Mining RDF Data for Property Axioms

Daniel Fleischhacker, Johanna Völker, and Heiner Stuckenschmidt[⋆]

KR & KM Research Group, University of Mannheim, Germany
{daniel,johanna,heiner}@informatik.uni-mannheim.de

Abstract. The Linked Data cloud grows rapidly as more and more knowledge bases become available as Linked Data. Knowledge-based applications have to rely on efficient implementations of query languages like SPARQL, in order to access the information which is contained in large datasets such as DBpedia, Freebase or one of the many domain-specific RDF repositories. However, the retrieval of specific facts from an RDF dataset is often hindered by the lack of schema knowledge, that would allow for query-time inference or the materialization of implicit facts. For example, if an RDF graph contains information about films and actors, but only *Titanic starring Leonardo_DiCaprio* is stated explicitly, a query for all movies Leonardo DiCaprio acted in might not yield the expected answer. Only if the two properties *starring* and *actedIn* are declared inverse by a suitable schema, the missing link between the RDF entites can be derived. In this work, we present an approach to enriching the schema of any RDF dataset with property axioms by means of statistical schema induction. The scalability of our implementation, which is based on association rule mining, as well as the quality of the automatically acquired property axioms are demonstrated by an evaluation on DBpedia.

Keywords: Linked Data, Ontology Learning, OWL2, Property Characteristics

1 Motivation

Currently, the amount of information which is available as Linked Data on the Web is growing rapidly. Large repositories like DBpedia or Freebase already contain data on many different topics, and thus serve as crystallization points for the inclusion of new data and domain-specific knowledge repositories. Its tremendous growth in terms of both size and coverage makes Linked Data more and more appealing for various types of applications. However, a problem of many linked data sets remains the lack of an appropriate schema that could be used to infer implicit information and to validate the consistency of newly added data.

The semantic web architecture foresees formal ontologies as a basis for deriving implicit information and data verification. The creation of such ontologies, however, is a complex and time-consuming process and the size and heterogeneity of linked data sources renders the manual creation of formal ontologies an unrealistic scenario. There are some efforts for creating schema information for linked data. Examples are

[⋆] Johanna Völker is financed by a Margarete-von-Wrangell scholarship of the European Social Fund (ESF) and the Ministry of Science, Research and the Arts Baden-Württemberg.

the DBpedia ontology[1] and more recently `schema.org`. These schemas including the mentioned ones, however, are rather lightweight and make very little use of the expressiveness of ontology languages such as OWL.[2] As a result, these schemas are of limited value for logical inference and semantics-based verification.

Recent work suggests the use of inductive learning methods for creating ontological knowledge from the linked data itself, thus avoiding a manual creation. Given the amount of linked data available, it is pretty clear that the use of the full expressive power of OWL is not a realistic option. In previous work, we have therefore focussed on inductive methods for creating schema information in the EL fragment of OWL 2 [7, 26]. This fragment, while having nice computational properties, has a serious drawback with respect to describing linked data in the large as it has rather limited support for exploiting property characteristics. As most of the information in linked data is about relations between objects, knowing more about the nature of these relations a main source of inferential information. While some relevant aspects like domain and range of properties have already been investigated in our previous work, others are still missing.

In our work, we target the RL fragment of OWL 2 which combines means for deductive query answering and the ability to detect inconsistencies in the data. This paper specifically addresses characteristics of relations that have not been addressed in our previous work, thus closing the gap between the EL and the RL fragment of OWL 2. Further, we provide a systematic manual evaluation of the extracted axioms which was not done in [26].

This paper is structured as follows. In Section 2, we will give an overview about similar and related work. Afterwards, in Section 3, we introduce the basics of OWL 2 RL and the most relevant notions of Association Rule Mining. Thereafter, in Section 4, our methods for learning property characteristics from Linked Data are presented whose results are evaluated in Section 5, where we present our methodology for the evaluation and its results. Afterwards, we draw conclusions from our work and present possible directions for future work.

2 Related Work

Our work is most closely related to earlier research in the area of *linked data mining*, as well as to previous approaches that have been developed to support the knowledge acquisition process by means of *association rule learning*. In the following, we give an overview of related work in these areas also including mentions of the very few approaches to *learning property axioms* that have been described in the literature.

Early work in the field of linked data or **semantic web mining** [25] includes methods based on systematic generalization [6] and clustering [20]. Grimnes et al. [9] present an ILP-based approach to mining descriptions of groups of people from FOAF data. More recent work focuses on RDF data integration and the automated discovery of

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

[2] For example, the DBpedia ontology version 3.6 only contains domain and range restrictions for object properties but not other object property characteristics.

links. For example, Parundekar et al. [23] present an approach to generating correspondences between classes that are defined in terms of different RDF vocabularies. Based on the degree of overlap between the extensions of complex or atomic classes in two schemas, they decide whether these classes are equivalent, disjoint or subsuming each other. Many of these approaches 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. More recently, the DL-learner [14] has been applied to several RDF knowledge bases [10].

While the most well-known application of **association rule mining** is the analysis of shopping carts in recommender systems, there has also been a significant amount of research on mining for association rules in textual or RDF-based data. Textual data obtained from labels and annotation properties is exploited, e.g., by David [4], who uses association rule mining for detecting correspondences between classes in different ontologies. Mädche and Staab [19] as well rely upon natural language input, in order to enrich an ontology with non-taxonomic relations. By considering each sentence as a transaction with items being the subject and object of this sentence, they compute the association strength of any two concepts in an ontology. An approach to mining frequent patterns (e.g. < `Terrorist Group, participate, Financial Crime` >) from large sets of RDF triples is described, e.g., by Jiang and Tan [12]. Netbot and Berlanga [21] use association rule learning for discovering causal relations in the medical domain, while Lorey et al. [17], finalists of the most recent Billion Triple Challenge, discover common usage patterns in Linked Open data. By means of both positive and negative association rule mining the latter compare these patterns to existing schema definitions, in order to indicate potential modeling errors. Finally, in our own previous work, we have used association rule mining for learning class definitions [26] and disjointness axioms [7].

As indicated above there is a considerable amount of work in the area of knowledge discovery from both textual and structured data. However, only few researchers have addressed the problem of automatically generating **property axioms**. Following early work by Lin and Pantel [15], Lin et al. [16] developed a method for automatically identifying functional relationships in natural language text. A more comprehensive framework for discovering various property characteristics has been proposed by Del Vasto Terrientes et al. [5]. By matching lexico-syntactic patterns in in textual Web data they determine for each property in a given ontology, whether it is symmetric, reflexive, transivitive, functional, or inverse to some other property. Axioms involving negation, such as asymmetry or disjointness, are not covered by their approach. Further research on learning functional characteristics of relations [24] focuses on the extraction of entailment rules, e.g., in the form of Horn clauses. Though largely based on ILP and statistical relevance measures, these approaches are highly related to ours in that they follow the paradigm of inductive and statistical learning, in order to populate a knowledge base with logical implications, e.g.

```
IsHeadquarteredIn(Company, State) :- IsBasedIn(Company, City) ∧
                 IsLocatedIn(City, State);
```

Previous research on learning property axioms from RDF data includes the work by Lausen et al. [13], who suggest the use of SPARQL for stating and checking several kinds of constraints on properties in RDF graphs (i.e. property subsumption, domain and range restrictions, minimum and maximum cardinalities, as well as unary foreign key and unary key constraints). More recently, an approach specifically developed to facilitate the detection of abnormalities or modeling errors in RDF data has been proposed by Yu and Heflin [27]. The clustering-based approach computes functional dependencies like, for example, `publisher` $\circ$ `country` $\rightarrow$ `language`, which can be interpreted as "The publisher's country determines the langugage of that published work". However, none of the aforementioned approaches has yet been developed into a framework for enhancing RDF repositories with richly axiomatized OWL schemas or concise property definitions.

3 Foundations

In this section, we describe some basic notions of OWL 2,[3] focussing on the RL fragment and the semantics of object property expressions. Afterwards, we give a short overview of the mining of association rules, which is needed to understand our approach to learning property characteristics.

3.1 OWL 2 RL

With the introduction of OWL 2 as successor of OWL, many new features got introduced into the Web Ontology Language. In particular, regarding the characteristics of properties, more expressivity became available by the introduction of property axioms such as disjoint properties or asymmetry. Additionally, with OWL 2 several so-called profiles have been introduced, each of providing a restriction on the full expressivity of OWL 2 in favor of certain computational guarantees. In particular, OWL 2 RL is aimed at providing as much expressive power as possible without becoming intractable. Thus, this special profile is well-suited for being used in querying scenarios on larger datasets which also provide expressive schemas. With respect to the property axioms, the main restriction of OWL 2 RL is that it does not support reflexive properties.

In the following, we give an overview about the semantics of OWL 2 RL. Since it is based on the description logic $\mathcal{SROIQ}$, the semantics is defined inductively starting from a set of concept (or class) names N_C, a set of property (or role) names N_R and a set of individual (or instance) names N_I. The actual semantics are then defined by means of an interpretation $\mathcal{I} = (\triangle, \cdot^{\mathcal{I}})$. The non-empty set $\triangle^{\mathcal{I}}$ is called the *domain* of $\mathcal{I}$, and it contains the individuals. The interpretation function $\cdot^{\mathcal{I}}$ maps concept names $c \in N_C$ to a $c^{\mathcal{I}} \subseteq \triangle^{\mathcal{I}}$, property 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 element $a^{\mathcal{I}} \in \triangle^{\mathcal{I}}$. Using these definitions, Table 1 and 2 give the semantics of OWL 2 required to understand the contents of our paper.

Given these basic semantics, OWL 2 itself enforces some additional global restrictions on the axioms which are described in Horrocks et al. [11] Furthermore, OWL 2 RL

[3] `http://www.w3.org/TR/owl2-overview/`

Table 1. The parts of the description logic $\mathcal{SROIQ}$ as supported by the OWL 2 RL profile.

Name	Syntax	Semantics
top	$\top$	$\triangle^{\mathcal{I}}$
bottom	$\bot$	$\emptyset$
complement	$\neg C$	$\triangle^{\mathcal{I}} \setminus C^{\mathcal{I}}$
conjunction	$C \sqcap D$	$C^{\mathcal{I}} \cap D^{\mathcal{I}}$
disjunction	$C \sqcup D$	$C^{\mathcal{I}} \cup D^{\mathcal{I}}$
nominals	$\{x_1\} \sqcup \cdots \sqcup \{x_n\}$	$\{x_1^{\mathcal{I}}\} \cup \cdots \cup \{x_n^{\mathcal{I}}\}$
existential restriction	$\exists r.C$	$\{x \in \triangle^{\mathcal{I}} \vert \exists y \in \triangle^{\mathcal{I}} : (x,y) \in r^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}\}$
self operator	$\exists r.\,\text{Self}$	$\{(x,x) \in r\}$
universal restriction	$\forall r.C$	$\{x \in \triangle^{\mathcal{I}} \vert \forall y \in \triangle^{\mathcal{I}} : (x,y) \in r^{\mathcal{I}} \Rightarrow y \in C^{\mathcal{I}}\}$
minimum cardinality	$\geqslant nr.C$	$\vert\{x \in \triangle^{\mathcal{I}} \vert \#\{y \in \triangle^{\mathcal{I}} \vert (x,y) \in r^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}\} \geq n\}$
maximum cardinality	$\leqslant nr.C$	$\vert\{x \in \triangle^{\mathcal{I}} \vert \#\{y \in \triangle^{\mathcal{I}} \vert (x,y) \in r^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}\} \leq n\}$
GCI	$C \sqsubseteq D$	$C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$
RI	$r_1 \circ ... \circ r_k \sqsubseteq r$	$r_1^{\mathcal{I}} \circ ... \circ r_k^{\mathcal{I}} \subseteq r^{\mathcal{I}}$

Table 2. Property Axioms available in OWL 2 RL.

Name	Syntax	Semantics
Inverse Property	r^-	$\{(x,y) \vert (y,x) \in r^{\mathcal{I}}\}$
Symmetry	$\text{Sym}(r)$	$(x,y) \in r^{\mathcal{I}} \Rightarrow (y,x) \in r^{\mathcal{I}}$
Asymmetry	$\text{Asy}(r)$	$(x,y) \in r^{\mathcal{I}} \Rightarrow (y,x) \notin r^{\mathcal{I}}$
Transitivity	$\text{Tra}(r)$	$(x,y) \in r^{\mathcal{I}} \wedge (y,z) \in r^{\mathcal{I}} \Rightarrow (x,z) \in r^{\mathcal{I}}$
Irreflexivity	$\text{Irr}(r)$	$\{(x,x) \vert x \in \triangle^{\mathcal{I}}\} \cap r^{\mathcal{I}} = \emptyset$
Property Disjointness	$\text{Dis}(r,s)$	$r^{\mathcal{I}} \cap s^{\mathcal{I}} = \emptyset$
Functionality	$\text{Fun}(r)$	$(x,y) \in r^{\mathcal{I}} \wedge (x,z) \in r^{\mathcal{I}} \Rightarrow y = z$
Inverse Functionality	$\text{Ifu}(r)$	$(x,z) \in r^{\mathcal{I}} \wedge (y,z) \in r^{\mathcal{I}} \Rightarrow x = y$

adds several restrictions to disallow object property reflexivity as well as several syntactic restrictions which guarantuee its computational properties as anonymous individuals no longer have to be considered during inference. For example, it is not allowed to use $\top$ on the right-hand side of a subsumption axiom or in object property domain and range axioms. The full set of restrictions for OWL 2 RL is described in the profile specification of OWL 2.[4]

Having a look at the current ontologies found throughout the Internet, we can observe that OWL 2 is only rarely used, this even holds for property axioms which have already been available in OWL itself. Thus, there is not much experience with these axioms and the complexity of their usage by human modelers. To simplify the under-

[4] `http://www.w3.org/TR/owl2-profiles/#OWL_2_RL`

standing of these axiom types, we provide some examples for possible applications of these property axiom types in Table 3.

Table 3. Examples for object property axioms.

Axiom	Description
hasParent $\equiv$ *hasChild*$^{-}$	If a is the parent of b then b must be the child of a.
$\mathrm{Sym}($*isFriendOf*$)$	Formalizes the understanding that friendship is mutual. Thus, if a is a friend of b then b is also a friend of a.
$\mathrm{Asy}($*isCommanderOf*$)$	If a is the commander of b then b cannot be the commander of a at the same time.
$\mathrm{Tra}($*hasAncestor*$)$	If a has an ancestor b and b has an ancestor c then also a has the ancestor c.
$\mathrm{Irr}($*isGreaterThan*$)$	No a can be greater than itself, i.e., for no a the statement $a > a$ holds.
$\mathrm{Dis}($*isParentOf*, *isChildOf*$)$	If a is the parent of b then a cannot be the child of b at the same time.

3.2 Association Rule Mining

Our approach to generating property axioms bases on association rule mining. Originally association rule mining has been developed for analyzing shopping carts of huge shops for products commonly bought in combination to enable better recommendations. Due to these use cases association rule mining algorithms are developed for large and sparse datasets and thus well-suited to process the massive amounts of information found in the Linked Data cloud.

Formally, association rule mining operates on a transaction database $D = (t_1, t_2, \ldots, t_n)$ containing transactions $t_j \subseteq I$ where I is the set of all possible items. For computationally processing the data, transaction databases are usually represented as so-called transaction tables where each line corresponds to one transaction and contains the numerical IDs of all items belonging to the specific transaction.

For generating association rules from a transaction database, it is first required to search it for frequent itemsets. For this purpose, there are a number of different algorithms. A commonly used one is the Apriori algorithm [1]. Frequent itemsets are itemsets $X \subseteq I$ exceeding a certain support value

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

which is the number of transactions containing all the items from X. Based on these frequent itemsets, association rules of the form $A \rightarrow B$ are generated. Their quality regarding the specific transaction database is assessed by means of the confidence value

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

which corresponds to the ratio of transactions the specific association rule is valid for compared to the number of transaction it could be applied to. Using this confidence value, the generated rules can be filtered to guarantee a certain level of quality for the specific dataset.

For our work, an extension to the association rule mining framework described above is required. Usually, association rules are used to capture positive implications, i.e., the existence of one item implies the existence of another one. For the extraction of negative rules, e.g., $\neg A \rightarrow B$ or $\neg A \rightarrow \neg B$, which represent implications depending on the absence of items in the transactions, special algorithms have been proposed [2, 18, 28]. The most basic method to also include absent items into the generation of association rules, is the introduction of artificial items for each original item A representing its complement $\neg A$, i.e., a transaction contains this item $\neg A$ if and only if it does not contain the original item A. This enables the direct application of the common algorithms on the resulting transaction database.

4 Method

We use the association rule mining approaches introduced in the previous section to learn OWL axioms from instance data acquired from a Linked Data repository. In previous works [7, 26], association rules have been used to learn concept-centric axioms like subsumption and concept disjointness and some property-centric ones like property subsumption as well as domain and range restrictions. In this paper, we focus on learning property axioms that have been neglected previously. We first have to translate the instance data contained in a Linked Data Repository into a suitable transaction database. Afterwards, we are able to run the Apriori algorithm on the transaction databases before generating the OWL axioms from the discovered association rules. This workflow is depicted in Figure 1. In the following, we first give an example which illustrates the different steps of the workflow for generating symmetric object properties. Having illustrated the complete workflow for one case, we will only provide shorter descriptions for all other cases.

4.1 Workflow Example: Object Property Symmetry

The basis for using statistical schema induction is obviously a dataset of sufficient size. We assume this dataset to be available through a SPARQL endpoint so that we are able to extract the required information using queries.

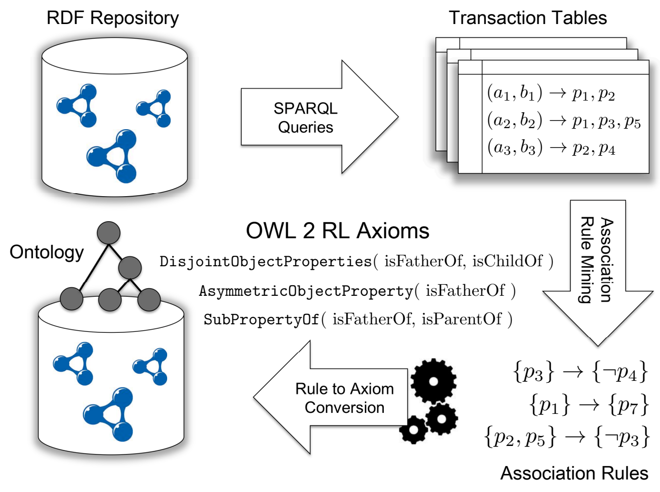
Terminology Acquisition: In the terminology acquisition phase, we gather information about the concepts, properties and instances contained in the data set. During this phase, we also introduce unique numerical identifiers for the terminology.

First, we extract all concepts used in the RDF dataset. This is done querying the endpoint for

```
SELECT DISTINCT ?c WHERE {[] a ?c}
```

The resulting class URIs are filtered by their namespace so that only those classes are taken which belong to the dataset itself. In particular, this filters out the RDF, RDFS and OWL vocabulary. Each of the remaining concepts gets assigned a unique numerical identifier.

Afterwards, the instances are extracted. These are extracted from the SPARQL endpoint by means of the query



RDF Repository
SPARQL Queries
Transaction Tables
$(a_1, b_1) \to p_1, p_2$
$(a_2, b_2) \to p_1, p_3, p_5$
$(a_3, b_3) \to p_2, p_4$
Association Rule Mining
OWL 2 RL Axioms
Ontology
DisjointObjectProperties(isFatherOf, isChildOf)
AsymmetricObjectProperty(isFatherOf)
SubPropertyOf(isFatherOf, isParentOf)
Rule to Axiom Conversion
$\{p_3\} \to \{\neg p_4\}$
$\{p_1\} \to \{p_7\}$
$\{p_2, p_5\} \to \{\neg p_3\}$
Association Rules

Fig. 1. Workflow for generating expressive ontologies out of Linked Data

```
SELECT DISTINCT ?i WHERE {?i a <CLASS URI>}
```

where `CLASS URI` is successively replaced with each class URI extracted the step before. Thus, we only extract instances of relevant concepts.

All of these instances get a unique numerical identifier assigned like it has already been done by Völker and Niepert [26]. Next, a crucial step for working with properties, we generate all possible pairs of instances. To limit the load put on the endpoint this is done locally. Since the number of all instance pairs might be too large to efficiently process all of them, we only consider pairs of instances (i_1, i_2) for which a property exists which connects both directly. This is done querying the endpoint for

```
ASK {<URI INSTANCE 1> ?p <URI INSTANCE 2>}
```

after inserting the respective instance URIs into the query template. It is also worth mentioning that the order of the instances in the pair is relevant and also pairs are generated which consist of the same instance twice as seen in Table 4.

So far, the process is the same for all types of property axioms. For the generation of property symmetry, we next extract all properties contained in the RDF dataset using the SPARQL query

```
SELECT DISTINCT ?p WHERE {[] ?p []}
```

which is then filtered for the respective URI prefix like `http://dbpedia.org` for the DBpedia dataset. After assigning the identifiers, we have a URI to ID assignment

Table 4. Table of instance pairs. The URI prefix `http://dbpedia.org/resource/` has been omitted for all instances.

Pair	Instance 1	Instance 2
1	`Pepsi`	`Pepsi`
2	`Pepsi`	`Coca-Cola`
3	`Coca-Cola`	`Pepsi`
4	`Pepsi`	`United_States`
...	...	...

Table 5. Property URI to ID Table

Object Property URI	Property ID	Inverse Property ID
`http://dbpedia.org/ontology/related`	20	21
`http://dbpedia.org/ontology/origin`	22	23
`http://dbpedia.org/ontology/highestState`	24	25
`http://dbpedia.org/ontology/coolingSystem`	26	27
...	...	...

like in the *Property ID* column of Table 5. For symmetry, we also need another ID to be introduced which we will refer to as *Inverse Property ID* later on.

Transaction Table Generation: During the next acquisition phase, the actual transaction tables are generated. To generate symmetry axioms, a transaction table is generated whose rows represent the pairs of instances acquired in the previous phase. For each pair of instances, the endpoint is asked for all properties connecting both instances

```
SELECT DISTINCT ?p
WHERE {<URI INSTANCE 1> ?p <URI INSTANCE 2>}
```

This leads to a list of properties holding between both instances for each of which the corresponding property identifier is looked up and added to the transaction row. In addition, for each property URI returned by the query

```
SELECT DISTINCT ?p
WHERE {<URI INSTANCE 2> ?p <URI INSTANCE 1>}
```

the inverse property identifier is also added to the current transaction table row. This detailed representation is usually very sparse, i.e., they contain many zero values. This fact is utilized for the serialized representation of the transaction tables which only contains rows for transactions actually containing values and only those values which represent items contained in the transaction. Furthermore, the relation between the transaction rows and the instance pairs accounting for it is not necessary with regard to the process of association rule mining and thus not maintained in the serialized table. Table 6 shows such a table containing only the identifiers necessary in the serialized format which is also the format to pass to association rule mining systems.

Table 6. Serialization of Transaction Table for Object Property Symmetry. × marks all properties used between the given pair of instances.

	related[20]	origin[22]	related⁻[21]	origin⁻[23]	
(Pepsi, Pepsi)					...
(Pepsi, Coca-Cola)	×		×		...
(Coca-Cola, Pepsi)	×		×		...
(Pepsi, United_States)		×			...
(United_States, Pepsi)				×	...
...	...	...	...	...	...

Association Rule Mining: The serialized format of the transaction tables acts as input for association rule mining tools like the apriori miner by Borgelt [3]. Depending on the settings like thresholds values, the execution of this tools leads to results like

$$\texttt{related}^{-}\texttt{[21]} \rightarrow \texttt{related[20]}(50, 100)$$

$$\texttt{related[20]} \rightarrow \texttt{related}^{-}\texttt{[21]}(50, 100)$$

where the numbers in square brackets are the properties' IDs in the terminology table, and the values in round parentheses provide the support and confidence for the corresponding association rule as described in Section 3.2.

Axiom Generation: After their creation, the results from the association rule mining phase are parsed for specific patterns (see Table 7) which indicate the desired axiom types. In case of property symmetry, we try to observe patterns like $\{p\} \Rightarrow \{p^{-}\}$, i.e., the first identifier is a normal property identifier and the second one an inverse property identifier for the same property. The parsed association rules are then filtered by applying a user-specified confidence threshold to the confidence values gained during mining the association rules.

Note, that we do no special treatment of property axioms being symmetric themselves like property disjointness or inverse properties, although these may be indicated by multiple association rules as exemplified above. Rather, we apply the same approach as for all other axiom types and generate a property axiom if at least one of the mined association rules has the exceeds the confidence threshold.

If an association rule fits in a specific pattern and exceeds the confidence threshold, the symmetry axiom matching this association rule is generated. For the association rule pattern given above, the corresponding symmetry axiom would be $\mathrm{Sym}(p)$ annotated with the confidence value of the association rule. In our example, only the second association rule fits into the required pattern. Since the confidence is 100, which is the maximum confidence, it would not be filtered out by any settings for confidence thresholds. The resulting axiom is a symmetry axiom for the property `related`.

$$\mathrm{Sym}(\texttt{related})$$

4.2 Overview of other Property Axioms

In this section, we present the transaction tables used for the property axiom types apart from property symmetry. This is first done in a descriptive way and then summarized in

Table 7. This table also provides information on the conversion from association rules to axioms. The foundation for most transaction tables we use in the following is an itemset I containing items for all properties r in the repository. Each transaction in the table represents one possible pair of instances $(a,\ b)$ and contains all property-items for properties p for which $a\,r\ \ b$ holds in the repository. We refer to this itemset and this style of transaction contents as *default* itemset resp. transactions.

Object Property Subsumption and Disjointness: Like for transitivity as well as domain and range shown beneath, object property subsumption has already been introduced by Völker and Niepert [26] before. It is very similar to basic concept subsumption regarding the transaction table. Here, we extend the approach to also cover object property disjointness by introducing artificial *complement-items* (written as $\neg r$) into the default itemset I for each property. Each transaction contains the complement-item for a property if and only if it does not contain the corresponding property-item. This enables us to generate association rules for object property subsumption, i.e., an association rule between two property-items, and object property disjointness, i.e., an association rule between a complement-item and a property-item.

Object Property Transitivity: The representation of transitivity in transaction tables is a bit more complex than for subsumption or disjointness. The itemset I contains the default items described above. In addition, it contains items for each two-property compositions of the same property, i.e., $r \circ r$. A transaction is extended by the item for $r \circ r$ if and only if $a\ (r \circ r)\ b$.

Object Property Domain and Range: For the mining of domain and range axioms, the itemset I does not contain the default items but one item for each concept used in the repository. For each property r in the repository, we add a universal-existence item ($\exists r.\top$) and another universal-existence item for the inverse of r (written as $\exists r^{-}.\top$). Each transaction represents one instance in the repository. The concept items C are added to the transaction if the corresponding instance is an instance of this concept. Furthermore, for the domain restriction, the universal-existence items are added to a transaction if the corresponding property is used with this instance as subject. For range restrictions, if the corresponding instance is used as an object of the property, $\exists r^{-}.\top$ is added to the transaction.

Object Property Symmetry, Asymmetry and Inverse: For these characteristics, we add the inverse-property (symmetry and inverse property) and complement-items (asymmetry) for each property into I. In addition to the default items, each transaction gets the complement-items as for the disjointness or the inverse-property item as for symmetry. Thus, we are able to capture the inverse direction of properties.

Object Property Functionality and Inverse Functionality: In contrast to all other variants, for functionality we only use the default contents for I but not for the transactions. Both characteristics are treated very similar but each gets its own transaction table. The default itemset is extended by functionality resp. inverse functionality items for each property. Each transaction represents a single instance and gets added a property-item if and only if the instance is used as subject (for functionality) resp. domain (inverse functionality) of the corresponding property. The (inverse) functionality item for a specific property is added to the transaction if the property is only used at most once with the given instance as domain resp. range. Thus, we are able to get associations between the usage of a property and its (inverse) functional usage.

Object Property Reflexivity and Irreflexivity: Even if OWL 2 RL does not support property reflexivity, we are aware of the fact that it might be useful for some use cases. Therefore, we also give the transaction tables for reflexivity here. We capture reflexivity by introducing another item into the default itemset I, the equality item $t_{a=b}$. For irreflexivity, we also add the complement-items as for property disjointness into I and these complement-items are also added to the transactions using the same approach as for property disjointness. Furthermore, we extend each transaction by the equality item if $a = b$ holds for the corresponding item pair (a, b). Using these transactions, we are able to get relations between the existence of a specific property between two instances and their equality.

Table 7. Summary of Transaction Table Generation. Each row in the transaction table either corresponds to one instance $a \in N_I$ or a tuple of instances $(a, b) \in N_I \times N_I$. A concept item C_i is added to the transaction if $a \in C_i^{\mathcal{I}}$ for the corresponding instance a. A property item r_i is added to the transaction if the corresponding property holds between the instance tuple, i.e., $a\ r_i\ b$, the property complement item $\neg r_i$ is added if $a\ r_i\ b$ does not hold. r_i^- is added if $b\ r_i\ a$, $\neg r_i^-$ if not $b\ r_i\ a$. $t_{a=b}$ is contained in a transaction if $a = b$. $r_i \circ r_i$ is contained in a transaction if $a\ r_i\ x$ and $x\ r_i\ b$ for an arbitrary instance $x \in N_I$. $\exists r.\top$ is added if the instance a is used as subject of r, $\exists r^-.\top$ if it is used as object. The items $t_{sub_{\leqslant 1}}(r)$ and $t_{obj_{\leqslant 1}}(r)$ are added to a transaction if the current instance a is used at most once as subject resp. object of the property r.

Axiom Type	Transaction Table	Association Rule
$\mathrm{Sym}(r)$	$(a, b) \rightarrow r_1, \ldots, r_n, r_1^-, \ldots, r_n^-$	$\{r\} \Rightarrow \{r^-\}$
$r_i \sqsubseteq \neg r_j$	$(a, b) \rightarrow r_1, \ldots, r_n, \neg r_1, \ldots, \neg r_n$	$\{r_i\} \Rightarrow \{\neg r_j\}$
$\mathrm{Ref}(r)$	$(a, b) \rightarrow r_1, \ldots, r_n, t_{a=b}$	$\{t_{a=b}\} \Rightarrow \{r\}$
$\mathrm{Irr}(r)$	$(a, b) \rightarrow \neg r_1, \ldots, \neg r_n, t_{a=b}$	$\{t_{a=b}\} \Rightarrow \{\neg r\}$
$\mathrm{Asy}(r)$	$(a, b) \rightarrow r_1, \ldots, r_n, \neg r_1^-, \ldots, \neg r_n^-$	$\{r\} \Rightarrow \{\neg r^-\}$
$r_i \sqsubseteq r_j^-$	$(a, b) \rightarrow r_1, \ldots, r_n, r_1^-, \ldots, r_n^-$	$\{r_i\} \Rightarrow \{r_j^-\}$
$\mathrm{Fun}(r)$	$a \rightarrow \exists r_1.\top, \ldots, \exists r_1.\top, t_{sub_{\leqslant 1}}(r_1), \ldots, t_{sub_{\leqslant 1}}(r_n)$	$\{\exists r_1.\top\} \Rightarrow \{t_{sub_{\leqslant 1}}(r)\}$
$\mathrm{Ifu}(r)$	$a \rightarrow \exists r_1^-.\top, \ldots, \exists r_n^-.\top, t_{obj_{\leqslant 1}}(r_1), \ldots, t_{obj_{\leqslant 1}}(r_n)$	$\{\exists r_1^-.\top\} \Rightarrow \{t_{obj_{\leqslant 1}}(r)\}$
$r_i \sqsubseteq r_j$	$(a, b) \rightarrow r_1, \ldots, r_n$	$\{r_i\} \Rightarrow \{r_j\}$
$r \circ r \sqsubseteq r$	$(a, b) \rightarrow r_1, \ldots, r_n, r_1 \circ r_1, \ldots, r_n \circ r_n$	$r_i \circ r_i \Rightarrow r_i$
$\exists r.\top \sqsubseteq C$	$a \rightarrow C_1, \ldots, C_l, \exists r_1.\top, \ldots, \exists r_n.\top$	$\{\exists r.\top\} \Rightarrow \{C\}$
$\exists r^-.\top \sqsubseteq C$	$a \rightarrow C_1, \ldots, C_l, \exists r_1^-.\top, \ldots, \exists r_n^-.\top$	$\{\exists r^-.\top\} \Rightarrow \{C\}$

5 Experiments

For evaluating our approach, we applied it to DBpedia, one of the largest and most well-known datasets available in the Linked Data cloud. Besides its size, another advantage of DBpedia is that it is based on data from Wikipedia and thus nearer to the knowledge we commonly work with. Thus, axioms generated by our method are easier to evaluate for their correctness than, for example, medical data would be.

5.1 Setting

We run our miner on the DBpedia dataset version 3.7 as released in November 2011 and built from Wikipedia dumps from late July 2011. This DBpedia dataset was made available through a local SPARQL endpoint running the Virtuoso software in version 06.01.3127. The transaction table generation and the conversion from association rules into the corresponding OWL 2 RL axioms has been implemented in the GOLD Miner system,[5] while the actual association rule mining was done by the apriori miner [3].

We performed the experiments applying three different confidence thresholds, namely 0.5, 0.75 and 1.0, to examine in how far higher confidence thresholds lead to a higher degree of correct axioms. The support threshold was fixed at an absolute value of 1 which actually means that we considered all association rules without enforcing a certain support level.

All experiments have been conducted on a Linux system equipped with an Intel Core i7 3.07GHz quad-core processor and 24 GB main memory. The most time-consuming step during the whole experiments phase, is the creation of the transaction tables. Depending on the specific property axiom type the table is generated for, this step may need several hours to finish. We do not provide exact numbers here, since the runtime greatly depends on the performance of the SPARQL endpoint providing access to the data and thus is not transferabble to other datasets. The size of the created transaction table files varies to a great extent, starting from about 6 MB for the table containing data to deduce domain or range axioms from to up to 22 GB for the property disjointness table. Nevertheless, the actual execution time of the association rule mining on these datasets is in the magnitude of minutes or even seconds. The mining step took about 20 minutes for processing the tables containing nearly 17 million transactions in total. The numbers of axioms generated from the resulting association rules for the different confidence thresholds are given in Table 8. We concentrated on the axiom types newly introduced in this paper and supported in OWL 2 RL. Therefore, characteristics like property subsumption and reflexivity have not been generated.

Table 8. Total number of generated axioms for given confidence thresholds.

minimum confidence	**0.5**	**0.75**	**1.0**
AsymmetricObjectProperty	435	410	384
DisjointObjectProperties	180,876	180,850	178,660
FunctionalObjectProperty	354	268	67
InverseObjectProperties	6	3	0
SymmetricObjectProperty	4	2	0
TransitiveObjectProperty	246	219	141
Total	181,921	181,752	179,252

5.2 Manual Evaluation

Expert Evaluation For the generated axioms, we did several manual evaluations to assess the performance of our approach. First, we randomly chose a subset of 40 axioms

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

Table 9. Evaluation results for subsets of generated axioms with **confidence threshold 0.5**. Listing the number of axioms evaluated by the annotators (# eval), axioms rated as correct (# corr), accuracy (Acc.) and the inter-annotator agreement by means of Fleiss' Kappa. Axioms evaluated by experts are not contained in the crowd evaluation, '-' shows that all generated axioms have been evaluated by experts.

Axiom Type	**Experts**				**Crowd**			
	# eval	# corr	Acc.	Fleiss	# eval	# corr	Acc.	Fleiss
AsymmetricObjectProperty	40	37	0.93	0.96	98	96	0.97	0.81
InverseObjectProperties	6	2	0.33	0.85	-	-	-	-
DisjointObjectProperties	40	39	0.98	0.99	261	258	0.99	0.90
TransitiveObjectProperty	40	2	0.05	0.91	159	32	0.20	0.51
FunctionalObjectProperty	40	9	0.23	0.80	175	48	0.27	0.51
SymmetricObjectProperty	4	3	0.75	0.95	-	-	-	-
Total	170	92	0.54	0.91	697	437	0.63	0.70

Table 10. Evaluation results for axiom subset with **confidence threshold 0.75**.

Axiom Type	**Experts**				**Crowd**			
	# eval	# corr	Acc.	Fleiss	# eval	# corr	Acc.	Fleiss
AsymmetricObjectProperty	24	22	0.92	0.96	93	92	0.99	0.82
InverseObjectProperties	2	2	1.00	1.00	-	-	-	-
DisjointObjectProperties	40	39	0.98	0.99	261	258	0.99	0.90
TransitiveObjectProperty	35	1	0.03	0.91	141	29	0.21	0.50
FunctionalObjectProperty	23	9	0.39	0.78	138	40	0.29	0.51
SymmetricObjectProperty	2	2	1.00	1.00	-	-	-	-
Total	126	75	0.60	0.92	633	419	0.66	0.72

per property axiom type or all axioms if less than 40 were generated. These axioms were evaluated by three ontology engineers for correctness on a two-valued scale, consisting of the categories *correct* and *wrong*. For the evaluation, the axioms were presented to the experts in OWL 2 Functional Syntax as well as by means of a natural language sentence like "*If A* `thirdDriverCountry` *B holds then B* `thirdDriverCountry` *A must not hold*" trying to clarify the semantics more. Furthermore, the annotators had access to the DBpedia page describing the relevant object properties. Since in some cases the naming of the object property alone is not enough to understand its meaning, the annotators had access to the results of the SPARQL query

```
SELECT ?a ?b WHERE {?a <PROPERTY URI> ?b}
```

The results of this part of the evaluation are contained in the *Expert* columns of Table 9, 10 and 11, each of which contains the result for one specific confidence threshold. These tables contain the numbers of evaluated and correct axioms, the accuracy of the learned axioms and the inter-annotator agreement determined by means of averaging the absolute values of Fleiss' kappa [8] for all axioms.

There are several observations which can be made from these numbers. First of all, it is noticeable that changes with regard to the confidence values did not have great influence on the accuracy of our results. Most accuracy values did not change considerably when applying a higher confidence threshold. High values confidence values stayed nearly the same for all thresholds and only lower confidence values saw a slight rise in some cases. Another observation is that low accuracy values like for functional properties came with a lower inter-annotator agreement. Only the property transitivity com-

Table 11. Evaluation results for axiom subset with **confidence threshold 1.0**. No inverse or symmetric properties have been generated for this threshold.

Axiom Type	**Experts**				**Crowd**			
	# eval	# corr	Acc.	Fleiss	# eval	# corr	Acc.	Fleiss
AsymmetricObjectProperty	21	19	0.90	0.95	89	89	1.0	0.81
DisjointObjectProperties	40	39	0.97	0.99	259	256	0.99	0.9
TransitiveObjectProperty	25	0	0.00	0.90	88	19	0.22	0.50
FunctionalObjectProperty	8	2	0.25	0.63	26	9	0.35	0.54
Total	94	60	0.64	0.93	462	373	0.81	0.79

bined higher values with respect to agreement with extremely low values in accuracy. This is one aspect in which we investigated further. For example, we had a closer look at the object property `birthPlace` for which an obviously wrong transitivity characteristic was generated.[6] One result set exhibiting the basic pattern of transitivity are the instances `Brian_Vandborg`, `Denmark` and `Peter_Snejbjerg`. Even though both `Brian_Vandborg` and `Peter_Snejbjerg` are humans, the `birthPlace` relation between both instances is verifiably valid in DBpedia. All in all, this results in the pattern shown in Figure 2. A closer look at the Wikipedia article disclosed that *Brian Vandborg* was born in *Snejbjerg, Denmark*. In this case, we could identify two sources for the errors. First, the word *Snejbjerg* is linked incorrectly to the person *Peter Snejbjerg* and second, the DBpedia parser extracts *Denmark* as a `birthPlace` on its own. Actually, the country should be already contained in the city assigned as `birthPlace` and thus would be redundant information.

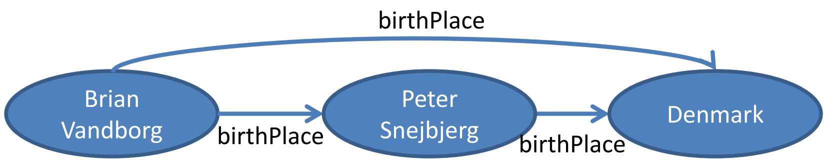


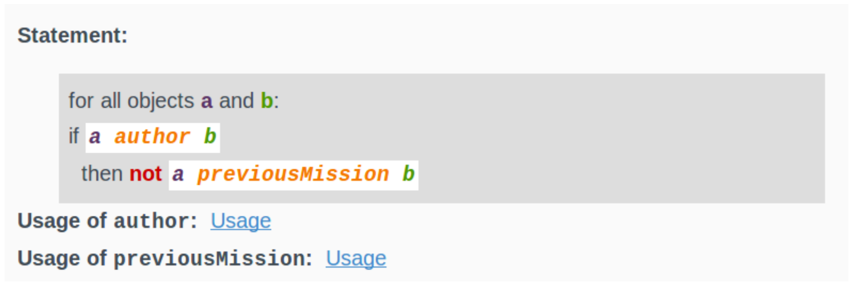
Fig. 2. Graphical representation of triples leading to wrong transitive property axioms.

We also identified more similar errors in DBpedia leading to many errors in our learned transitivity axioms. Not all of these were traceable to a Wikipedia error which means that the DBpedia extractor itself introduced errors in the DBpedia data. However, this also illustrates that our learned axioms are helpful for discovering errors in datasets.

[6] For this investigation, we used the SPARQL query whose results are available at the shortened URL `http://is.gd/xfyuZW` as provided by the DBpedia endpoint.

Errors in Wikipedia and DBpedia are also to the reason for our approach not to generate any irreflexivity axioms since many instances have erroneous self-references.

Crowd Evaluation Since the number of generated axioms is very high, an extensive evaluation by experts is hardly doable. To have a more extensive evaluation and also to assess the possibilities and limits of this evaluation method, we performed an evaluation using crowdsourcing. We chose CrowdFlower which provides quality assurance services for crowdsourcing [22] but delegates the actual work to platforms like Amazon Mechanical Turk. These quality assurance services need a set of gold standard answers to assess the trust of workers. For this purpose, we used those axioms the experts agreed on when evaluating. In addition, we randomly chose learned axioms from the three confidence levels and let them evaluate by the crowdsourcing workers. To simplify the task for people without ontology engineering experience, we provided the axioms as in Figure 3. Accompanied by a description of the task and some examples, the workers also had access to *Usage examples* which were retrieved from DBpedia. The results for the different confidence thresholds are presented in the *Crowd* column of Table 9, 10 and 11. It has to be mentioned that the sets of axioms to be evaluated by experts and by crowd workers are disjoint. Thus, there are no additional axioms evaluated by crowd workers for axioms with very few learned occurences.


Statement:

for all objects **a** and **b**:
if ***a author b***
then **not** ***a previousMission b***

Usage of author: Usage
Usage of previousMission: Usage

Fig. 3. Axiom evaluation task as presented to the crowdsourcing participants.

A major problem during the crowdsourced evaluation has been the naming of the DBpedia properties. Many properties are confusingly named without a real chance to deduce their correct usage only from their name. For example, the property `training` is used to connect an athlete or sports team to the corresponding training location even if the naming implies a usage like, e.g., connecting the trainer and the trainee. This was also one problem mentioned by the crowd workers themselves.

All in all, the crowdsourcing results are very similar to the ones we got from the expert evaluation. Therefore, we are confident that our expert results also hold for the larger dataset and not only the evaluated subset. We let an expert evaluate 190 crowd-evaluated axioms again and got an agreement of 80 % between the crowd and experts. Furthermore, crowdsourcing seems to be a proper mean for evaluating such results.

Even if we did no experiments on datasets different from DBpedia for this specific approach up to now, in previous experiments, we demonstrated the applicability of statistical schema induction to DBpedia as well as to the `data.gov.uk` data set [26]. Thus, we are certain that the approach presented here will perform comparably well on other datasets.

6 Conclusion & Outlook

In this paper, we have presented an approach to mining RDF data for various types of property axioms. By thus offering efficient means for enriching RDF knowledge bases by OWL 2 RL schemas we have laid the foundation for precise and tractable query answering on the Linked Data cloud. Further use cases for statistical schema induction include the inference-based diagnosis, debugging of linked data sources and repair of modeling flaws in RDF knowledge bases, as well as automated data integration.

In the near future, we will investigate these application scenarios, and evaluate our approach on other data sets, such as Freebase.[7] In order to reduce the computational overhead for generating the transaction tables, we will develop an incremental mining approach that can deal with frequent changes to the underlying RDF graph. Finally, we have already started to extend our approach in a way that it discovers further types of axioms such as cardinality constraints, for example. The discovery of axioms involving datatype properties is also possible with minor adaptations to our current methods for terminology acquisition and rule mining.

References

1. Agrawal, R., Srikant, R.: Fast algorithms for mining association rules. In: Proc. of the 20th Intl. Conference on Very Large Data Bases (VLDB). pp. 487–499. Morgan Kaufmann (1994)
2. Antonie, M.L., Zaïane, O.R.: Mining positive and negative association rules: An approach for confined rules. In: Proceedings of the 8th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD). pp. 27–38 (2004)
3. Borgelt, C., Kruse, R.: Induction of association rules: Apriori implementation. In: Proc. of the 15th Conference on Computational Statistics (COMPSTAT). pp. 395–400. Physica Verlag (2002)
4. David, J., Guillet, F., Briand, H.: Association rule ontology matching approach. International Journal on Semantic Web and Information Systems 3(2), 27–49 (2007)
5. Del Vasto Terrientes, L., Moreno, A., Sánchez, D.: Discovery of relation axioms from the web. In: Proceedings of the 4th international conference on Knowledge science, engineering and management. pp. 222–233. KSEM'10, Springer-Verlag, Berlin, Heidelberg (2010)
6. Delteil, A., Faron-Zucker, C., Dieng, R.: Learning ontologies from rdf annotations. In: 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)
7. Fleischhacker, D., Völker, J.: Inductive learning of disjointness axioms. In: Proceedings of the 2011 Confederated international conference on On the move to meaningful internet systems - Volume Part II. pp. 680–697. OTM'11, Springer-Verlag, Berlin, Heidelberg (2011)
8. Fleiss, J.L.: Measuring nominal scale agreement among many rater. Psychological Bulletin 76, 378–382 (1971)
9. Grimnes, G.A., Edwards, P., Preece, A.D.: Learning meta-descriptions of the FOAF network. In: Proceedings of the 3rd International Semantic Web Conference (ISWC). LNCS, vol. 3298, pp. 152–165. Springer (November 2004)
10. Hellmann, S., Lehmann, J., Auer, S.: Learning of OWL class descriptions on very large knowledge bases. Intl. Journal on Semantic Web and Information Systems 5(2), 25–48 (2009)

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

11. Horrocks, I., Kutz, O., Sattler, U.: The even more irresistible SROIQ. In: Proceedings of the 10th International Conference on Principles of Knowledge Representation and Reasoning 2006. pp. 57–67. AAAI Press (2006)
12. Jiang, T., Tan, A.H.: Mining rdf metadata for generalized association rules. In: Proceedings of the 17th international conference on Database and Expert Systems Applications (DEXA). LNCS, vol. 4080, pp. 223–233. Springer-Verlag, Berlin, Heidelberg (2006)
13. Lausen, G., Meier, M., Schmidt, M.: Sparqling constraints for rdf. In: Proceedings of the 11th international conference on Extending database technology (EDBT): Advances in database technology. pp. 499–509. ACM, New York, NY, USA (2008)
14. Lehmann, J.: DL-Learner: learning concepts in description logics. Journal of Machine Learning Research (JMLR) 10, 2639–2642 (2009)
15. Lin, D., Pantel, P.: Dirt – discovery of inference rules from text. In: Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining (KDD). pp. 323–328. ACM, New York, NY, USA (2001)
16. Lin, T., Mausam, Etzioni, O.: Identifying functional relations in web text. In: Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing. pp. 1266–1276. Association for Computational Linguistics, Cambridge, MA (October 2010)
17. Lorey, J., Abedjan, Z., Naumann, F., Böhm, C.: Rdf ontology (re-)engineering through large-scale data mining. In: International Semantic Web Conference (ISWC) (November 2011), finalist of the Billion Triple Challenge
18. Morzy, M.: Efficient mining of dissociation rules. In: Proceedings of the 8th international conference on Data Warehousing and Knowledge Discovery. pp. 228–237. Springer-Verlag, Berlin, Heidelberg (2006)
19. Mädche, A., Staab, S.: Discovering conceptual relations from text. In: Proceedings of the 14th European Conference on Artificial Intelligence (ECAI). pp. 321–325. IOS Press (2000)
20. Mädche, A., Zacharias, V.: Clustering ontology-based metadata in the semantic web. In: Proceedings of the 6th European Conference on Principles of Data Mining and Knowledge Discovery (PKDD). LNCS, vol. 2431, pp. 348–360. Springer (2002)
21. Nebot, V., Berlanga, R.: Mining association rules from semantic web data. In: Proc. of the 23rd Intl. Conference on Industrial Engineering and other Applications of Applied Intelligent Systems (IEA/AIE), Part II. LNAI, vol. 6097, pp. 504–513. Springer (2010)
22. Oleson, D., Sorokin, A., Laughlin, G.P., Hester, V., Le, J., Biewald, L.: Programmatic gold: Targeted and scalable quality assurance in crowdsourcing. In: Human Computation. AAAI Workshops, vol. WS-11-11. AAAI (2011)
23. Parundekar, R., Knoblock, C.A., Ambite, J.L.: Linking and building ontologies of linked data. In: Proc of the 9th Intl. Semantic Web Conference (ISWC) (November 2010)
24. Schoenmackers, S., Etzioni, O., Weld, D.S., Davis, J.: Learning first-order horn clauses from web text. In: Proceedings of the International Conference on Empirical Methods in Natural Language Processing (EMNLP). pp. 1088–1098. ACL (October 2010)
25. 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)
26. Völker, J., Niepert, M.: Statistical schema induction. In: The Semantic Web: Research and Applications. LNCS, vol. 6643, pp. 124–138. Springer Berlin / Heidelberg (2011)
27. Yu, Y., Heflin, J.: Extending functional dependency to detect abnormal data in rdf graphs. In: Proceedings of the 10th International Semantic Web Conference (ISWC). LNCS, vol. 7031, pp. 794–809. Springer (October 2011)
28. Zhang, C., Zhang, S.: Association rule mining: models and algorithms. Springer Berlin / Heidelberg (2002)