DBpedia Ontology Enrichment for Inconsistency Detection

Gerald Töpper
Hasso Plattner Institute
Prof.-Dr.-Helmert-Str. 2–3
14482 Potsdam, Germany
gerald.toepper@student.hpi.uni-potsdam.de

Magnus Knuth
Hasso Plattner Institute
Prof.-Dr.-Helmert-Str. 2–3
14482 Potsdam, Germany
magnus.knuth@hpi.uni-potsdam.de

Harald Sack
Hasso Plattner Institute
Prof.-Dr.-Helmert-Str. 2–3
14482 Potsdam, Germany
harald.sack@hpi.uni-potsdam.de

ABSTRACT

In recent years the Web of Data experiences an extraordinary development: an increasing amount of Linked Data is available on the World Wide Web (WWW) and new use cases are emerging continually. However, the provided data is only valuable if it is accurate and without contradictions. One essential part of the Web of Data is DBpedia, which covers the structured data of Wikipedia. Due to its automatic extraction based on Wikipedia resources that have been created by various contributors, DBpedia data often is error-prone. In order to enable the detection of inconsistencies this work focuses on the enrichment of the DBpedia ontology by statistical methods. Taken the enriched ontology as a basis the process of the extraction of Wikipedia data is adapted, in a way that inconsistencies are detected during the extraction. The creation of suitable correction suggestions should encourage users to solve existing errors and thus create a knowledge base of higher quality.

Categories and Subject Descriptors

H.3.5 [**Information Storage and Retrieval**]: On-line Information Services; I.2.4 [**Computing Methodologies**]: Knowledge Representation Formalisms and Methods—*Semantic Networks*

Keywords

DBpedia, Linked Data, Data Cleansing, Ontology Enrichment

1. INTRODUCTION

With the continuous growth of Linked Data on the World Wide Web (WWW) and the increase of web applications that consume Linked Data, the quality of Linked Data resources has become a relevant issue. Recent initiatives (e. g. the Pedantic Web group[1]) uncovered various defects and flaws in Linked Data resources.

As one essential multi-domain part of the Web of Data, DBpedia is broadly used and its data quality affects a wide range of web applications. But due to the automatic extraction based on Wikipedia resources that have been created by a large number of non-expert users its data is partly incorrect or incomplete.

In this work we propose an approach to identify inconsistencies in the DBpedia dataset based on an improved DBpedia ontology. Therefore, the ontology is enriched with axioms that have been identified with methods from the field of *Inductive Logical Programming* (ILP). By applying the enriched ontology during the extraction process it is possible to induce contradictions that point to incorrect facts, which demand for correction either in the original Wikipedia page, in the DBpedia ontology, or in the mappings[2] that have been used for extraction.

The paper is organized as follows: Section 2 recapitulates previous work in the field of error detection and correction in Linked Data as well as ontology enrichment. The proposed methods applied for enriching the DBpedia ontology are presented in Section 3. This ontology is then used for the detection of inconsistencies, which has been integrated into the DBpedia Extraction Framework[3] as presented in Section 4. Section 5 assesses the results achieved for ontology enrichment and error detection. Section 6 summarizes the achievements of this work and provides an outlook for further research.

2. RELATED WORK

The detection of inconsistencies and errors within Linked Data is subject of various studies recently. Hogan et al. have focused on different types of errors referring to accessibility, syntactical correctness, and consistency of published RDF data [1]. For these error types proposals have been made, how publishers are able to avoid and users are able to deal with those errors. Similarly, in [2] a system has been presented, which efficiently creates statistics according to accessibility of RDF documents and SPARQL endpoints as well as syntactical errors. The system periodically updates the statistics and publishes it on the web[4].

A related approach to detect inconsistencies in Linked Data by means of logical reasoning is presented by Péron et al. in

[1] http://pedantic-web.org/

I-SEMANTICS 2012, 8th Int. Conf. on Semantic Systems Sept. 5–7, 2012, Graz, Austria.

[2] DBpedia mappings wiki: http://mappings.dbpedia.org/
[3] http://sourceforge.net/projects/dbpedia/develop
[4] http://stats.lod2.eu/

[3], which searches for inconsistency patterns using SPARQL queries applied to the deductive closure of the graph of all inferred rules. Their definition of an inconsistency regards an absent type assignment as inconsistent, whereas our approach additionally demands classes to be explicitly disjoint. Furthermore, we automatically enhance the underlying ontology beforehand with such disjointness constraints and missing domain and range restrictions.

However, in this work the detection of inconsistencies within DBpedia is highlighted, which is achieved by automatic semantic enrichment of the underlying ontology. Apart from the manual enrichment of ontologies many approaches have been developed for the purpose of automatic enrichment. Popular methods for the automatic construction of ontologies from semi- and unstructured data are *Natural Language Processing* (NLP) and *Machine Learning* (ML). Our approach follows existing methods from *Inductive Logic Programming* (ILP), a subfield of ML, which deduces a hypothesis based on background knowledge in terms of positive and negative examples.

In [4], which uses ILP, association rules – well-known in the field of data mining – are constructed and translated into axioms. In that context axioms are created, which express a subclass relationship or specify the domain, range or transitivity of a role. Existing `rdf:type` statements serve as a basis for the generation of the transaction tables, from which the association rules are produced. Association rules are exploited as well in [5], where disjoint classes are identified.

ILP is also applied in the DL-Learner framework[5], which learns complex class descriptions on the basis of instances [6]. ORE[6] is using that framework as a foundation and learns axioms, which express a subclass relationship or a class equivalence. An integrated reasoner identifies resulting inconsistencies within the ontology. Both the inconsistency and suggestions on how to solve it are presented to the user, where one possible solution is the deletion of an acquired axiom [7].

Haase et al. describe an approach that keeps generated ontologies consistent [8]. For this purpose the automatically acquired knowledge is enriched with annotations, which comprise a declaration about the correctness and the relevance of the axioms. Subsequently arising inconsistencies are handled immediately. The result is a knowledge base, which is consistent in any case and semantically correct.

In [9] four different approaches for handling inconsistencies in changing ontologies have been surveyed: the evolution of a consistent ontology, the reparation of inconsistencies, reasoning in the presence of inconsistencies, and multi-version reasoning. With respect to the DBpedia ontology the second variant – repairing the inconsistencies through a continuous process consisting of the identification and the resolution of inconsistencies – seems to be feasible. Due to the huge amount of data in the DBpedia knowledge base ignoring the inconsistencies is unavoidable in some cases, why the third variant would apply.

[5]http://dl-learner.org
[6]http://ore-tool.net

Crowdsourcing is another approach that allows to apply human intelligence for the detection of errors in knowledge bases. The quiz *WhoKnows?* [10] generates questions out of the mapping-based dataset of DBpedia. In case that a question appears odd, the player has the chance to report this. Facts, i. e. RDF triples, applied in frequently reported questions indicate erroneous data. In [11] a technique is presented, which uses this information for the creation of patch requests. As a use case *WhoKnows?* was extended in a way that a player is able to state, which fact in DBpedia is obviously wrong or eventually missing. According to this information a suitable patch request is generated and submitted to a central repository. The collected patch requests can be applied to the corresponding dataset and thus improve the quality of the data.

3. UPGRADING THE DBPEDIA ONTOLOGY

Multiple levels of errors might occur in automatically generated RDF knowledge bases. The first and most obvious class are *syntactic errors* in RDF that can be detected with the help of a simple RDF parser/validator. There are also some syntactic errors in DBpedia, e. g. incorrect date information.

One level above are *logical errors*. They are caused by contradicting RDF triples and can be identified with the help of a reasoner, such as Pellet[7] or Fact++[8]. To produce a logical contradiction the negation of facts or the construction of binding constraints must be possible. With RDF(S) alone the construction of plain logical contradictions is not possible.

The upper level *semantic errors* comprise facts that are not corresponding to facts in the real world. These inconsistencies are most hard to detect since they are – so far – usually logically correct and demand real world knowledge for identification. Semantic errors represent the majority of flaws in the DBpedia dataset. They emerge from the difficulty to parse the broad range of Wikipedia infoboxes correctly. An example of such an error is the fact

```
dbp:2666_%28novel%29 dbo:publisher dbp:Barcelona .
```

This RDF triple refers to the fact that the novel "2666" has been published by Barcelona, which is obviously wrong, because Barcelona is a city while a publisher should be a person or a publishing company. As shown in Figure 1 this fact results from the composition of the Wikipedia infobox[9] that quotes "Editoral Anagrama" as the actual publisher of the book and in absence of a Wikipedia entry for "Editoral Anagrama" an additional link to Barcelona, the residence of the publishing company, is mentioned. Since links are preferentially treated by the automatic extractor, `dbp:Barcelona` is choosen for the extracted triple's object.

Our approach to detect such semantic inconsistencies automatically is to transform semantic errors into logical ones by extending the axioms of the underlying ontology in order

[7]http://clarkparsia.com/pellet/
[8]http://owl.man.ac.uk/factplusplus/
[9]c. f. http://en.wikipedia.org/wiki/2666?oldid=480743891

Figure 1: Extract of the infobox in the article *2666*

to cause a logical contradiction that can then be recognised by a reasoner.

How this can be achieved, will be demonstrated using the example mentioned above. Apart from the extracted triple additional information is known about the property `dbo:publisher` and the object entity `dbp:Barcelona`:

```
dbp:2666_%28novel%29 dbo:publisher dbp:Barcelona .
dbo:publisher rdfs:range dbo:Company .
dbp:Barcelona rdf:type dbo:Settlement .
```

So far, this tripleset is not inconsistent. Instead, a reasoner would deduce `dbo:Company` as a new `rdf:type` for `dbp:Barcelona`, which is no contradiction to the original type `dbo:Settlement`, as long as both classes are not marked disjoint. By adding the disjointness axiom

```
dbo:Company owl:disjointWith dbo:Settlement .
```

the inconsistency of the tripleset can be deduced.

Concerning formal restrictions the current state of the DBpedia ontology is a stumbling block, since there are no domain and range restriction for a number of properties and no class disjointness axioms included. In the following, methods are presented to derive this missing information from the DBpedia dataset. For this task the English DBpedia 3.7 dataset [12] has been used. In order to improve the DBpedia ontology those methods determine new domain and range restrictions as well as class disjointness axioms that can be added to the ontology.

3.1 Property domain restrictions

To be able to deduce inconsistencies related to property domains, domain restrictions have to be specified explicitly. However, approximately 16% of all properties in the DBpedia ontology lack the declaration of an `rdfs:domain`-value. Additionally a few properties exist, whose domain does not correspond to the common usage of the property in the ABox. Thus, for all properties of the DBpedia ontology new domains have been determined. Subsequently a metric is presented that is based on the ABox of the ontology.

Let $KB = \{(s\,p\,o) : s \in E \cup C \wedge p \in P \wedge o \in E \cup L \cup C\}$ be the knowledge base, E the set of all entities, P the set of all properties, $p \in P$, L the set of all Literals, C the set of all classes and $c \in C$. The metric $md_{p,c}$ indicates whether class c is the domain of property p. As shown in equation 1, it is calculated from the number of triples $(s\,p\,o) \in KB$, whose subject $s \in E$ belongs to the class c, relatively to the number of triples $(s\,p\,o) \in KB$, whose subject $s \in E$ belongs to any class more specific than `owl:Thing` (abbreviated `T` in the equations). The equations use the abbreviation $(s\,\mathtt{a}\,c)$ for expressing the fact that the entity $s \in E$ belongs to the class c.

$$md_{p,c} = \frac{|\{(s\,p\,o) : (s\,p\,o) \in KB \wedge (s\,\mathtt{a}\,c) \in KB\}|}{|\{(s\,p\,o) : (s\,p\,o) \in KB \wedge (s\,\mathtt{a}\,d) \in KB \wedge d \neq \mathtt{T}\}|} \quad (1)$$

Because of the fact that the subjects that occur with the property p, do belong to more than one `rdf:type`, a set of classes with their respective $md_{p,c}$-value is generated. The higher the value for $md_{p,c}$ the more frequent is the usage of property p for subjects that do belong to class c. The class with the highest value (max_{md_p}) is defined as the domain of the property. If there are multiple classes possessing the highest value and those classes are in a subclass relationship, the most specific class will be declared as the domain. Some properties are applied universally in different domains, which is why a certain class as an `rdfs:domain` is not determinable. Due to the fact that only atomic classes can be specified in the DBpedia mappings wiki, `owl:Thing` is the only possible domain for those generic properties. In case max_{md_p} lies under a certain threshold τ_{md}, `owl:Thing` is defined as the domain of the property.

The appropriate threshold $\tau_{md} = 0.96$ has been determined via a randomized analysis. For most properties max_{md_p} equals 1.0 meaning that such a property is non-generic and a concrete class is determinable as a domain. In order to obtain a meaningful sample all properties with $max_{md_p} = 1.0$ are ignored for the creation of the sample. Out of the remaining properties 10% respectively 15 properties are randomly chosen. After the manual classification, whether these properties are generic or a concrete class is determinable, the threshold τ_{md} is determined taking the classification and the max_{md_p}-value of these properties into account. Using this threshold the domain for 1,363 DBpedia properties has been determined, whereas 107 of them are used rather generically, which is why `owl:Thing` is selected as their domain.

3.2 Property range restrictions

The metric $mr_{p,c}$, which indicates whether c is the range of an object property p, is quite similar to the approach shown before that identifies the domain of the property.

Let KB again be the knowledge base, E the set of all entities, OP the set of all object properties, $p \in OP$, C the set of all classes and $c \in C$. The metric $mr_{p,c}$ calculates from the number of triples $(s\,p\,o) \in KB$, whose object $o \in E$ belongs to the class c, relatively to the number of triples $(s\,p\,o) \in KB$, whose object $o \in E$ belongs to any more specific class than `owl:Thing`:

$$mr_{p,c} = \frac{|\{(s\,p\,o) : (s\,p\,o) \in KB \wedge (o\,\mathtt{a}\,c) \in KB\}|}{|\{(s\,p\,o) : (s\,p\,o) \in KB \wedge (o\,\mathtt{a}\,d) \in KB \wedge d \neq \mathtt{T}\}|} \quad (2)$$

On the basis of the highest value (max_{mr_p}) and the threshold τ_{mr} a range of the property p is specified, which can either be a class of the DBpedia ontology or `owl:Thing`.

The threshold $\tau_{mr} = 0.77$, which has been investigated by means of a randomized analysis, seems appropriate. The procedure for finding the threshold is comparable to the one

applied to the domain of properties. With this threshold 592 DBpedia properties have been classified in terms of their range. Due to their generality 82 of them have received the range `owl:Thing` only.

3.3 Class disjointness axioms

For the recognition of disjoint classes the *Vector Space Model* (VSM) [13] from *Information Retrieval* (IR) has been applied. It can be used for measuring the relevance of a document with respect to a query as well as the similarity of two documents. Likewise the similarity of two ontology classes can be computed, from which the disjointness of both classes is deduced in case their similarity value computes below a given threshold.

In IR similar documents contain equivalent terms. Analogously, entities of similar classes occur more frequently with the same properties. An RDF triple containing an object property as a predicate has both an entity as a subject and an entity as an object. Semantically there is a difference whether an entity occurs as a subject or as an object in association with a property. Thus for every object property the corresponding inverse property has to be considered.

Let DP be the set of all datatype properties and OP the set of all object properties. Consequently $IOP = \{p' : p' \equiv p^{-1} \wedge p \in OP\}$ denotes the set of all inverse object properties. The set of all properties P results from the union of all datatype, object and inverse object properties: $P = DP \cup OP \cup IOP = \{p_1, p_2, \ldots, p_k, \ldots, p_n\}$. Furthermore, let $C = \{c_1, c_2, \ldots, c_i, \ldots, c_m\}$ be the set of all classes. The class vector $\vec{v}_{c_i} = (w_{c_i,p_1}, w_{c_i,p_2}, \ldots, w_{c_i,p_k}, \ldots, w_{c_i,p_n})$ is an n-dimensional vector that corresponds to the class c_i, whereby w_{c_i,p_k} represents the weight of a property p_k in a class c_i.

In IR the weight of a term in a document for a given set of documents can be determined based on the term frequency-inverse document frequency (TF-IDF). Comparably the weight $w_{c,p}$ of a property p in a class c can be calculated from the product of the property frequency (PF) and the inverse class frequency (ICF).

$$w_{c,p} = pf_{c,p} * icf_p \tag{3}$$

In this context the PF $pf_{c,p}$ is the absolute frequency of a property p along with the entities of the class c. Let KB be the knowledge base, then the PF equals the number of triples $(spo) \in KB$, whose subject s belongs to the class c.

$$pf_{c,p} = |\{(s\,p\,o) : (s\,p\,o) \in KB \wedge (s\,\mathtt{a}\,c) \in KB\}| \tag{4}$$

In IR the sublinear TF scaling is used to take into account that multiple occurrences of the same term in a document do not imply a multiplex relevance of a single occurrence. Similarly, a property, which occurs with a number of entities of the same class, is not as many times relevant for the class than a property, which occurs with only one entity of the class. For that reason the sublinear PF scaling is applied, which calculates from the logarithm of the PF $pf_{c,p}$.

$$wpf_{c,p} = \begin{cases} 1 + \log pf_{c,p} & \text{, if } pf_{c,p} > 0 \\ 0 & \text{, else} \end{cases} \tag{5}$$

The ICF icf_p measures the general relevance of a property p for the complete knowledge base. Let KB be the knowledge base, then the ICF equals the logarithm of the ratio between the number of all classes C and the number of the classes c, whose entities occur with the property p. In case a property p occurs with entities of only a few classes, the ICF is accordingly higher:

$$icf_p = \log \frac{|C|}{|\{c : c \in C \wedge (s\,p\,o) \in KB \wedge (s\,\mathtt{a}\,c) \in KB\}|} \tag{6}$$

Analogous to the document similarity in IR, the similarity of two classes c_i und c_j can be computed by means of the cosine similarity of their class vectors $\vec{v}_{c_i}$ and $\vec{v}_{c_j}$:

$$sim_{c_i,c_j} = \frac{\vec{v}_{c_i} * \vec{v}_{c_j}}{|\vec{v}_{c_i}||\vec{v}_{c_j}|} = \frac{\sum_{k=1}^{n} w_{c_i,p_k} w_{c_j,p_k}}{\sqrt{\sum_{k=1}^{n} w_{c_i,p_k}^2} \sqrt{\sum_{k=1}^{n} w_{c_j,p_k}^2}} \tag{7}$$

The similarity value sim_{c_i,c_j} of two classes c_i und c_j is normalized between 0 und 1, since $w_{c_i,p_k} >= 0$ und $w_{c_j,p_k} >= 0$ holds for all p_k. A random sample has pointed out that $\tau_{sim} = 0.17$ seems to be an appropriate threshold.

Consequently all class pairs (c_i, c_j) with $sim_{c_i,c_j} <= 0.17$ can be considered as disjoint, which results in 37,091 disjoint pairs of classes identified in the DBpedia ontology.

4. DETECTION OF INCONSISTENCIES

The *MappingExtractor* as part of the DBpedia Extraction Framework extracts the content of infoboxes and tables of a Wikipedia article and maps the resulting data onto the DBpedia ontology. As an outcome RDF triples are generated, which represent the statements in the infoboxes and tables as well as the ontology class of a distinct entity (via `rdf:type`). The identification of inconsistencies in property `rdfs:domain` and `rdfs:range` has been implemented and integrated within the extraction process of a Wikipedia dump. The extracted data are examined after all the pages of the Wikipedia dump have been processed. That is because checking the consistency of the range of a property in an RDF triple needs the information about the `rdf:type` of the object, which might be extracted later.

During extraction all the triples generated by the *MappingExtractor* are stored within the main memory. After the extraction has finished consistency checking is performed in parallel. For every single triple it has to be verified whether an `rdf:type` of the subject and the `rdfs:domain`-value of the property are disjoint. Analogously, it is verified for all triples containing an object property, whether an `rdf:type` of the object and the `rdfs:range`-value of the predicate are disjoint. In case a violation of this rule is detected a suggestion

to correct the inconsistency will be created. Currently, the suggestions are stored within a database for further manual processing. Due to the huge amount of data it is not feasible to apply a reasoner such as Pellet for the consistency checks. Instead, violations are identified expediently in the way that has been described in the first place by an own implementation. The complete consistency check of all triples extracted from the Wikipedia dump requires six minutes on a server with 16 cores[10] and 96 GB main memory.

For the violation of the domain of an RDF property the following solution variants are conceivable:

1. (D1) Map the template property onto another ontology property.
2. (D2) Remove the classes' disjointness axiom.
3. (D3) Change the domain of the ontology property to `owl:Thing`.

In some cases the inconsistency results from an incorrectly used property (D1), i. e. the property was intended for and is usually used in a different context. Basically, within the DBpedia mappings wiki the template property of an infobox is mapped onto an ontology property, whose meaning does not conform to the meaning of the template property. In order to solve these inconsistencies appropriate properties have to be determined, whose domain and range fit to the `rdf:type` of the subject and the `rdf:type` of the object of the inconsistent triple.

Moreover, the class that represents the `rdf:type` of the subject and the class that represents the domain of the property might not be disjoint (D2). Consequently the removal of the disjointness axiom will be suggested.

Another reason for the inconsistencies might be caused by the fact that the ontology property is generic (D3), i. e. it can be applied in various domains and no distinct class can be determined as `rdfs:domain`. Due to the fact that only atomic classes can be specified in the DBpedia mappings wiki, only `owl:Thing` qualifies as a feasible `rdfs:domain`-value.

For the violation of the range of a property the following suggestions are generated:

1. (R1) Create a link to the appropriate article in the articles infobox.
2. (R2) Delete the value or associate the value with another template property.
3. (R3) Map the template property onto another ontology property.
4. (R4) Remove the classes' disjointness axiom.
5. (R5) Change the range of the ontology property to `owl:Thing`.

[10] 16x Intel(R) Core(TM) i7 CPU 930 @ 2.80 GHz

One reason for a range-violation is caused by the fact that a linked article in a value of an infobox is confused with the actual article (R1). This happens due to the existence of homonyms or similar spellings of terms, which identify different real world entities or abstract concepts. In order to distinguish these ambiguities disambiguation sites[11] exist that list all articles with same or similar spelling. This information, which can be obtained by applying the *DisambiguationExtractor*, is used for the validation of a potentially correct article. For each article, whose title has a similar spelling to the one provoking the range-violation, it is verified that the `rdf:type` of the corresponding DBpedia entity suits to the range of the property that belongs to the validated triple. In case of a successful verification the article is correct.

Furthermore, the case might occur that information does not fit to the template property within an infobox (R2). Deleting the value or associating the value with another more appropriate template property might eliminate the inconsistency.

The three remaining correction suggestions (R3, R4, R5) are created in a comparable way to the violation of the domain of a property.

5. EVALUATION

According to the two step approach, the evaluation has been performed for the enrichment task and detection task separately.

5.1 Enrichment

The enrichment of the DBpedia ontology as stated in Section 3 embraces the domain and range of properties and the disjointness of classes. The evaluation of the suggested metrics has been carried out by means of random samples. Statistical features provide information, whether the random samplings are sufficient to express representative statements about the population. The mapping-based dataset of the English DBpedia in the version 3.7 has been used as a basis for the evaluation.

For all properties occurring in the ABox a **domain restriction** has been determined with the aid of the metric $md_{p,c}$ – either a class of the DBpedia ontology or the class `owl:Thing`. Out of the 1,363 properties 5% are randomly chosen und the correctness of their classification is checked[12]. Table 1 shows the results of the random sample.

Table 1: Results of the classification of the domain of DBpedia properties

N	n	t_p	f_p	$\widehat{pr}$	95% confidence interval
1,363	68	67	1	0.985	$[0.957, 0.999]$

In this context N represents the population, n the sample size, t_p the number of correctly classified properties, f_p the

[11] http://en.wikipedia.org/wiki/Wikipedia:Disambiguation

[12] All validations have been performed by the authors. Agreements have been reached by referring to the respective template and property description pages in Wikipedia and DBpedia, and by majority vote.

number of wrongly classified properties and $\widehat{pr}$ the estimated value for the precision. The confidence interval indicates that 95 times out of 100 the actual precision, which refers to the population, lies within the stated interval boundaries. The results show that if a domain is suggested, approximately only one out of 100 suggestions is incorrect.

Such a result is due to the fact that the class of the entity that occurs in the subject of a triple arises from the infobox of the corresponding Wikipedia article. The only false positive example within the sample is caused by a general problem of both the metrics $md_{p,c}$ and $mr_{p,c}$. The property `dbo:productionStartYear` denotes the year, in which the production of a thing is started. While it is commonly used in relation to the entites of the class `dbo:MeanOfTransportation`, it occurs infrequently with the entities of the class `dbo:AutomobileEngine`. Since $max_{md_p} >= \tau_{md}$ holds for the property, the domain is defined as `dbo:MeanOfTransportation`. In practice the property is a generic one and its domain should be `owl:Thing`.

In Section 3 a **range restriction** has been assigned to 592 properties. Approximately 10% of these properties have been randomly chosen for manually verification of the classification. Table 2 shows the results of the random sample.

Table 2: Results of the classification of the range of DBpedia properties

N	n	t_p	f_p	$\widehat{pr}$	95% confidence interval
592	59	51	8	0.864	$[0.781, 0.948]$

The estimated precision is $\widehat{pr} \approx 0.864$. The reason for a lower result in comparison with the precision regarding the domain can be explained by the diversity of information stated in the values of the Wikipedia infoboxes. In some cases it is not determinable, which kind of information should be associated with a certain template property. Often there is no information in the description site for an infobox. Another problem is that some entities do not belong to a more specific `rdf:type` than `owl:Thing`. Thus, the range defined by the metric $mr_{p,c}$, is somehow skewed.

Out of the 37,091 class pairs that have been declared as **disjoint classes** approximately 0,5% of the pairs are taken as a random sample. Subsequently their classification is checked. Table 3 shows the result of the random sample.

Table 3: Results of the classification of the disjointness of two classes

N	n	t_p	f_p	$\widehat{pr}$	95% confidence interval
37,091	185	183	2	0.989	$[0.974, 1.0]$

The estimated precision for the metric $mr_{p,c}$ amounts to $\widehat{pr} \approx 0.989$, which depends heavily on the chosen threshold τ_{sim}. A higher threshold results in the declaration of more actually disjoint classes, but has also negative consequences for the precision value. The two false positive examples relate to potential exceptions. For instance it is conceivable that a `dbo:FigureSkater` aims at a career as a `dbo:Politician` after the career as a sportsman. The fact that the entities of both classes infrequently occur with the same properties in the ABox leads to a low similarity value $sim_{c_i,c_j} \approx 0.056$.

5.2 Detection

The consistency check examined 3,110,392 entities, whereas the majority of 3,060,898 resources showed to be consistent. The remaining 49,494 entities have been classified as inconsistent having 60,602 inconsistencies. In 12,218 cases the inconsistency results from a domain restriction violation of a used property, 40,404 inconsistencies result from range restriction violations.

For the violations regarding both the domain and range restriction a sample of each 100 inconsistencies has been reviewed manually, which of the proposed correction suggestions conduct to a reasonable removal of the inconsistency.

Figure 2 shows the ratio of the different suggestions, which have been applied to eliminate **domain restriction violations**.

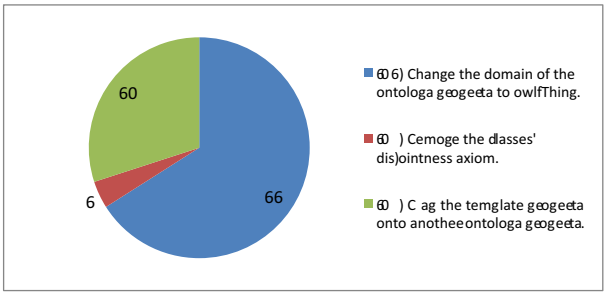

Figure 2: Proportion of the correction suggestions leading to the removal of the domain restriction violations

The actual reason for the majority of the inconsistencies is an erroneous definition of the TBox of the ontology. In two third of all cases a more specific class than `owl:Thing` is declared as a domain, whereas `owl:Thing` would be the correct one. In another four cases the axiom declaring a class pair being disjoint is wrong. Explanations concerning flaws in the DBpedia ontology have been discussed earlier in the evaluation. In 30 cases the correction suggestion, which recommends to change the mapping from the template property to the ontology property, leads to the removal of the inconsistency. Exemplary the property `dbo:division` has the class `dbo:Species` as a domain and therefore should specify the division of a species. But the property is similarly used for representing the branch of a company. Consequently the domain of the property could be set to `owl:Thing` but there exists a semantic difference, whether the division of a species or the branch of a company is meant. Thus the semantic correct solution is to map each template property onto a separate ontology property.

Figure 3 shows the ratio of the different suggestions, which are applied in order that **range restriction violations** are eliminated.

Comparable to domain restriction violations some inconsistencies originate from a flawed DBpedia ontology. In 28

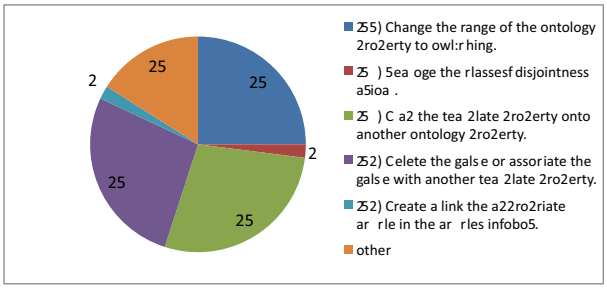

Figure 3: Proportion of the correction suggestions leading to the removal of the range restriction violations

cases the mapping from a template property onto an ontology property is wrong - a corresponding example has been discussed in the evaluation of the domain restriction violations. In 27 cases the deletion of the value or the association of the value with another template property leads to the elimination of the inconsistency. For example the movie *Kamen Rider J* uses the infobox *film*, in which the template property *producer* refers to a company. The template property *producer* maps onto the ontology property `dbo:producer`, whose range is `dbo:Person`. Due to the fact that the class of all companies is disjoint to the class of all persons, an inconsistency occurs. Additionally the infobox also owns the template property *studio*, which serves for stating the companies that produce a movie. Accordingly, this is a fitting template property, which could be associated to the company. Sometimes such a well fitting template property does not exist. Concerning this matter only the deletion of the value is conductive, which can involve the loss of relevant data. In two cases the linking of a suggested article in the infobox value leads to the removal of the inconsistency. The suggested article has been determined by the disambiguation site of the article linked in the infobox value. Exemplary the Australian actor `dbp:John_Jarratt` has the movie `dbp:Australia_%282008_film%29` as a birthplace, whereas the country `dbp:Australia` would be the correct value. The following explanations refer to corrections that cannot be accomplished by means of the generated suggestions. In Figure 3 they are denoted by the category *other*. Frequently the actually correct article cannot be suggested for linking in the infobox, since no corresponding disambiguation site exists. In other cases the actual article cannot be linked as an infobox value, since it simply does not exist in Wikipedia. Instead, a related article is linked, which can lead to an inconsistency.

6. CONCLUSION AND OUTLOOK

In this paper an approach is presented to identify inconsistent triples during the extraction process of the DBpedia dataset which may lead to a higher quality extraction. Therefore, the DBpedia ontology has been improved by extracting further axioms concerning properties' domain and range restrictions, and class disjointness from the original DBpedia dataset. The applied methods performed with reasonably high precision, which allows to use the enriched ontology[13] for other purposes, too.

For now, the correction of inconsistencies needs to be performed manually according to the suggestions. Therefore, a user interface for managing the correction suggestions will be provided that might report supplementary information, for instance, the solution that minimizes the number of remaining inconsistencies, to support the decision of the user. To decide automatically, which triple is in charge of inducing the inconsistency, statistical methods are conceivable, while further research is needed.

The realized experiments base on the dump of DBpedia version 3.7, but are also applicable to succeeding versions. For future application the error detection needs to be adapted to the DBpedia Live extraction process [14], which provides online updates of DBpedia according to the changes in the Wikipedia articles and mappings. In such a scenario, an iterative approach can be followed to obtain a consistent DBpedia incrementally.

7. REFERENCES

[1] Hogan, A., Harth, A., Passant, A., Decker, S., Polleres, A.: Weaving the pedantic web. In: Linked Data on the Web Workshop (LDOW 2010) at WWW 2010. Volume 628., CEUR Workshop Proceedings (2010) 30–34

[2] Demter, J., Auer, S., Martin, M., Lehmann, J.: LODStats – an extensible framework for high-performance dataset analytics. (To appear)

[3] Péron, Y., Raimbault, F., Ménier, G., Marteau, P.F.: On the detection of inconsistencies in RDF data sets and their correction at ontological level. In: Proceedings of the 10th International Semantic Web Conference (ISWC 2011). (2011)

[4] Völker, J., Niepert, M.: Statistical schema induction. In Grobelnik, M., Simperl, E., eds.: Proceedings of the 8th Extended Semantic Web Conference (ESWC 2011). ESWC'11, Heraklion, Crete, Greece, Springer (2011) 124–138

[5] Fleischhacker, D., Völker, J.: Inductive learning of disjointness axioms. In: On the Move to Meaningful Internet Systems: OTM 2011. Volume 7045. Springer (2011) 680–697

[6] Lehmann, J.: DL-Learner: learning concepts in description logics. Journal of Machine Learning Research (JMLR) **10** (2009) 2639–2642

[7] Lehmann, J., Bühmann, L.: ORE – a tool for repairing and enriching knowledge bases. In: Proceedings of the 9th International Semantic Web Conference (ISWC 2010). Volume 6497 of Lecture Notes in Computer Science., Berlin/Heidelberg, Springer (2010) 177–193

[8] Haase, P., Völker, J.: Ontology learning and reasoning – dealing with uncertainty and inconsistency. In Costa, P.C., D'Amato, C., Fanizzi, N., Laskey, K.B., Laskey, K.J., Lukasiewicz, T., Nickles, M., Pool, M., eds.: Uncertainty Reasoning for the Semantic Web I. Volume 5327. Springer, Berlin, Heidelberg (2008) 366–384

[13]The enriched DBpedia ontology can be accessed via http://purl.org/hpi/dbpedia_enriched.owl

[9] Haase, P., van Harmelen, F., Huang, Z., Stuckenschmidt, H., Sure, Y.: A framework for handling inconsistency in changing ontologies. In: Proceedings of the 4th International Semantic Web Conference (ISWC 2005). Volume 3729., Springer (2005) 353–367

[10] Waitelonis, J., Ludwig, N., Knuth, M., Sack, H.: WhoKnows? – evaluating linked data heuristics with a quiz that cleans up DBpedia. International Journal of Interactive Technology and Smart Education (ITSE) **8**(3) (2011) 236–248

[11] Knuth, M., Hercher, J., Sack, H.: Collaboratively patching linked data. In: Proceedings of 2nd International Workshop on Usage Analysis and the Web of Data (USEWOD 2012), co-located with the 21st International World Wide Web Conference 2012 (WWW 2012), Lyon, France (April 2012)

[12] DBpedia: DBpedia 3.7 downloads (September 2011) accessed Nov 28., 2011.

[13] Manning, C.D., Raghavan, P., Schütze, H.: Introduction to Information Retrieval. Cambridge University Press, New York, NY, USA (2008)

[14] Hellmann, S., Stadler, C., Lehmann, J., Auer, S.: DBpedia Live extraction. In: On the Move to Meaningful Internet Systems: OTM 2009. Volume 5871. Springer (2009) 1209–1223