Revealing the Conceptual Schemas of RDF Datasets

Subhi Issa(✉), Pierre-Henri Paris, Fayçal Hamdi, and Samira Si-Said Cherfi

CEDRIC - Conservatoire National des Arts et Métiers,
292 Rue Saint Martin, Paris, France
{subhi.issa,faycal.hamdi,samira.cherfi}@cnam.fr,
pierre-henri.paris@upmc.fr

Abstract. RDF-based datasets, thanks to their semantic richness, variety and fine granularity, are increasingly used by both researchers and business communities. However, these datasets suffer a lack of completeness as the content evolves continuously and data contributors are loosely constrained by the vocabularies and schemes related to the data sources. Conceptual schemas have long been recognized as a key mechanism for understanding and dealing with complex real-world systems. In the context of the Web of Data and user-generated content, the conceptual schema is implicit. In fact, each data contributor has an implicit personal model that is not known by the other contributors. Consequently, revealing a meaningful conceptual schema is a challenging task that should take into account the data and the intended usage. In this paper, we propose a completeness-based approach for revealing conceptual schemas of RDF data. We combine quality evaluation and data mining approaches to find a conceptual schema for a dataset, this model meets user expectations regarding data completeness constraints. To achieve that, we propose LOD-CM; a web-based completeness demonstrator for linked datasets.

Keywords: Conceptual modeling · Completeness · Model quality · Conceptual schema mining · Schema mining

1 Introduction

Data became a strategic asset in the information-driven world. One of the challenges for companies and researchers is to improve the display and understandability of the data they manage and use.

However, exploiting and using open linked data, even if it is more and more accessible, is not an easy task. Data is often incomplete and lacks metadata. This means that the quality of published data is not as good as we could expect leading to a low added value and a low reliability of the derived conclusions. In [10], the authors believe that existing approaches which describe datasets focus on their statistical aspects rather than on capturing conceptual information.

P. Giorgini and B. Weber (Eds.): CAiSE 2019, LNCS 11483, pp. 312–327, 2019.
https://doi.org/10.1007/978-3-030-21290-2_20

A conceptual schema is an abstraction of a reality that can serve as a vehicle for understanding, communicating, reasoning and adding knowledge about this reality.

In traditional information system development, conceptual modeling is driven by intended usage and needs. In the Web of Data, as in all user-generated content, data is rather use-agnostic [15]. Consequently, data is represented from many individual viewpoints and the overall semantics, although necessary to reasoning of data, is missing. We believe that a conceptual schema that creates an abstract representation upon data would help to overcome the disparity of visions and will reveal the underlying semantics [17]. Let us consider, for instance, that we have a collaboratively built dataset. In this case, the traditional top down vision of a predefined schema is no more applicable. Both data and underlying schema evolve continuously, as data are described by several communities with different views and needs. In this situation, a conceptual schema, defined as an abstract and consensual representation about the reality that is derived from requirements, could not be applied. The challenge is then to find a way to create a suitable conceptual schema having instances as a starting point.

In this paper, we are interested in conceptual modeling of RDF Data (Resource Description Framework) [13]. Our objective is to define an approach for deriving conceptual schemas from existing data. The proposed solution should cope with the essential characteristics of a conceptual schema that are the ability to make an abstraction of relevant aspects from the universe of discourse and the one of meeting user's requirements [19]. The approach we propose in this paper takes into account the two facets; namely the universe of discourse represented by the data sources, and the user's needs represented by the user's decisions during the conceptual schema construction. As the model should express the meaningful state of the considered dataset, we rely on a mining approach leading to taking into consideration the data model from a more frequent combination of properties. The relevancy of properties is handled by integrating a completeness measurement solution that drives the identification of relevant properties [6,9]. To meet user's requirements, we propose to construct the conceptual schema on a *scratch card* manner where the user decides about the parts of the conceptual schema to reveal according to her needs and constraints. The main contributions are:

1. We use a mining approach to infer a model from data, as we consider that no predefined schema exists. The underlying assumption is that the more frequent a schema is, the more representative for the dataset it is.
2. We introduce a novel approach, called *LOD-CM*, for Conceptual Model mining based on quality measures, and, in this paper, on completeness measures as a way to drive the conceptual schema mining process.

The remainder of this paper is organized as follows: Sect. 2 summarizes a related literature on the subject while Sect. 3 details the mining-based approach for RDF data conceptual modeling. This section explains the tight link with the completeness quality criterion. Section 4 presents two use cases of LOD-CM. Finally, Section 5 draws conclusions and future research directions.

2 Related Work

RDF data is described as sets of statements called *triples*. A triple $< s, p, o >$ is a fact where a subject s has a property p, and the property value is the object o. As an example, $<$ *England*, *capital*, *London* $>$ means that London is the capital city of England. Understanding and reasoning about this data requires at least knowledge about its abstract model. Consequently, schema discovery has attracted several researchers originating from several communities. The research directions address objectives such as efficient storage, efficient querying, navigation through data or semantic representation, etc.

Completeness of Linked Data is one of the most important data quality dimension [1]. This dimension is defined as the degree to which all required information is present in a particular dataset [23]. We have to know that a reference schema (or a gold standard) should be available to compare against a given dataset.

In the database community, the question was how to store this kind of data. Levandoski et al. [14] proposed a solution that derives a classical relational scheme from an RDF data source in order to accelerate the processing of queries. In the FlexTable method [22], authors proposed to replace RDF triples by RDF tuples resulting from the unification of a set of triples having the same subject. All these approaches do not target a human readable schema and are more concerned with providing suitable structure for a computer processing of data.

The Semantic Web community is more aware of data semantics through the usage of *ontologies* and *vocabularies*. Several semi-automatic or automatic proposals, mainly based on classification, clustering, and association analysis techniques are proposed. In [21] a statistical approach based on association rules mining allows generating ontologies from RDF data. Other works, such as those presented in [2,12,18], are closer to modeling. Authors propose to derive a data structure using a clustering algorithm. After a manual labeling of clusters representing groups of frequent properties, a schema is derived. These approaches, however, do not consider user's needs and preferences and the derived schema is the result of automatic preprocessing, apart from the labeling task.

In traditional conceptual modeling, models are generally derived from user's requirements. However, with the increasing use of external data sources in information systems, there is a need to apply a bottom-up modeling from instances. This is motivated by the expressiveness and the analysis facilities that conceptual models could provide for such data. Similarly to our bottom-up approach, [16] proposed a conceptual modeling grammar based on the assumption that instances play a major role while human beings try to represent the reality. In [15] authors presented a set of principles for conceptual modeling within structured user-generated content. The authors highlighted the problem of quality in such produced content. They focused on the importance of capturing relevant properties from instances. However, the proposal does not provide an explicit solution for deriving such models. Concerning unstructured data, we can cite [3] where authors addressed the problem of deriving conceptual models based on regular-expression pattern recognition.

Recognizing that conceptual modeling is a powerful tool for data understanding, our proposal addresses the problem of deriving a conceptual schema from RDF data. By exploring instances, our approach integrates a completeness measurement as a quality criterion to ensure the relevancy of the derived schema as data from RDF data sources is the result of a free individual publication effort. The result would be a conceptual schema enriched with completeness values.

3 Conceptual Schemas Derivation

To illustrate our proposed approach, let us consider a user willing to obtain a list of artists with their names and birth places from an RDF data source; To do so, she can write the following SPARQL query[1]:

```
SELECT * WHERE {
      ?actor rdf:type dbo:Actor .
      ?actor foaf:name ?name .
      ?actor dbo:birthPlace ?birthPlace .}
```

Writing such a query is much more difficult in a Linked Open Data (LOD) source context than in a relational database one. In a relational context, the database schema is predefined and the user writing the query is aware of it. In a LOD context, in addition to the fact that the schema does not exist, there is another problem related to data completeness. Actually, the expressed query returns only the list of actors having values for all the properties listed in the query. In our example, only actors having values for both *foaf:name* and *dbo:birthPlace* are included in the result. Knowing that at most 74% of actors have a value for *dbo:birthPlace*, the user should probably appreciate getting this information to add for example `OPTIONAL` to the second pattern of the query and obtain more results. Besides, she would be aware of the fact that the result is complete to a certain degree (i.e. *dbo:birthPlace* is present in only 74% of actors).

To tackle these two problems, we propose an approach that aims to help "revealing" a conceptual schema from a LOD RDF source. This conceptual schema is driven by the user for both its content and completeness quality values. In the context of the Web of Data, most of the datasets published in the Web are described by models called, in the linked data jargon, vocabularies (or ontologies). However, these models are not used in a prescriptive manner. Consequently, a person who publishes data is not constrained by the underlying ontology leading to sparse descriptions of concepts. For example, the category *Actor* from DBpedia has around 532 properties that are not equally relevant.

From these observations, it is clear that checking data (instances) is necessary to infer a relevant model that can be used to guarantee, for example, an expected completeness value. The approach that we propose deals with this issue through an iterative process which infers a conceptual schema complying the expected completeness. Figure 1 gives an overview of this process.

[1] Performed on: http://dbpedia.org/sparql.

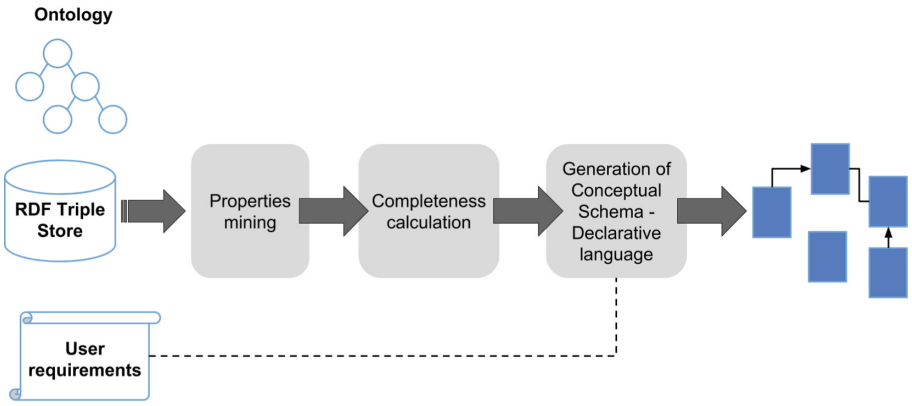

Fig. 1. The *LOD-CM* workflow

The process of inferring a conceptual schema goes through four steps: First, a subset of data that corresponds to the user's scope is extracted from the triple store (cf. Sect. 3.1). This subset is then transformed into transactions and a mining algorithm is applied. In our approach, for efficiency reasons, we chose the well-known FP-growth algorithm [7,8] (any other itemset mining algorithm could obviously be used). From the generated frequent itemsets, only a subset of these frequent itemsets, called "Maximal" [4,5,11], is captured. This choice is motivated by the fact that, on the one hand, we are interested in the *expression* of the frequent pattern and, on the other hand, the number of frequent patterns could be exponential when the transaction vector is very large (cf. Sect. 3.2). $\mathcal{MFP}$ is the set containing all maximal frequent patterns. Each pattern in $\mathcal{MFP}$ is then used to calculate the completeness of each transaction (regarding the presence or absence of the pattern) and, hence, the completeness of the whole dataset regarding this pattern. The final completeness value will be the average of all completeness value calculated for each $\mathcal{MFP}$ pattern (cf. Sect. 3.3). Finally, based on the completeness value and $\mathcal{MFP}$ that guarantees this value, a conceptual schema is generated. The classes, the attributes, and the relations of the model will be tagged with the completeness value (cf. Sect. 3.4). All these steps are integrated in an iterative process in such a way that the user could choose some parts in the generated model to refine. The data corresponding to the parts to refine is then extracted from the triple store, and the same steps are carried out to generate a new model.

In the following subsections, we give a detailed description of each step of the workflow.

3.1 Scope and Completeness Specification

In this step, a subset of data is extracted from the triple store. This subset could correspond to a category or a set of categories such as *Actor*, *Film* or

Organization. This defines what we call the user's scope that corresponds to the categories that the user plans to use in a query, to the information she wants to explore or any kind of usage based on data consumption.

The user is also asked to indicate the degree of the desired completeness. Indeed, properties for a given category are not equally valuated. For example, for the category *Artist*, the property *foaf:name* has a value for 99% of the instances whereas the *dbo:birthPlace* property has a value for at most 74% of the instances. Our approach gives the possibility to express a constraint on the completeness values desired for mined properties and associations. Once the categories are identified, the data is converted into transaction vectors and a mining algorithm is applied to obtain a set of frequent itemsets.

Table 1 illustrates some instances of the *Film* category in the form of triples, taken from DBpedia. Each Category is described by a set of properties (predicates) and an instance of this category could have a value for all the properties or only for a subset of these properties. This subset is called transaction. Table 2 represents the set of transactions constructed from the triples of Table 1.

More formally, let us define a dataset $\mathcal{D}$ to be the triple (C, I_C, P), where C is the set of categories (e.g., *Film*, *Artist*), I_C is the set of instances for categories in C (e.g., *The_Godfather* is an instance of the *Film* category), and $P = \{p_1, p_2, ..., p_n\}$ is the set of properties (e.g $director(Film, Person)$).

Let $\mathcal{T} = \{t_1, t_2, ..., t_m\}$ be a set of transactions with $\forall k, 1 \leq k \leq m : t_k \subseteq P$ be a vector of transactions over P, and $E(t_k)$ be the set of items in transaction t_k. Each transaction is a set of properties used in the description of the instances of the subset $\mathcal{I}' = \{i_1, i_2, ..., i_m\}$ with $\mathcal{I}' \subseteq I_C$ (e.g., properties used to describe the *The_Godfather* instance are: *director* and *musicComposer*). We consider $\mathcal{CP}$ the completeness of $\mathcal{I}'$ against properties used in the description of each of its instances.

3.2 Properties Mining

In the RDF model, all statements having the same subject represent the same category that will be modeled in our approach by a class. The related properties could consequently constitute, either the attributes (properties) of the classes or relationships to other classes, when the property value (the object in the triple $< s, p, o >$) refers to another category. In this step, the objective is to find the properties patterns that are the most shared by the subset of instances extracted from the triple store related to the same category. This set will be then used to calculate a completeness value regarding these patterns.

Let $\mathcal{D}(C, I_C, P)$ be a dataset (stored in the triple store) and $\mathcal{I}'$ be a subset of data (instances) extracted from $\mathcal{D}$ with $\mathcal{I}' \subseteq I_C$. We first initialize $\mathcal{T} = \emptyset$, $\mathcal{MFP} = \emptyset$. For each $i \in \mathcal{I}'$ we generate a transaction t. Indeed, each instance i is related to values (either resources or literals) through a set of properties. Therefore, a transaction t_k of an instance i_k is a set of properties such that $t_k \subseteq P$. Transactions generated for all the instances of $\mathcal{I}'$ are then added to the $\mathcal{T}$ set.

Example 1. Referring Table 1, let $\mathcal{I}'$ be a subset of instances such that: $\mathcal{I}' = \{The\_Godfather, Goodfellas, True\_Lies\}$. The set of transaction $\mathcal{T}$ would be:

$$\mathcal{T} = \{\{director, musicComposer\}, \{director, editing\}, \{director, editing, musicComposer\}\}$$

The objective is then to compute the set of frequent patterns $\mathcal{FP}$ from the transaction vector $\mathcal{T}$.

Table 1. A sample of triples from DBpedia

Subject	Predicate	Object
The_Godfather	director	Francis_Ford_Coppola
The_Godfather	musicComposer	Nino_Rota
Goodfellas	director	Martin_Scorsese
Goodfellas	editing	Thelma_Schoonmaker
True_Lies	director	James_Cameron
True_Lies	editing	Conrad_Buff_IV
True_Lies	musicComposer	Brad_Fiedel

Table 2. Transactions created from triples

Instance	Transaction
The_Godfather	director, musicComposer
Goodfellas	director, editing
True_Lies	director, editing, musicComposer

Definition 1. *(*Pattern*) Let $\mathcal{T}$ be a set of transactions. A pattern $\hat{P}$ is a sequence of properties shared by one or several transactions t in $\mathcal{T}$. It is sometimes called an* itemset*.*

For any pattern $\hat{P}$, let $E(\hat{P})$ be the corresponding set of items (constitutes, in our case, of properties), and $T(\hat{P}) = \{t \in \mathcal{T} \mid E(\hat{P}) \subseteq E(t)\}$ be the corresponding set of transactions. $E(\hat{P})$ designates the *expression* of $\hat{P}$, and $|T(\hat{P})|$ the *support* of $\hat{P}$. A pattern $\hat{P}$ is frequent if $\frac{1}{|\mathcal{T}|}|T(\hat{P})| \geq \xi$, where ξ is a user-specified threshold.

Example 2. Referring Table 2, let $\hat{P} = \{director, musicComposer\}$ and $\xi = 60\%$. $\hat{P}$ is frequent as its relative support (66.7%) is greater than ξ.

To find all the frequent patterns $\mathcal{FP}$, we used, as we mentioned above, the FP-growth itemsets mining algorithm. However, according to the size of the transactions vector, the FP-growth algorithm could generate a very large $\mathcal{FP}$ set. As our objective is to see how a transaction (a description of an instance) is *complete* against a set of properties, we focus on the pattern *expression* (in terms of items it contains) instead of its *support*.

For completeness calculation, we need to select a pattern to serve as a reference schema. This pattern should present a right balance between frequency and expressiveness, therefore we use the concept, called "Maximal" frequent patterns, to find this subset. Thus, to reduce $\mathcal{FP}$, we generate a subset containing only "Maximal" patterns.

Definition 2. *($\mathcal{MFP}$) Let $\hat{P}$ be a frequent pattern. $\hat{P}$ is maximal if none of its proper superset is frequent. We define the set of Maximal Frequent Patterns $\mathcal{MFP}$ as:*

$$\mathcal{MFP} = \{\hat{P} \in \mathcal{FP} \mid \forall \hat{P'} \supsetneq \hat{P} : \frac{|T(\hat{P'})|}{|\mathcal{T}|} < \xi\}$$

Example 3. Referring Table 2, let $\xi = 60\%$ and the set of frequent patterns $\mathcal{FP}$ = {{*director*},{*musicComposer*},{*editing*}, {*director*,*musicComposer*},{*director*,*editing*}}. The $\mathcal{MFP}$ set would be:

$$\mathcal{MFP} = \{\{director, musicComposer\}, \{director, editing\}\}$$

3.3 Completeness Calculation

In this step, we carry out for each transaction a comparison between its corresponding properties and each pattern of the $\mathcal{MFP}$ set (regarding the presence or the absence of the pattern). An average is, therefore, calculated to obtain the completeness of each transaction $t \in \mathcal{T}$. Finally, the completeness of the whole $t \in \mathcal{T}$ will be the average of all the completeness values calculated for each transaction.

Definition 3. *(*Completeness*) Let $\mathcal{I}'$ be a subset of instances, $\mathcal{T}$ the set of transactions constructed from $\mathcal{I}'$, and $\mathcal{MFP}$ a set of maximal frequent pattern. The completeness of $\mathcal{I}'$ corresponds to the completeness of its transaction vector $\mathcal{T}$ obtained by calculating the average of the completeness of $\mathcal{T}$ regarding each pattern in $\mathcal{MFP}$. Therefore, we define the completeness $\mathcal{CP}$ of a subset of instances $\mathcal{I}'$ as follows:*

$$\mathcal{CP}(\mathcal{I}') = \frac{1}{|\mathcal{T}|} \sum_{k=1}^{|\mathcal{T}|} \sum_{j=1}^{|\mathcal{MFP}|} \frac{\delta(E(t_k), \hat{P}_j)}{|\mathcal{MFP}|} \tag{1}$$

$$\textit{such that: } \hat{P}_j \in \mathcal{MFP}, \textit{ and}$$

$$\delta(E(t_k), \hat{P}_j) = \begin{cases} 1 \textit{ if } \hat{P}_j \subset E(t_k) \\ 0 \textit{ otherwise} \end{cases}$$

The Algorithm 1 shows the pseudo-codes for calculating $\mathcal{CP}(\mathcal{I}')$.

Example 4. Let $\xi = 60\%$. The completeness of the subset of instances in Table 1 regarding $\mathcal{MFP} = \{\{director, musicComposer\}, \{director, editing\}\}$ would be:

$$\mathcal{CP}(\mathcal{I}') = (2 * (1/2) + (2/2))/3 = 0.67$$

This value corresponds to the completeness average value for the whole dataset regarding the inferred patterns in $\mathcal{MFP}$.

Algorithm 1. Completeness calculation

Input: $\mathcal{D}$, $\mathcal{I}'$, ξ
Output: $\mathcal{CP}(\mathcal{I}')$
  for each $i \in \mathcal{I}'$ **do**
    $t_i = \lvert p_1\ p_2\ \ldots\ p_n \rvert$
    $\mathcal{T} = \mathcal{T} + t_i$
  ▷ *Properties mining*
  $\mathcal{MFP} = Maximal(FP\text{-}growth(\mathcal{T}, \xi))$
  ▷ *Using equation 1*
  return $CP(\mathcal{I}') = CalculateCompleteness(\mathcal{I}', \mathcal{T}, \mathcal{MFP})$

3.4 Generation of Enriched Conceptual Schemas

In this step, the goal is to generate a conceptual schema enriched with the completeness values calculated in the previous step. The $\mathcal{MFP}$ patterns used to get the completeness values are transformed into a class diagram. Figure 2 illustrates the user's interface of our LOD-CM web service. Using the graphical interface[2], the user can choose her own constraints. The web service permits the

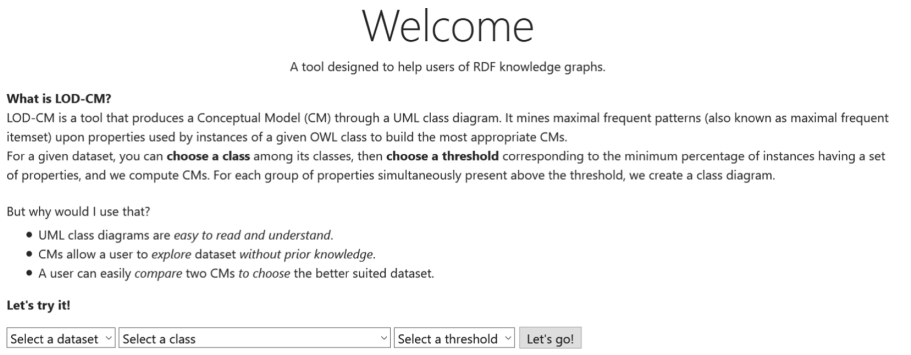

Fig. 2. LOD-CM main interface

[2] http://cedric.cnam.fr/lod-cm.

user to enter the class name in the text box and the user may select the threshold completeness she wants to apply. Currently, our demo supports DBpedia.

After the user selects the class name and desired completeness and clicks Submit button, the algorithm runs to find the attributes, relationships and the missed domains/ranges based on user's constraints.

The structure of the model is constructed regarding the definitions of the patterns properties in the ontology describing the dataset. Figure 3 represents a class diagram derived by our approach, from a set of films extracted from DBpedia.

A First Iteration: In this example, the expectation of the user is a model that guarantees at least 50% of completeness. To generate the model, the first step consists of obtaining the set of properties $p \in \bigcup_{j=1}^{n} E(\hat{P}_j)$, and $\hat{P}_j \in \mathcal{MFP}$ that composes the union of all the $\mathcal{MFP}$, mined from the extracted subset, with a minimum support $\xi = 50\%$. For this example, the set of properties are: {*director, label, name, runtime, starring, type*}, {*director, label, name, starring, type, writer*} and {*label, name, runtime, type, writer*}. OWL distinguishes between two main categories of properties: (i) datatype properties, where the value is a data literal, and (ii) object properties, where the value is an individual (i.e., an other instance with its own type). Each property is considered as an attribute (e.g. name) of the class or a relationship (e.g. director) with another class, depending on the nature of the value. Therefore, according to the nature of the value of each property, it is considered as an attribute of the class or a relationship with another class. Two types of links will be used during generating of conceptual schemas: inheritance and association links. Inheritance link describes the relation between the class and the superclass, and association link describes the relation between two classes and point to the property. A dotted link was added to illustrate that a class has been inferred to complete the relationship. For this reason, based on the approach that has been proposed in [20], we infer missed domains (and/or ranges) of properties. In our example, the names of the classes and the inheritance links between the classes are derived from categories names and organization described in the ontology of the data source DBpedia. We do not derive new names nor new organization of the classes as the conceptual schema should conform to the data used. Indeed, even if the derived conceptual schema is not satisfactory from conceptual modeling principles, it should faithfully reflect the reality of data while taking into account user preferences. Finally, the diagram is enriched by the completeness values calculated in the previous step. These values are associated to each component of the model.

A Second Iteration: A new iteration is triggered when the user chooses to get more details about a part of the model (e.g. the *Artist* class). In this case, a new query is executed on the triple store to extract data corresponding to this part. The previous three steps are then executed in order to generate a new model integrating the new desired details. Figure 4 shows an example that details a

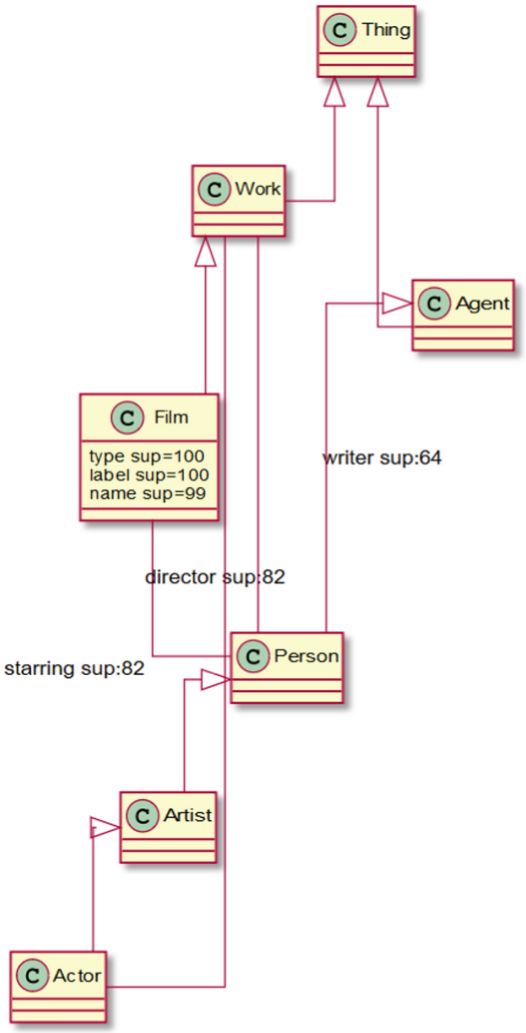

Fig. 3. The *Film* conceptual schema as a class diagram

part of the model from Fig. 3. In this example, a set of classes, relationships and attributes are added to the *Artist* class with corresponding completeness values. This way of revealing the conceptual schema is similar to a magnifying glass that allows the user navigating around a targeted concept, here the *Film* category.

The output of our algorithm is a file written in a declarative language. The file includes the chosen category, the attributes, and the relationships tagged by completeness values. We use PlantUML[3] to transfer this generated file into a picture to illustrate it to the user.

[3] http://plantuml.com/.

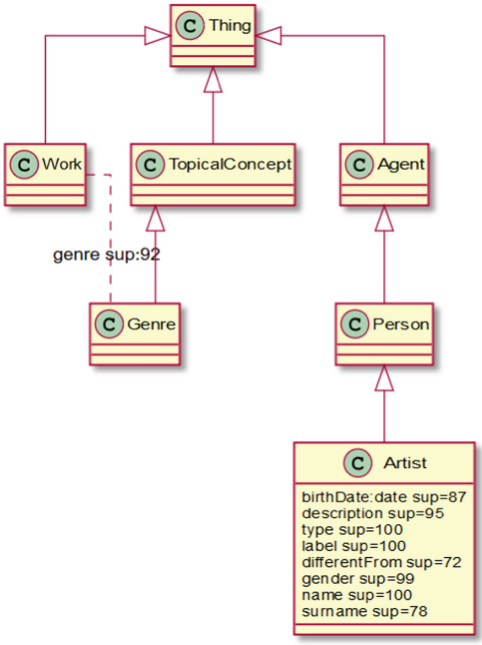

Fig. 4. The *Artist* diagram class

4 Use Cases

The objective of the Linked Open Data cloud is to enable large-scale data integration, so that we can have a contextually relevant Web and find quick answers to a much wider range of questions. LOD-CM is a web-based completeness demonstrator for linked datasets. It is used to display data related to the chosen class of a dataset. In this section, we provide a brief summary of two use cases related to schema discovery based on user's needs. The displayed model could help the user to understand the schema and discover the related properties. LOD-CM only supports DBpedia at the moment.

1. **Class diagram to facilitate data browsing**
   LOD-CM aims basically to visualize the discovered schema based on user's requirements. Suppose a user wants to find the directors and budgets of a list of films. Actually, 82% of films have a director in DBpedia dataset. In addition, only 15% of films have budget value for the same dataset. Only the list of films that have the properties (director and budget) will be displayed (i.e., at most 15% of the films). The outcome model could help the user to present the properties that are related to the chosen class and that are greater than a specified threshold. Besides, it illustrates the relation between the concerned classes such as the classes *Person* and *Film* that are linked

by the property *director*. Furthermore, the model illustrates the inheritance relationship such as *Artist* is subclass of *Person*.

2. **Discovering a subset of MFP**
   As mentioned in Sect. 3.2, our goal is also to find the set of properties that can be used together in the query and does not exceed the selected threshold. For example, for the *Film* class with 60% of completeness, there are four sets of properties that are greater than 60% {{type, name, label, director, writer}, {type, name, label, director, runtime}, {type, name, label, director, starring},{type, name, label, runetime, starring}}. For this reason, our LOD-CM interface enables the user to check the desired properties that appear in the returned model. It should be noted that the property which does not achieve the completeness threshold with other selected properties will be inactivated, such as *starring* and *writer* in our previous example. This case could help the user to be sure that the returned results for its query with this set of properties are equal or greater than the desired threshold.

Table 3. DBpedia number of predicates by classes and thresholds

Class/threshold	0.1	0.3	0.5	0.7	0.9
Film	18	12	7	6	3
Settlement	18	14	8	5	4
Organisation	18	4	4	3	3
Scientist	19	16	12	9	5

Finally, Table 3 shows the number of properties we get (at the end of our pipeline) for several classes according to several thresholds. The lower the threshold is, the more properties there are, obviously. Thus, lower thresholds produce more complex conceptual schemas but with more noise. Hence, this tool can help the user to find the right balance between those two.

5 Conclusion

We have presented an approach for revealing conceptual schemas from RDF data sources. The approach is an iterative process that computes a plausible model from the data values. We have shown how to automatically extract schema and represent it as a model from a data source using a user-specified threshold. The inferred model takes into account the data and the user quality expectations. The result is a conceptual schema enriched by both completeness values as a relevancy indicator on the elements of the models, and existence constraints that inform about how often these elements co-exist or co-appear in the real data.

The elements composing the model (classes, relationships and properties) are obtained by applying a mining algorithm with an underlying assumption stating that the more frequent a schema is, the more relevant it is. The user can decide on the desired completeness, the parts of the data for which the model will be inferred and the possibility to focus on a different category through an iterative process. Currently, our demo supports only the DBpedia dataset.

We have provided several use cases to demonstrate the usefulness of such a tool. We believe that it can help in the discovery of a new dataset and its internal structure, therefore, it can help in the adoption of LD datasets.

Our analysis revealed some interesting characteristics allowing the characterization of the sources and the behavior of the community that maintains each of the data sources. The results show the rich opportunities of analysis offered by our approach and underlying outputs.

In the future, we plan to investigate the role of conceptual modeling in an integration context where the universe of discourse is not only one data source but an integrated system upon several Linked Open Data. We plan to make more datasets available and allow the user to easily compare two conceptual schemas side by side (from two datasets). We believe that the ability to compare two conceptual schemas of two datasets side by side can help to choose the one that is best suited for its use.

References

1. Batini, C., Scannapieco, M.: Erratum to: data and information quality: dimensions, principles and techniques. In: Batini, C., Scannapieco, M. (eds.) Data and Information Quality. DSA, pp. E1–E1. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-24106-7_15
2. Christodoulou, K., Paton, N.W., Fernandes, A.A.A.: Structure inference for linked data sources using clustering. In: Hameurlain, A., Küng, J., Wagner, R., Bianchini, D., De Antonellis, V., De Virgilio, R. (eds.) Transactions on Large-Scale Data- and Knowledge-Centered Systems XIX. LNCS, vol. 8990, pp. 1–25. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-46562-2_1
3. Embley, D.W., Liddle, S.W.: Big data—conceptual modeling to the rescue. In: Ng, W., Storey, V.C., Trujillo, J.C. (eds.) ER 2013. LNCS, vol. 8217, pp. 1–8. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-41924-9_1
4. Gouda, K., Zaki, M.J.: Efficiently mining maximal frequent itemsets. In: Proceedings of the 2001 IEEE International Conference on Data Mining, ICDM 2001, pp. 163–170. IEEE Computer Society, Washington, DC (2001)
5. Grahne, G., Zhu, J.: Efficiently using prefix-trees in mining frequent itemsets. In: Goethals, B., Zaki, M.J. (eds.) FIMI 2003, Frequent Itemset Mining Implementations, Proceedings of the ICDM 2003 Workshop on Frequent Itemset Mining Implementations, 19 December 2003, Melbourne, Florida, USA, CEUR Workshop Proceedings, vol. 90. CEUR-WS.org (2003)
6. Hamdi, F., Cherfi, S.S.S.: An approach for measuring rdf data completeness. BDA 2015 Gestion de Données-Principes, Technologies et Applications 29 septembre au 2 octobre 2015 Île de Porquerolles p. 32 (2015)

7. Han, J., Pei, J., Yin, Y.: Mining frequent patterns without candidate generation. In: Chen, W., Naughton, J.F., Bernstein, P.A. (eds.) Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data, 16–18 May 2000, Dallas, Texas, USA, pp. 1–12. ACM (2000)
8. Han, J., Pei, J., Yin, Y., Mao, R.: Mining frequent patterns without candidate generation: a frequent-pattern tree approach. Data Min. Knowl. Discov. **8**(1), 53–87 (2004)
9. Issa, S., Paris, P.-H., Hamdi, F.: Assessing the completeness evolution of DBpedia: a case study. In: de Cesare, S., Frank, U. (eds.) ER 2017. LNCS, vol. 10651, pp. 238–247. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-70625-2_22
10. Jain, P., Hitzler, P., Yeh, P.Z., Verma, K., Sheth, A.P.: Linked data is merely more data. In: Linked Data Meets Artificial Intelligence, Papers from the 2010 AAAI Spring Symposium, Technical Report SS-10-07, Stanford, California, USA, 22–24 March 2010 (2010). http://www.aaai.org/ocs/index.php/SSS/SSS10/paper/view/1130
11. Bayardo Jr., R.J: Efficiently mining long patterns from databases. In: Haas, L.M., Tiwary, A. (eds.) SIGMOD 1998, Proceedings ACM SIGMOD International Conference on Management of Data, 2–4 June 1998, Seattle, Washington, USA, pp. 85–93. ACM Press (1998)
12. Kellou-Menouer, K., Kedad, Z.: Schema discovery in RDF data sources. In: Johannesson, P., Lee, M.L., Liddle, S.W., Opdahl, A.L., López, Ó.P. (eds.) ER 2015. LNCS, vol. 9381, pp. 481–495. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-25264-3_36
13. Klyne, G., Carroll, J.J.: Resource description framework (RDF): Concepts and abstract syntax (2006)
14. Levandoski, J.J., Mokbel, M.F.: RDF data-centric storage. In: IEEE International Conference on Web Services, ICWS 2009, pp. 911–918. IEEE (2009)
15. Lukyanenko, R., Parsons, J.: Principles for modeling user-generated content. In: Johannesson, P., Lee, M.L., Liddle, S.W., Opdahl, A.L., López, Ó.P. (eds.) ER 2015. LNCS, vol. 9381, pp. 432–440. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-25264-3_32
16. Lukyanenko, R., Parsons, J., Samuel, B.M.: Representing instances: the case for reengineering conceptual modelling grammars. Eur. J. Inf. Syst. **28**(1), 68–90 (2019)
17. Olivé, A.: Conceptual Modeling of Information Systems. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-39390-0
18. Pham, M., Passing, L., Erling, O., Boncz, P.A.: Deriving an emergent relational schema from RDF data. In: Gangemi, A., Leonardi, S., Panconesi, A. (eds.) Proceedings of the 24th International Conference on World Wide Web, WWW 2015, Florence, Italy, 18–22 May 2015, pp. 864–874. ACM (2015). http://doi.acm.org/10.1145/2736277.2741121
19. Rolland, C., Prakash, N.: From conceptual modelling to requirements engineering. Ann. Softw. Eng. **10**(1–4), 151–176 (2000)
20. Töpper, G., Knuth, M., Sack, H.: Dbpedia ontology enrichment for inconsistency detection. In: Proceedings of the 8th International Conference on Semantic Systems, pp. 33–40. ACM (2012)
21. Völker, J., Niepert, M.: Statistical schema induction. In: Antoniou, G., et al. (eds.) ESWC 2011. LNCS, vol. 6643, pp. 124–138. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-21034-1_9

22. Wang, Y., Du, X., Lu, J., Wang, X.: FlexTable: using a dynamic relation model to store RDF data. In: Kitagawa, H., Ishikawa, Y., Li, Q., Watanabe, C. (eds.) DASFAA 2010. LNCS, vol. 5981, pp. 580–594. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-12026-8_44
23. Zaveri, A., et al.: Quality assessment methodologies for linked open data. Semant. Web J. (2013, submitted)