Statistical Schema Induction

Johanna Völker⋆ and Mathias Niepert

KR & KM Research Group,
University of Mannheim, Germany
`{voelker,niepert}@informatik.uni-mannheim.de`

Abstract. While the realization of the Semantic Web as once envisioned by Tim Berners-Lee remains in a distant future, the Web of Data has already become a reality. Billions of RDF statements on the Internet, facts about a variety of different domains, are ready to be used by semantic applications. Some of these applications, however, crucially hinge on the availability of expressive schemas suitable for logical inference that yields non-trivial conclusions. In this paper, we present a statistical approach to the induction of expressive schemas from large RDF repositories. We describe in detail the implementation of this approach and report on an evaluation that we conducted using several data sets including DBpedia.

Keywords: Linked Data, Ontologies, Association Rule Mining.

1 Introduction

The generation of ontologies from formal and semi-formal data, frequently called *Semantic Web Mining* [29], has been studied for several years within the Semantic Web community. Recently d'Amato [11] et al. suggested the term *Ontology Mining* for "all those activities that allow to discover hidden knowledge from ontological knowledge bases." This line of research is partly motivated by the crucial role ontologies play for reasoning-based applications, and by the knowledge acquisition bottleneck that is caused by the enormous efforts it takes to build highly axiomatized logical theories.

However, as argued by Auer and Lehmann [2], ontologies derived from RDF repositories can also bring major benefits for the Web of Data. Although it would be foolish to consider ontologies (or generally speaking "schemas") a panacea for all the problems currently plaguing the Web of Data, they can help to ease integration, querying and maintenance of RDF datasets. By providing conceptual descriptions of RDF graphs ontologies might facilitate, for instance, the discovery of links between disconnected data sets, or enable the detection of contradictory facts spread across the cloud of Linked Open Data. Unlike Jain et al. [18], for example, we do not believe that it will be feasible or desirable to squeeze every RDF repository under a single top-level ontology such as SUMO – and those people who are currently contributing to the success of the Linked Open Data initiative by publishing their data as RDF triples will certainly not be willing to

---

adhere to any prescribed schema. Still, if we were able to automatically generate such a schema for any given RDF repository, we would be able to provide people with formal semantics of the terminology people use for talking about their data, and possibly prepare the grounds for new types of applications.

The more RDF data becomes available the more promising seems the use of inductive, i.e. bottom-up, methods which facilitate the construction of ontologies from given facts. Inductive methods to acquiring schema-level knowledge from RDF data sources have been shown to be effective, e.g., by Lehmann et al. [17] (for an excellent overview of various types of inductive methods in the context of Linked Data see [11]). We can roughly distinguish between logical and statistical methods: While statistical methods based on conceptual clustering, for instance, tend to be more scalable and robust with respect to noisy or uncertain data, logical methods such as Inductive Logic Programming are often inferior when it comes to the generation of highly axiomatized ontologies.

In this paper, we propose an approach to generating ontologies from RDF datasets that we will refer to as *Statistical Schema Induction* (SSI). After giving a brief overview of related work (cf. Section 2), we elaborate on the theoretical foundations of our approach, including the basics of association rule mining. In Section 3, we also introduce the EL profile of OWL 2, which provides us with the logical basis for constructing ontologies that are both reasonably expressive and computationally tractable. Section 4 describes in detail the implementation of our approach, before we provide the reader with details about the experiments we conducted in order to evaluate this approach on several real-world datasets (cf. Section 5). Finally, in Section 6, we conclude with a summary and an outlook to possible future work.

2   Related Work

Our approach follows previous work in the field of ontology generation from formal and semi-formal data, e.g., in the form of RDF or OWL knowledge bases. Early approaches in this line of research rely upon systematic generalization [13] or clustering [24]. Later Grimnes et al. [14] suggested an ILP-based approach to generating descriptions of groups of people from FOAF profiles.

ILP, short for *Inductive Logic Programming*, is a type of machine learning that combines machine learning and logic programming techniques in order to derive logical theories from examples (i.e. assertions) and background knowledge. Common to all ILP-based approaches is that they adhere to the paradigm of induction – a form of inference that draws general conclusions from specific instances, assuming that the latter exemplify a general truth. ILP-based methods have successfully been applied to the problem of concept learning and ontology induction, e.g., by Cohen and Hirsh [10], but only very few implementations are commonly available one of those being the DL-Learner by Jens Lehmann [22]. In recent experiments, Hellmann et al. [17] applied the DL-Learner to several RDF knowledge bases, in order to generate definitions of classes from the YAGO ontology, for instance. Unlike our implementation, the DL-Learner uses positive

and negative examples (i.e. members and non-members of these classes) randomly sampled, e.g., from DBpedia. Another particularly interesting approach has been proposed by Cimiano et al. [9], who generate intentional descriptions of the factoid answers (e.g. sets of individuals) that are returned by queries to a given knowledge base. These intentional descriptions consist of concept expressions obtained by bottom-up generalization.

Further extensional approaches to generating or refining ontologies based on given facts can be found in the area of *Formal Concept Analysis* (FCA) or Relational Exploration, respectively. OntoComP developed by Baader et al. [5] supports knowledge engineers in the acquisition of axioms expressing subsumption between conjunctions of named classes. A similar method for acquiring domain-range restrictions of object properties has been proposed later by Rudolph [27]. In both cases, hypotheses about axioms potentially missing in the ontology are generated from existing as well as from interactively acquired assertions. One of the biggest challenges for research on both FCA and ILP is uncertain and noisy input in the form of background knowledge or examples. While Auer and Lehmann [2] suggest to face this challenge by higher degrees of user interaction, we rely on statistical methods that are both robust and scalable enough to handle huge sets of Linked Data – an important prerequisite for compensating the lack of negative examples, which are not taken into account by our mining algorithms. The expressiveness of ontologies acquired by means of Statistical Schema Induction is comparable to those produced by ILP-based methods (mostly variants of $\mathcal{ALC}$) and higher than what can obtained by Relational Exploration (i.e. $\mathcal{FLE}$).

In the field of ontology learning from natural language text, we find our approach related, e.g., to methods for inducing taxonomies by means of hierarchical clustering of context vectors [8] as well as to early approaches to extracting non-taxonomic relations by *Association Rule Mining* [23]. The discovery of association rules has also been shown to facilitate the generation of ontologies from folksonomies [19] and semantic annotations in text documents [21].[1] An efficient algorithm for computing sets of association rules from RDF data was suggested by Jiang and Tan [20], while Nebot and Berlanga [25] use association rules to discover causal relations in RDF-based medical data.

Association rules have also been applied in the area of ontology matching as in the AROMA system, for example [12]. Most closely related to our approach is recent work by Parundekar et al. [26], who consider containment relationships between sets of class instantiations for producing alignments between several linked data repositories, including DBpedia. While their approach could as well be used to suggest refinements for a single ontology, they currently only acquire mappings which express subsumption or equivalence between so-called *restriction classes* roughly corresponding to $C \sqcap \exists r.D$ class expressions. In order to determine the type of correspondence between a given pair of restriction classes, Parundekar et al. rely on thresholds applied to measures of extensional overlap.

---

[1] Note that Association Rule Mining is similar to FCA in so far as every rule with a confidence of 1.0 directly corresponds to an implication in a formal context, and hence there has been some research on using FCA for Association Rule Mining [28].

3   Preliminaries

3.1   OWL 2 EL

The EL profile of the Web Ontology Language OWL 2 captures the expressivity of many ontologies in the life sciences and other application domains. In OWL 2 EL, which is based on the description logic $\mathcal{EL}^{++}$[3], reasoning services such as consistency and instance checking can be performed in time that is polynomial with respect to the number of axioms. Therefore, OWL 2 EL is well-suited for applications employing ontologies that contain very large numbers of classes, properties, and axioms. Several efficient reasoning algorithms are available.

Description logics define concept descriptions inductively by a set of constructors, starting with a set N_C of concept (or class) names, a set N_R of role (or property) names, and a set N_I of individual names. Concept descriptions and role inclusions in $\mathcal{EL}^{++}$ are build with the constructors depicted in Figure 1. We will write a and b to denote individual names; r and s to denote role names; and C and D to denote concept descriptions. The semantics of the concept descriptions in $\mathcal{EL}^{++}$ are defined in terms of an interpretation $\mathcal{I} = (\triangle^{\mathcal{I}}, \cdot^{\mathcal{I}})$. The domain $\triangle^{\mathcal{I}}$ of this interpretation is a non-empty set of individuals and the interpretation function $\cdot^{\mathcal{I}}$ maps concepts names $A \in N_C$ to a subset $A^{\mathcal{I}}$ of $\triangle^{\mathcal{I}}$, role names $r \in N_R$ to a binary relation $r^{\mathcal{I}} \subseteq \triangle^{\mathcal{I}} \times \triangle^{\mathcal{I}}$, and each individual name $a \in N_I$ to an individual $a^{\mathcal{I}} \in \triangle^{\mathcal{I}}$. The extension of $\cdot^{\mathcal{I}}$ to arbitrary concept descriptions is recursively defined as shown in Table 1.

Table 1. The description logic $\mathcal{EL}^{++}$ *without* nominals and concrete domains

Name	Syntax	Semantics
top	$\top$	$\triangle^{\mathcal{I}}$
bottom	$\bot$	$\emptyset$
conjunction	$C \sqcap D$	$C^{\mathcal{I}} \cap D^{\mathcal{I}}$
existential restriction	$\exists r.C$	$\{x \in \triangle^{\mathcal{I}} | \exists y \in \triangle^{\mathcal{I}} : (x, y) \in r^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}\}$
GCI	$C \sqsubseteq D$	$C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$
RI	$r_1 \circ ... \circ r_k \sqsubseteq r$	$r_1^{\mathcal{I}} \circ ... \circ r_k^{\mathcal{I}} \subseteq r^{\mathcal{I}}$

Role inclusion (RI) axioms generalize axiom types that occur often in ontology applications such as role hierarchies $r \sqsubseteq s$ and transitive roles, which can be expressed by the axiom $r \circ r \sqsubseteq r$. Also note that the bottom concept in combination with generalized concept inclusion axioms (GCIs) can be used to express disjointness of complex concept descriptions. Furthermore, it is possible to model both range and domain restrictions [4] in OWL 2 EL.

In this work, we will focus on axiom types that are captured by the EL profile of OWL 2. This way we are able to learn many of the axioms used in practical, large-scale ontologies while being able to employ efficient reasoning algorithms.

3.2   Association Rule Mining

Association rules are a very simple but useful form of implication patterns. Consider the example in Table 2. The rows represent individuals and the columns represent the types occurring in DBpedia. A value of 1 in field (i, j) indicates that individual i is of type j. We are now interested in mining meaningful rules that provide evidence for certain axioms in the hidden schema. The framework of association rules was originally developed for large and sparse datasets such as transaction databases of international supermarket chains. A typical dataset in such a setting can have up to 10^{10} transactions (rows) and 10^6 attributes (columns). Hence, the mining algorithms developed for these applications are also applicable to the large data repositories in the open Linked Data cloud.

Let $I = \{i_1, i_2, ..., i_n\}$ be a set of n binary attributes. These attributes are usually referred to as *items*. In the open Linked Data setting items correspond to types occurring in the repository. Let $D = (t_1, t_2, ..., t_m)$ be a list of transactions (the transaction database) with each t_i a subset of I. Each transaction (that is, each row in the database) corresponds to one individual. The entry in column j has value 1 if the individual is of type i_j and 0 otherwise. The support $supp(X)$ of an itemset $X \subseteq I$ is defined as the number of transactions in the data set which contains the itemset X:

$$supp(X) = |\{t_i \in D : X \subseteq t_i\}|$$

Now, the frequent itemset mining problem is the following: Given the set I of items, a transaction database D over S, and a nonnegative threshold τ, determine the set of items whose support is at least τ. The most widely used algorithm for mining frequent itemsets is the Apriori algorithm [1].

Association rules are often used to gain a deeper understanding of the regularities and patterns of large data sets. An association rules is an implication of the form $X \Rightarrow Y$ with X and Y itemsets. While there is an exponential amount of potential association rules most state of the art algorithms take advantage of the sparsity of the transaction database and prune the search space whenever possible. In the majority of the cases, the output of frequent itemset algorithms such as Apriori is further processed to derive all association rules with a particular minimum support. The *confidence* of an association rule which is sometimes also called the *accuracy* is defined as follows:

$$conf(A \Rightarrow B) = \frac{supp(A \cup B)}{supp(A)}$$

The confidence value of an association rule can be viewed as a frequency-based maximum-likelihood estimate of the conditional probability of B occurring in the data given that A occurs in the data. The basic idea of the presented work is that association rules with a high confidence value correspond to certain OWL 2 EL axioms. For instance, a high confidence value for the association rule $A \Rightarrow B$ with A and B being RDF types would provide evidence for the validity of the subsumption axiom $A \sqsubseteq B$ because most resources that are of `rdf:type` A are also of `rdf:type` B.

Table 2. Example of a transaction database in the context of the DBpedia dataset

IRI	Comedian	Artist	Person	Airport	Building	Place	Animal
Jerry_Seinfeld	1	1	0	0	0	0	0
Black_Bird	0	0	0	0	0	0	1
Chris_Rock	1	1	1	0	0	0	0
Robin_Williams	1	0	1	0	0	0	0
JFK_Airport	0	0	0	1	1	1	0
Hancock_Tower	0	0	0	0	1	1	0
Newark_Airport	0	0	0	1	1	1	0

Example. Let us assume that the table in Figure 2 is the transaction database for a fragment of the DBpedia data set. Then, we have for instance that $supp(\{Comedian, Artist\}) = 2$, $supp(\{Comedian\}) = 3$, $supp(\{Artist, Person\})$ = 1, $supp(\{Airport, Building\} = supp(\{Airport, Place\}) = supp(\{Airport, Building, Place\}) = 2$, and $supp(\{Building, Place\}) = 3$. Furthermore, some of the association rules and their confidence values are $conf(\{Comedian\} \Rightarrow \{Artist\}) = \frac{2}{3}$ and $conf(\{Airport\} \Rightarrow \{Building\}) = \frac{2}{2} = 1.0$.

4   Statistical Schema Induction

In the following, we describe in detail our approach to inducing or enriching the schema of an RDF repository through its SPARQL endpoint. Our implementation of this approach is based on the assumption that the semantics of any RDF resource, such as a predicate for example, is revealed by patterns we can observe when considering the usage of this resource in the repository. While the general methodology of detecting such patterns by means of association rule mining can be applied to virtually any RDF repository with minor modifications, certain characteristics of a underlying RDF graph certainly facilitate the induction of a schema. We will discuss some of these characteristics in Section 5, where we detail on the experimental evaluation we conducted on different datasets.

The overall process of SSI can be summarized as follows (cf. Figure 1):

1. First, we acquire the *terminology*, i.e. the non-logical vocabulary of the OWL ontology to be constructed, by posing SPARQL queries to the repository's endpoint (cf. Section 4.1). The result of this step is a set of relational database tables containing the URIs of all those RDF resources which we assume to correspond to classes and properties.[2] Note that we also assign unique identifiers to certain combinations of resources (e.g. any pair of two predicates r_1 and r_2) as we would like use those for building complex class or property expressions (e.g. $r_1 \circ r_2$).

---

[2] In order to identify potential classes and properties in an RDF graph, we rely on heuristics similar to those suggested by Bechhofer and Volz [6] and taken up later by Hellmann et al. [17].
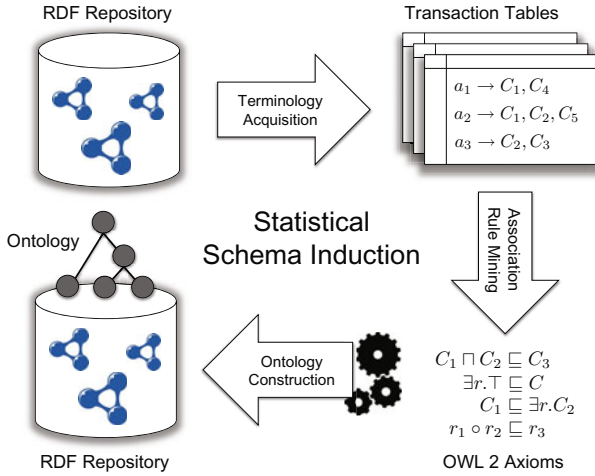
Fig. 1. Worfklow of the Statistical Schema Induction framework

2. Second, we construct the *transaction tables*, which we need in order to mine the dataset for the various kinds of OWL axioms (cf. Section 4.2). Each transaction, i.e. row in a transaction table, corresponds to a resource or a pair of resources, respectively. While for every single resource the items in a transaction are named or complex "classes" according to our terminology, a pair of resources always maps to the set of predicates or predicate chains (i.e. "property expressions") they are linked with. We then mine the transaction tables for *association rules*.
3. Finally, every association rule gets translated into an OWL 2 EL axiom. The support and the confidence values of the rules are taken into account when the ontology is constructed in a fully automatic manner (cf. Section 4.3).

4.1   Terminology Acquisition

Named classes. Initially, we gather information about those resources which are likely to represent *classes* $C \in \mathsf{N_C}$ in the ontology that we would like to generate. We do so by means of a SPARQL query motivated by a simple heuristic: every object of an `rdf:type` statement is a class, whereas the subjects of all such statements provide us with the instances of these classes.[3] Since the syntax and semantics of RDF do not constrain the use of `rdf:type` statements nor otherwise enforce a division into assertional and terminological level, we cannot expect this heuristic to work for every RDF graph. However, as we will see in Section 5, it yields good results for several of the most well-known datasets. Every resource supposed to be a class or an individual gets assigned a unique numerical identifier and is stored in a relational database.

---

[3] We only consider explicit `rdf:type` statements, hence ignore those which are entailed e.g. by `owl:sameAs` statements, which often have an unclear semantics [16].

While the names of the "individuals" (i.e. those resources explicitly stated to have an `rdf:type`) are not relevant for our approach as the individuals anyway will not become part of the schema, we have to be able to uniquely identify them in order to construct the transactions table (see further below). In cases where the overall number of resources considered individuals is too high for storing them locally, one should consider the use of sampling heuristics (as suggested, e.g., by d'Amato et al. [11]) or of a Map-Reduce infrastructure on top of a distributed file system. The latter would actually work very well with our approach as the computation of the transactions tables from the data gathered at this stage can be conducted in a completely parallel way.

Object properties. In a similar way, we collect the names of all those RDF resources which we assume to represent *object properties* $r \in \mathsf{N_R}$ in the ontology: Every predicate of an RDF triple which belongs to the DBpedia namespace and whose object is linked to another resource by means of an `rdf:type` statement is considered an object property. Again, we store both, all of these predicates and the unique pairs of resources linked by any of these predicates, in our database.

Class expressions. Now that we have acquired the basic terminology, i.e. the names of all resources to denote named classes or properties, we can turn to complex class and property expressions. Since we want to be able to also mine the dataset for domain and range restrictions such as $\exists r.\top \sqsubseteq C$, for example, we have to assign unique identifiers to the following types of class expressions: $\exists r.C$, $\exists r.\top$ and $\exists r^{-1}.\top$ for each $r \in \mathsf{N_R}$ and $C \in \mathsf{N_C}$.[4] Note that there can be resources which are not explicitly stated to be of some type (cf. named classes) while still fitting some of these class expressions (e.g. because the respective resource is linked to another one by virtue of a property r). For this reason, we also need to extend our initially computed set of identifiers for potential individuals.

Property chains. Finally, as motivated in Section 3.1, we would like to acquire *transitivity* axioms for all the predicates (i.e. potential object properties) in the dataset. Transitivity can be expressed by a particular type of property chain inclusion axiom, namely $r \circ r \sqsubseteq r$ for $r \in \mathsf{N_R}$. Therefore, we again create a database table for mapping each property chain expression to a unique identifier and, similarly as for the class expressions, assign a new identifier to a pair of resources whenever this pair is not linked directly by any object property but only by a property chain of the aforementioned shape.

4.2   Association Rule Mining

Before we can start mining for association rules as described in Section 4.3, we have to create *transaction tables* for the various types of axioms that we would like to become part of the ontology. Figure 3 gives an overview of the types of axioms covered by our current implementation and the corresponding transaction tables. For example, axioms of the type $C \sqsubseteq D$ (that is, those expressing subsumption between atomic classes) can be mined from a transaction table whose

---

[4] Range restrictions can be expressed in OWL 2 EL, even though inverse properties are not included in the profile [4].

Table 3. Each row in a transaction table either corresponds to a resource a or pair of resources (a, b) for $a, b \in \mathsf{N_I}$, which are assumed to represent an individual or a pair of individuals, respectively. We output the identifier of C iff a is linked to a resource named C by means of an `rdf:type` statement, and the identifier of r iff a and b are connected by the predicate r. Likewise, if a row in the transaction table contains the identifier of a class expression $\exists r.C$, for example, this means that the corresponding resource a is linked to another resource of `rdf:type` C by virtue of r.

Axiom Type	Transaction Table	Association Rule
$C \sqsubseteq D$	$a \to C_1, ..., C_n$ for $a \in \mathsf{N_I}$	$\{C_i\} \Rightarrow \{C_j\}$
$C \sqcap D \sqsubseteq E$	$a \to C_1, ..., C_n$ for $a \in \mathsf{N_I}$	$\{C_i, C_j\} \Rightarrow \{C_k\}$
$D \sqsubseteq \exists r.C$	$a \to C_1, ..., C_l, \exists r_1.C_{11}, ..., \exists r_m.C_{mn}$ for $a \in \mathsf{N_I}$	$\{C_k\} \Rightarrow \{\exists r_j.C_{jk}\}$
$\exists r.C \sqsubseteq D$	$a \to C_1, ..., C_l, \exists r_1.C_{11}, ..., \exists r_m.C_{mn}$ for $a \in \mathsf{N_I}$	$\{\exists r_j.C_{jk}\} \Rightarrow \{C_i\}$
$\exists r.\top \sqsubseteq C$	$a \to C_1, ..., C_l, \exists r_1.\top, ..., \exists r_m.\top$ for $a \in \mathsf{N_I}$	$\{\exists r_j.\top\} \Rightarrow \{C_i\}$
$\exists r^{-1}.\top \sqsubseteq C$	$a \to C_1, ..., C_l, \exists r_1^{-1}.\top, ..., \exists r_m^{-1}.\top$ for $a \in \mathsf{N_I}$	$\{\exists r_j^{-1}.\top\} \Rightarrow \{C_i\}$
$r \sqsubseteq s$	$(a, b) \to r_1, ..., r_n$ for $(a, b) \in \mathsf{N_I} \times \mathsf{N_I}$	$\{r_i\} \Rightarrow \{r_j\}$
$r \circ r \sqsubseteq r$	$(a, b) \to r_1, ..., r_n, r_1 \circ r_1, ... r_n \circ r_n$ for $(a, b) \in \mathsf{N_I} \times \mathsf{N_I}$	$\{r_i \circ r_i\} \Rightarrow \{r_i\}$

rows correspond to those individuals having at least one `rdf:type` $C \in \mathsf{N_C}$. Each row in this table contains the identifiers of those classes $C_1, ..., C_n$ the respective individual belongs to, and can be determined by a simple SPARQL query (`SELECT distinct ?c WHERE <a> a ?c`) followed by a lookup in the previously built terminology tables (cf. Section 4.1). Note that axioms that involve the same types of class or property expressions can be mined from the same transaction tables. For instance, both the $D \sqsubseteq \exists r.C$ and the $\exists r.C \sqsubseteq D$ axioms are mined from the same transaction database. Thus, we only need 6 transaction databases in order to mine the dataset for the axiom types listed in Table 3.

4.3   Ontology Construction

As indicated by Figure 3, the association rules mined from the various transaction tables can be translated into OWL 2 EL axioms in a relatively straightforward way. The confidence and support value for each of the association rules provides us with a measure of certainty, which we take into account when constructing the ontology.[5]

Following [15], we pursue a simple greedy strategy for adding the acquired axioms to an initially empty or to an existing OWL ontology that we would like to refine: First, we sort all of the generated axioms in descending order based on their certainty values. Then we add them to the ontology one by one, checking the coherence of the ontology after the addition of each axiom. In case any of the classes in the ontology becomes incoherent, e.g. because an axiom states disjointness

---

[5] By setting a confidence threshold of 1.0 we obtain an ontology that perfectly fits the data, i.e. which does not contain any inconsistencies if merged with the factoid knowledge in the RDF repository.

between two classes which subsume each other according to a previously added axiom, we retract the most recent axiom and continue with the next one.

5   Evaluation

In the following, we report on several experiments we conducted in order to evaluate Statistical Schema Induction with real-world data. The results of these experiments are available online and can be downloaded from a dedicated web page.[6] By the first experiment based on the DBpedia dataset (cf. Section 5.1) we were aiming to gain some insights regarding the quality of the generated ontologies as compared to existing, manually constructed schemas. The comparatively large size of the DBpedia dataset makes it a good benchmark for the scalability of our implementation. In a second experiment (see Section 5.2), we assessed the feasibility of our approach on a set of smaller RDF datasets, all of them taken from `data.gov.uk`.

Both of these experiments were carried out on a an AMD 64bit DualCore computer with 2,792 MHz and 8 GB RAM. The Java-based implementation of our approach makes use of various publicly available libraries for database access (MySQL 5.0.51), ontology management (Pellet 2.2.1 and OWL API 3.0.0) and Linked Data querying (Jena 2.6.3). In addition, we applied the Apriori implementation by Borgelt and Kruse [7] to mine the association rules.

```
1110   1325   6293   0      1      144
4065   4665   4695   1146   6330   6973   64   185
1146   6330   6973   64     68     141
3235   6668   6769   3242   5049   6673   3907   2   66
1110   1325   6293   0      73     144
```

Fig. 2. Textual serialization of a transaction table

The input to this implementation were textual serializations of the transaction tables (cf. Table 3). Figure 2 shows an excerpt from an input file that enables us to acquire axioms of the form $C \sqsubseteq \exists r.D$ and $\exists r.C \sqsubseteq D$ for the DBpedia dataset (cf. Section 5.1). The items in each transaction like 144 (*Place*) or 3907 ($\exists language.Language$), for example, are the identifiers of those named or complex classes the respective individual belongs to. Note that we only need to compute transactions for those individuals which are known to be members of at least one named or complex class. For the DBpedia dataset, the number of rows in the various transaction tables varied between 5,217,133 (transitivity axioms) and 1,477,796 (domain and range restrictions).

From the computed association rules, we generated the types of axioms listed by Table 3. Moreover, we generated disjointness axioms ($C \sqcap D \sqsubseteq \bot$) between classes with more than 100 instances that do not have any individuals in common. The confidence values for these axioms were generated by normalizing the product of the number of instances for each pair of classes.

---

[6] `http://code.google.com/p/gold-miner/`

τ	# axioms	recall	precision	F_1 score
1.0	365	0.997	0.992	0.995
0.9	373	0.997	0.971	0.983
0.8	381	0.997	0.950	0.973

(a) Without support threshold

τ	# axioms	recall	precision	F_1 score
1.0	339	0.926	0.991	0.957
0.9	347	0.926	0.968	0.947
0.8	354	0.926	0.949	0.937

(b) With support threshold of 10

Fig. 3. Recall, precision and F_1 values for **subsumption axioms** between atomic classes for varying thresholds on the confidence values

τ	# axioms	recall	precision	F_1 score
1.0	950	0.900	0.808	0.852
0.9	1143	0.946	0.655	0.774
0.8	1181	0.946	0.683	0.793

(a) Without support threshold

τ	# axioms	recall	precision	F_1 score
1.0	821	0.790	0.821	0.805
0.9	1036	0.835	0.576	0.682
0.8	1092	0.838	0.558	0.670

(b) With support threshold of 10

Fig. 4. Recall, precision and F_1 values for **domain restriction axioms** for varying thresholds on the confidence values

5.1 DBpedia

We evaluated the generated ontology by comparing it to the DBpedia ontology[7] (version 3.5.1), which we considered the most natural gold standard. The DBpedia ontology was created by a manual mapping of 1,055 Wikipedia infobox templates to 259 named classes. Besides these classes, the ontology comprises 602 object properties, 674 datatype properties, 257 explicit subsumption axioms as well as 459 domain and 482 range restrictions. 1,477,796 of the roughly 3.4 million "things" (i.e. RDF resources representing Wikipedia articles) are explicitly classified with regard to the DBpedia ontology.

However, as the expressivity of the DBpedia ontology is relatively low (it equals the complexity of the $\mathcal{ALF}(D)$ description logic), we only considered those types of axioms that are common to both ontologies when comparing the two schemas: subsumption between named classes and property restrictions. Tables 3, 4, and 5 list the results of the schema induction process for various thresholds on the confidence values. The time needed to compute the association rules was less than 5 seconds for the largest transaction table, which confirms the scalability of the Apriori algorithm to the large Linked Data repositories. The recall and precision scores are computed *relative* to the DBpedia ontology. Hence, not all false positives and false negatives with respect to the DBpedia ontology would necessarily be incorrect in terms of a more complete gold standard. Note that for the comparison of the two ontologies we did not only consider the explicit axioms, but all the *inferable* class subsumption and property restriction axioms.

---

[7] `http://wiki.dbpedia.org/Ontology`

τ	# axioms	recall	precision	F_1 score
1.0	401	0.392	0.666	0.494
0.9	740	0.712	0.655	0.682
0.8	868	0.790	0.620	0.695

(a) Without support threshold

τ	# axioms	recall	precision	F_1 score
1.0	206	0.258	0.854	0.396
0.9	740	0.580	0.512	0.544
0.8	840	0.658	0.604	0.630

(b) With support threshold of 10

Fig. 5. Recall, precision and F_1 values for **range restriction axioms** for varying thresholds on the confidence values

5.2   Other Datasets

In order to test the applicability of Statistical Schema Induction to RDF datasets other than DBpedia (in particular to datasets that do not come with any existing schema yet), we performed a second evaluation experiment based on the RDF repository of `data.gov.uk`. In this experiment, we focused on the taxonomy of named classes and merely generated axioms of the form $C \sqsubseteq D$. Due to time constraints and the lack of a proper gold standard for this dataset we only report our most important findings, and refer the reader to the aforementioned website that we setup for our experiments.

Without any changes to our implementation, we were able to compute appropriate transaction tables for five subsets of `data.gov.uk` each of these subsets corresponding to a public sector:[8] *reference*, *eduction*, *ordnance*, *transport* and *finance*. For three of them we obtained a proper class hierarchy,[9] while closer inspection of the other two datasets (*ordnance* and *finance*) revealed that none of the resources in the dataset was stated to be of more than one `rdf:type`.

One might argue that the redundancy caused by multiple type statements cannot be assumed to be common in real-world datasets. However, note that the kind of axioms involving complex class expressions or any of the property subsumption axioms could still be acquired in this case. Moreover, whenever we do not find multiple `rdf:type` statements for all of the RDF resources (e.g. in the case of the *ordnance* dataset of `data.gov.uk` only very few resources have more than one type), we could pursue the following bootstrapping-like strategy: First, we mine the dataset for domain-range restrictions of the predicates that we assume to represent object properties. For example, we might find the domain of a certain predicate r to be of type C, that is $\exists r.\top \sqsubseteq C$. Then, we use these restrictions in order to "classify" all of the resources in the RDF graph. In particular, adhering to the OWL semantics of domain-range restrictions, we can infer that each resource that has the property r must be of type C and likewise for the range. Those additional type statements could finally help to induce the hierarchy of named classes.

---

[8] As the *legislation* part of `data.gov.uk` uses only a very limited set of identifiers for the objects of `rdf:type` statements, we were unable to acquire a sufficiently rich terminology for this dataset.

[9] For example, the axioms *DeputyDirector* $\sqsubseteq$ *CivilServicePost* and *MinisterialDepartment* $\sqsubseteq$ *Department* were mined from the *reference* part of the `data.gov.uk` dataset.

Without applying this strategy, we obtained 64 classes and 47 axioms for *education*, 62 classes and 137 axioms for *transport*, 20 classes and 17 axioms for *reference*, 29 classes and 3 axioms for *ordnance*, as well as 41 classes and 0 axioms for *finance*, where "axioms" refers to explicit subsumption axioms ($C \sqsubseteq D$).

6   Conclusion and Future Work

In this paper, we introduced statistical schema induction as a means to generating ontologies from RDF data. Our approach based on association rule mining has been tested on several real-world datasets. While the first results are actually very promising, we are well aware of the fact that our implementation could be improved in various respects.

As future work we envision, for example, an adaptation of our approach to more expressive description logics. In particular, we will extend our implementation to also capture property disjointness ($r \sqsubseteq \neg s$), inverse properties ($r \equiv s^{-1}$) and cardinality restrictions (e.g. $C \sqsubseteq \; \leq 1r.\top$). Furthermore, we would like to facilitate a more efficient construction of the transaction tables by appropriate sampling strategies or a Map-Reduce framework for distributed computation. Another very promising way to increase the scalability of our approach could be the use of incremental methods for adapting a generated ontology to subsequent changes in the underlying dataset. As long as we can assume these changes to be strictly monotonic, the necessary adaptations to the transaction tables will be linear in time, and efficient algorithms for mining association rules could suggest appropriate ontology refinements within a few seconds at most. It is thus tempting to imagine, for example, an on-the-fly refinement of the DBpedia ontology that keeps it synchronized with the DBpedia live dataset. Finally, we are confident that these optimizations as well as existing instance mapping techniques will facilitate the application of Statistical Schema Induction across even larger and more heterogenous fragments of the Linked Data cloud.

References

1. Agrawal, R., Srikant, R.: Fast algorithms for mining association rules in large databases. In: Proceedings of 20th International Conference on Very Large Data Bases, pp. 487–499. Morgan Kaufmann, San Francisco (1994)
2. Auer, S., Lehmann, J.: Creating knowledge out of interlinked data. Semantic Web 1(1-2), 97–104 (2010)
3. Baader, F., Brandt, S., Lutz, C.: Pushing the $\mathcal{EL}$ envelope. In: Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence IJCAI, Edinburgh, UK. Morgan-Kaufmann Publishers, San Francisco (2005)
4. Baader, F., Brandt, S., Lutz, C.: Pushing the EL envelope further. In: Clark, K., Patel-Schneider, P.F. (eds.) Proceedings of the OWLED 2008 DC Workshop on OWL: Experiences and Directions (2008)

5. Baader, F., Ganter, B., Sertkaya, B., Sattler, U.: Completing description logic knowledge bases using formal concept analysis. In: Veloso, M.M. (ed.) Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI), pp. 230–235. AAAI Press, Menlo Park (2007)
6. Bechhofer, S., Volz, R.: Patching syntax in OWL ontologies. In: McIlraith, S.A., Plexousakis, D., van Harmelen, F. (eds.) ISWC 2004. LNCS, vol. 3298, pp. 668–682. Springer, Heidelberg (2004)
7. Borgelt, C., Kruse, R.: Induction of association rules: Apriori implementation. In: 15th Conference on Computational Statistics, pp. 395–400 (2002)
8. Cimiano, P., Hotho, A., Staab, S.: Comparing conceptual, divise and agglomerative clustering for learning taxonomies from text. In: Proceedings of the 16th European Conference on Artificial Intelligence (ECAI), pp. 435–439. IOS Press, Amsterdam (2004)
9. Cimiano, P., Rudolph, S., Hartfiel, H.: Computing intensional answers to questions – an inductive logic programming approach. Data & Knowledge Engineering 69(3), 261–278 (2010)
10. Cohen, W.W., Hirsh, H.: Learning the classic description logic: Theoretical and experimental results. In: Proceedings of the 4th International Conference on Principles of Knowledge Representation and Reasoning (KR), pp. 121–133. Morgan Kaufmann, San Francisco (1994)
11. d'Amato, C., Fanizzi, N., Esposito, F.: Inductive learning for the semantic web: What does it buy? Semantic Web 1(1-2), 53–59 (2010)
12. David, J., Guillet, F., Briand, H.: Association rule ontology matching approach. International Journal on Semantic Web and Information Systems 3(2), 27–49 (2007)
13. Delteil, A., Faron-Zucker, C., Dieng, R.: Learning ontologies from rdf annotations. In: Maedche, A., Staab, S., Nedellec, C., Hovy, E.H. (eds.) Proceedings of the 2nd Workshop on Ontology Learning (OL) at the 17th International Conference on Artificial Intelligence (IJCAI). CEUR Workshop Proceedings, vol. 38, CEUR-WS.org (2001)
14. Grimnes, G.A., Edwards, P., Preece, A.D.: Learning meta-descriptions of the FOAF network. In: McIlraith, S.A., Plexousakis, D., van Harmelen, F. (eds.) ISWC 2004. LNCS, vol. 3298, pp. 152–165. Springer, Heidelberg (2004)
15. Haase, P., Völker, J.: Ontology learning and reasoning — dealing with uncertainty and inconsistency (ISWC International Workshop, URSW 2005-2007, Revised Selected and Invited Papers). In: da Costa, P.C.G., d'Amato, C., Fanizzi, N., Laskey, K.B., Laskey, K.J., Lukasiewicz, T., Nickles, M., Pool, M. (eds.) URSW 2005 - 2007. LNCS (LNAI), vol. 5327, pp. 366–384. Springer, Heidelberg (2008)
16. Halpin, H., Hayes, P.J., McCusker, J.P., McGuinness, D.L., Thompson, H.S.: When owl:sameAs isn't the same: An analysis of identity in linked data. In: Patel-Schneider, P.F., Pan, Y., Hitzler, P., Mika, P., Zhang, L., Pan, J.Z., Horrocks, I., Glimm, B. (eds.) ISWC 2010, Part I. LNCS, vol. 6496, pp. 305–320. Springer, Heidelberg (2010)
17. Hellmann, S., Lehmann, J., Auer, S.: Learning of OWL class descriptions on very large knowledge bases. International Journal on Semantic Web and Information Systems 5(2), 25–48 (2009)
18. Jain, P., Hitzler, P., Yeh, P.Z., Sheth, A.P.: Linked data is merely more data. In: Proceedings of the AAAI Spring Symposium, Linked AI: Linked Data Meets Artificial Intelligence (2010)
19. Jäschke, R., Hotho, A., Schmitz, C., Ganter, B., Stumme, G.: Discovering shared conceptualizations in folksonomies. Journal of Web Semantics 6(1), 38–53 (2008)

20. Jiang, T., Tan, A.-H.: Mining RDF metadata for generalized association rules. In: Bressan, S., Küng, J., Wagner, R. (eds.) DEXA 2006. LNCS, vol. 4080, pp. 223–233. Springer, Heidelberg (2006)
21. Kuittinen, H., Tuominen, J., Hyvönen, E.: Extending an ontology by analyzing annotation co-occurrences in a semantic cultural heritage portal. In: Proceedings of the Workshop on Collective Intelligence (ASWC-CI) at the 3rd Asian Semantic Web Conference, ASWC (2008)
22. Lehmann, J.: DL-Learner: learning concepts in description logics. Journal of Machine Learning Research (JMLR) 10, 2639–2642 (2009)
23. Mädche, A., Staab, S.: Discovering conceptual relations from text. In: Horn, W. (ed.) Proceedings of the 14th European Conference on Artificial Intelligence (ECAI), pp. 321–325. IOS Press, Amsterdam (2000)
24. Maedche, A., Zacharias, V.: Clustering ontology-based metadata in the semantic web. In: Elomaa, T., Mannila, H., Toivonen, H. (eds.) PKDD 2002. LNCS (LNAI), vol. 2431, pp. 348–360. Springer, Heidelberg (2002)
25. Nebot, V., Berlanga, R.: Mining association rules from semantic web data. In: García-Pedrajas, N., Herrera, F., Fyfe, C., Benítez, J.M., Ali, M. (eds.) IEA/AIE 2010. LNCS, vol. 6097, pp. 504–513. Springer, Heidelberg (2010)
26. Parundekar, R., Knoblock, C.A., Ambite, J.L.: Linking and building ontologies of linked data. In: Patel-Schneider, P.F., Pan, Y., Hitzler, P., Mika, P., Zhang, L., Pan, J.Z., Horrocks, I., Glimm, B. (eds.) ISWC 2010, Part I. LNCS, vol. 6496, pp. 598–614. Springer, Heidelberg (2010)
27. Rudolph, S.: Acquiring generalized domain-range restrictions. In: Medina, R., Obiedkov, S. (eds.) ICFCA 2008. LNCS (LNAI), vol. 4933, pp. 32–45. Springer, Heidelberg (2008)
28. Stumme, G.: Efficient data mining based on formal concept analysis. In: Hameurlain, A., Cicchetti, R., Traunmüller, R. (eds.) DEXA 2002. LNCS, vol. 2453, pp. 534–546. Springer, Heidelberg (2002)
29. Stumme, G., Hotho, A., Berendt, B.: Semantic web mining: State of the art and future directions. Journal of Web Semantics 4(2), 124–143 (2006)