Entity-based Data Source Contextualization for Searching the Web of Data – Technical Report

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 search on the Web of data, systems have to combine *data from multiple sources*. However, to effectively fulfill user information needs, systems must be able to "look beyond" exactly matching data sources and offer information from additional/contextual sources (*data source contextualization*). For this, users should be *involved in the source selection process – choosing which sources contribute to their search results*. Previous work, however, solely aims at source contextualization for "Web tables", while relying on schema information and simple relational entities. Addressing these shortcomings, we exploit work from the field of data mining and show how to enable *Web data source contextualization*. Based on a real-world use case, we built a prototype contextualization engine, which we integrated in a system for searching the Web of data. We empirically validated the effectiveness of our approach – achieving performance gains of up to 29% over the state-of-the-art.

1 Introduction

The amount of RDF on the Web, such as Linked Data, RDFa and Microformats, is large and rapidly increasing. RDF data contains descriptions of entities, with each description being a set of triples: $\{\langle s, p, o\rangle\}$. A triple associates an entity (subject) s with an object o via a predicate p. A set of triples forms a data graph.

Querying of Distributed Data Sources. RDF is oftentimes highly *distributed*, with each data source comprising one or more RDF graphs, cf. Fig. 1.

Example. In the finance sector, catalogs like Eurostat[1] *or Worldbank*[2] *offer rich financial data about, e.g., GDP, which is spread across many sources.*

However, in order to provide the user with her desired information, a system has to *combine data such distributed data*. 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 (fragment). Previous works selected sources via indexes, e.g., [9, 13], link-traversal, e.g., [10, 13], or by using source meta-data, e.g., [6].

Data Source Contextualization. Existing approaches for source selection aim solely at a mapping of queries/query fragments to sources with *exactly matching* data [6, 9, 10, 13]. In particular, such works do not consider what sources are actually about and how they relate to each other.

Example. Consider a user searching for GDP rates in the EU. A traditional system may discover sources in Eurostat to comprise matching data. However, other sources offer contextual information concerning, e.g., the national debt.

[1] http://ec.europa.eu/eurostat/

[2] http://worldbank.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. They contextualize each other, as they feature varying observation values and properties for Germany's GDP. Src. 2 provides additional information, as it holds the German debt for 2010.

Note, contextual sources are actually not relevant to the user's query, but *relevant to her information need*. Thus, integration of these "additional" sources provides a user with broader results in terms of result dimensions (schema complement) and result entities (entity complement). See our 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, which are used for result computation. In particular, users should be *recommended contextual sources* at each step of the search process. After modifying the selected sources, the results may be reevaluated and/or the query expanded.

Unfortunately, recent work on data source contextualization focuses on Web tables [2], while using top-level schema such as Freebase. Further, the authors restrict data to a simple relational form. We argue that such a solution is not a good fit for the "wild" Web of data. In particular with regard to Linked Data, data sources frequently feature schema-less, heterogeneous entities.

Contributions. We provide the following contributions: (1) Previous work on Web table contextualization suffers from inherent drawbacks. Most notably, it is restricted to fixed-structured relational data and requires external schemata to provide additional information. Omitting these shortcomings, we present an entity-based solution for data source contextualization in the Web of data. Our approach is based on well-known data mining strategies and does not require schema information or data adhering to a particular form. To the best of our knowledge, this is the first work on RDF data source contextualization. (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 publicly available and is currently tested by a pilot customer.[3] (3) We conducted two user studies to empirically validate the effectiveness of the proposed approach: our system outperforms the state-of-the-art with up to 29%.

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

Outline. In Sect. 2, we present our use case. We give preliminaries in Sect. 3, outline the approach in Sect. 4, and present its implementation in Sect. 5. We discuss the evaluation in Sect. 6, related work in Sect. 7, and conclude in Sect. 8.

2 Use Case Scenario

In this section, we introduce a real-world use case to *illustrate challenges w.r.t. data source selection during searching the Web of data.* The scenario is provided by a pilot user in the financial industry and available online.[4]

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 these debts, most countries set up a plan to reduce their public budget deficits. 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 has 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 is "hidden" from the researcher, i.e., 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. *This way, the researcher may discover new sources more easily, as one source of interest links to another – allowing her to explore the space of sources.*

[4] `http://data.fluidops.net/resource/Demo_GDP_Germany`

3 Preliminaries

Data Model. Data in this work is represented as RDF:

Definition 1 (RDF Triple, RDF Graph). *Given a set of URIs $\mathcal{U}$, blank nodes $\mathcal{B}$, and a set of literals $\mathcal{L}$, $t = \langle s, p, o \rangle \in \mathcal{U} \cup \mathcal{B} \times \mathcal{U} \times (\mathcal{U} \cup \mathcal{L} \cup \mathcal{B})$ is a* RDF triple. *A* RDF graph *$\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is defined by vertices $\mathcal{V} = \mathcal{U} \cup \mathcal{L} \cup \mathcal{B}$ and a set of triples as edges $\mathcal{E} = \{\langle s, p, o \rangle\}$.*

A data graph $\mathcal{G}$ is distributed over multiple sources:

Definition 2 (RDF Data Source). *A data source $D_i \in \mathcal{D}$ is a set of n RDF graphs, i.e., $D_i = \{\mathcal{G}_1^i, \ldots, \mathcal{G}_n^i\}$, with $\mathcal{D}$ as set of all sources.*

Notice, the above definition abstracts from the data access, e.g., via HTTP GET requests. In particular, Def. 2 also covers Linked Data sources, see Fig. 1.

Entity Model. Given a data source D_i, an entity e is a subject that is identified with a URI d_e in $\mathcal{G}_j^i$. Entity e is described via a connected subgraph of $\mathcal{G}_j^i$ containing d_e (called $\mathcal{G}_e$). Subgraphs $\mathcal{G}_e$, however, may be defined in different ways [7]. For our work, we used the *concise bound description*, where all triples with subject d_e are comprised in $\mathcal{G}_e$. Further, if an object is a blank node, all triples with that blank node as subject are also included and so on [7].

Example. We have 3 subjects in Fig. 1 and each of them stands for an entity. Every entity description, $\mathcal{G}_e$, is a one-hop graph. For instance, the description for entity `es:data/tec0001` comprises all triples in Src. 1.

Kernel Functions. We compare different entities by comparing their descriptions, $\mathcal{G}_e$. For this, we make use of kernel functions [19]:

Definition 3 (Kernel function). *Let $\kappa : \mathcal{X} \times \mathcal{X} \mapsto \mathbb{R}$ denote a kernel function such that $\kappa(x_1, x_2) = \langle \varphi(x_1), \varphi(x_2) \rangle$, where $\varphi : \mathcal{X} \mapsto \mathcal{H}$ projects a data space $\mathcal{X}$ to a feature space $\mathcal{H}$ and $\langle \cdot, \cdot \rangle$ refers to the scalar product.*

Note, φ is not restricted, i.e., a kernel is constructed without prior knowledge about φ. We will give suitable kernels for entity descriptions, $\mathcal{G}_e$, in Sect. 4.

Clustering. We use k-means [11] as a simple and scalable algorithm for discovering clusters, C_i, of entities in data sources $\mathcal{D}$. It works as follows [11]:

(1) Choose k initial cluster centers, m_i. (2) Based on a dissimilarity function, dis, an indicator function is given as: $\mathbb{1}(e, C_i)$ is 1 if $dis(e, C_i) < dis(e, C_j), \forall j \neq i$ and 0 otherwise. That is, $\mathbb{1}(e, C_i)$ assigns each entity e to its "closest" cluster C_i. (3) Update cluster centers m_i and reassign, if necessary, entities to new clusters. (4) Stop if convergence threshold is reached, e.g., no (or minimal) reassignments occurred. Otherwise go back to (2). A key problem is defining a dissimilarity function for entities in $\mathcal{D}$. Using kernels we may define such a function – as we will show later, cf. Sect. 4.1.

Problem. We address the problem of finding *contextual data sources* for a given source. Contextual sources should: (a) Add *new entities* that refer to the same real world object as given ones (entity complement). (b) Add entities with same URI identifier, but have *new properties* in their description (schema complement). Note, (a) and (b) are not disjoint, i.e., some entities may be new and add additional properties.

Example. In Fig. 1 Src. 3 contextualizes Src. 1 in terms of both, entity as well as schema complement. That is, Src. 3 adds entity `wbi:NY.GDP.MKTP.CN`, while providing new properties and a different GDP value. In fact, even Src. 2 contextualizes Src. 1, as entities in both sources refer to the real world object "Germany" – one source captures the GDP, while the other describes the debt.

4 Entity-based Data Source Contextualisation

Approach. *The intuition behind our approach is simple: if data sources contain similar entities, they are somehow related. In other words, we rely on (clusters of) entities to capture the "latent" semantics of data sources.*

More precisely, we start by extracting entities from given data sources. In a second step, we apply clustering techniques, to mine for entity groups. Notice, for the k-means clustering we employ a kernel function as similarity measure. This way, we abstract from the actual entity representation, i.e., RDF graphs, and use a high-dimensional space (feature space) to increase data comparability/separability [19]. Last, we rely on entity clusters for relating data sources to each other, and compute a contextualization score based on these clusters. Note, only the last step is done online – all other steps/computations are offline.

Discussion. We argue our approach to allow for some key advantages: (1) We decouple representation of source content and source similarity, by relying on entities for capturing the overall source semantics. (2) Our solution is highly flexible as it allows to "plug-in" application-specific entity definitions/extraction strategies, entity similarity measures, and contextualization heuristics. That is, we solely require an entity to be described as a subgraph, $\mathcal{G}_e$, contained in its data source D. Further, similarity measures may be based on any valid kernel function. Last, various heuristics proposed in [2] could be adapted for our approach. (3) Exploiting clusters of entities allows for a scalable and maintainable approach, as we will outline in the following.

4.1 Related Entities

In order to compare data sources, we first compare their entities with each other. That is, we extract entities, measure similarity between them, and finally cluster them (Fig. 2). All of these procedures are offline.

Entity Similarity. We start by extracting entities from each source $D_i \in \mathcal{D}$: First, for scalability reasons, we go over all subjects in RDF graphs in D_i, and collect an entity sample, with every entity e having the same probability of being selected. Then, we crawl the concise bound description [7] for each entity e in the sample. For cleaning $\mathcal{G}_e$, we apply standard data cleansing strategies to fix, e.g., missing or wrong data types. Having extracted entities, we define a similarity measure relating pairs of entities. For this, we use two kinds of kernels: (1) kernels for structural similarities κ^s, and (2) those for literal similarities κ^l.

With regard to the former, we measure structural "overlaps" between entity descriptions, $\mathcal{G}'_e$ and $\mathcal{G}''_e$, using graph intersections [16]:

Definition 4 (Graph Intersection $\mathcal{G}' \cap \mathcal{G}''$). *Let the intersection between two graphs, denoted as $\bar{\mathcal{G}} = \mathcal{G}' \cap \mathcal{G}''$, be given by $\bar{\mathcal{V}} = \mathcal{V}' \cap \mathcal{V}''$, and $\bar{\mathcal{E}} = \mathcal{E}' \cap \mathcal{E}''$.*

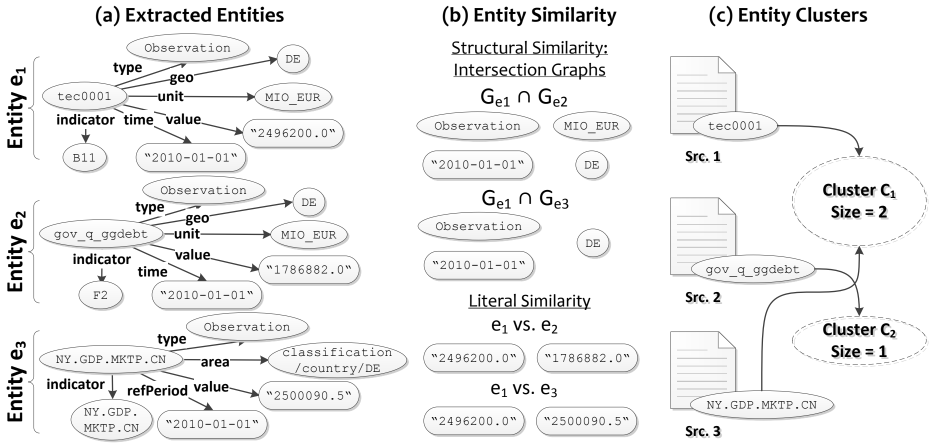


Fig. 2: (a) Extracted entities e_1, e_2, and e_3 from Fig. 1. (b) Structural similarities as intersection graphs $\mathcal{G}_{e1} \cap \mathcal{G}_{e2}$, and $\mathcal{G}_{e1} \cap \mathcal{G}_{e3}$. Literal similarities for entity e_1 vs. e_2, and e_1 vs. e_3. Note, exact matching literals are omitted, as they are covered already by the intersection graphs. Comparison of e_2 vs. e_3 is omitted due to space reasons. (c) Entities grouped in $k = 2$ clusters.

We aim at connected structures in $\mathcal{G}'_e \cap \mathcal{G}''_e$. Thus, we define a path [16]:

Definition 5 (Path). *Let a path in a graph $\mathcal{G}$ be defined as a sequence of vertices and triples $v_1, \langle v_1, p_1, v_2 \rangle, v_2, \ldots, v_n$, with $\langle v_i, p_i, v_{i+1} \rangle \in \mathcal{E}$, having no cycles. The path length is given by the number of contained triples. The set of all paths, up to length l, in $\mathcal{G}$ is denoted as $path_l(\mathcal{G})$.*

The corresponding path kernel is [16]:

Definition 6 (Path Kernel κ^s). *A path kernel is $\kappa^s_{l,\lambda}(\mathcal{G}_1, \mathcal{G}_2) = \sum_{i=1}^{l} \lambda^i \mid \{p \mid p \in paths_i(\mathcal{G}_1 \cap \mathcal{G}_2)\} \mid$, with $\lambda > 0$ as discount factor for path length.*

Note, [16] introduced further kernels, however, we found path kernels to be simple, and perform well in our experiments, cf. Sect. 6.

Example. Extracted entities from sources in Fig. 1 are given in Fig. 2-a. In Fig. 2-b, we compare the structure of `tec0001` (short: e_1) and `gov_q_ggdebt` (short: e_2). For this, we compute an intersection: $\mathcal{G}_{e1} \cap \mathcal{G}_{e2}$. This yields a set of 4 paths, each with length 0. The unnormalized kernel value is $\lambda^0 \cdot 4$.

For literal similarities, one can use different kernels κ^l on, e.g., strings or numbers [19]. The string subsequence kernel, κ^l_s, may be defined as [15]:

Definition 7 (String Subsequence Kernel κ^l_s). *Let Σ denote a vocabulary for strings, with each string s as finite sequence of characters in Σ. Let $s[i:j]$ denote a substring $s_i, \ldots, s_j$ of s. Further, let u be a subsequence of s, if indices $i = (i_1, \ldots, i_{|u|})$ exist with $1 \leq i_1 \leq i_{|u|} \leq |s|$ such that $u = s[i]$. The length $l(i)$ of subsequence u is $i_{|u|} - i_1 + 1$. Then, a kernel function κ^l_s is defined as sum over all common, weighted subsequences for strings s, t: $\kappa^l_s(s,t) = \sum_u \sum_{i:u=s[i]} \sum_{j:u=t[j]} \lambda^l(i) \lambda^l(j)$, with λ as decay factor.*

Example. For instance, strings "MI" and "MIO_EUR" share a common subsequence "MI" with $i = (1, 2)$. Thus, the unnormalized kernel is $\lambda^2 + \lambda^2$.

With regard to numerical values, a simple kernel is given by [19]:

Definition 8 (Real Numbers Kernel κ^l_n). *Let x and y denote real numbers, i.e., $x, y \in \mathbb{R}$, a simple kernel function is: $\kappa^l_n(x, y) = xy$.*

As literal kernels, κ^l, are only defined for two literals, we sample over every possible literal pair (with the same data type) for two given entities, and aggregate the needed kernels for each pair. Finally, we aggregate structure kernel, κ^s, and literal kernels, κ^l, resulting in one single kernel [19]:

$$\kappa(e', e'') = \kappa^s(\mathcal{G}_{e'}, \mathcal{G}_{e''}) \bigoplus_{o_1, o_2 \,\in\, sample(\mathcal{G}_{e'}, \mathcal{G}_{e''})} \kappa^l(o_1, o_2)$$

We use a weighted summation as aggregation $\oplus$ (we obtained weights experimentally). The dissimilarity between e' and e'' is given as Euclidean distance [20]:

$$dis^2(e', e'') \coloneqq \|\varphi(e') - \varphi(e'')\|^2 = \kappa(e', e') - 2\kappa(e', e'') + \kappa(e'', e'')$$

Entity Clustering. Using this dissimilarity measure, we may learn clusters of entities. Notice, our algorithm does not separate the input data (graphs $\mathcal{G}_e$), but instead its representation in the feature space. Based on [20], cluster center m_i in the feature space is: $m_i = \frac{1}{|C_i|} \sum \mathbb{1}(\varphi(e), C_i)\varphi(e)$. Distance between a projected entity $\varphi(e)$ and m_i is given by [20]:

$$dis^2(\varphi(e), m_i) = \|\varphi(e) - m_i\|^2 = \kappa(e, e) + f(e, C_i) + g(C_i), \text{ with}$$

$$f(e, C_i) \coloneqq -\frac{2}{|C_i|} \sum_j \mathbb{1}(\varphi(e_j), C_i)\kappa(e, e_j)$$

$$g(C_i) \coloneqq \frac{1}{|C_i|^2} \sum_j \sum_l \mathbb{1}(\varphi(e_j), C_i)\mathbb{1}(\varphi(e_l), C_i)\kappa(e_j, e_l)$$

Now, k-means can be applied as introduced in Sect. 3.

Example. In Fig. 2-c, we found two entity clusters. Here, structural similarity was higher for entities e_1 vs. e_2 than for e_1 vs. e_3. However, numerical similarity between e_1 vs. e_3 was stronger: "2496200.0" was closer to "2500090.5" as "1786882". As we weighted literal similarity to be more important than structural similarity, this leads to e_1 and e_3 forming one cluster. In fact, such a clustering is intuitive, as e_1 and e_3 is about GDP, while e_2 is concerned with debt.

4.2 Related Data Sources

Given a data source D', we score its contextualization w.r.t. another source D'', using entities contained in D' and D''. Note, in contrast to previous work [2], we do not rely on any kind of "external" information, such as top-level schema. Instead, we *solely exploit semantics as captured by entities.*

Contextualisation Score. Similar to [2], we compute two scores, $ec(D'' \mid D')$ and $sc(D'' \mid D')$, for a data source D'', given another source D'. The former is an indicator for the entity complement of D'' w.r.t. D'. That is, *how many*

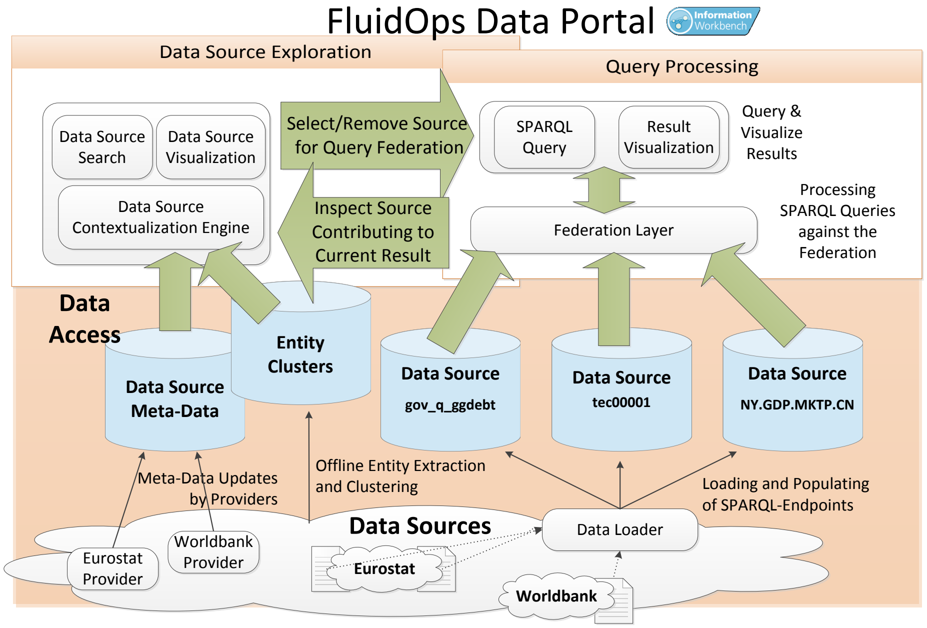

FluidOps Data Portal
Information Workbench
Data Source Exploration
Query Processing
Data Source Search
Data Source Visualization
Data Source Contextualization Engine
Select/Remove Source for Query Federation
Inspect Source Contributing to Current Result
SPARQL Query
Result Visualization
Query & Visualize Results
Federation Layer
Processing SPARQL Queries against the Federation
Data Access
Entity Clusters
Data Source Meta-Data
Data Source gov_q_ggdebt
Data Source tec00001
Data Source NY.GDP.MKTP.CN
Meta-Data Updates by Providers
Offline Entity Extraction and Clustering
Loading and Populating of SPARQL-Endpoints
Data Sources
Data Loader
Eurostat Provider
Worldbank Provider
Eurostat
Worldbank

Fig. 3: Our data portal offers two services: source space exploration and query processing. The source contextualization engine is integrated as a key component of the source space exploration. For this, data source meta-data is loaded, and entities are extracted/clustered. For query processing, each data source is mapped to a SPARQL endpoint, from which data is accessed via a data-loader.

new, similar entities does D'' contribute to given entities in D'. The latter score judges *how many new "dimensions"* are added by D'', to those present in D' (schema complement). Both scores are aggregated to a *contextualization score* for data source D'' given D'.

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

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

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.

Considering the schema complement score $sc : \mathcal{D} \times \mathcal{D} \mapsto [0, 1]$, we aim to add new dimensions (properties) to those already present. Thus, sc is given as:

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

with *props* as function that projects 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 simple, weighted summation:

$$cs(D'' \mid D') := {}^1\!/_2 \cdot ec(D'' \mid D') + {}^1\!/_2 \cdot sc(D'' \mid D')$$

Example. Assuming Src. 1 is given, $cs(D_2 \mid D_1) = cs(D_3 \mid D_1) = {}^1\!/_2$*, because* $ec(D_3 \mid D_1) = 1$ *and* $sc(D_3 \mid D_1) = 0$ *(the other way around for* $D_2 \mid D_1$*). These scores are meaningful, because* D_2 *adds additional properties (debt), while* D_3 *complements* D_1 *with entities (GDP). See also Fig. 2-c.*

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 $cluster()$) and cluster size $|C|$ is needed. Further, measure sc only requires an additional mapping of clusters to “contained” properties (function $props()$). All necessary statistics are easily kept in memory.

With regard to the 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 [1, 5, 20].

5 Searching the Web of Data via Source Contextualization

Based on our real-world use case (Sect. 2), we will now show how a source contextualization engine may be *integrated in a real-world query processing system* for the Web of data.

Overview. Towards an active *involvement of users in the source selection process*, we implemented a data portal, which offers a *data source space exploration* as well as a *distributed query processing* service.

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.

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 other relevant sources may be found via contextualization.

The data portal is based on the Information Workbench [8] and a running prototype is available.[5] Following our use-case (Sect. 2), we populated the system 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 data sources, which is used to load and populate the data sources locally. Every data source is stored in a triple store and is accessed via a SPARQL endpoint. See Fig. 3 for an overview.

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

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

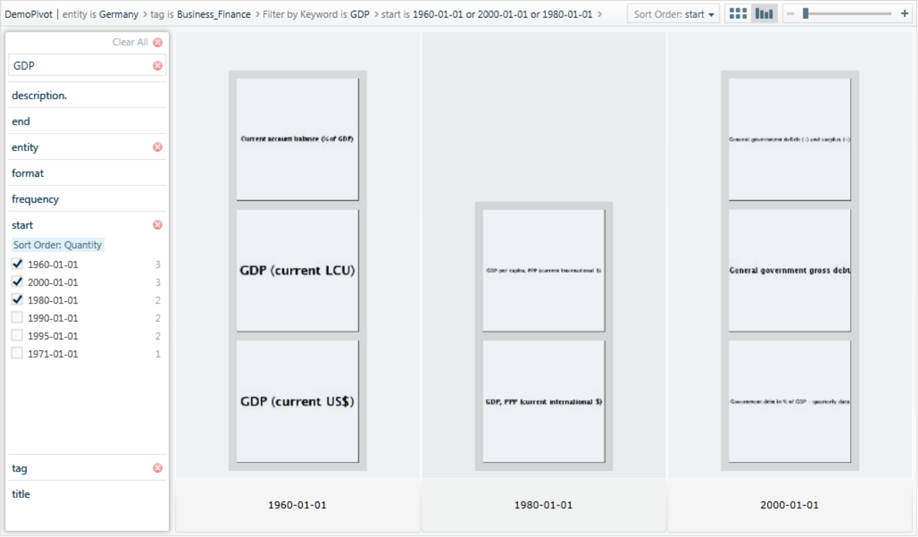

DemoPivot | entity is Germany > tag is Business_Finance > Filter by Keyword is GDP > start is 1960-01-01 or 2000-01-01 or 1980-01-01 >
Sort Order: start
Clear All
GDP
description.
end
entity
format
frequency
start
Sort Order: Quantity
☑ 1960-01-01 3
☑ 2000-01-01 3
☑ 1980-01-01 2
□ 1990-01-01 2
□ 1995-01-01 2
□ 1971-01-01 1
tag
title
Current account balance (% of GDP)
GDP (current LCU)
GDP (current US$)
GDP per capita, PPP (current international $)
GDP, PPP (current international $)
General government deficit (-) and surplus (+)
General government gross debt
Government debt in % of GDP - quarterly data
1960-01-01
1980-01-01
2000-01-01

Fig. 4: Faceted search exploration of data sources.

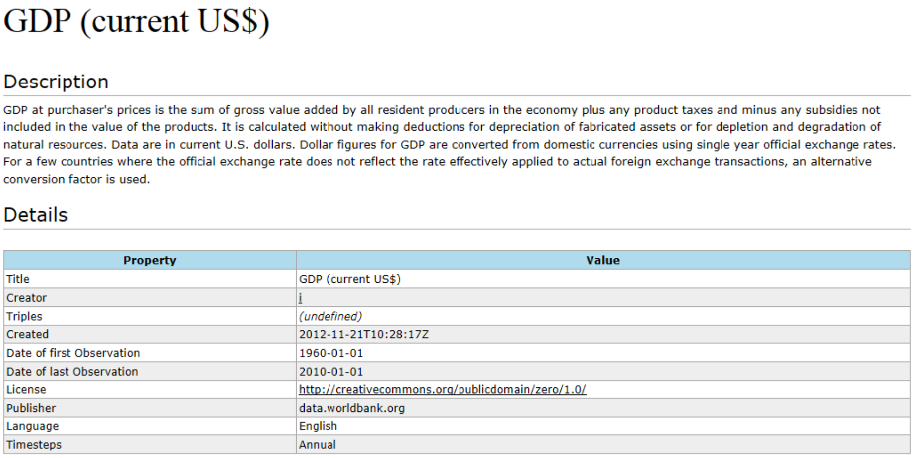
product”, yielding sources with matching words in their meta-data. If this query does not lead to sources suitable for her information need, a faceted search interface or a tag-cloud may be used. For instance, she refines her sources via entity “Germany” in a faceted search, cf. Fig. 4.

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$)` is given in Fig. 5. More details about source `GDP (current US$)` is given via an entity and schema overview, cf. Fig.6-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, see Fig.6-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 not only knows her relevant sources, but has also gained first insights into data schema and entities.

Processing Queries over Selected Sources. Via this second service (Fig. 3), we provide means for query processing over multiple (previously selected) sources. Say, a user has chosen `GDP (current US$)` as well as its contextualization source `GDP at Market Prices` (Fig. 6-c). 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



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	i
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. 5: Schema and textual information about source `GDP (current US$)`.

in a single source. Query processing is handled by the FedX engine [18], which enables efficient execution of federated queries over multiple data sources.

Following the example, a user may issue the SPARQL query given in Fig. 7. Here, she 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`.

Query results may be visualized using widgets offered by the Information Workbench. For instance, GDP information for Germany may be depicted as bar chart – as shown in Fig. 7.

6 Evaluation

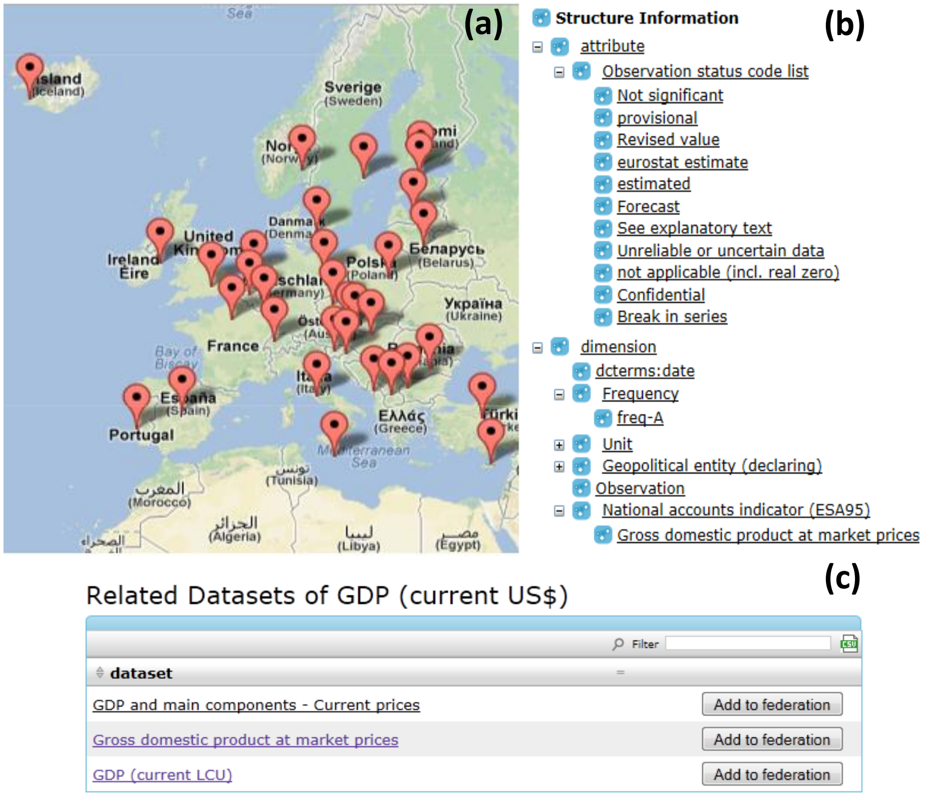
We now discuss evaluation results to analyze the *effectiveness* of our approach. We conducted two experiments: (1) Effectiveness w.r.t. a gold standard, thereby measuring the *accuracy of the contextualization score*. (2) Effectiveness w.r.t. data source search result augmentation. In other words, we ask: *How useful are top-ranked contextualization sources in terms of additional information?*

Participants. Experiment (1) and (2) were performed by two different groups, each comprising 14 users. Most participants had an IT background and/or were CS students. Three users came from the finance sector. The users vary in age, 22 to 37 years and gender. All experiments were unsupervised.

Data Sources and Clustering. Following our use case in Sect. 2, we employed Linked Data sources from Eurostat[6] and Worldbank[7]. Both provide a large set of data sources comprising statistical information: Eurostat holds $5K$ sources ($8000M$ triples), while Worldbank has $76K$ sources ($167M$ triples). Note,

[6] `http://eurostat.linked-statistics.org/`

[7] `http://worldbank.270a.info`



Structure Information

- attribute
  - Observation status code list
    - Not significant
    - provisional
    - Revised value
    - eurostat estimate
    - estimated
    - Forecast
    - See explanatory text
    - Unreliable or uncertain data
    - not applicable (incl. real zero)
    - Confidential
    - Break in series
- dimension
  - dcterms:date
  - Frequency
    - freq-A
  - Unit
  - Geopolitical entity (declaring)
  - Observation
  - National accounts indicator (ESA95)
    - Gross domestic product at market prices

Related Datasets of GDP (current US$)

Filter

dataset	
GDP and main components - Current prices	Add to federation
Gross domestic product at market prices	Add to federation
GDP (current LCU)	Add to federation

Fig. 6: (a+b) Source information for `GDP (current US$)` based on its entities and schema. (c) Contextualization sources for `GDP (current US$)`.

for Eurostat we simply used instances of `void:Dataset` as sources and for Worldbank we considered available dumps as sources. Further, sources varied strongly in their size and # entities. While small sources contained ≤ 10 entities, large sources featured $\geq 1K$ entities. For extracting/sampling of entities, we restricted attention to instances of `qb:Observation`. We employed the k-means algorithm with $k = 30K$ (chosen based on experiments with different values for k) and initiated the cluster centers via random entities.

Queries. Based on evaluation queries in [2], domain experts constructed 15 queries. A query listing is depicted in Table 1. This query load was designed to be "broad" w.r.t. the following dimensions: (1) We cover a wide range of topics, which may be answered by Eurostat and Worldbank. (2) "Best" results for each query are achieved by using multiple sources from both datasets. (3) # Sources and # entities relevant for a particular query vary.

Systems. We implemented entity-based source contextualization (`EC`) as described in Sect. 4. In particular, we made use of three kernels for capturing entity similarity: a path kernel, a substring kernel, and a numerical kernel [16, 19]. As

```
SELECT ?year ?gdp
WHERE {                                    UNION
  {                                         {
   ?obs1 a qb:Observation;                   ?obs2 a qb:Observation;
   wb-property:indicator                     qb:dataset es-data:tec00001;
         wbi-ci:NY.GDP.MKTP.CN;              es-property:geo es-dic:geo#DE;
   sdmx-dimension:refArea                    sdmx-dimension:timePeriod ?year;
         wbi-cc:DE;                          sdmx-measure:obsValue ?gdp.
   sdmx-dimension:refPeriod ?year;           FILTER(?year > "2010-01-01"^^xs:date)
   sdmx-measure:obsValue ?gdp.              }
  }                                        }
```

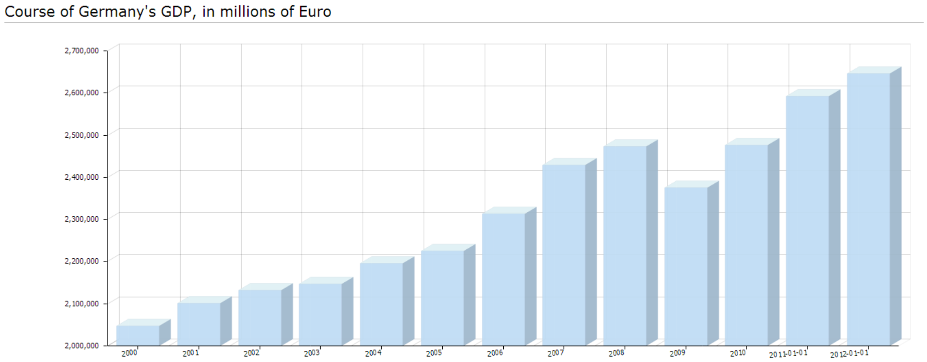


Fig. 7: Query and result visualization for Germany's GDP from 2000-2012. Data sources were selected during source exploration, see Fig. 6.

baselines we used two approaches: a schema-based contextualization SC [2] and a keyword-based contextualization KC. SC maps entities in data sources to top-level schemata and finds complementary sources based on schema as well as label similarities of mapped entities. More precisely, the baseline is split in two approaches: SC$_{ec}$ and SC$_{sc}$. SC$_{ec}$ aims at entity complements, i.e., data sources with similar schema, but different entities. SC$_{sc}$ targets data sources having complementary schema, but holding the same entities. Last, the KC approach treats every data source as a bag-of-words and judges the relevance of a source w.r.t. a query by the number and quality of its matching keywords.

6.1 Effectiveness of Contextualisation Score

Gold Standard. We calculated a simple gold standard that ranks pairs of sources based on their contextualization. That is, for each query in Table 1, we randomly selected one "given" source and 5 "additional" sources – all of which contained the query keywords. Then, 14 users were presented that given source and had to rank how well each of the 5 additional sources contextualizes it (scale on 0 - 5, where higher is better). To judge the sources' contents, users were given a schema description and a short source extract. We aggregated the user rankings, thereby obtaining a gold standard ranking.

Metric. We applied the footrule distance [12] as rank-distance indicator, which measures the accuracy of the evaluation systems vs. gold standard. Footrule distance is: $\frac{1}{k}\sum_{i=1,\ldots,k}|rank_a(i) - rank(i)|$, with $rank_a(i)$ and $rank(i)$ as approximated and gold standard rank for the "additional" source D_i.

	Keywords	*#Entities*	*#Sources*		*Keywords*	*#Entities*	*#Sources*
Q1	country debt	3,155	1,981	Q9	international trade	2,444	1,085
Q2	country gdp	4,265	1,969	Q10	national trade	2,971	1,199
Q3	country population	4,076	3,868	Q11	price indices	10,490	4930
Q4	energy consumption	5,194	651	Q12	prison population	5,135	629
Q5	fish species	2,323	157	Q13	research expenditure	5,287	201
Q6	fish areas	6,664	316	Q14	school leavers	2,722	637
Q7	forest area	3,141	707	Q15	unemployment rate	2,914	266
Q8	gas emissions	5,470	722	$\sum$		66,251	19,318

Table 1: Queries with their number of relevant entities and sources.

Results. Fig. 8-a/d give an overview over results of EC and SC. Note, we excluded the KC baseline, because it was too simplistic. That is, reducing sources to bags-of-words, one may only intersect these bags for two given sources. This yielded, however, no meaningful contextualization scores/ranking.

We noticed EC to lead to a good and stable performance over all queries (Fig. 8-a) as well as data source sizes (Fig. 8-d). Overall, EC could outperform SC$_{sc}$ by 5.7% and SC$_{ec}$ by 29%. Further, as shown in Fig. 8-d, while we observed a decrease in rank distance for all systems in source and entity sample size, EC yielded the best results. We explain these results with our fine-grained source semantics captured by entity clusters. Note, most of our employed contextualization heuristics are very similar to those from SC. However, we observed SC performance to strongly vary with the quality of its schema mappings. Given an accurate classification of entities contained in a particular source, SC was able to effectively relate that source with others. However, if entities were mapped to "wrong" concepts, it greatly affected computed scores. In contrast, our approach relied on clusters learned from instance data, thereby achieving a "more reliable" mapping from sources to their semantics (clusters).

On the other hand, we observed EC to result in equal or, for 2 outlier queries with many "small" sources, worse performance than SC$_{sc}$ (cf. Fig. 8-d). In particular, given query Q2, SC$_{sc}$ could achieve a better ranking distance by 20%. We explain such problematic queries with our simplistic entity sampling. Given small sources, only few entities were included in a sample, which unfortunately pointed to "misleading" clusters. Note, we currently only use a uniform random sample for selecting entities from sources. We expect better results with more refined techniques for discovering important entities for a particular source.

Last, we observed SC$_{ec}$ to lead to much less accurate rankings as SC$_{sc}$ and EC. This is due to exact matching of entity labels: SC$_{ec}$ did not capture common substrings (as EC did), but solely relied on a boolean similarity matching.

6.2 Effectiveness of Augmented Data Source Results

Metric. Following [2], we employ a *reordering of data source search results* as metric: (1) We obtain top 100 data sources via the KC baseline for every query. Here, the order is determined solely by number and quality of keyword hits in the sources' bags-of-words. (2) Via SC and EC we reorder the first 10 results:

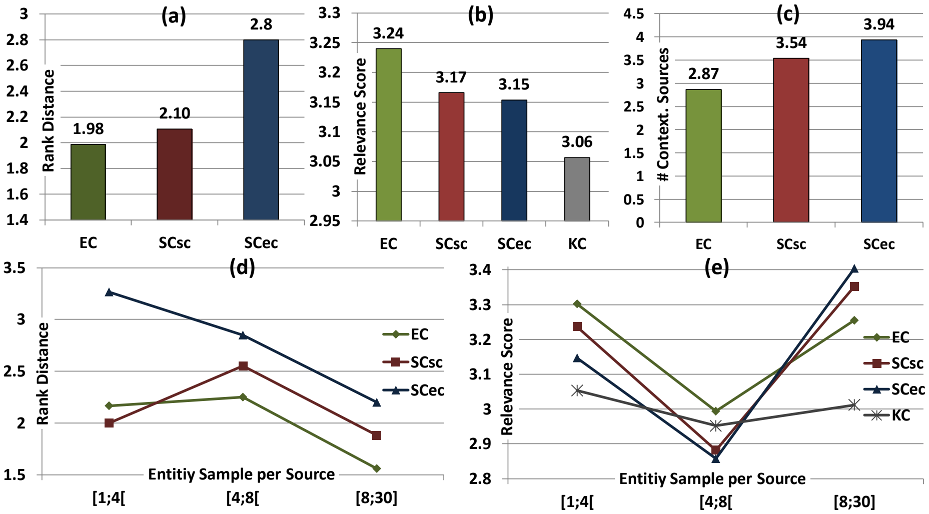


Fig. 8: (a) Exp-1: Rank distance as average over all queries. (b) Exp-2: Relevance score per source, as average over all queries. (c) Exp-2: # Sources for contextualization per source, as average over all queries. (d) Exp-1: Rank distance average over all queries vs. average # sampled entities per source. (e) Exp-2: Relevance score as average over all queries vs. average # sampled entities per source.

For each data source D_i in the top-10 results, we search the top-100 sources for a D_j that contextualizes D_i best. If we find a D_j with contextualization score higher than a threshold, we move D_j after D_i in the top-10 results.

This reordering procedure yields three different top-10 source rankings: one via KC and two (reordered ones) from SC/EC. For every top-10 ranking, users provided a relevance feedback for each source w.r.t. a given query, using a scale 0 - 5 (higher means "more relevant"). For this, users had a schema description and short source extract for every source. Last, we aggregated these relevancy judgments for each top-10 ranking and query.

Results. User relevance scores are depicted in Fig. 8-b/e. Overall, EC yielded an improved ranking, i.e., it ranked more relevant sources w.r.t. a given query. More precisely, EC outperforms SC_{sc} with 2.5%, SC_{ec} with 2.8% and KC with 6.2%, cf. Fig. 8-b. Furthermore, our EC approach led to stable results over varying source sizes (Fig. 8-e). However, for 2 queries having large sources in their top-100 results, we observed misleading a clustering of sampled entities. Such associated clusters, in turn, led to a bad contextualization, see Fig. 8-e.

Notice that all result reorderings, either by EC or by SC, yielded an improvement (up to 6.2%) over the plain KC baseline, cf. Fig. 8-b/e. These findings confirm results reported in [2] and clearly show the potential of data source contextualization techniques. In other words, such observations demonstrate the usefulness of contextualization – *participants wanted to have additional/contextualized sources, as provided by EC or SC.*

In Fig. 8-c we show the # sources returned for contextualization of the "original" top-10 sources (obtained via KC). EC returned 19% less contextualization sources than $\texttt{SC}_{sc}$ and 28% less than $\texttt{SC}_{ec}$. Unfortunately, as contextualization is a "fuzzy" criteria, we could not determine the total number of relevant sources to be discovered in the top-100. Thus, no recall factor can be computed. However, combined with relevance scores in Fig. 8-b, # sources gives a precision-like indicator. That is, we can compute the average relevance gain per contextualization source: 1.13 EC, 0.89 $\texttt{SC}_{sc}$, and 0.77 $\texttt{SC}_{sc}$, see Fig. 8-b/c. Again, we explain the good "precision" of EC with the fine-grained clustering, providing better and more accurate data source semantics.

Last, it is important to note that, while Exp. 1 (Sect. 6.1) and Exp. 2 (Sect. 6.2) differ greatly in terms of their setting, our observations were still similar: EC outperformed both baselines due to its fine-grained entity clusters.

7 Related Work

Closest to our approach is work on contextual Web tables [2]. However, [2] focuses on flat entities in tables, i.e., entities adhere to a simple and fixed relational structure. In contrast, we consider entities to be subgraphs contained in data sources. Further, we do not require any kind of "external" information. Most notably, we do not use top-level schemata.

Another line of work is concerned with query processing over distributed RDF data, e.g., [6, 9, 10, 13]. During source selection, these approaches frequently exploit indexes, link-traversal, or source meta-data, for mapping queries/query fragments to sources. Our approach is *complementary*, as it enables systems to involve 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 (data) search has received much attention [4]. In fact, some works aim at recommendations for sources to be integrated, e.g., [3, 14, 17]. Here, the goal is to give recommendations to identify (integrate) identical entities and schema elements across different data sources.

In contrast, we target a "fuzzy" form of integration, i.e., we do not give exact mappings of entities or schema elements, but merely measure whether or not sources contain entities that might be "somehow" related. In other words, our contextualization score indicates whether sources *might* refer to similar entities and *may* provide contextual information. Furthermore, our approach does not require data sources to adhere to a known schema – instead, we exploit data mining strategies on instance data (entities).

8 Conclusion

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. Further, we empirically validated the effectiveness of our approach via two user studies. In fact, while relying on simplistic scoring heuristics and no top-level schema, we still outperformed state-of-the-art with up to 29%.

References

1. R. Chitta, R. Jin, T. C. Havens, and A. K. Jain. Approximate kernel k-means: solution to large scale kernel clustering. In *SIGKDD*, 2011.
2. A. Das Sarma, L. Fang, N. Gupta, A. Halevy, H. Lee, F. Wu, R. Xin, and C. Yu. Finding related tables. In *SIGMOD*, 2012.
3. H. R. de Oliveira, A. T. Tavares, and B. F. Lóscio. Feedback-based Data Set Recommendation for Building Linked Data Applications. In *I-SEMANTICS*, 2012.
4. A. Doan, A. Y. Halevy, and Z. G. Ives. *Principles of Data Integration.* Morgan Kaufmann, 2012.
5. S. Fausser and F. Schwenker. Clustering large datasets with kernel methods. In *ICPR*, 2012.
6. O. Görlitz and S. Staab. SPLENDID: SPARQL Endpoint Federation Exploiting VOID Descriptions. In *COLD Workshop*, 2011.
7. G. A. Grimnes, P. Edwards, and A. Preece. Instance based clustering of semantic web resources. In *ESWC*, 2008.
8. P. Haase, M. Schmidt, and A. Schwarte. The Information Workbench as a Self-Service Platform for Linked Data Applications. In *COLD Workshop*, 2011.
9. 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.
10. O. Hartig, C. Bizer, and J. Freytag. Executing SPARQL Queries over the Web of Linked Data. In *ISWC*, 2009.
11. A. K. Jain, M. N. Murty, and P. J. Flynn. Data clustering: a review. *ACM Computing Surveys*, 1999.
12. M. Kendall and J. D. Gibbons. *Rank Correlation Methods.* 1990.
13. G. Ladwig and T. Tran. Linked Data Query Processing Strategies. In *ISWC*, 2010.
14. L. A. P. P. Leme, G. R. Lopes, B. P. Nunes, M. A. Casanova, and S. Dietze. Identifying Candidate Datasets for Data Interlinking. In *ICWE*, 2013.
15. H. Lodhi, C. Saunders, J. Shawe-Taylor, N. Cristianini, and C. Watkins. Text classification using string kernels. *J. Mach. Learn. Res.*, 2002.
16. U. Lösch, S. Bloehdorn, and A. Rettinger. Graph kernels for RDF data. In *ESWC*, 2012.
17. A. Nikolov and M. d'Aquin. Identifying Relevant Sources for Data Linking using a Semantic Web Index. In *LDOW Workshop*, 2011.
18. A. Schwarte, P. Haase, K. Hose, R. Schenkel, and M. Schmidt. FedX: Optimization Techniques for Federated Query Processing on Linked Data. In *ISWC*, 2011.
19. J. Shawe-Taylor and N. Cristianini. *Kernel Methods for Pattern Analysis.* 2004.
20. R. Zhang and A. Rudnicky. A large scale clustering scheme for kernel K-Means. In *Pattern Recognition*, 2002.