TripleCheckMate: A Tool for Crowdsourcing the Quality Assessment of Linked Data

Dimitris Kontokostas[1], Amrapali Zaveri[1], Sören Auer[2], and Jens Lehmann[1]

[1] AKSW/BIS, Universität Leipzig, Germany, `http://aksw.org`
`{lastname}@informatik.uni-leipzig.de`
[2] CS/EIS, Universität Bonn, `http://www.iai.uni-bonn.de/~auer`
`auer@cs.uni-bonn.de`

Abstract. Linked Open Data (LOD) comprises of an unprecedented volume of structured datasets on the Web. However, these datasets are of varying quality ranging from extensively curated datasets to crowdsourced and even extracted data of relatively low quality. We present a methodology for assessing the quality of linked data resources, which comprises of a manual and a semi-automatic process. In this paper we focus on the manual process where the first phase includes the detection of common quality problems and their representation in a quality problem taxonomy. The second phase comprises of the evaluation of a large number of individual resources, according to the quality problem taxonomy via crowdsourcing. This process is implemented by the tool *TripleCheckMate* wherein a user assesses an individual resource and evaluates each fact for correctness. This paper focuses on describing the methodology, quality taxonomy and the tools' system architecture, user perspective and extensibility.

Keywords: Data Quality, Linked Data, DBpedia

1 Introduction

The advent of semantic web technologies, as an enabler of Linked Open Data (LOD), has provided the world with an unprecedented volume of structured data currently amounting to 50 billion facts represented as RDF triples. Although publishing large amounts of data on the Web is certainly a step in the right direction, the data is only as usable as its quality. On the Data Web, we have varying quality of information covering various domains. There are a large number of high quality datasets (in particular in the life-sciences domain), which are carefully curated over decades and recently published on the Web. There are, however, also many datasets, which were extracted from unstructured and semi-structured information or are the result of some crowdsourcing process, where large numbers of users contribute small parts. DBpedia [5,3,4] is actually an example for both - a dataset extracted from the result of a crowdsourcing process. Hence, quality problems are inherent in DBpedia. This is not a problem per se, since quality usually means fitness for a certain use case [1]. Hence, even datasets with quality problems might be useful for certain applications, as long as the quality is in the required range.

In this paper, we first describe a data quality assessment methodology, which comprises of a manual and a semi-automatic process (Section 2). However, we focus only on the manual process of quality assessment of DBpedia in this article and refer the readers to [6] for details on the semi-automatic process. The first phase includes the detection of common quality problems and their representation in a comprehensive taxonomy of potential quality problems. The second phase comprises of the evaluation of a large number of individual resources, according to the quality problem taxonomy, using *crowdsourcing* in order to evaluate the type and extent of data quality problems occurring in DBpedia. Each represented fact is evaluated for correctness by each user and, if found problematic, annotated with one of 17 pre-defined quality criteria. This process is accompanied by a tool, *TripleCheckMate* wherein a user assesses an individual resource and evaluates each fact for correctness, which is the main focus of this paper (Section 3). With this study we not only aim to assess the quality of DBpedia but also to adopt a methodology to improve the quality in future versions by regularly providing feedback to the DBpedia maintainers to fix these problems.

2 Quality Assessment Methodology

In this section, we describe a generalized methodology for the manual assessment and subsequent data quality improvement of resources belonging to a dataset. This methodology consists of the following four steps: (1) Resource selection, (2) Evaluation mode selection, (3) Resource evaluation and (4) Data quality improvement. In the following, we describe these steps in more detail.

Step I: Resource selection In this first step, the resources belonging to a particular dataset are selected. This selection can be performed in three different ways:

- *Per Class:* select resources belonging to a particular class
- *Completely random:* a random resource from the dataset
- *Manual:* a resource selected manually from the dataset

Choosing resources per class (e.g. animal, sport, place etc.) gives the user the flexibility to choose resources belonging to only those classes she is familiar with. However, when choosing resources from a class, the selection should be made in proportion to the number of instances of that class. Random selection, on the other hand, ensures an unbiased and uniform coverage of the underlying dataset. In the manual selection option, the user is free to select resources with problems that she has perhaps previously identified.

Step II: Evaluation mode selection The assignment of the resources, selected in Step I, to a person can be accomplished in the following three ways:

- *Manual:* the selected resources are assigned to a person (or group of individuals) who will then proceed to manually evaluate the resources individually.

- *Semi-automatic:* selected resources are assigned to a semi-automatic tool which performs data quality assessment employing some form of user feedback.
- *Automatic:* the selected resources are given as input to an automatic tool which performs the quality assessment without any user involvement.

Step III: Resource evaluation In case of manual assignment of resources, the person (or group of individuals) evaluates each resource individually to detect the potential data quality problems. In order to support this step, a quality assessment tool can be used which allows a user to evaluate each individual triple belonging to a particular resource.

Step IV: Data quality improvement After the evaluation of resources and identification of potential quality problems, the next step is to improve the data quality. There are at least two ways to perform an improvement:
- Direct: editing the triple, identified to contain the problem, with the correct value
- Indirect: using the Patch Request Ontology[3] [2] which allows gathering user feedbacks about erroneous triples.

A systematic review done in [7] identified a number of different data quality dimensions (criteria) applicable to Linked Data. After carrying out an initial data quality assessment on DBpedia (as part of the first phase of the manual assessment methodology), the problems identified were mapped to this list of identified dimensions. In particular, *Accuracy, Relevancy, Representational-consistency and Interlinking* were identified to be problems affecting a large number of DBpedia resources. Additionally, these dimensions were further divided into categories and sub-categories.

Table 1 gives an overview of these data quality dimensions along with their categories and sub-categories. Moreover, the table specifies whether the problems are specific to DBpedia (marked with a ✔) or could potentially occur in any RDF dataset. For example, the sub-category *Special template not properly recognized* is a problem that occurs only in DBpedia due to the presence of specific keywords in Wikipedia articles that do not cite any references or resources (e.g. {{Unreferenced stub—auto=yes}}). On the other hand, the problems that are not DBpedia specific can occur in any other datasets.

3 TripleCheckMate

TripleCheckMate is a tool built specifically for the purpose of the Quality Assessment Methodology (cf. Section 2). The tool is released as open source[4] under the Apache License. A successful use case of the tool can be seen in the *DBpedia*

[3] `http://141.89.225.43/patchr/ontologies/patchr.ttl#`
[4] `https://github.com/AKSW/TripleCheckMate`

Dimension	**Category**	**Sub-category**	**DBpedia specific**
Accuracy	Triple incorrectly extracted	Object value is incorrectly extracted	–
		Object value is incompletely extracted	–
		Special template not properly recognized	✔
	Datatype problems	Datatype incorrectly extracted	–
	Implicit relationship between attributes	One fact encoded in several attributes	✔
		Several facts encoded in one attribute	–
		Attribute value computed from another attribute value	✔
Relevancy	Irrelevant information extracted	Extraction of attributes containing layout information	✔
		Redundant attribute values	–
		Image related information	✔
		Other irrelevant information	–
Represensational-Consistency	Representation of number values	Inconsistency in representation of number values	–
Interlinking	External links	External websites	–
	Interlinks with other datasets	Links to Wikimedia	–
		Links to Freebase	–
		Links to Geospecies	–
		Links generated via Flickr wrapper	–

Table 1. Data quality dimensions, categories and sub-categories identified in the DBpedia resources. The DBpedia specific column denotes whether the problem type is specific only to DBpedia (tick) or could occur in any RDF dataset.

Evaluation Campaign[5] where 58 users evaluated a total of 792 resources (521 distinct) and 2928 distinct triples. A thorough description of that work can be found in [6].

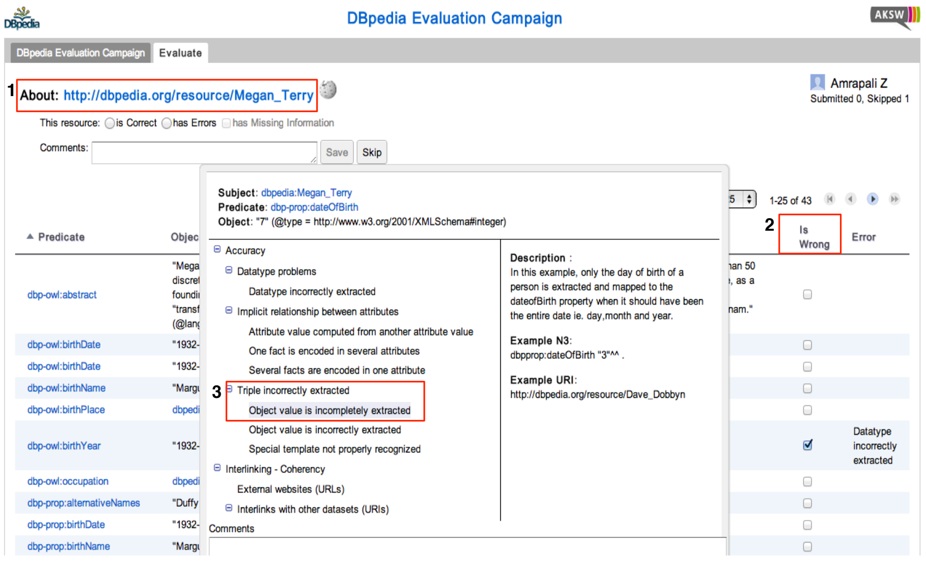
In the following, we describe TripleCheckMate from a user perspective (Section 3.1) as well as the system architecture (Section 3.2) and extensibility of the tool (Section 3.3).

3.1 Overview

The design of TripleCheckMate is aligned with the methodology described in Section 2, in particular with Steps 1–3. To use the tool, the user is required to authenticate herself, which not only prevents spam but also helps in keeping track of her evaluations. After authenticating herself, she proceeds with the selection of a resource (Step 1). She is provided with three options: (i)*per class*, (ii)*completely random* and (iii)*manual* (as described in Step I of the assessment methodology).

After selecting a resource, the user is presented with a table showing each triple belonging to that resource on a single row. Step 2 involves the user evaluating each triple and checking whether it contains a data quality problem. The link to the original Wikipedia page for the chosen resource is provided on top of the page which facilitates the user to check against the original values. If the triple contains a problem, she checks the box "is wrong". Moreover, she is provided with a taxonomy of pre-defined data quality problems (cf. Table 1) where she assigns each incorrect triple to a problem. If the detected problem does not

[5] `http://nl.dbpedia.org:8080/TripleCheckMate/`


DBpedia Evaluation Campaign

DBpedia Evaluation Campaign | Evaluate

1 About: http://dbpedia.org/resource/Megan_Terry

This resource: ◯ is Correct ◯ has Errors ☐ has Missing Information

Comments: Save Skip

Amrapali Z
Submitted 0, Skipped 1

Subject: dbpedia:Megan_Terry
Predicate: dbp-prop:dateOfBirth
Object: "7" (@type = http://www.w3.org/2001/XMLSchema#integer)

- Accuracy
  - Datatype problems
    - Datatype incorrectly extracted
  - Implicit relationship between attributes
    - Attribute value computed from another attribute value
    - One fact is encoded in several attributes
    - Several facts are encoded in one attribute
  - 3 Triple incorrectly extracted
    - Object value is incompletely extracted
    - Object value is incorrectly extracted
    - Special template not properly recognized
- Interlinking - Coherency
  - External websites (URLs)
  - Interlinks with other datasets (URIs)

Comments

Description :
In this example, only the day of birth of a person is extracted and mapped to the dateofBirth property when it should have been the entire date ie. day,month and year.

Example N3:
dbpprop:dateOfBirth "3"^^ .

Example URI:
http://dbpedia.org/resource/Dave_Dobbyn

1-25 of 43

Predicate	Object	2 Is Wrong	Error
dbp-owl:abstract	"Mega discre foundi "transf (@lan … han 50 …, as a … nam."	☐	
dbp-owl:birthDate	"1932-	☐	
dbp-owl:birthDate	"1932-	☐	
dbp-owl:birthName	"Margu	☐	
dbp-owl:birthPlace	dbpedi	☐	
dbp-owl:birthYear	"1932-	✓	Datatype incorrectly extracted
dbp-owl:occupation	dbpedi	☐	
dbp-prop:alternativeNames	"Duffy	☐	
dbp-prop:birthDate	"1932-	☐	
dbp-prop:birthName	"Margu	☐	

Fig. 1. Screenshot of the TripleCheckMate data quality assessment tool. First, a user chooses a resource. Second, she is displayed with all triples belonging to that resources and evaluates each triple individually to detect quality problems. Third, If she finds a problem, she marks it and associates it with a relevant problem category. A demo[8]and a screencast[9]of the tool are available.

match any of the existing types, she has the option to provide a new type and extend the taxonomy. After evaluating one resource, the user saves the evaluation and proceeds to choosing another random resource and follow the same procedure.

An important feature of the tool is to allow measuring of inter-rater agreements. This means, when a user selects a random method (*Any* or *Class*) to choose a resource, there is a 50% probability that she is presented with a resource that was already evaluated by another user. This probability as well as the number of evaluations per resource is configurable. Allowing many users evaluating a single resource not only helps to determine whether incorrect triples are recognized correctly but also to determine incorrect evaluations (e.g. incorrect classification of problem type or marking correct triples as incorrect), especially when crowdsourcing the quality assessment of resources.

3.2 Architecture

TripleCheckMate is built with *Java* using the *Google Web Toolkit*[10] (GWT) development toolkit. GWT allows one to build and optimize complex browser-

[9] http://nl.dbpedia.org:8080/TripleCheckMate-Demo
[9] http://www.youtube.com/watch?feature=player_embedded&v=l-StthTvjFI
[10] https://developers.google.com/web-toolkit/

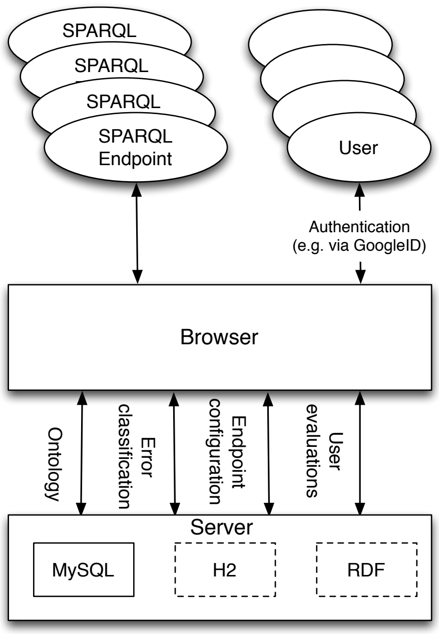


Fig. 2. Architecture of the TripleCheckMate tool.

based applications as it provides a static typed programming interface (Java) and compiles the output to native cross-browser HTML+Javascript. A Java Web Server (Apache Tomcat or Jetty) is used as a backend along with a MySQL database engine.

Figure 2 depicts the general TripleCheckMate architecture. In order to minimize the dependencies and make TripleCheckMate as portable as possible, all the application logic is built on the frontend. The applications' backend is used only to store and retrieve evaluation related data. The database schema of the backend is depicted in Figure 3.

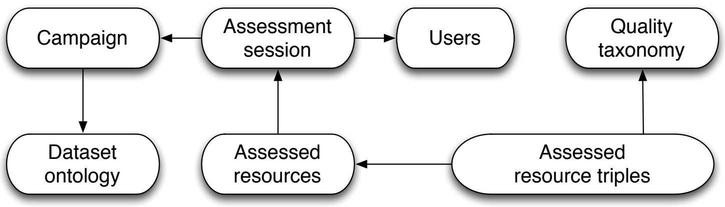


Fig. 3. The backend database schema where the arrows depict the foreign key constrains relations between the tables.

When the user enters the application, the available campaign and all the relevant configuration is loaded. In the future we plan to support multiple simultaneous campaigns. The user authentication is performed through the *Google OAuth 2.0*[11] protocol. If the user enters the application for the first time, her entry is transparently stored in the users table. A new session is created every time a user enters the application and all the evaluations in that session are associated. In order to speed-up the *per class* resource selection option, the class hierarchy is cached in the *Dataset ontology* table.

Finally, our data model separates the general resource evaluation (*Assessed resources*) from the detailed triple evaluation (*Assessed resource triples*). The rationale for this are cases where the user wants to mark a resource as completely correct or comment on missing information. In *Assessed resource triples* we store detailed evaluations at the triple level and assign errors to triples based on the *Quality taxonomy.*

Technically, after every complete resource evaluation, the evaluation results are submitted to the server and the user statistics are re-aggregated for an up-to-date contributor ranking. The communication to the server is performed with RPC requests facilitated by the GWT developer toolkit. Finally, for the SPARQL Endpoint communication we implemented a very lightweight client-side SPARQL framework where we encode SPARQL queries in *GET* requests and parse results in the JSON-LD format.

3.3 Extensibility

Although TripleCheckMate was initially built for the purpose of the *DBpedia Evaluation Campaign*, it can meanwhile be easily customized to support any arbitrary (open or closed) dataset with a SPARQL endpoint.

Most of the configurations lie in the database. The SPARQL endpoint configuration is stored in the *campaign* table. The Dataset ontology and *Quality Taxonomy* hold the respective data as tree structures and, thus, can be easily changed to any ontology or taxonomy. The database connection configuration is stored in property files. Finally, visual end-user changes can be performed directly into the HTML and GWT Layout files.

As a data store backend, we support MySQL at the moment, although any JDBC compatible database can be used. The embedded in-memory H2 database is also supported as it provides a JDBC interface. We plan to ship TripleCheckMate with a preconfigured H2 database, as a standalone evaluation tool. Finally, RDF as data store is also a feature our community base expressed interest to implement.[12]

[11] `https://developers.google.com/accounts/docs/OAuth2`

[12] `https://groups.google.com/d/topic/dbpedia-data-quality/rkXfR1BR4uY/discussion`

4 Conclusions and Future Work

In this paper, we described *TripleCheckMate*, a tool for crowdsourcing the assessment of Linked Data. We discussed the architecture, usability and extensibility of the tool. Additionally, we described a data quality assessment methodology which included a quality problem taxonomy. The methodology and taxonomy are integral parts of the tool. The tool has already been successfully tested in assessing the quality of DBpedia and can be easily configured to work with any dataset that provides a SPARQL endpoint.

In future versions of the tool, we will include further support for the methodology outlined in Section 2 by directly integrating semi-automatic methods, which can then filter those triples of a resource, which are most likely to cause problems. We will investigate whether this can improve the efficiency of the quality assessment. Moreover, we also plan to include support for the patch ontology [2] as an output format.

Acknowledgment

This work was supported by grants from the EU's 7th Framework Programme provided for the projects LOD2 (GA no. 257943), DIACHRON (GA no. 601043) and GeoKnow (GA no. 318159).

References

1. J. Juran. *The Quality Control Handbook*. McGraw-Hill, New York, 1974.
2. M. Knuth, J. Hercher, and H. Sack. Collaboratively patching linked data. *CoRR*, 2012.
3. J. Lehmann, C. Bizer, G. Kobilarov, S. Auer, C. Becker, R. Cyganiak, and S. Hellmann. DBpedia - a crystallization point for the web of data. *Journal of Web Semantics*, 7(3):154–165, 2009.
4. J. Lehmann, R. Isele, M. Jakob, A. Jentzsch, D. Kontokostas, P. N. Mendes, S. Hellmann, M. Morsey, P. van Kleef, S. Auer, and C. Bizer. Dbpedia - a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic Web Journal*, 2013. Under review.
5. M. Morsey, J. Lehmann, S. Auer, C. Stadler, and S. Hellmann. DBpedia and the Live Extraction of Structured Data from Wikipedia. *Program: electronic library and information systems*, 46:27, 2012.
6. A. Zaveri, D. Kontokostas, M. A. Sherif, L. Bühmann, M. Morsey, S. Auer, and J. Lehmann. User-driven quality evaluation of dbpedia. In *To appear in Proceedings of 9th International Conference on Semantic Systems, I-SEMANTICS '13, Graz, Austria, September 4-6, 2013*. ACM, 2013.
7. A. Zaveri, A. Rula, A. Maurino, R. Pietrobon, J. Lehmann, and S. Auer. Quality assessment methodologies for linked open data. Under review, available at http://www.semantic-web-journal.net/content/quality-assessment-methodologies-linked-open-data.