Discovering Related Data Sources in Data-Portals

Andreas Wagner[†], Peter Haase[‡], Achim Rettinger[†], and Holger Lamm[‡]

[†] Karlsruhe Institute of Technology and [‡] fluid Operations
{a.wagner,rettinger}@kit.edu, peter.haase@fluidops.com

Abstract. To allow effective querying on the Web of data, systems frequently rely on *data from multiple sources* for answering queries. For instance, a user may wish to combine data from sources comprised in different statistical catalogs. Given such federated queries, in order to enable an *interactive exploration* of results, systems must allow user involvement during *data source selection*. That is, a user should be able to choose data sources contributing to query results, thereby allowing to refine/expand current findings. For this, one needs effective recommendations for data sources to be picked: *data source contextualization*. Recent work, however, solely aims at source contextualization for "Web tables", while heavily relying on schema information and simple table structures. Addressing these shortcomings, we exploit work from the field of data mining and show how to enable effective Web data source contextualization. Based on a real-world finance use-case, we built a contextualization engine, which we integrated into a Web search system, our *data portal*, for accessing statistics data sets.

1 Introduction

The amount of RDF data available on the Web today, such as Linked Data[1], RDFa and Microformats, is large and rapidly increasing.[2] RDF data contains descriptions of entities, with each description being a set of triples. A triple associates an entity identifier (subject) with an object via a predicate. A set of triples forms a data graph.

RDF data is oftentimes highly *distributed*, with each data source comprising one or more RDF graphs (Fig. 1). Most notably, Linked Data as well as Web-accessible SPARQL endpoints have contributed to this development.

Integrated Querying of Multiple Data Sources. In order to fulfill information needs over multiple, distributed data sources, a number of issues need to addressed. These range from the ability to discover and identify relevant data sources, to the ability to integrate them and finally to support querying them in a transparent manner.

[1] http://www.w3.org/DesignIssues/LinkedData.html
[2] http://webdatacommons.org

Src. 1: `tec00001` (Eurostat).

```
es:data/tec0001
 rdf:type qb:Observation;
 es-prop:geo es-dic:DE;
 es-prop:unit es-dic:MIO_EUR;
 es-prop:indic_na es-dic:B11;
 sd:time "2010-01-01"^^xs:date;
 sm:obsValue "2496200.0"^^xs:
      double.
```

Src. 2: `gov_q_ggdebt` (Eurostat).

```
es:data/gov_q_ggdebt
 rdf:type qb:Observation;
 es-prop:geo es-dic:DE;
 es-prop:unit es-dic:MIO_EUR;
 es-prop:indic_na es-dic:F2;
 sd:time "2010-01-01"^^xs:date;
 sm:obsValue "1786882.0"^^xs:
      decimal.
```

Src. 3: `NY.GDP.MKTP.CN` (Worldbank).

```
wbi:NY.GDP.MKTP.CN
 rdf:type qb:Observation;
 sd:refArea wbi:classification/country/DE;
 sd:refPeriod "2010-01-01"^^xs:date;
 sm:obsValue "2500090.5"^^xs:double;
 wbprop:indicator wbi:classification/indicator/NY.GDP.MKTP.CN .
```

Fig. 1: Src. 1/3 describes Germany's GDP in 2010. Note, they contextualize each other, as they feature varying observation values respectively properties. Src. 2 also provides additional information w.r.t. Src. 1, as it holds the German debt in 2010. Src. 1-3 each contain one entity: `es:data/tec0001`, `es:data/gov_q_ggdebt`, and `wbi:NY.GDP.MKTP.CN`. Every entity description, $\mathcal{G}_e$, equals the entire graph contained in its source.

Say a user is interested in economic data. Here, catalogs like Eurostat[3] or Worldbank[4], offer rich statistical information about, e.g., GDP, spread across many sources. However, these data sources are very specific, and in order to provide the user with her desired information, a system has to *combine data from multiple sources*. Processing queries in such a manner requires knowledge about what source features which information. This problem is commonly known as *source selection*: a system chooses data sources relevant for a given query and query fragment, respectively. Previous works selected sources by means of indexes, e.g., [8, 11], link-traversal, e.g., [9, 11], or by using available source metadata, e.g., [5].

Data Source Contextualization. Existing approaches for source selection aim solely at a mapping of queries/query fragments to *sources featuring exactly matching data* [5, 8, 9, 11]. In particular, such works do not consider "source semantics", i.e., what sources are actually about and how they relate to each other. For instance, consider a user searching for GDP rates in the EU. A traditional system may discover sources in Eurostat to comprise matching data. At the same time, other sources offer *contextual* information concerning, e.g., the national debt. Notice, such sources are actually not relevant to the user's query,

[3] `http://ec.europa.eu/eurostat/`
[4] `http://worldbank.org`

but *relevant to her information need*. Integration of these additional sources for contextualization of known, relevant sources, provides a user with broader results in terms of result dimensions (schema complement) respectively result entities (entity complement). See also an example in Fig. 1.

For enabling systems to identify and integrate sources for contextualization, we argue that *user involvement during source selection* is a key factor. That is, starting with an initial search result obtained via, e.g., a SPARQL or keyword query, a user should be able to choose and change sources used for result computation. In particular, users should be recommended contextual sources at each step of the search process. After modifying the selected sources, results may be reevaluated and/or queries expanded.

Recent work on data source contextualization focuses on Web tables [4], while using top-level schema such as Freebase[5]. Further, they restrict data to a simple table-structured form. We argue that such a solution is not a good fit for the "wild" Web of data. In particular, considering Linked Data, data sources frequently feature schema-less data and/or high-dimensional, heterogeneous entities. Targeting the Web of data, we propose an approach based on well-known techniques from the field of data mining. That is, we extract a sample of entities from each data source and learn clusters of entities. Then, we exploit the constructed clusters as a description for data sources, and *find contextual sources via similarity measures between entity clusters*.

Contributions. In this work, we provide the following contributions:

(1) We present an entity-based solution for data source contextualization in the Web of data. This engine is based on well-known data mining strategies, and does not require schema information or data adhering to a particular form.

(2) We implemented our system, the *data-portal*, based on a real-world use case, thereby showing its practical relevance and feasibility. A prototype version of this portal is freely available and currently tested by a pilot customer.[6]

Outline. In Sect. 2, we present a real-world use case. In Sect. 3, we outline our contextualization engine, before we discuss the data portal system in Sect. 4. We present related work in Sect. 5. We conclude with Sect. 6.

2 Use Case Scenario

In this section, we introduce a real-world use case to illustrate *challenges and opportunities in contextualizing data sources*. The scenario is situated in financial research, provided by a pilot user in a private bank.

In their daily work, financial researchers heavily rely on a variety of open and closed Web data sources in order to provide prognoses of future trends. A typical example is the analysis of government debt. During the financial crisis in 2008-2009, most European countries made high debts. To lower doubts about repaying

[5] `http://www.freebase.com/`

[6] `http://data.fluidops.net/`

these debts, most countries set up a plan to reduce their public budget deficits. The fulfillment of these plans is essential for the Euro zone's development.

To analyze such plans, a financial researcher requires an overview of public revenue and expenditure in relation to the gross domestic product (GDP). To measure this, she needs information about the deficit target, the revenue/-expenditure/deficit and GDP estimates. This information is publicly available, provided by catalogs like Eurostat and Worldbank. However, it is spread across a huge space of sources. That is, there is no single source satisfying her information needs, instead data from multiple sources have to be identified and combined.

To start her search process, a researcher may give "gross domestic product" as keyword query. The result is GDP data from a large number of sources. At this point, data source selection "hidden" from the researcher, and sources are solely ranked via number and quality of keyword hits. However, knowing *where her information comes from* is critical. In particular, she may want to restrict and/or know the following meta-data:

- General information about the data source, e.g., the name of the author and a short description of the data source contents.
- Information about entities contained in the data source, e.g., the single countries of the European Union.
- Description about the dimensions of the observations, e.g., the covered time range or the data unit of the observations.

By means of faceted search, the researcher finally restricts her data source to `tec00001` (Eurostat, Fig. 1) featuring "Gross domestic product at market prices". However, searching the data source space in such a manner requires extensive knowledge. Further, the researcher was not only interested in plain GDP data – she was also looking for *additional information*.

For this, a system should *suggest data sources* that might be of interest, based on sources known to be relevant. These *contextual sources* may feature related, additional information w.r.t. current search results/sources. For instance, data sources containing information about the GDP of further countries or with a different temporal range. By such means, the researcher may discover new sources more easily, as one source of interest links to another – allowing her to *explore the space of sources*.

3 Contextualisation Engine

In this section, we outline an approach for Web data source contextualization.

For this, we conceive a data source $D \in \mathcal{D}$ as set of multiple RDF graphs, with $\mathcal{D}$ as set comprising all sources in the data space. Further, an entity e is given by an RDF instance contained in source D, and described by a subgraph $\mathcal{G}_e$ in D [6], see also Fig. 1.

Related Entities. The intuition behind our approach is simple: *if data sources contain similar entities, they are somehow related*. In other words, we rely on entities to capture the "latent" semantics of data sources. That is, we em-

ploy offline procedures as follows: we (1) extract entities, (2) measure similarity between them, and (3) cluster them.

(1) *Entity Extraction.* We start by extracting entities from each source D. First, for scalability reasons, we go over all entities in data graphs in D and collect a sample, with every entity having the same probability of being selected. For each selected entity e, we crawl its surrounding subgraph – resulting in a graph $\mathcal{G}_e$ that describes e [6]. For cleaning $\mathcal{G}_e$, we apply standard data cleansing strategies to fix, e.g., missing or wrong data types.

(2) *Entity Similarity.* In a second step, we define a dissimilarity measure, dis, between two entities based on previous work on kernel functions for RDF [12]. That is, for a given entity pair e' and e'', we count common substructures in $\mathcal{G}'_e$ and $\mathcal{G}''_e$. The more "overlapping structures" between the two graphs are found, the lower we score the dissimilarity between e' and e''. In addition to the structural characteristics of entities, we also consider their literal dissimilarity. For entities e' and e'' we pairwise compare their literals by means of string and numerical kernels, respectively [16]. The former counts the number of common substrings, given a literal pair from e'/e''. The latter, on the other hand, computes the numerical distance between two literals associated with e' and e''. We aggregate these three different dissimilarity measures for e' and e'' via kernel aggregation strategies [16]. Intuitively, such a kernel aggregation combines multiple kernels using, e.g., weighted summation.

(3) *Entity Clustering.* Last, we apply clustering techniques to mine for entity groups. More precisely, we aim at discovering clusters, C_j, comprising similar entities, which may or may not originate from the same source. Thus, clusters relate sources by relating their contained entities. We use k-means [10] as a well-known and simple algorithm for computing entity clusters. k-means adheres to four steps [10]. (a) Choose k initial cluster centers m_i. (b) Based on above dissimilarity function, dis, an indicator function is given as: $\mathbb{1}(e, C_j)$ is 1 iff $dis(e, C_j) < dis(e, C_z), \forall j \neq z$, and 0 otherwise. Intuitively, $\mathbb{1}(\cdot, \cdot)$ assign each entity e to its "closest" cluster C_j. (c) Update cluster centers m_i, and reassign (if necessary) entities to new clusters. (d) Stop if convergence threshold is reached, e.g., no (or minimal) reassignments occurred. Otherwise go back to (b).

Contextualisation Score. Similar to [4], given a source D', we compute two scores, $ec(D'' \mid D')$ and $sc(D'' \mid D')$, for quantifying the contextualization of D' via a second source D''. Both scores are aggregated to a contextualization score for data source D'' given D'.

The former is an indicator for the entity complement of D'' w.r.t. D'. That is, ec asks: *how many new, similar entities* does D'' contribute to given entities in D'? The latter score, sc, measures *how many new "dimensions"* are added by D'', compared to those already present in D' (schema complement). In contrast to [4], however, we do not rely on any kind of "external" information, such as top-level schema. Instead, we *solely exploit semantics as captured by entities.*

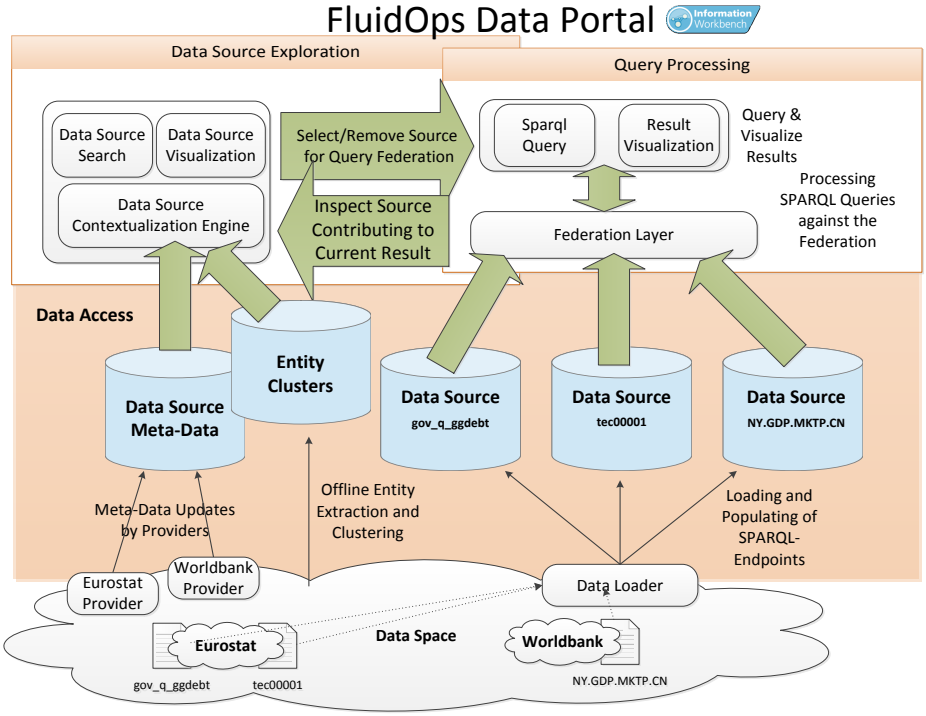


Fig. 2: The *data portal* system features two kinds of services: source space exploration, and query processing. For the former, our source contextualization engine is integrated as a key component. Overall, source space exploration requires source meta-data as well as entity clusters to be available. Entity clusters are computed as an offline process, while meta-data may be updated frequently during runtime. On the other hand, query processing distributes query fragments via a federation layer. Each fragment is evaluated over one or more sources. For this, each data source is mapped to a SPARQL endpoint, for which data is accessed via a data-loader. For our running example, the necessary sources are loaded via three endpoints: `gov_q_ggdebt`, `tec00001`, and `NY.GDP.MKTP.CN`.

Let us first define an entity complement score $ec : \mathcal{D} \times \mathcal{D} \mapsto [0, 1]$. In the most simplistic manner, we may measure ec by counting the overlapping clusters between both sources:

$$ec(D'' \mid D') \coloneqq \sum_{C_j \in\, cluster(D')} \frac{\mathbb{1}(C_j, D'')|C_j|}{|C_j|}$$

with *cluster* as function mapping data sources to clusters their entities are assigned to. Further, let $\mathbb{1}(C, D)$ by an indicator function, returning 1 if cluster C is associated with data source D via one or more entities in D.

Considering the schema complement score, $sc : \mathcal{D} \times \mathcal{D} \mapsto [0,1]$, we aim to count new dimensions (properties) that are introduced by D''. Thus, a simple realization of sc may be given by:

$$sc(D'' \mid D') \coloneqq \sum_{C_j \in\, cluster(D'')} \frac{|props(C_j) \setminus \bigcup_{C_i \in\, cluster(D')} props(C_i)|}{|props(C_j)|}$$

with *props* as function projecting a cluster C to a set of properties, where each property is contained in a description of an entity in C.

Finally, a contextualization score cs is obtained by a monotonic aggregation of ec and sc. In our case, we apply a weighted summation:

$$cs(D'' \mid D') \coloneqq 1/2 \cdot ec(D'' \mid D') + 1/2 \cdot sc(D'' \mid D')$$

Runtime Behavior and Scalability. Regarding online performance, i.e., computation of contextualization score cs, given the offline learned clusters, we aimed at simple and lightweight heuristics. For ec only an assignment of data sources to clusters (function $cluster(D)$), and cluster size $|C|$ is needed. Further, measure sc only requires an additional mapping of clusters to "contained" properties (function $props(C)$). All necessary statistics are easily kept in memory.

With regard to offline clustering behavior, we expect our approach to perform well, as existing work on kernel k-means clustering showed such approaches to scale to large data sets, e.g., [3, 18].

4 Source Contextualization in the Data-Portal

We have implemented the presented algorithms (Sect. 3) for data source contextualization in a *data-portal*, enabling on demand access to data sources from a number of open statistic data catalogs. Based on our real-world use case, we show how the source contextualization is used within this portal.

Overview. Towards an active *involvement of users in the source selection* process, we implemented a contextualization engine and integrated it in a system offering two services: *source space exploration* and *distributed query processing*,

Using the former, users may explore the space of sources, i.e., search and discover data sources of interest. Here, the contextualization engine fosters discovery of relevant sources during exploration. The query processing service, on the other hand, allows queries to be federated over multiple sources. See also Fig. 2 for an overview.

Interaction between both services is tight and user-driven. In particular, sources discovered during source exploration may be used for answering queries. On the other hand, sources employed for result computation may be inspected, and via contextualization other relevant sources may be found.

The data portal is based on the Information Workbench [7], and a running prototype is available.[7] Following our use-case (Sect. 2), we populated the system

[7] `http://data.fluidops.net/`

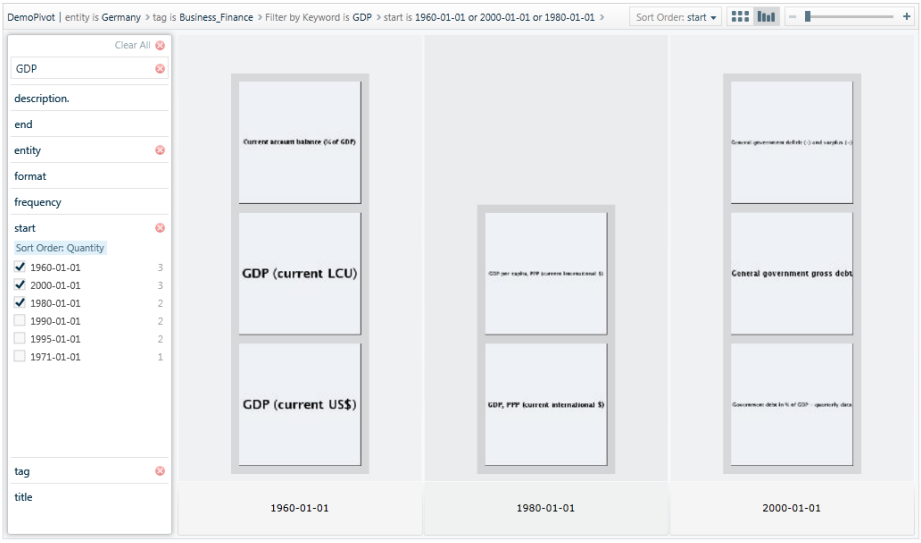


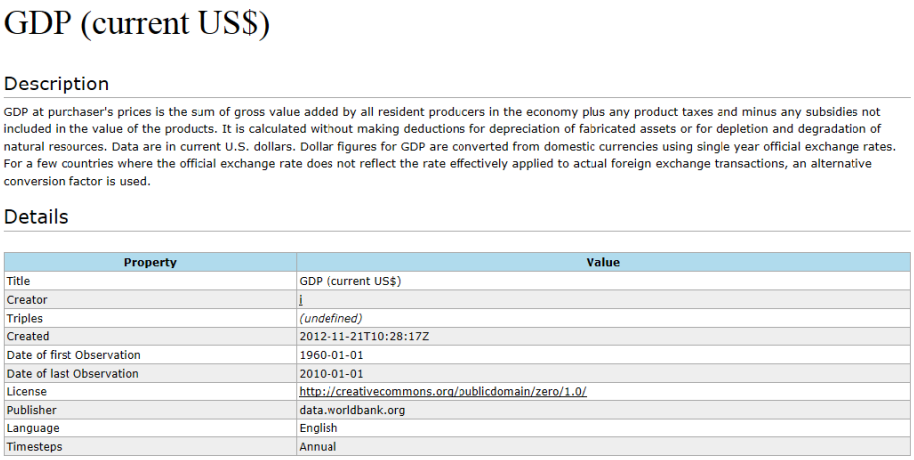
Fig. 3: Faceted search exploration of data sources.

with statistical data/sources from Eurostat and Worldbank. This population involved an extraction of meta-data from data catalogs, represented using the VoID and DCAT vocabularies. The meta-data includes information about the accessibility of the actual data sources, which is used in a second step to load and populate the data sources locally. Every data source is stored in a triple store and accessible via a dedicated SPARQL endpoint. Overall, we have a total of more than 10000 data sources available.

Source Exploration and Selection. A typical search process starts with looking for "the right" sources. That is, a user begins with exploration of the data source space. For instance, she may issue a keyword query "gross domestic product", yielding sources with matching words in their meta-data. If this query does not lead to sources suitable for her information need, a faceted search interface or a tag-cloud may be used. For instance, she refines her sources via entity "Germany" in a faceted search, Fig.3.

Once the user discovered a source of interest, its structure as well as entity information is shown. For example, a textual source description for `GDP (current US$)` is given in Fig. 4. More details about source `GDP (current US$)` is given via an entity and schema overview, respectively (Fig.5-a/b). Note, entities used here have been extracted by our approach, and are visualized by means of a map. Using these rich source descriptions, a user can get to know the data and data sources before issuing queries.

Further, for every source a ranked list of contextualization sources is given. For `GDP (current US$)`, e.g., source `GDP at Market Prices` is recommended, Fig.5-c. This way, the user is guided from one source of interest to another. At any point, she may *select a particular source for querying*. Eventually, she



GDP (current US$)

Description

GDP at purchaser's prices is the sum of gross value added by all resident producers in the economy plus any product taxes and minus any subsidies not included in the value of the products. It is calculated without making deductions for depreciation of fabricated assets or for depletion and degradation of natural resources. Data are in current U.S. dollars. Dollar figures for GDP are converted from domestic currencies using single year official exchange rates. For a few countries where the official exchange rate does not reflect the rate effectively applied to actual foreign exchange transactions, an alternative conversion factor is used.

Details

Property	Value
Title	GDP (current US$)
Creator	j
Triples	*(undefined)*
Created	2012-11-21T10:28:17Z
Date of first Observation	1960-01-01
Date of last Observation	2010-01-01
License	http://creativecommons.org/publicdomain/zero/1.0/
Publisher	data.worldbank.org
Language	English
Timesteps	Annual

Fig. 4: Textual information for source `GDP (current US$)`.

not only knows her relevant sources, but has also gained first insights into data schema and entities.

Processing Queries over Selected Sources. In this second component, we provide means for issuing and processing queries over multiple (previously selected) sources. Say, a user has chosen `GDP (current US$)` as well as its contextualization source `GDP at Market Prices` (Fig. 5-d). Due to her previous exploration, she knows that the former provides the German GDP from 2000 - 2010, while the second one features GDP from years 2011 and 2012 in Germany.

Knowing data sources that contain the desired data, the user may simply add them to the federation by clicking on the corresponding button. The federation can then be queried transparently, i.e., as if the data was physically integrated in a single source. Query processing is handled by the FedX engine [15], which enables efficient execution of federated queries over multiple data sources.

Following the example, the user may issue the SPARQL query shown in Fig. 6. Here, the user combined data from the sources `GDP (current US$)` and `GDP at market prices`. That is, while GDP data from 2000 to 2010 was retrieved from source `GDP (current US$)`, GDP information for the years 2011 and 2012 was loaded from the data source `GDP at market prices`.

The results of this query can be visualized using widgets offered by the Information Workbench. For instance, as shown in Fig.7, GDP information for Germany may be depicted as bar chart.

Future Applications of the Contextualization Engine. Besides the current usage of source contextualization, we see further applications in the future. In particular, the learned entity clusters may be used for data source search result visualization, or even for visualization of SPARQL query results. Further, SPARQL results could be ranked based on data source contextualization sources and user inputs for source selection, respectively.

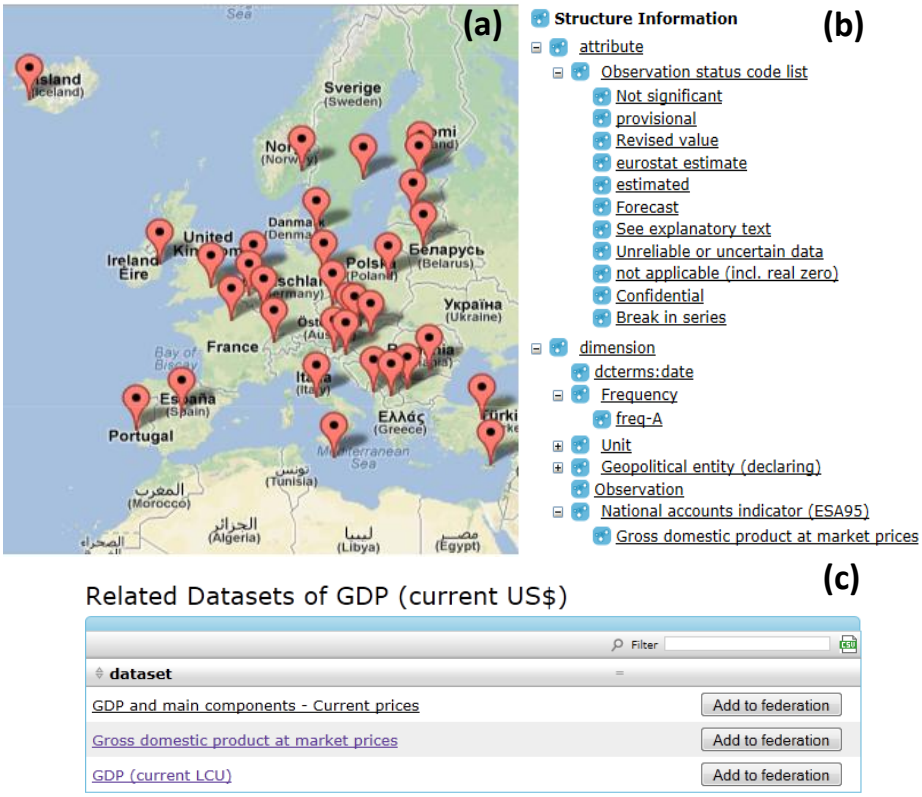


Fig. 5: (a+b) Source information for `GDP (current US$)` based on its entities and schema, respectively. (c) Contextualization sources for `GDP (current US$)`.

```
SELECT ?year ?gdp
WHERE {
{
 ?obs1 rdf:type qb:Observation ;
 wb-property:indicator
      wbi-ci:NY.GDP.MKTP.CN ;
 sdmx-dimension:refArea
      wbi-cc:DE ;
 sdmx-dimension:refPeriod ?year ;
 sdmx-measure:obsValue ?gdp .
 }
```

```
UNION
{
  ?obs2 rdf:type qb:Observation ;
  qb:dataset es-data:tec00001 ;
  es-property:geo es-dic:geo#DE ;
  sdmx-dimension:timePeriod ?year ;
  sdmx-measure:obsValue ?gdp .
  FILTER(?year > "2010-01-01"^^xsd:date)
 }
}
```

Fig. 6: Query asking for Germany's GDP from 2000-2012. Relevant sources, `GDP (current US$)` and `GDP at Market Prices`, were selected during source exploration.

5 Related Work

Closest to our approach is recent work on "finding related tables" on the Web [4]. In fact, our notion of entity and schema complement is adopted from that

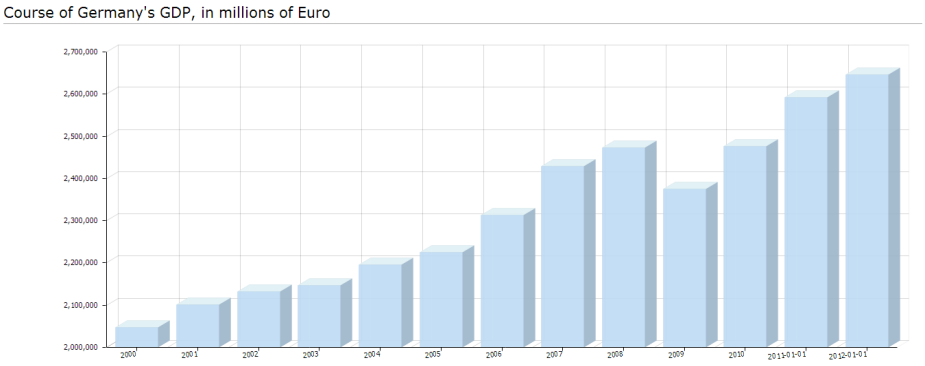


Fig. 7: Visualization of results for query in Fig. 6.

paper. However, [4] focuses on flat entities in Web tables, i.e., entities adhere to a simple and fixed relational structure. In contrast, we consider entities as subgraphs contained in Web data sources. Further, we do not require any kind of "external" information. Most notably, we do not use top-level schema. We argue that relying on such information would limit the applicability of our approach.

Also related are approaches on data source recommendation for source linking, e.g., [13, 14]. Here, given a source D, the task is to find (and rank) other sources sharing same contents, in order to interlink such data sources with D. Existing works commonly exploit keyword search, ontology matching, or user feedback/information. In contrast, our contextualization engine does not depend on user or schema information. Instead, it exploits clusters of entities, learned from the data, and based on structural and literal similarities. Most importantly, however, our goals differs: recommendation for source linking aims at discovering *exactly the same entities* across sources. Instead, we aim at finding either completely new entities, which are somehow related to known/relevant entities (entity complement), or same entities that feature different properties (schema complement), i.e., provide additional information.

Another line of work is concerned with query processing over distributed RDF data, e.g., [5, 8, 9, 11]. During source selection, these approaches frequently exploit indexes or source meta-data, for mapping queries/query fragments to sources. Our approach is complementary, as it enables systems for involve their users during source selection. We outlined such an extension of the traditional search process, as well as its benefits throughout the paper.

Last, data integration for Web search has received much attention. Some works target rewriting queries, e.g., [2, 19], while others rely on keyword search, reducing queries and sources to bags-of-words, e.g., [1, 17]. We target, however, a "fuzzy" form of integration, i.e., we do not give exact mappings of entities, but merely measure whether sources contain entities that might be "somehow" related. That is, our contextualization score indicates whether sources might refer to similar entities, and may provide different data for these entities.

6 Conclusion and Future Work

We presented a novel approach for Web data source contextualization. For this, we adapted well-known techniques from the field of data mining. More precisely, we provide a framework for source contextualization, to be instantiated in an application-specific manner. By means of a real-world use-case and prototype, we show how source contextualization allows for user involvement during source selection. Based on our use-cases and data-portal system, we plan to conduct empirical experiments for validating the effectiveness of our approach. In fact, we aim at a comparison with related work from the field of web-table contextualization, as discussed in Sect. 5.

References

1. R. Blanco, P. Mika, and S. Vigna. Effective and efficient entity search in rdf data. In *ISWC*, 2011.
2. A. Cali, D. Lembo, and R. Rosati. Query rewriting and answering under constraints in data integration systems. In *IJCAI*, 2003.
3. R. Chitta, R. Jin, T. C. Havens, and A. K. Jain. Approximate kernel k-means: solution to large scale kernel clustering. In *SIGKDD*, 2011.
4. A. Das Sarma, L. Fang, N. Gupta, A. Halevy, H. Lee, F. Wu, R. Xin, and C. Yu. Finding related tables. In *SIGMOD*, 2012.
5. O. Görlitz and S. Staab. SPLENDID: SPARQL Endpoint Federation Exploiting VOID Descriptions. In *COLD Workshop*, 2011.
6. G. A. Grimnes, P. Edwards, and A. Preece. Instance based clustering of semantic web resources. In *ESWC*, 2008.
7. P. Haase, M. Schmidt, and A. Schwarte. The information workbench as a self-service platform for linked data applications. In *COLD Workshop*, 2011.
8. A. Harth, K. Hose, M. Karnstedt, A. Polleres, K. Sattler, and J. Umbrich. Data summaries for on-demand queries over linked data. In *WWW*, 2010.
9. O. Hartig, C. Bizer, and J. Freytag. Executing SPARQL queries over the web of linked data. In *ISWC*, 2009.
10. A. K. Jain, M. N. Murty, and P. J. Flynn. Data clustering: a review. *ACM Computing Surveys*, 1999.
11. G. Ladwig and T. Tran. Linked data query processing strategies. In *ISWC*, 2010.
12. U. Lösch, S. Bloehdorn, and A. Rettinger. Graph kernels for rdf data. In *ESWC*, 2012.
13. A. Nikolov and M. d'Aquin. Identifying relevant sources for data linking using a semantic web index. In *Workshop on Linked Data on the Web*, 2011.
14. L. A. P. Paes Leme, G. R. Lopes, B. P. Nunes, M. Casanova, and S. Dietze. Identifying candidate datasets for data interlinking. In *ICWE*, 2013.
15. A. Schwarte, P. Haase, K. Hose, R. Schenkel, and M. Schmidt. FedX: Optimization Techniques for Federated Query Processing on Linked Data. In *ISWC*, 2011.
16. J. Shawe-Taylor and N. Cristianini. *Kernel Methods for Pattern Analysis*. 2004.
17. H. Wang, Q. Liu, T. Penin, L. Fu, L. Zhang, T. Tran, Y. Yu, and Y. Pan. Semplore: A scalable IR approach to search the Web of Data. *JWS*, 2009.
18. R. Zhang and A. Rudnicky. A large scale clustering scheme for kernel k-means. In *Pattern Recognition*, 2002.
19. X. Zhou, J. Gaugaz, W.-T. Balke, and W. Nejdl. Query relaxation using malleable schemas. In *SIGMOD*, 2007.