Exposing the underlying schema of LOD sources

Fabio Benedetti, Sonia Bergamaschi, Laura Po
Università di Modena e Reggio Emilia - Dipartimento di Ingegneria "Enzo Ferrari" - Italy
firstname.lastname@unimore.it

Abstract—**The Linked Data Principles defined by Tim-Berners Lee promise that a large portion of Web Data will be usable as one big interlinked RDF database. Today, with more than one thousand of Linked Open Data (LOD) sources available on the Web, we are assisting to an emerging trend in publication and consumption of LOD datasets. However, the pervasive use of external resources together with a deficiency in the definition of the internal structure of a dataset causes many LOD sources are extremely complex to understand.**

In this paper, we describe a formal method to unveil the implicit structure of a LOD dataset by building a (Clustered) Schema Summary. The Schema Summary contains all the main classes and properties used within the datasets, whether they are taken from external vocabularies or not, and is conceivable as an RDFS ontology. The Clustered Schema Summary, suitable for large LOD datasets, provides a more high level view of the classes and the properties used by gathering together classes that are object of multiple instantiations.

I. Introduction

The LOD Cloud contains more then one thousand of interlinked datasets and several billions of RDF triples. The size of a LOD dataset can vary widely, but on average it contains between thousands and millions of triples[1]. These numbers are still rapidly growing encouraged by the linking open data community and by the open government data initiatives. As greater amounts of data become available through LOD cloud, the expected consumption increases and this encourages new data publications, establishing a virtuous cycle.

Understanding a large and unfamiliar LOD dataset becomes a key challenge for its consumption. Nevertheless, it is often difficult to get the overall view of a large dataset and its metadata are often missing. This becomes even more problematic when users with limited experience encounter a large and complex dataset.

The Semantic Web has provided a schema language such as RDF Schema (RDFS) and an ontology definition language as OWL which allow for adding rich semantics to the dataset. However, not all the LOD datasets make an extensive use of RDFS and OWL, that is, their information about the structure is not explicitly defined. This behavior is primarily due to automatic translation of dataset to the RDF data model from other data model (i.e. Relational data model). The use of external resources together with the lack of intensional knowledge cause that many LOD sources are extremely complex to understand, since the classes and properties used are not described in the intensional knowledge within the dataset, but are hidden in the thousands of instances that represent the extensional knowledge. Moreover the huge size of several LOD datasets is difficult to explore, thus a synthetic view on these sources is needed.

These observations are the motivations behind our work of defining a model and a tool for the creation of a representative schema for a LOD source, developing a framework for LOD source visualization, navigation and querying [1] [2] that facilitates the analysis and comparison of LOD datasets. The central idea of our methodology is the Schema Summary (SS), a concise view built over the LOD dataset. The SS has been recognized as an effective tool to facilitate LOD understanding by helping users quickly make sense of an unfamiliar dataset and explore the instances by defining visual query [3] [4]. However for huge LOD datasets, due to many classes that must be represented, the SS becomes complex and useless. In these cases, a more high-level view could simplify the design and allow navigation and understanding of even big datasets.

In this paper, we further contribute to the LOD summarization process by defining a clustering procedure to shrink the representative schema. We define the Clustered Schema Summary (CSS) that exploits the multiple class instantiation of LOD sources (i.e. the declaration of instances as members of more than one class). Multiple class instantiation is a common practice in knowledge bases and also in LOD datasets, since it could offer a modelling solution for most situation were an instance may be consider as being member of two or more classes. Both SS and CSS enable summarization at different granularities. The SS conveys the implicit structure of the LOD dataset, by displaying the main classes and the properties used among them. While, the CSS provide a further “contraction” of the SS by gathering together classes which concur in the instantiation of the same instances and computing the central class that best identifies each group.

Several techniques of schema summarization has been applied in the last few years to different data models with the purpose of increase the usability and the comprehension of the dataset where they are applied. The Ontology Summarization techniques [5]–[7] usually produce as output a ranked list of the most important concepts identified in the ontology. The main drawbacks of these techniques are their summaries do not represent the structure of the source; moreover these techniques are applied to ontologies of small size containing just intensional content. Differently from these ontologies, the LOD datasets have a more heterogeneous content and a bigger size. In [8], [9], summarization techniques have been applied on vocabularies coming from the LOD cloud. The limitation of these works is that they always rely on a schema available in RDFS format, while several datasets, lacking of intensional knowledge, remain excluded. The main difference

This work has been accomplished in the framework of a PhD program organized by the Global Grant Spinner 2013 and funded by the European Social Fund and the Emilia Romagna Region.

[1] http://lod-cloud.net/state/

of our method is that the summary produced need just the extensional content, so it can be applied on every dataset belonging to the LOD cloud.

The remainder of this paper is organized as follows. In Section II, we formally define the Schema Summary, the Clustered Schema Summary and report a comparative example of SS and CSS. An preliminary evaluation of our methodology is reported in section III, while conclusions are sketched in Section IV.

II. MODEL DEFINITION

Each RDF graph is composed by a set of vertices V and a set of labeled edges E. The vertices can be divided in 3 disjoint sets: the URIs U, the blank nodes B and literals L. Two vertices connected by an edge represent a statement. Each statement is stored into a <*subject*,*predicate*,*object*> triple, where *subject* $\in U \cup B$, *object* $\in V$ and *predicate* $\in E$. We can define the whole RDF graph as a set of triples RG.

Definition 1: $RG \subseteq (U \cup B) \times E \times V$

Each triple belonging to an RDF graph defines a relation between two nodes, and the kind of relation is made explicit through the value of the property. In particular, the RDF property *rdf:type* is used to state that a certain resource is an instance of a class. We define the set of classes as Cs.

Definition 2: $Cs = \{c | \text{<}i,rdf{:}type,c\text{>} \in RG \wedge i \in (U \cup B)\}$

Usually, the triples contained in an RDF graph depict two kind of knowledge: *intensional* and *extensional* knowledge. The RDFS/OWL triples used to define a vocabulary or an ontology describe the intensional knowledge. The instances and the connections between them represent the extensional knowledge. Each of the triples of the extensional knowledge belongs to one of these three main patterns: Subject Class (Sc), Subject Class to literal (Scl) and Object Class (Oc).

Definition 3: $Sc = \{(c,p) | \text{<}i,rdf{:}type,c\text{>} \in RG \wedge \text{<}i,p,u\text{>} \in RG \wedge i \in (U \cup B) \wedge u \in (U \cup B)\}$

Definition 4: $Scl = \{(c,p) | \text{<}i,rdf{:}type,c\text{>} \in RG \wedge \text{<}i,p,l\text{>} \in RG \wedge i \in (U \cup B) \wedge l \in (L)\}$

Definition 5: $Oc = \{(c,p) | \text{<}i,rdf{:}type,c\text{>} \in RG \wedge \text{<}u,p,i\text{>} \in RG \wedge i \in (U \cup B) \wedge u \in (U \cup B)\}$

The Sc, Scl and Oc patterns unveil all the classes and the properties used within the dataset, even if they are not explicitly defined in the intensional knowledge. These information enrich the comprehension of the source itself and are used as input in the building of the SS/CSS. The Index Extraction module [2], developed within the LODeX tool [1], is responsible to extract these patterns and to count their occurrences in the RDF graph. Then, for each LOD dataset, we can build its corresponding Schema Summary.

Definition 6: A ***Schema Summary*** *S, derived from a RDF dataset, is a pseudograph: S = <C, P, s, o, A, m,*Σ_l*, l, count>, where:*

- *C contains a set of c, where c is a class of the RDF dataset; the elements of C represent the nodes of the pseudograph;*
- *P contains the properties between the classes of the RDF dataset; the elements of P represent the edges of the pseudograph;*
- *s: P → C is a function that assigns to each property p ∈ P its source class c ∈ C;*
- *o: P → C is a function that assigns to each property p ∈ P its object class c ∈ C;*
- *A contains all the attributes of the classes of the RDF dataset;*
- *m: A → C is a function that maps each attribute a ∈ A to the class c ∈ C to which it belongs;*
- Σ_l *is the finite alphabet of the available labels;*
- *l:* $(C \cup P \cup A) \rightarrow \Sigma_l$ *is a function that assigns to each class, property or attribute its label;*
- *count:* $(C \cup P \cup A) \rightarrow \mathbb{N}$ *is a function that assigns to each property or attribute the number of times it appears in the RDF dataset, and to each class the number of instances of the class itself.*

The *SC*, *SCl*, *OC*, *Cs* and the function $count()$, able to return the number of occurrences for each pattern, are the input of the SS generation algorithm, while the output is a pseudograph S (in [3], we reported in details the process of generation and querying of a SS.).

The SS is an effective model to represent the classes and properties within a LOD source, however, when dozen of classes are instantiated, it becomes fruitless. In this circumstances, a new way to group together similar classes might enhance the comprehension of the schema. The CSS has been defined to accomplish this task, i.e. grouping together similar classes and producing a more synthetic schema of the source.

Multiple class instantiation is a common practice in LOD datasets; a node of the RDF graph can be, at the same time, instance of more than a class. We call *partial cluster of classes* (PC) a set of classes that concur in the multiple instantiation of the same resource:

Definition 7: $PC(i) = \{c | \text{<}i,rdf{:}type,c\text{>} \in RG \wedge i \in (U \cup B)\}$

Each $PC(i) \subseteq C$ and, by examining all the instances in a RG graph, we find different PC. The collection of all the PC that occur in a RG graph is called *family of PC*, $\mathfrak{C}$:

Definition 8: $\mathfrak{C} = \{PC(i) : \forall i \in (B \cup U)\}$

$\mathfrak{C}$ contains a particular family of sets able to generate all the other sets. We call this family, *family of super sets* ($\mathfrak{S}^2$), and we define it as follow:

Definition 9: $\mathfrak{S} = \{ST \in \mathfrak{C} : \nexists PC \in \mathfrak{C} \wedge PC \supset ST\}$

For each set $st \in \mathfrak{S}$, a class $ca \in st$ must be elected to represent the entire set of classes. This class is called *candidate agent of the superset*. For each superset, we choose as candidate agent the class with the highest number of instances.

[2] We defined an algorithm able to extract $\mathfrak{S}$ starting from $\mathfrak{C}$. The pseudo-code of the algorithm is available at http://www.dbgroup.unimo.it/lodexCluster.

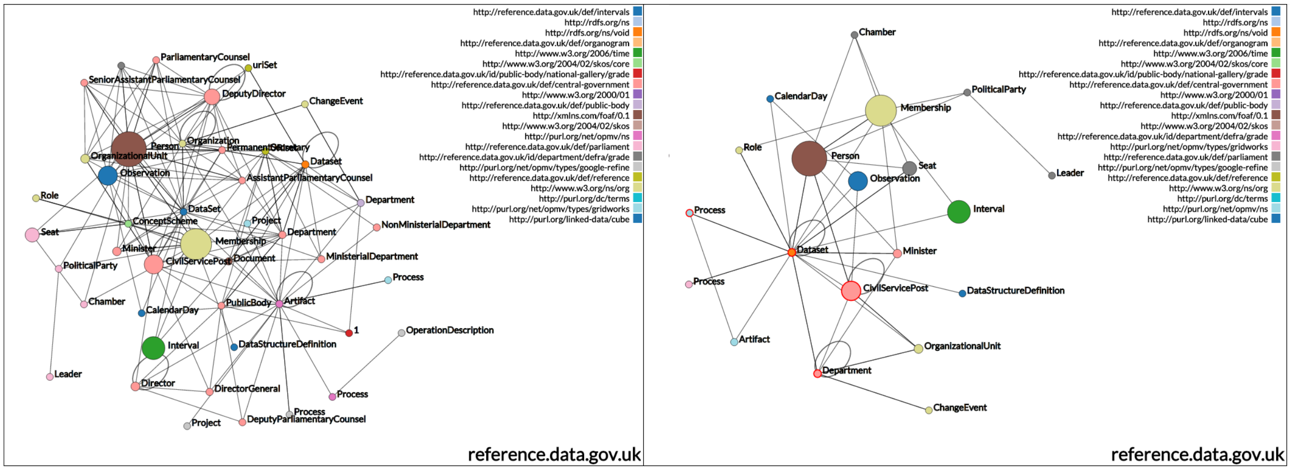
ParliamentaryCounsel
uriSet
SeniorAssistantParliamentaryCounsel
DeputyDirector
ChangeEvent
Organization
Person
PermanentSecretary
OrganizationalUnit
Dataset
Observation
AssistantParliamentaryCounsel
Role
Department
DataSet
Project
NonMinisterialDepartment
ConceptScheme
Seat
Department
Membership
Minister
Document
MinisterialDepartment
CivilServicePost
PoliticalParty
Process
Chamber
PublicBody
Artifact
CalendarDay
1
OperationDescription
Interval
DataStructureDefinition
Leader
Director
DirectorGeneral
Process
Process
Project
DeputyParliamentaryCounsel
http://reference.data.gov.uk/def/intervals
http://rdfs.org/ns
http://rdfs.org/ns/void
http://reference.data.gov.uk/def/organogram
http://www.w3.org/2006/time
http://www.w3.org/2004/02/skos/core
http://reference.data.gov.uk/id/public-body/national-gallery/grade
http://reference.data.gov.uk/def/central-government
http://www.w3.org/2000/01
http://reference.data.gov.uk/def/public-body
http://xmlns.com/foaf/0.1
http://www.w3.org/2004/02/skos
http://purl.org/net/opmv/ns
http://reference.data.gov.uk/def/parliament
http://reference.data.gov.uk/id/department/defra/grade
http://purl.org/net/opmv/types/google-refine
http://reference.data.gov.uk/def/reference
http://www.w3.org/ns/org
http://purl.org/dc/terms
http://purl.org/net/opmv/types/gridworks
http://purl.org/linked-data/cube
reference.data.gov.uk
Chamber
CalendarDay
PoliticalParty
Membership
Role
Person
Seat
Leader
Observation
Process
Interval
Dataset
Minister
Process
CivilServicePost
DataStructureDefinition
Artifact
OrganizationalUnit
Department
ChangeEvent
http://reference.data.gov.uk/def/intervals
http://rdfs.org/ns
http://rdfs.org/ns/void
http://reference.data.gov.uk/def/organogram
http://www.w3.org/2006/time
http://www.w3.org/2004/02/skos/core
http://reference.data.gov.uk/id/public-body/national-gallery/grade
http://reference.data.gov.uk/def/central-government
http://www.w3.org/2000/01
http://reference.data.gov.uk/def/public-body
http://xmlns.com/foaf/0.1
http://www.w3.org/2004/02/skos
http://reference.data.gov.uk/id/department/defra/grade
http://purl.org/net/opmv/types/gridworks
http://reference.data.gov.uk/def/parliament
http://purl.org/net/opmv/types/google-refine
http://reference.data.gov.uk/def/reference
http://www.w3.org/ns/org
http://purl.org/dc/terms
http://purl.org/net/opmv/ns
http://purl.org/linked-data/cube
reference.data.gov.uk

Fig. 1. Representation using LODeX of the SS (left) and the CSS (right) generated from the "Reference data for linked UK government" dataset.

The set of all the candidate agents is called CA. The function $ca : CA \rightarrow \mathfrak{S}$ assigns to each candidate agent the corresponding super set. Now that some classes have been grouped together, we can produce a new SS with a reduced number of classes, that we called Clustered Schema Summary (CSS). The formal definition of the CSS is given in the following.

*Definition 10: A **Clustered Schema Summary** CS for a RDF dataset, derived from the Schema Summary S = <Cs, P, s, o, A, m, Σ_l, l, count>, is a pseudograph: CS = <Cs', P, s, o, A, m, Σ_l, l, count, $\mathfrak{S}$, ca>, where*

- *P, s, o, A, m,Σ_l, l, count are the same elements defined in the Schema Summary S;*
- *Cs' contains the classes represented in the CSS,* $Cs' = Cs - \{st | st \in ST : \forall ST \in \mathfrak{S}\} + CA$
- *$\mathfrak{S}$ is the family of superset;*
- *$ca : CA \rightarrow \mathfrak{S}$ is the candidate agent assignment function.*

We deem that a good summarization algorithm may produce an output that is compatible and comparable with the input. For this reason, we provide an algorithm able to translate the SS/CSS in an RDFS ontology that embodies the structure of the RDF dataset[3]. This translation is not completely lossless; by using the RDFS primitives only, it is not possible to exhibit all the information contained in the SS/CSS. In particular, we lose the number of occurrences of attributes, properties and instances (the function *count(e)*). However, an RDFS ontology describing the structure of the RDF Graph can be very useful and it can be portable to other applications.

To demonstrate the potential of the methodology, we introduce an example showing the SS and CSS built on the reference.data.gov.uk dataset, i.e. a source that contains reference data for linked UK government data, composed by 59K triples, 9K instances and 50 classes. Figure 1 reports, side by side, the SS[4] and the CSS[5] visualized through LODeX. The SS is composed by 44 nodes while the CSS is composed by 20 nodes of which 4 are candidate classes that represent a cluster. In this example, the CSS contains less than the 50% of the SS nodes. Table I lists the clusters that have been automatically generated; it is possible to see that the class selected as candidate agent, in each cluster, correctly represents the whole set of classes. This is even more evident in the last two clusters, where the candidate agents (*CivilServicePost* and *Department*) represent a generalization of the classes contained in the set.

TABLE I. CLUSTERS OF CLASSES IN THE "REFERENCE DATA FOR LINKED UK GOVERNMENT" DATASET (*uk.gov* STANDS FOR HTTP://REFERENCE.DATA.GOV.UK/DEF/)

http://purl.org/net/opmv/ns#Process
http://purl.org/net/opmv/types/google-refine#Process

http://rdfs.org/ns/void#Dataset
http://purl.org/linked-data/cube#DataSet
http://purl.org/net/opmv/ns#Artifact
uk.gov:reference/URIset
uk.gov:reference/uriSet
http://www.w3.org/2004/02/skos/core#ConceptScheme
http://xmlns.com/foaf/0.1/Document

***uk.gov*:central-government/CivilServicePost**
uk.gov:central-government/AssistantParliamentaryCounsel
uk.gov:central-government/DeputyDirector
uk.gov:central-government/DeputyParliamentaryCounsel
uk.gov:central-government/Director
uk.gov:central-government/DirectorGeneral
uk.gov:central-government/ParliamentaryCounsel
uk.gov:central-government/PermanentSecretary
uk.gov:central-government/SeniorAssistantParliamentaryCounsel
http://reference.data.gov.uk/id/public-body/national-gallery/grade/1

***uk.gov*:central-government/Department**
uk.gov:central-government/MinisterialDepartment
uk.gov:central-government/NonMinisterialDepartment
uk.gov:central-government/PublicBody
uk.gov:public-body/Department
http://www.w3.org/ns/org#Organization

[3] The pseudo-code of the algorithm is available at http://www.dbgroup.unimo.it/lodexCluster.

[4] Please use Chrome to access to this url: http://dbgroup.unimo.it/lodex2/testCluster#!/schemaSummary/328.

[5] Please use Chrome to access to this url: http://dbgroup.unimo.it/lodex2/testCluster#!/cSchemaSummary/328

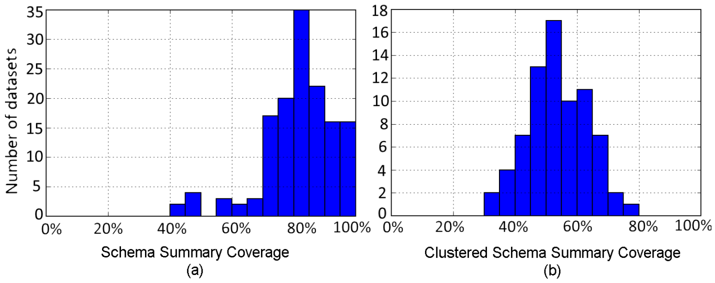

Fig. 2. Coverage of SS and CSS

III. Preliminary results

The evaluation of ontology summarization techniques is a quite controversial topic in literature. In fact, these techniques are usually designed to summarize an ontology with a purpose, so their evaluation is focused on the achievement of this goal. Nevertheless, Li and Motta in [10] tried to provide a systematic view of the different evaluation measures. We chose and adapted to our scenario one of the metrics prosed to estimate the quality of the summarized ontology we built: the corpus coverage ontology evaluation, that scores the ontology appropriateness to cover the topic of the corpus.

The datasets used to evaluate our methodology are taken from DataHub[6]. For each of the SPARQL endpoints listed in DataHub, we make use of the Index Extraction module [2] of LODeX to extract the patterns needed to generate the SS/CSS and the corresponding RDFS ontologies. Currently, 206 (34.39 %) out of the 599 LOD datasets listed on Data Hub are SPARQL 1.1 compatible datasets. We were able to generate the SS on 185 datasets [3] and the CSS on 90 datasets (only half of the inspected datasets contained multiple class instantiations)[7].

The SS supplies a synthetic representation of the extensional knowledge, so that we can think to the concept of *coverage* as what percentage of the instances in the source RDF graph (i.e.*corpus* int the original definition) are represented though the SS/CSS. Moreover, the SS can be translated into an RDFS ontology.

To evaluate the coverage of SS and CSS the function *count* is crucial. This function reports how many time a particular pattern appears in the source graph. The number of triples that are represented though the SS/CSS, n_{repr}, is approximatively computed by summing the number of occurrences of each property and each attribute and the number of class instances that appear in the SS/CSS. The ratio of n_{repr} to the total number of triples of the RDF graph, n_{tot}, give us the coverage.

Definition 11: $coverage = n_{repr}/n_{tot}$

In Figure 2 (a), it is shown the distribution of the coverage for 140 SS[8]. We obtained an high coverage, with an average value of 80%. An exception is given for 9 datasets that obtain a coverage under the 60%. By examining these cases, we discovered that the datasets describe an ontology in which the intensional knowledge is predominant. The coverage distribution for the CSS shows an average value of 54%. This clearly shows a predictable result that, by using the CSS, we are not able to represent some instances of the clustered classes.

IV. Conclusion

This paper has proposed a formal method able to unveil the implicit structure of a LOD dataset by building a Schema Summary (SS) and, a more compressed, Clustered Schema Summary (CSS). The SS exposes all the main classes and properties used within the datasets, either they are taken from external vocabularies or not. The CSS provides a more high level view of the classes and the properties used, it exploits the multiple class instantiations to generate clusters of classes and decrease the overall size of the graph. Both SS and CSS have shown an high coverage of the LOD source on which they are applied. The most crucial aspect, for future developments, is the generation of SPARQL queries on a CSS. This solicits a mapping functionality to convert a query on the CSS to a SPARQL query on the LOD endpoint.

References

[1] F. Benedetti, S. Bergamaschi, and L. Po, “A visual summary for linked open data sources,” in *Proceedings of the ISWC 2014 Posters & Demonstrations Track a track within the 13th International Semantic Web Conference, ISWC 2014, Riva del Garda, Italy, October 21, 2014.*, ser. CEUR Workshop Proceedings, vol. 1272, 2014, pp. 173–176.

[2] ——, “Online index extraction from linked open data sources,” in *Proceedings of the Second International Workshop on Linked Data for Information Extraction (LD4IE 2014) co-located with the 13th International Semantic Web Conference (ISWC 2014), Riva del Garda, Italy, October 20, 2014.*, ser. CEUR Workshop Proceedings, vol. 1267, 2014, pp. 9–20.

[3] ——, “Visual querying lod sources with lodex,” to appear in the 8th International Conference on Knowledge Capture, K-CAP 2015.

[4] ——, “Lodex: A tool for visual querying linked open data,” to appear in the 14th International Semantic Web Conference ISWC 2015 (Posters & Demos).

[5] S. Peroni, E. Motta, and M. dAquin, “Identifying key concepts in an ontology, through the integration of cognitive principles with statistical and topological measures,” in *The Semantic Web*. Springer, 2008, pp. 242–256.

[6] N. Li, E. Motta, and M. d’Aquin, “Ontology summarization: an analysis and an evaluation,” 2010, proceedings of the International Workshop on Evaluation of Semantic Technologies (IWEST 2010), [held in conjunction with] 9th International Semantic Web Conference (ISWC2010).

[7] G. Wu, J. Li, L. Feng, and K. Wang, “Identifying potentially important concepts and relations in an ontology,” in *The Semantic Web-ISWC 2008*. Springer, 2008, pp. 33–49.

[8] X. Zhang, G. Cheng, W.-Y. Ge, and Y.-Z. Qu, “Summarizing vocabularies in the global semantic web,” *Journal of Computer Science and Technology*, vol. 24, no. 1, pp. 165–174, 2009.

[9] G. Cheng, F. Ji, S. Luo, W. Ge, and Y. Qu, “Biprank: ranking and summarizing rdf vocabulary descriptions,” in *The Semantic Web*. Springer, 2012, pp. 226–241.

[10] N. Li and E. Motta, “Evaluations of user-driven ontology summarization,” in *Knowledge Engineering and Management by the Masses*. Springer, 2010, pp. 544–553.

[6] http://datahub.io/

[7] A online demo of some of the SS and CSS extracted is available here (please use Chrome) :http://dbgroup.unimo.it/lodex2/testCluster.

[8] The number of datasets is decreased, since some SPARQL endpoints do not supply the number of triples.