findings for 8 DBpedia chapters endpoints where it's easy to note the absence of SKOS data, and less presence of data modeled using `time` vocabulary. The Table also shows the differences in the standard vocabularies used to convert the wikipedia data into RDF across different chapters.

[16] `http://data.lirmm.fr/ontologies/chart`

Category	number	Percentage
GEO DATA	97	21.84%
EVENT DATA	16	3.60%
TIME DATA	27	6.08%
SKOS DATA	2	0.45%
ORG DATA	48	10.81%
PERSON DATA	59	13.28%
STAT DATA	29	6.6%

Table 2. Classification of the endpoints according to the datatype detected with our SPARQL generic queries

Endpoint	event	geo	org	person	skos	time
dbpedia.org	0	1	1	1	0	0
de.dbpedia.org	0	1	1	1	0	0
el.dbpedia.org	1	1	1	1	0	0
fr.dbpedia.org	1	1	1	1	0	1
ja.dbpedia.org	1	1	1	1	0	0
live.dbpedia.org	1	1	1	1	0	1
nl.dbpedia.org	1	1	1	1	0	0
pt.dbepdia.org	1	1	1	1	0	0

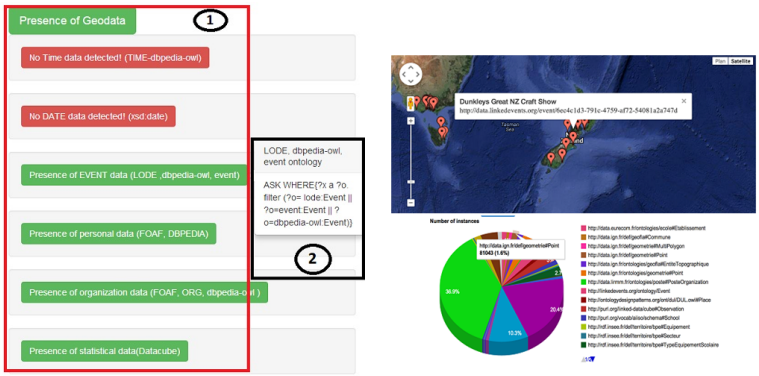
Table 3. Categories detected in some *dbpedia* endpoint domains, where "1" is the presence and "0" the absence of the given type of category.

6.3 Implementation

A first prototype, implemented with javascript and the Bootstrap framework[17], is available at `http://semantics.eurecom.fr/datalift/rdfViz/apps/`, as a proof of concept. We aim at providing a lightweight tool for lay users to quickly understand what the data is about and so that they get first visualizations based on categories detected in the datasets. We also reuse *sgvizler* [8] for generating charts according to the categories retrieved by the wizard. In the current implementation, the user can enter any SPARQL endpoint, and with a "click", the user can receive the list of categories detected together with sample resources. In the second step, the wizard retrieves the properties from the objects and subjects part of `owl:sameAs` links. The last step shows different tabs with the summary of the previous steps, the visualizations available for each categories, and a report both in human and machine readable formats. Figure 2 depicts a sample visualization generated by the wizard for geo data and statistics data.

The system can be used in any tool consuming Linked Data in which the complexity of SPARQL analysis and visualizations of RDF datasets is hidden to the lay users, with the benefits of showing that information encoded in triples is not only "beautiful", but also useful in the sense of traditional wizard-based tools.

[17] `http://getbootstrap.com/`



(a) Overview of the first step report on categories detected:(1) categories wizard, for geodata and stats data detected,(2) vocabularies used.

(b) Two views generated by the LD wizard, for geodata and stats data detected,(2) vocabularies used.

Fig. 2. Categories detected and visualization generated by the Linked Data visualization wizard in the case of EventMedia endpoint service.

7 Conclusion and Future Work

In this paper we have presented an approach for creating visualizations on top of Linked Data. We first defined seven categories of objects worth viewing in a dataset, and we propose to associate them with common Linked Data vocabularies. We then present a description of the main components of LDVizWiz: a Linked Data Visualization Wizard. We describe a lightweight implementation in javascript as a *proof-of-concept* of our proposal, with the benefits to be usable on-line or being extensible. We advocate that such a tool can be easily integrated in any workflow for publishing and linking data on the web, like Datalift[18] or the GeoKnow stack[19]. We plan to use a more exhaustive set of vocabularies in our generic queries for detecting those categories, plugging directly the wizard to the LOV catalogue. Regarding the aggregation properties, it can be extended to take into account other semantic relations (e.g: `skos:exactMatch`). Additionally, we plan to make an evaluation of the prototype and compare it to related tools such as the ones aiming to build a dataset profile.

8 Acknowledgments

This work has been partially supported by the French National Research Agency (ANR) within the Datalift Project, under grant number ANR-10-CORD-009.

[18] `http://www.datalift.org`
[19] `http://generator.geoknow.eu/`

Bibliography

[1] C. Buil-Aranda, A. Hogan, J. Umbrich, and P.-Y. Vandenbussche. Sparql web-querying infrastructure: Ready for action? In *The Semantic Web–ISWC 2013*, pages 277–293. Springer, 2013.

[2] A.-S. Dadzie and M. Rowe. Approaches to visualising linked data: A survey. *Semantic Web Journal*, 2(2):89–124, 2011.

[3] A. de Leon, F. Wisniewki, B. Villazón-Terrazas, and O. Corcho. Map4rdf - Faceted Browser for Geospatial Datasets. In *Using Open Data: policy modeling, citizen empowerment, data journalism (PMOD'12)*, 2012.

[4] J. Klimek, J. Helmich, and M. Neasky. Application of the Linked Data Visualization Model on Real World Data from the Czech LOD Cloud. In 6^{th} *International Workshop on the Linked Data on the Web (LDOW'14)*, 2014.

[5] D. L. Phuoc, A. Polleres, C. Morbidoni, M. Hauswirth, and G. Tummarello. Rapid semantic web mashup development through semantic web pipes. In 18^{th} *International World Wide Web Conference (WWW'09)*, Madrid, Spain, 2009.

[6] P. E. Salas, M. Martin, F. M. D. Mota, K. Breitman, S. Auer, and M. A. Casanova. Publishing statistical data on the web. In *Proceedings of 6th International IEEE Conference on Semantic Computing*, IEEE 2012. IEEE, 2012.

[7] B. Shneiderman. The Eyes Have It: A Task by Data Type Taxonomy for Information Visualizations. In *IEEE Symposium on Visual Languages '96, IEEE, Los Alamos, CA (September 1996)*, pages 336–343, 1996.

[8] G. M. Skjaeveland. Sgvizler: A javascript wrapper for easy visualization of sparql result sets. In 9^{th} *Extended Semantic Web Conference (ESWC'12)*, 2012.

[9] C. Stadler, M. Martin, and S. Auer. Exploring the Web of Spatial Data with Facete. In *Companion proceedings of 23rd International World Wide Web Conference (WWW)*, pages 175–178, 2014.

[10] P.-Y. Vandenbussche and B. Vatant. Metadata Recommendations For Linked Open Vocabularies. OKFN, 2012. `http://lov.okfn.org/dataset/lov/Recommendations_Vocabulary_Design.pdf`.

[11] P.-Y. Vandenbussche, B. Vatant, and L. Rozat. Linked open vocabularies: an initiative for the web of data. In *QetR Workshop*, Chambery, France, 2011.

[12] C. Ware. *Information Visualization, Second Edition: Perception for Design.* Morgan Kaufmann Publishers Inc.; 2 edition (April 21, 2004), San Francisco, CA, USA, 2014.