RDF Ontology (Re-)Engineering through Large-scale Data Mining

Johannes Lorey Ziawasch Abedjan Felix Naumann Christoph Böhm
firstname.lastname@hpi.uni-potsdam.de

Hasso Plattner Institute, Potsdam, Germany

Abstract. As Linked Open Data originates from various sources, leveraging well-defined ontologies aids integration. However, oftentimes the utilization of RDF vocabularies by data publishers differs from the intended application envisioned by ontology engineers. Especially in large-scale datasets as presented in the Billion Triple Challenge a significant divergence between vocabulary specification and usage patterns can be observed. This may impede the goals of the Web of Data in terms of discovering domain-specific information in the Semantic Web. In this work, we identify common misusage patterns by employing frequency analysis and rule mining and propose reengineering suggestions.

1 Introduction

The concept of Linked Open Data (LOD) enables information providers to create and share linkable data across the Web. Besides the increasing number of LOD sources, there also exist a number of rules for publishing Linked Data [2]. However, consuming and integrating LOD still necessitates thorough analysis and study of the data sources, because individual data publishers may have different understandings or knowledge of useful vocabulary definitions. Here, reusable knowledge bases and ontologies facilitate understanding and integrating multiple data sources. They provide metadata to define the domains and ranges of properties of or taxonomical relationship between resources.

Nevertheless, there is a discrepancy between well-defined ontologies and the principles of LOD. While ontologies try to comprise homogeneous data sources and define their structure, LOD is meant to be flexible and independent of other sources. Thus, integration of LOD data, ontology discovery and matching pose major challenges for the pervasion of the Web of Data. While there do exist best practices for how to publish Linked Data using common ontologies [5], our analysis shows that due to various reasons certain misusage patterns occur frequently. This partly stems from the fact that ontologies may either be too specific or too generic. In addition, custom namespace-specific properties may be added to an ontology concept when need arises. It is likely that at least some of these properties are redundant, as data publishers are unaware of one another's additions.

Redesign of ontologies and their taxonomies requires analysis and mining of the underlying data. For example, if we observe that for a large number

of instances of type `http://xmlns.com/foaf/0.1/Agent`, the assigned property `http://xmlns.com/foaf/0.1/gender` is not set, one could conclude misplacement of this predicate and shift it to a more appropriate subclass, e.g., `http://xmlns.com/foaf/0.1/Person`. Addressing these quality issues of an ontology can be considered a form of schema analysis. Here, a schema defines the set of predicates whose domain is a specific concept within the ontology base.

We use association rule mining for schema discovery and evaluate the mismatch of extension and intension of the schemata inferred from the data against defined specification of the ontologies. Mining both positive and negative rules enables us to detect properties likely to appear in combination as well as exclusive of one another. This in turn helps to evaluate which properties are indeed appropriately declared for certain classes and which ones could instead be defined for other (sub-)classes.

There have been case studies about the pervasiveness and usage of certain RDF vocabularies, especially the "Friend of a Friend"[1] (FOAF) ontology [3][4], which are related to our work. However, we analyze a heterogenous large-scale dataset utilizing various ontologies, we consider extensions to the original specifications, and we use a data mining approach to construct frequent itemsets instead of focusing on individual properties or vocabularies only.

The rest of this paper is organized as follows: In Sec. 2 we introduce a detailed problem statement. Section 3 illustrates the workflow of our approach to evaluate the appropriateness of ontologies based on usage patterns. The results in Sec. 4 indicate some exemplary findings in the BTC 2011 dataset before Sec. 5 concludes this paper and presents an outlook on future work. Please note that for brevity and readability, we use a number of prefix abbreviations when presenting RDF resources. These abbreviations are defined in Listing 3 in the Appendix.

2 Problem Statement

We identified two basic divergences of how the actual usage of RDF vocabularies differs from their specification. Firstly, certain classes may be *overspecified*: A number of properties are declared for a particular class, however they are are rarely (if ever) used for real-world instances of this class. On the other hand, a class may also be *underspecified*, when in real-world data certain properties are used frequently even though they are not specified by the vocabulary.

To illustrate the problem, consider Listing 1. This listing shows an excerpt of the `http://dbpedia.org/ontology/Settlement` class definition. If a geolocation data provider decides to employ this specification for her data, she might be confused about how to set proper values for some of the properties. The properties in lines 2 and 3 are more or less intuitively applicable to all instances of `:Settlement`[2], whereas others seem only useful for a strict subset of instances (such as `:scottishName` in line 4).

[1] `http://xmlns.com/foaf/spec/`

[2] For prefix abbreviations, please refer to Listing 3 in the Appendix.

```
:Settlement rdfs:subClassOf :Place .
:winterTemperature rdfs:domain :Settlement .
:summerTemperature rdfs:domain :Settlement .
:scottishName rdfs:domain :Settlement .
:distanceToEdinburgh rdfs:domain :Settlement .
```

Listing 1: Specification of `http://dbpedia.org/ontology/Settlement`.

However, none of the properties of this class (or any of its parent classes `:PopulatedPlace`, `:Place`, and `http://www.w3.org/2002/07/owl#Thing`) model the latitude and longitude of a settlement although these two properties are set for many instances of class `:Settlement` in DBpedia, e.g., via the `dbprop:latitude` and `dbprop:longitude` predicates, respectively. Overall, for a more intuitive vocabulary definition it would be advantageous to introduce more specific subclasses of `:Settlement` (e.g., `:ScottishSettlement`) and push down rather 'restricted' properties (e.g., `:scottishName`) to them. Additionally, some of the properties that are already set for a large number of the instances of `:Settlement` (such as `dbprop:latitude`) can be included in the definition of the class. A possible alternative is presented in Listing 2.

In general, over- or underspecification may cause confusion which ontology and which classes therein to adopt when publishing RDF data. Using vocabularies that are not intended for certain data or unwarranted extensions to existing ontologies limits machine readability and thus impedes the benefits of Linked Data. The goal of our work is to identify recurring misuse of ontologies and suggest possible ways to overcome these problems by reengineering ontologies.

```
:Settlement rdfs:subClassOf :Place .
:winterTemperature rdfs:domain :Settlement .
:summerTemperature rdfs:domain :Settlement .
dbpprop:latitude rdfs:domain :Settlement .
dbpprop:longitude rdfs:domain :Settlement .
:ScottishSettlement rdfs:subClassOf :Settlement .
:scottishName rdfs:domain :ScottishSettlement .
:distanceToEdinburgh rdfs:domain :ScottishSettlement .
```

Listing 2: A possible overhaul of `http://dbpedia.org/ontology/Settlement`.

3 Ontology Verification

Our approach for evaluating the use of ontology structures in the real world consists of four steps:

1. Extract typed concepts from the data (i.e., instances declaring `rdf:type`).
2. Retrieve relevant ontological information for all types found in the first step.
3. Perform data mining on all instances of each extracted relevant concept.
4. Compare the discovered usage patterns with class structures of the ontology.

For evaluating and comparing the usage patterns and the appropriateness of ontologies in a given dataset we need to scan the data twice: First, we need to discover which concept types have been used in the data and how the corresponding classes are defined for these concepts. In the next scan, for each concept we

analyze what predicates (besides `rdf:type`) are used for instances of this type. Finally, pattern analysis reveals frequent predicate patterns as well as positive and negative association rules between predicates. These rules in turn can be used to evaluate the appropriateness of ontology definitions in general and the assignment of certain types to entities. In the following, we describe Steps 2, 3, and 4 in more detail.

3.1 Ontology Retrieval

To retrieve the ontological specification of the approx. 200,000 discovered classes in the BTC dataset, we combined two approaches: First, we extracted the specifications for all types set by at least one instance from within the BTC data and afterwards gathered missing type definitions by performing lookups on the URI of each remaining class. Using the information available in the BTC data, we discovered class definitions for 90.97% of all the instances in the BTC dataset. By adding class definitions available online, this percentage was increased to 90.99%.

Judging from these numbers, a reasonable amount of TBox data is already available in the BTC 2011 crawl, especially for common RDF classes. Furthermore, we discovered that many online lookups of the missing class definitions resulted in HTTP 404 response codes. As pointed out in Sec. 1, this might be caused by data publishers not employing suitable vocabularies and rather defining illegitimate ontological structures, in this case leading to unresolvable URIs. Again, one solution to this problem might be providing more intuitive ontologies.

3.2 Predicate Analysis and Mining

Having identified the relevant concepts in the dataset, we evaluated the actual usage of predicates associated with the instances of each class. Trivially, this can be achieved by determining the frequency of each distinct predicate for the instances within the dataset. The result of this analysis is the set of the most frequent and therefore presumably relevant properties of a specific type.

A more general approach is to discover frequent sets of predicates. A frequent set of predicates is a set of predicates that co-occur for a minimum number of instances of a specific type. Similar to mining frequent itemsets from a transaction database [1], this number is defined as *minimum support* s. A set of predicates holds support s if s% of instances in the dataset involve all the predicates of this set. Moreover, we can detect dependencies between frequent sets of predicates as positive and negative association rules. A positive association rule $X \rightarrow Y$ states that the predicates in Y depend on the predicates in X. The confidence *conf* of such an association rule is the conditional probability $P(Y|X)$. Denoting the support of a set X as $supp(X)$, $conf(X \rightarrow Y)$ is computed as $supp(X \cup Y)/supp(X)$. Relevant rules are those that hold *minimum confidence* c.

A negative association rule of the form $X \rightarrow \neg Y$ denotes the conditional probability of the absence of Y given X. Respectively, $supp(X \cup \neg Y)$ denotes how many instances from the dataset involve the predicates of X but no predicate of Y. Negative associations between predicates might imply that these predicates

have alternating meanings and describe different categories of instances. Another reason for predicates to occur exclusively from one another is that they may have similar meanings, e.g., `foaf:lastName` and `foaf:familyName`.

3.3 Ontology Evaluation

In the last step, we compare the retrieved ontology definitions with the predicate patterns extracted from the data. For each ontology class we analyze:

- Predicates that are part of the class definition but rarely used in the data.
- Predicates that occur significantly often for instances of the specific type but are not included in the actual class (or superclass) definition.
- Dependencies between predicates that are defined for a certain class and those that are not.

By these three analysis steps we are able to categorize the mismatch of ontology class definition and the existing data with regard to over- or underspecification of the class definition as illustrated above. Overspecification occurs when predicates are defined by the class definition but are only set in a small number of instances. A special case of overspecification can be identified by means of negative rules that partition the instances into two different clusters implying that possibly two subclasses may be created, as will be exemplarily illustrated in Sec. 4. We define underspecification as many frequent predicates occurring in the dataset without being part of the class (or any superclass) definition. Again, association rules and disjoint frequent sets of predicates might justify the introduction of subclasses.

4 Results

In the BTC 2011 corpus we discovered 213,382 distinct type classes specified by 441,461,669 individual instances (which can be of more than one type). Of these instances, `foaf:Person` is the most common type, accounting for 362,590,928 typed entities. Additionally, we discovered around 150 properties whose `rdfs:domain` is a `foaf:Person`, whereas the original specification declares only 16 properties for this class.

For a first analysis of vocabulary usage, we extracted instances of several common types and matched their properties with the ones of the original ontology specification. Table 1a lists examples of overspecification. Consider the property `foaf:plan`: only one instances of a `foaf:Person` has a value set for this predicate in the entire BTC 2011 dataset. Some of the specified properties of `mo:MusicArtist` are never used at all (e.g., `mo:activity_start`).

On the other hand, Tab. 1b lists several properties that are commonly used for `foaf:Person`, but are not specified in the ontology for various reasons. These reasons include undefined attributes (e.g., `foaf:member_name`) or general misusage (e.g., `foaf:image`). For the other classes in Tab. 1b, we detected different properties that were declared and used in the context of several large-scale datasets and thus have an overall high frequency among instances of these types.

Resource	#instances
foaf:Agent	310,529
• foaf:yahooChatID	0
• foaf:tipjar	0
foaf:Person	362,590,928
• foaf:geekcode	7
• foaf:plan	1
mo:MusicArtist	310,529
• mo:activity_end	0
• mo:activity_start	0

(a) Examples for Overspecification.

Resource	#instances
foaf:Person	362,590,928
• foaf:member_name	19,083,000
• foaf:tagLine	19,062,451
• foaf:image	18,033,515
foaf:OnlineAccount	2,938,416
• sioc:account_of	2,411,233
• sioc:follows	1,484,445
mo:MusicArtist	310,529
• openvocab:sortLabel	108,073

(b) Examples for Underspecification.

Table 1: Over- and underspecification of common types in the BTC 2011 dataset.

Next, we mined association rules on instances of some of the common types to detect patterns that support the categorization of the class definition and data mismatch. Table 2 shows some interesting rules with confidence $\geq 90\%$ among predicates that occur for the instances of the types `mo:MusicArtist` and `foaf:Person`. Here, a strong dependency between several predicates defined for a `foaf:Person` can be observed. For `mo:MusicArtist`, predicates of the original ontology (e.g., `mo:remixed`) are also used frequently in combination with predicates from other namespaces (e.g., `http://vocab.org/bio/0.1/#event`).

foaf:Person	
foaf:image	→ foaf:nick
foaf:gender	→ foaf:weblog
foaf:weblog	→ foaf:member_name
mo:MusicArtist	
mo:image	→ foaf:depiction
http://purl.org/dc/terms/description	→ foaf:homepage
mo:remixed	→ http://vocab.org/bio/0.1/#event
(foaf:name, foaf:page, mo:member_of)	→ mo:musicbrainz

Table 2: Association rule examples with $c \geq 90\%$ and $s \geq 0.2\%$

Table 3 illustrates some negative rules with $c \geq 90\%$ that we extracted from the same data. The first rule for `mo:MusicArtist` shows an obvious hint for establishing two disjoint classes for instrumental artists and vocalists. Both predicates belong to the same ontology source and do not have contradicting meanings, but in the BTC data the set of vocalists and the set of instrumentalists are nearly disjunct. The negative association rule between `foaf:homepage`, `mo:myspace`, and `foaf:page` might imply that different data publishers use different properties for describing the same resource (i.e., an artist's web page). Looking at the rule examples for `foaf:Person`, one can recognize that most negative associations are between synonyms or similar resources such as `foaf:name` and `foaf:nick`, `foaf:homepage` and `foaf:weblog`, or `foaf:image` and `foaf:img` (as pointed out earlier, `foaf:image` is misused as a predicate very often). We discovered many negative association rules between predicates from different namespaces, although the associated instances are of the same type.

As mentioned earlier, we consider the crawl of the Billion Triple Challenge a unique snapshot of the Web of Data. Therefore, in contrast to individual

foaf:Person		
foaf:weblog	→ ¬	foaf:homepage
foaf:image	→ ¬	foaf:img
foaf:name	→ ¬	(foaf:nick, foaf:gender)
mo:MusicArtist		
mb:isInstrumentalArtistOf	→ ¬	mb:isVocalistOf
foaf:page	→ ¬	(foaf:homepage, mo:myspace)

Table 3: Negative rule examples with $c \geq 90\%$ and $s \geq 0.2\%$

data sources, the rules outlined above give some indication about a broad misconception of certain ontological structures. Instead of single data publishers misusing defined vocabularies, this might hint at a required design overhaul of these vocabularies. For more results of our experiments, please visit `http://www.hpi.uni-potsdam.de/naumann/projects/btc/btc2011.html`.

5 Conclusions and Future Work

In this work, we presented an approach to evaluate the utilization of several widely used ontologies based on the usage patterns presented in the Billion Triple Challenge 2011 dataset. We identified two misconceptions of the vocabulary definition in terms of its application: over- and underspecification. As illustrated in Sec. 4, there are mismatches between the intentions of certain well-known ontology specifications and how they are employed. Reasons for this may include

- illegitimate usage of the vocabulary by data publishers (e.g., because of a lack of knowledge of specification details),
- the ontology has evolved over time and needs to be reengineered as class definitions are hardly suitable for current real-world data, and
- general design flaws in the vocabulary, including term ambiguity or confusing schema definitions.

We leave open for future work an automated approach to propose reengineering steps for vocabularies based on the observed usage patterns. Additionally, we want to further the proper usage of ontologies by aiding data publishers in selecting the correct attributes and class definitions when preparing their data.

References

1. R. Agrawal and R. Srikant. Fast Algorithms for Mining Association Rules in Large Databases. In *Proceedings of the International Conference on Very Large Databases (VLDB)*, pages 487–499, Santiago de Chile, Chile, 1994.
2. C. Bizer, T. Heath, and T. Berners-Lee. Linked Data - The Story So Far. *International Journal on Semantic Web and Information Systems*, 5(3):1–22, 2009.
3. L. Ding, L. Zhou, T. Finin, and A. Joshi. How the Semantic Web is Being Used: An Analysis of FOAF Documents. In *Proceedings of the 38th Annual Hawaii International Conference on System Sciences*, page 113.3, Big Island, HI, USA, 2005. IEEE Computer Society.
4. J. Golbeck and M. Rothstein. Linking Social Networks on the Web with FOAF: A Semantic Web Case Study. In *Proceedings of the 23rd National Conference on Artificial Intelligence*, pages 1138–1143, Chicago, IL, USA, 2008. AAAI Press.
5. T. Heath and C. Bizer. *Linked Data: Evolving the Web into a Global Data Space*. Morgan & Claypool, 1st edition, 2011.

Appendix: Evaluation Criteria we have met

The BTC crawl provides a unique snapshot of the Linked Data universe. Among other aspects, it reflects how well-established ontologies are used by data providers to publish their data and how this usage differs from the intended application. Consequently, this knowledge may prove beneficial for ontology engineers to revise and alter the vocabularies provided.

In our work, we exploited the large-scale dataset to gain insight into how intuitive these ontologies are by comparing their specification with actual usage patterns. For this, we examined all 441,461,669 instances with type information in the BTC dataset. We were able to discover ontological specifications for almost all instances (90.99%) by either extracting it directly from the BTC data or looking it up online.

The techniques for contrasting type specification with actual utilization introduced in this paper are not only applicable to the Billion Triple Challenge 2011 dataset, but all large-scale RDF datasets. In particular, this approach is best suited for heterogenous data where the employed vocabularies are applied beyond the datasets they were originally designed for.

We believe that by continuously applying the concepts introduced in this work to large-scale snapshots of the Web of Data as presented in the Billion Triple Challenge and monitoring the results, over time the quality of ontologies can be improved (in terms of how intuitive these vocabularies are). As the problem of usage pattern analysis is equivalent to classical frequency analysis in transaction databases, rule mining approaches are well-suited. The employed algorithms also benefit from the fact that the total number of items (i.e., predicates) is much smaller than in the original use case.

We present additional results and resources regarding this submission at `http://www.hpi.uni-potsdam.de/naumann/projects/btc/btc2011.html`. For some of our analysis and experiments, we employed a distributed open-source RDF store developed in our group named HDRS[3]. The store uses the Apache Hadoop framework[4] to store RDF data in a distributed environment while providing means to efficiently query the available information.

```
@prefix : <http://dbpedia.org/ontology/> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema\#> .
@prefix dbpprop : <http://dbpedia.org/property/> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns\#> .
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
@prefix mo: <http://purl.org/ontology/mo/> .
@prefix sioc: <http://rdfs.org/sioc/ns\#> .
@prefix openvocab: <http://open.vocab.org/terms/> .
@prefix mb: <http://dbtune.org/musicbrainz/resource/vocab/> .
```

Listing 3: RDF prefix abbreviations used in this document.

[3] http://code.google.com/p/hdrs/
[4] http://hadoop.apache.org/