QUANT - Question Answering Benchmark Curator

Ria Hari Gusmita[1(✉)], Rricha Jalota[1], Daniel Vollmers[1], Jan Reineke[1], Axel-Cyrille Ngonga Ngomo[1,2], and Ricardo Usbeck[1,2]

[1] University of Paderborn, 33098 Paderborn, Germany
{ria.hari.gusmita,rricha.jalota,daniel.vollmers,jan.reineke,
axel.ngonga,ricardo.usbeck}@uni-paderborn.de
[2] University of Leipzig, 04109 Leipzig, Germany
{ngonga,usbeck}@informatik.uni-leipzig.de

Abstract. Question answering engines have become one of the most popular type of applications driven by Semantic Web technologies. Consequently, the provision of means to quantify the performance of current question answering approaches on current datasets has become ever more important. However, a large percentage of the queries found in popular question answering benchmarks cannot be executed on current versions of their reference dataset. There is a consequently a clear need to curate question answering benchmarks periodically. However, the manual alteration of question answering benchmarks is often error-prone. We alleviate this problem by presenting QUANT, a novel framework for the creation and curation of question answering benchmarks. QUANT supports the curation of benchmarks by generating smart edit suggestions for question-query pair and for the corresponding metadata. In addition, our framework supports the creation of new benchmark entries by providing predefined quality checks for queries. We evaluate QUANT on 653 questions obtained from QALD-1 to QALD-8 with 10 users. Our results show that our framework generates reliable suggestions and can reduce the curation effort for QA benchmarks by up to 91%.

Keywords: Benchmark · Question answering · Knowledge base

1 Introduction

Question answering (QA) engines are at the core of an increasing number of human computer interfaces, including personal assistants and chatbots [9]. The development of accurate QA frameworks for (RDF) knowledge graphs has hence become an endeavor of increasing importance and popularity [14,16]. Consequently, the provision of means to evaluate the performance of QA systems on *current datasets* is critical to (1) monitor the improvement of the state of art over past approaches and (2) provide realistic insights in relevant improvements for question answering systems on current challenges found in datasets. Benchmark series such as the Question Answering on Linked Data (QALD) series [15] address

M. Acosta et al. (Eds.): SEMANTiCS 2019, LNCS 11702, pp. 343–358, 2019.
https://doi.org/10.1007/978-3-030-33220-4_25

this need for objective evaluation. They support QA researchers and developers by providing new versions of their benchmarks periodically. However, maintaining high-quality and current benchmark datasets is a challenging endeavor. In particular, changes in the knowledge base underlying the benchmarks (as well as metadata annotation errors) lead to a large proportion of the queries in previous benchmarks not being executable on current versions of datasets. Table 1 gives an overview of the extend of the degradation of the QALD benchmarks over time. A significant proportion of the SPARQL queries that were not modified over time degraded (i.e., could not be executed) with newer versions of the knowledge base underlying QALD. For example, more than 30% of the QALD-4 benchmark cannot be executed on DBpedia 2014, which was release a mere year after the publication of QALD-4.

Table 1. Degradation of QALD benchmarks against various versions of DBpedia (in %). The numbers in brackets indicate total number of questions.

DBpedia version	QALD-1 (44)	QALD-2 (87)	QALD-3 (88)	QALD-4 (177)	QALD-5 (262)	QALD-6 (350)	QALD-7 (215)	QALD-8 (219)
3.6	18.18							
3.7	25.00	16.09						
3.8	31.82	20.69	17.05					
3.9	54.55	41.38	40.90	25.99				
2014	50.00	39.08	40.90	30.50	24.43			
2015–04	36.36	27.58	23.86	18.08	13.74			
2015–10	36.36	26.44	23.86	18.08	12.59	10.57		
2016–04	36.36	26.44	25.00	20.90	14.88	14.00	4.19	
2016–10	43.18	33.33	32.95	25.99	20.23	20.00	12.09	0

Addressing the challenge of updating a QA benchmark to the current schema of a dataset is a tedious, time-consuming and error-prone endeavor (see Sect. 4 for numbers). In this paper, we alleviate this problem by providing *QUANT, a framework for the intelligent creation and curation of QA benchmarks*. QUANT regards the i^{th} version B_i of a QA benchmark as a pair (D_i, Q_i) composed of a dataset D_i and a set of questions Q_i. One of the core functions of QUANT is the generation of intelligent suggestions for benchmark curators (i.e., users annotating and improving a QA benchmark): Given a query $q_{ij} \in Q_i$ with zero results on D_k with $k > i$ (i.e., on a newer version of D_i), QUANT's suggestions aim to provide a small number of modifications to q_{ij}, such that the modified q_{ij} i.e. q'_{ij}, can be executed on D_k with non-zero results. We call this modification process for queries *porting* the queries from version i to version k. With these smart suggestions, QUANT aims (1) to ensure that queries from B_i can be reused for B_k (e.g., as training queries) and (2) to speed up the curation process as compared to the commonly used manual and text-editor-based creation and curation process [15]. To achieve this goal, QUANT (1) supports *the creation of SPARQL queries* answering a particular information need as well as the execution of said query against a predefined endpoint or knowledge base.

Moreover, QUANT checks (2) *the validity of benchmark metadata* as well as (3) the *spelling and grammatical correctness of questions* across multiple languages both in their natural-language query and keyword form.

To demonstrate the usability of QUANT and the efficiency of the smart suggestions, we performed two extensive evaluation campaigns. First, we analyzed the performance gain using QUANT over the tradition manual curation process with 3 experts. The results show that we decreased the required curation time by 91% while keeping the inter-rater agreement at 0.82. Second, we used QUANT to create a new joint benchmark from 8 QALD datasets. The smart suggestions were accepted by 83.75% of the users on average, indicating their usefulness. The novel, large and high-quality QA benchmark dataset, called QALD-9, is available at https://github.com/ag-sc/QALD/tree/master/9/data.

2 Related Work

The work on QUANT is related to three research areas, namely (1) workshops and evaluation campaigns, (2) datasets for QA over knowledge graphs and (3) curation tools for benchmarks.

2.1 Workshops and Evaluation Campaigns

A number of challenges and campaigns attracting researchers as well as industry practitioners to QA have seen the light of day over the last two decades. Since 1998, the TREC conference, especially the QA track [17], aims to provide domain-independent evaluations over large, unstructured corpora. The CLEF campaigns on information retrieval has a more than 10-year tradition in evaluating IR systems [1]. The well-known QALD (Question Answering over Linked Data) [15] campaign, currently running in its 9th instantiation, is a diverse evaluation series which include questions, of which the answer can be computed (1) based on a single RDF knowledge base, (2) by combining RDF and textual data, (3) using several knowledge bases. The benchmarks cover several domains, including encyclopedia knowledge and music. Given that this series of benchmarks is openly available and widely used ([5,7] points to 30 systems, which were evaluated using QALD), we will use the QALD datasets to evaluate QUANT.

2.2 QA over Knowledge Graphs

Other QA datasets emerged apart from the above-mentioned challenges. LCQuAD [13] is one of the largest QA over knowledge bases benchmarks with 5000 questions and their corresponding SPARQL queries over the 2016-04 version of DBpedia.[1] It also provides a framework for generating natural language questions and their corresponding SPARQL queries, minimizing the domain expert intervention. However, these questions are often grammatically incorrect and

[1] http://dbpedia.org.

require manual paraphrasing. Out of the 5000 LCQuAD SPARQL queries, 2570 queries could not be answered by the 2016-10 DBpedia version and interestingly, 456 queries were not answered by the 2016-04 version. We performed this evaluation before LCQuAD was updated.[2] Free917 [3] and WebQuestions[3] are widely used in the Semantic Web as well as Deep Learning community. Cai and Yates [3] manually created the Free917 dataset consisting of 917 questions and their logical forms, tailored to around 600 Freebase properties. Berant et al. [2] generated the WebQuestions dataset by using the Google Suggest API to collect 1M questions and got a subset of them (100K) labeled on Freebase by Amazon Mechanical Turk works. Yih et al. [18] built the WebQuestionsSP dataset[4] by re-annotating the WebQuestions dataset. WebQuestionsSP, unlike its parent dataset, contains the natural language questions, their semantic parses in the form of SPARQL queries and the derived answers. For annotating the dataset with SPARQL queries, they designed a dialog-like user interface to fasten the process which is unfortunately no longer available.

2.3 Curation Tools for Benchmarks

Since both manual curation and crowd-sourcing for benchmark creation are tedious and time-consuming tasks, there is a need for tools that speed up the process while reducing annotation errors. Jha et al. [8] built Eaglet, a semi-automatic benchmark curation tool for named entity recognition and entity linking (NER/EL). The framework checks for anomalies in a gold standard, based on the rules derived from the existing gold standards for annotating documents for NER/EL. Duan et al. [6] introduced an RDF storage benchmark generator to convert any dataset into a benchmark dataset (to reduce the gap between real and benchmark RDF data) for evaluating the performance of RDF stores, by formulating the benchmark generation problem as an integer programming problem. It is capable of generating data that resembles the characteristics (structuredness, size, and content) of real datasets with user-specified data properties. Lance [11] is a domain-independent, generic benchmark generator for Instance Matching systems; supports semantics-aware transformations with varying degrees of difficulty and creates a weighted gold standard for a better evaluation of the performance of instance matching tools. The interface accepts user-provided specifications to generate a benchmark. All the above-mentioned tools are similar to QUANT in the sense that they make benchmark generation easier for end users and employ strategies derived from an analysis on previous gold standards to improve the quality of the resulting dataset. However, to the best of our knowledge, there is no tool similar to QUANT in the domain of QA over knowledge bases.

[2] http://lc-quad.sda.tech/.
[3] https://goo.gl/93iqgC.
[4] http://aka.ms/WebQSP.

3 Approach

3.1 Architecture and Workflow

QUANT has a modular design comprising (1) a preprocessing module to eliminate duplicates in case several datasets get loaded, (2) a machine translation module to automatically translate text into 10 languages, (3) a keyword generation module to make the QA datasets suitable even for keyword-based information retrieval evaluations, and finally (4) a curation module to serve smart suggestions as can be seen in Fig. 1.

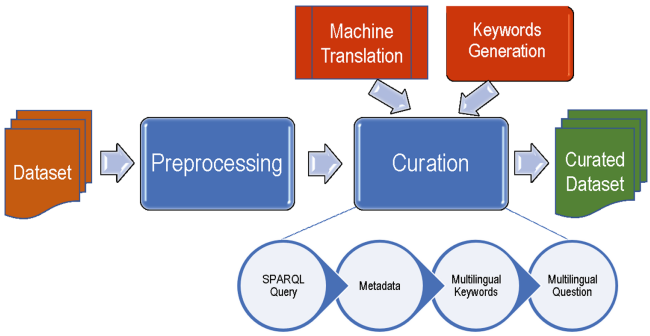

Fig. 1. QUANT's modular architecture

A curation process consists of (a) a user logging into QUANT, (b) determining an endpoint URI and version for the target knowledge base, (c) uploading a (QALD-JSON formatted) dataset. A user can then either d) create, (e) delete or (f) curate questions using smart suggestions. Finally, a user can export the curated dataset into the widely accepted QALD-JSON format [16]. Examples of a dataset formatted using QALD-JSON are available herein.[5]

3.2 Smart Suggestions

The most distinctive features of QUANT which enhance the overall curation productivity are smart suggestions for every attribute (SPARQL query or metadata) in case they contain a wrong value. That is, the system automatically detects the presence of potentially incorrect entries and offers hints pertaining to how to correct them. In the subsequent paragraphs, we explain how we provide smart suggestions for (a) question to SPARQL mappings as well as for (b) metadata attributes, and (c) question or keyword translations.

[5] https://github.com/ag-sc/QALD/tree/master/9/data.

SPARQL Suggestion. There can be various reasons for a SPARQL query (S) that worked on a previous version of the knowledge base to not work against another version or another endpoint. If QUANT is not able to fetch results from the current SPARQL endpoint, it activates the SPARQL Correction and Suggestion curation module. QUANT either suggests a new SPARQL query or renders the failure case if the correction module fails, to allow for a manual curation by the user. The cases that we applied for SPARQL Correction have been described below.

- Missing prefixes: QUANT first checks whether the query (S) fails due to missing prefixes. We call a prefix missing if it has been used in the query but not been defined in the beginning. An example of such a query is,

```
select ?s where { res:New_Delhi dbo:country ?s .}
```

  where the corrected query is:

```
PREFIX dbo: <http://dbpedia.org/ontology/>
PREFIX res: <http://dbpedia.org/resource/>
select ?s where { res:New_Delhi dbo:country ?s .}
```

  Henceforth, we assume that a query contains the correct prefixes.
- Predicate change: For every triple in the query with either a known subject or object (`pre:knownEntity` in the example below), we check if the predicate changed in the underlying knowledge base. The original predicate is preserved and a new SPARQL query (S') is formed to search for all the predicates that are associated with either the known subject or the known object. However, if in a SPARQL triple, both subject and object are unknown, then all previous triples (if present) (`prevSubject prevPredicate prevObject.`) are added to the new SPARQL query (S'). That is, if the triple being checked is of the form `?s pre:Predicate ?o` and there exists a triple preceding it, that provides the value for either the unknown subject (`?s`) or the unknown object (`?o`) , then in the new SPARQL query (S'), the previous triple is used to limit the search space for predicate testing. The new SPARQL query (S') can have one of the following forms.
  First form:

```
select ?p where { pre:knownEntity ?p ?o }
```

  Second Form:

```
select ?p where { ?s ?p pre:knownEntity }
```

  Third Form:

```
select ?p where { ?unknownSubj prevPredicate prevObject. ?
    ↪ unknownSubj ?p ?o.} or
select ?p where { prevSubject prevPredicate ?unknownObj. ?s ?p
    ↪ ?unknownObj.} or
```

```
select ?p where { ?unknownSubj prevPredicate prevObject. ?s ?p
    ↪ ?unknownSubj.} or
select ?p where { prevSubject prevPredicate ?unknownObj. ?
    ↪ unknownObj ?p ?o.}
```

All the resulting predicates that match or contain the original predicate's label are stored. By replacing the original predicate with each of these stored predicates in the original SPARQL query, we check if the query produces non-zero results. If it works, we suggest this newly formed query to the user. Note, if we need to apply the third case, it is a match against all predicates in the knowledge base that arise from the result of the previous triple(s).

- Predicate Missing: If none of the resulting predicates match or do not contain the original predicate's label, the user is informed about the missing predicate in the triple for manual curation.
- Entity Change: Each known entity (subject or object) in the triple is checked in the knowledge base. In DBpedia, if the entity is not found and belongs to a YAGO class, we append 'Wikicat' (which is a YAGO-specific update on DBpedia's later versions) at the beginning of the entity label and check again. If this new YAGO class is present in the knowledge base, we check if the SPARQL query works with it and suggest it to the user, if it does. If the missing entity is not a YAGO class, we check if there is a redirection on DBpedia for this entity by using the following SPARQL query:

```
select ?redirect where
{ <entityToBeChecked> dbo:wikiPageRedirects ?redirect. }
```

If a redirect is found, the updated SPARQL query is tested against the endpoint and suggested to the user, if it returns an answer. Note, we were aware that this method is highly tailored towards DBpedia but can be adapted to any Linked Data knowledge graph using standard attributes such as `owl:sameAs` and `skos:related`.

- Entity Missing: The user is informed about the missing entity in the triple if the procedures to find an alternative entity fails.

If there are no suggestions generated after performing the checks above, QUANT permutes the order of the triple patterns within the conjunctive clauses to which they belong and reruns the SPARQL correction pipeline. While the order of the triple patterns in such clauses does not matter, it does affect the search space when we test for entities and predicates. Hence, by changing the order of triples, we can either narrow down or broaden the search space and increase the probability of correcting the SPARQL query and returning a suggestion.

The following examples depict SPARQL suggestions or messages returned by QUANT when it receives an outdated query.

- Entity change:
  Degraded SPARQL query:

```
SELECT ?uri WHERE
{ ?uri rdf:type yago:CapitalsInEurope }
```

QUANT suggestion:

```
SELECT ?uri WHERE
{ ?uri rdf:type yago:WikicatCapitalsInEurope }
```

– Predicate missing:
Degraded SPARQL query:

```
SELECT ?uri WHERE
{ ?subject rdfs:label "Tom Hanks".
?subject foaf:homepage ?uri }
```

QUANT suggestion:

```
The predicate foaf:homepage is missing in ?subject foaf:
    ↪ homepage ?uri
```

– Predicate change, Query Permutation:
Degraded SPARQL query:

```
SELECT ?date WHERE
{ ?website rdf:type onto:Software .
?website onto:releaseDate ?date .
?website rdfs:label "DBpedia" . }
```

QUANT suggestion:

```
SELECT ?date WHERE
{ ?website rdf:type onto:Software .
?website rdfs:label "DBpedia" .
?website dbp:latestReleaseDate ?date . }
```

Metadata Suggestion. QA benchmark metadata can be used to tailor benchmarks to the needs or research directions that a QA system follows, e.g., to ignore questions which need aggregation operations or to especially focus on them [5,7]. QUANT provides formal checks for the metadata entries found in QA benchmarks.

For example, the *answer type* tag corresponds to the data type of the answer returned by the SPARQL endpoint. There are five possible data types (i.e., Boolean, Date, Number, Resource, and String). If the existing value of answer type is not suitable for the returned answer, QUANT will suggest the correct one based on a regular expression. The *aggregation* tag defines whether the SPARQL query contains one or more aggregation functions such as COUNT, SUM, AVG, MIN, MAX, SAMPLE, GROUP_CONCAT, VECTOR_AGG, and

COUNT DISTINCT.[6] If the SPARQL query contains at least one of these functions, *aggregation* must be set to true, otherwise it is set to false. QUANT detects the presence of these functions in the query and suggests the correct value. The *hybrid* metadata entry describes whether it is required to search not only the Linked Data knowledge base but also textual data to produce an answer. The SPARQL query of a hybrid question will mostly contain the phrase `text:query` or `if:contains` in it. In this case, this attribute must be set to True. *onlydbo* is a binary flag which states whether the SPARQL query contains URIs which belong exclusively to the DBpedia namespace. QUANT examines all the URIs, both long forms and abbreviations,[7] in the SPARQL query to check if they belong to DBpedia and suggests the correct value for this field. *out-of-scope* tag denotes a SPARQL query that is not able to retrieve answers from a SPARQL endpoint or when the answers are not semantically correct. If this is the case, *out-of-scope* must be set to true.

Multilingual Questions and Keywords Suggestion. To enable multilingual QA evaluation campaigns and foster more active research in this area, QA benchmarks are often made available in several languages. However, translating queries across languages in a consistent way entails a significant amount of manual effort. In the case of missing or incomplete translations, QUANT first applies stopwords and question-word removal techniques to generate missing keywords. Here, we rely on technique similar to those implemented in FOX [12]. Secondly, our framework applies an automated machine translation tool called Translate Shell[8] to provide translation-suggestions in 10 other languages for both questions and keywords. As machine translation is not perfect, the completion of the final translation remains the curator's task.

Figure 2 shows a screenshot of the framework which displays curation process.

4 Evaluation

Our evaluation had three goals: (1) compare the curation time using QUANT with manual curation time, (2) investigate the effectiveness of smart suggestions, and (3) determine how capable QUANT is in providing a high-quality benchmark dataset.

4.1 Efficiency Evaluation

First, we analyzed the performance gain using QUANT versus a manual curation. Our annotators were three graduate CS students with a good working knowledge of Linked Data. To avoid any inherent bias, the three graduate students

[6] https://www.w3.org/TR/sparql11-query/#Aggregates.
[7] https://www.w3.org/TR/2013/REC-sparql11-query-20130321/#sparqlSyntax.
[8] https://www.soimort.org/translate-shell/.

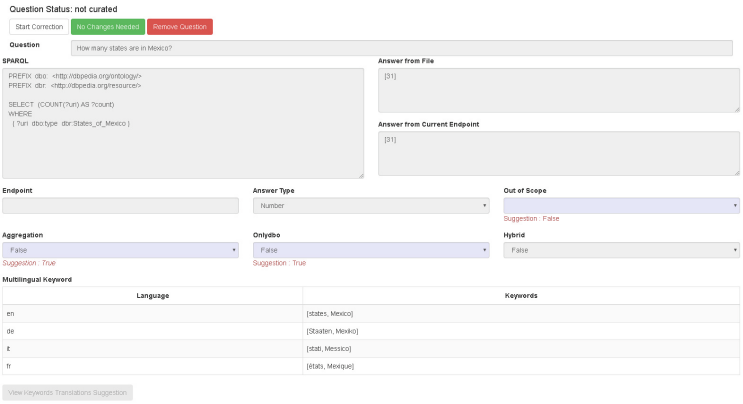

Fig. 2. Screenshot of QUANT's curation process

worked sequentially and without any prior knowledge of the data. They had to curate 50 questions manually and subsequently curate 50 different questions using QUANT. The results show that we decreased the needed time by 91% while keeping the inter-rater agreement from two of the users at 0.82, which stands for almost perfect agreement [4]. On average, users needed 23 min (between 22 and 25 min) using QUANT as opposed to 278 min (between 240 and 330 min) on average (more than 10×) using a manual curation approach.

4.2 Smart Suggestion Evaluation

Second, we used QUANT to create a new joint benchmark by joining the past 8 QALD datasets together and unifying them. We divided 10 expert users (Ph.D. students and senior researchers) into 5 pairs. The members of each pair had to curate exactly the same questions. The first four user pairs curated 130 questions, while the user pair worked on another 133 questions. This resulted in 653 questions, see also Sect. 5.

We monitored the number and types of suggestions accepted by the users throughout the curation process. Our evaluation results show that from 2380 suggestions provided by QUANT in total the acceptance rate from all the users was 81.04% on average (see Fig. 3). As seen in Fig. 4, most users accepted suggestions for the out-of-scope metadata, which, after correcting the SPARQL query, entailed a change to the questions' metadata.[9] Keyword and question translation suggestions yielded the second and third highest acceptance rates. We got higher acceptance rates (over 90%) mostly on questions from the later versions of QALD (QALD-7 and QALD-8), i.e., by users 1, 2, 9, and 10. Despite the fact that the questions are handed out to the curators in chronological order, we saw no effect of this ordering in the acceptance rate. Interestingly, the acceptance

[9] Note that if a user changed the SPARQL query manually using the hint from the suggestion, it is not added to the statistic.

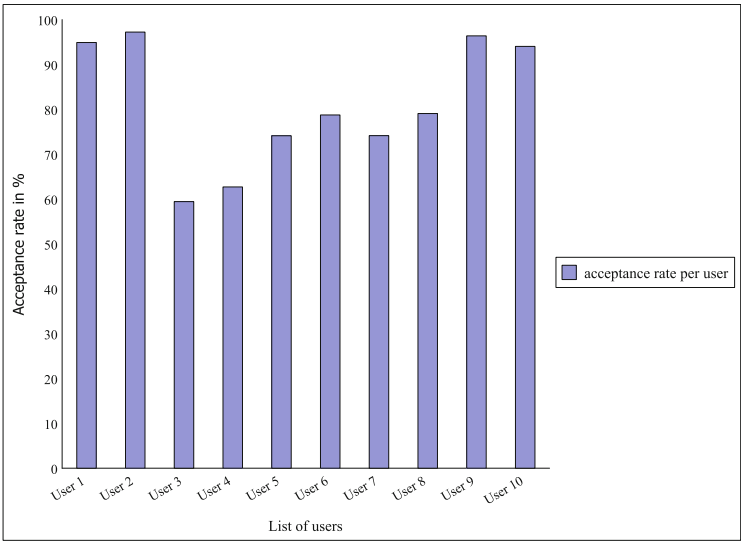

Fig. 3. Acceptance rate of all users

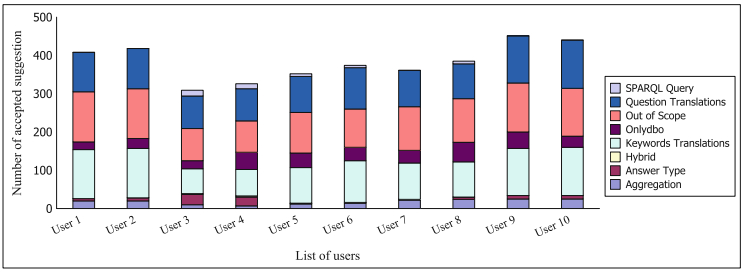

Fig. 4. Number of accepted suggestions for each attribute from all users

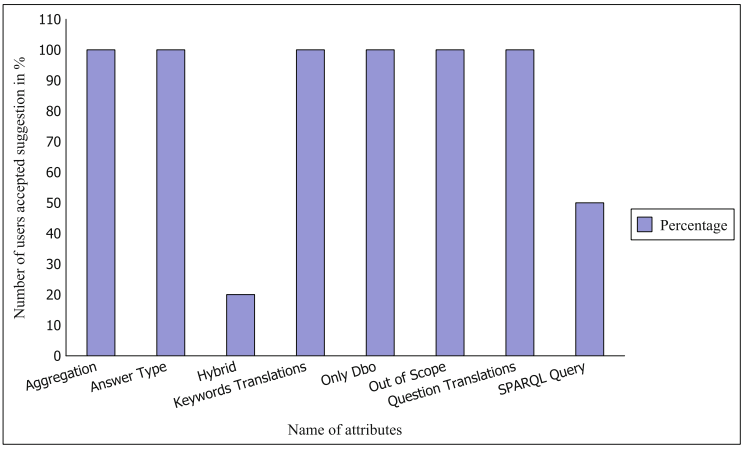

Fig. 5. Number of users accepted suggestions for each attribute

rate is independent of the number of suggestions. 83.75% of the users accepted QUANT's smart suggestions on average, see Fig. 5. However, the hybrid metadata attribute and the SPARQL suggestions were only accepted by 2 and 5 users respectively. We were also interested to know how many attributes were changed without using smart suggestions and redefined by users directly. During the evaluation with 10 users there were 4 attributes changed without using the suggestions, see Fig. 6. These are answer type, onlydbo, out-of-scope, and SPARQL query.

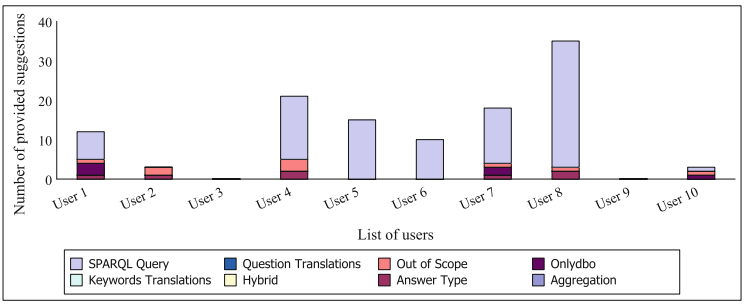

Fig. 6. Number of attributes whose value are provided by users

Finally, we computed the inter-rater agreement between each pair of users which shared the same questions. Our results are shown in Table 2 and suggest from very good to almost perfect agreement among the users [4]. This is very positive result as it suggests that our framework provides consistently helpful suggestions to its users.

Table 2. Inter-rater agreement over 5 annotator pairs curating at least 130 questions

Group	Inter-rater agreement
1st two-users	0.97
2nd two-users	0.72
3rd two-users	0.88
4th two-users	0.77
5th two-users	0.96

5 QALD-Specific Analysis

In total, there are 1924 questions where 1442 questions are training data and 482 questions are test data across the different versions of QALD we considered. So far, novel QALD train datasets were created by merging the test and training

questions of the previous QALD version. The test dataset for a new QALD version is normally based on completely new questions extracted from search engine or chatbot log files [15]. It can be seen in Fig. 7 that the real distribution of QALD-train dataset in almost all versions unfortunately does not represent the ideal distribution. The change of the knowledge base contributes in the sense it causes several questions become unanswerable so that they have to be removed from dataset.

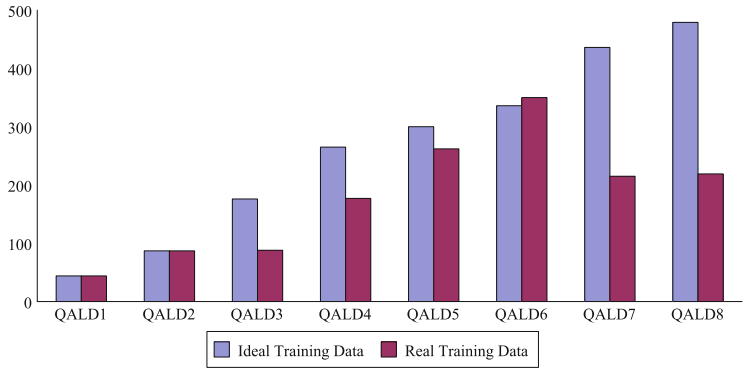

Fig. 7. Ideal and real distribution of QALD training data in all versions

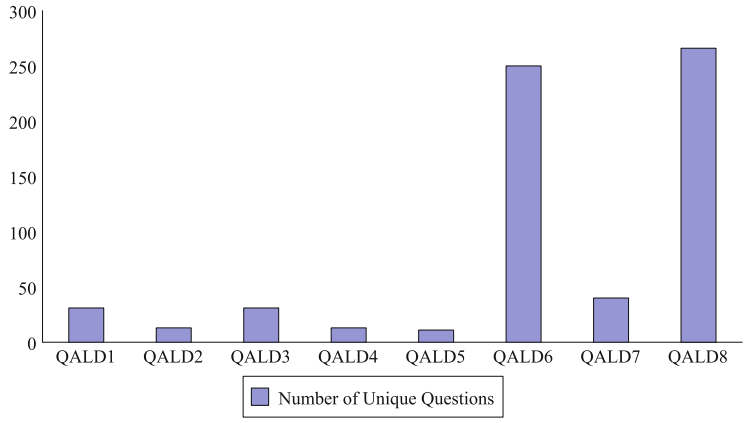

Fig. 8. Distribution of unique questions in all QALD versions

Our analysis discovered there are many exact duplicates, i.e., questions which were exactly the same in all attributes, in most QALD versions. We solved this problem by taking the one from latest version as it is more mature with respect to correctness and completeness of the question's attributes. Furthermore, there were only 655 unique questions as seen in Fig. 8. Sequentially, we removed 2 semantically similar questions so that finally we have 653 questions in total.

After applying the smart suggestions via 10 expert users, QUANT was able to produce a QALD-JSON formatted dataset of 558 total benchmark questions, increasing the size of QALD compared to QALD-8 by 110.6%. This dataset forms the QALD-9 dataset [10]. In particular, questions previously marked as out-of-scope in past challenges were curated such that they are now a valid question, and are thus treated as novel questions in this new QALD dataset.

6 Conclusion and Future Work

QUANT's evaluation highlights the need for better datasets and their maintenance. The degradation of datasets linked to the growing amount of Linked Data-based knowledge bases builds a barrier to novel research methods which are demanding large amounts of high-quality training data. We were able to show that QUANT speeds up the curation process by up to 91%. Furthermore, we saw that smart suggestions motivate users to engage in more attribute corrections than if there were no hints, compare Figs. 4 and 6. Also, we pointed out that we need to invest more time into SPARQL suggestions as only 5 users accepted them. This low acceptance rate is due to the tremendous changes in the underlying ontologies from one version to the other. Of course, we plan to support more file formats based on our internal library.[10]

Acknowledgments. This work was supported by the German Federal Ministry of Transport and Digital Infrastructure (BMVI) in the project LIMBO (no. 19F2029I) and by the German Federal Ministry of Education and Research (BMBF) in the project SOLIDE (no. 13N14456) within 'KMU-innovativ: Forschung für die zivile Sicherheit' in particular 'Forschung für die zivile Sicherheit'.

References

1. Agosti, M., Nunzio, G.M.D., Dussin, M., Ferro, N.: 10 years of CLEF data in DIRECT: where we are and where we can go. In: Proceedings of the 3rd International Workshop on Evaluating Information Access, EVIA 2010, National Center of Sciences, Tokyo, Japan, 15 June 2010, pp. 16–24 (2010)
2. Berant, J., Chou, A., Frostig, R., Liang, P.: Semantic parsing on freebase from question-answer pairs. In: Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, EMNLP 2013, A meeting of SIGDAT, a Special Interest Group of the ACL, Grand Hyatt Seattle, Seattle, Washington, USA, 18–21 October 2013, pp. 1533–1544 (2013)
3. Cai, Q., Yates, A.: Large-scale semantic parsing via schema matching and lexicon extension. In: ACL 2013, Volume 1: Long Papers, Sofia, Bulgaria, 4–9 August 2013, pp. 423–433 (2013)
4. Chaturvedi, S.R.B.H.: Evaluation of inter-rater agreement and inter-rater reliability for observational data: an overview of concepts and methods. J. Indian Acad. Appl. Psychol. **41**(3), 20–27 (2015)

[10] https://github.com/dice-group/QUANT.

5. Diefenbach, D., López, V., Singh, K.D., Maret, P.: Core techniques of question answering systems over knowledge bases: a survey. Knowl. Inf. Syst. **55**(3), 529–569 (2018)
6. Duan, S., Kementsietsidis, A., Srinivas, K., Udrea, O.: Apples and oranges: a comparison of RDF benchmarks and real RDF datasets. In: Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2011, Athens, Greece, 12–16 June 2011, pp. 145–156 (2011)
7. Höffner, K., Walter, S., Marx, E., Usbeck, R., Lehmann, J., Ngomo, A.N.: Survey on challenges of question answering in the semantic web. Semant. Web **8**(6), 895–920 (2017)
8. Jha, K., Röder, M., Ngonga Ngomo, A.-C.: All that glitters is not gold – rule-based curation of reference datasets for named entity recognition and entity linking. In: Blomqvist, E., Maynard, D., Gangemi, A., Hoekstra, R., Hitzler, P., Hartig, O. (eds.) ESWC 2017. LNCS, vol. 10249, pp. 305–320. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-58068-5_19
9. Malyshev, S., Krötzsch, M., González, L., Gonsior, J., Bielefeldt, A.: Getting the most out of wikidata: semantic technology usage in Wikipedia's knowledge graph. In: Vrandečić, D., et al. (eds.) ISWC 2018. LNCS, vol. 11137, pp. 376–394. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-00668-6_23
10. Usbeck, R., Gusmita, R.H., Ngomo, A.C.N., Saleem, M.: 9th challenge on question answering over linked data (QALD-9). In: Semdeep/NLIWoD@ ISWC, pp. 58–64 (2018)
11. Saveta, T., Daskalaki, E. Flouris, G., Fundulaki, I., Ngomo, A.N.: LANCE: a generic benchmark generator for linked data. In: Proceedings of the ISWC 2015 Posters and Demonstrations Track co-located with the 14th International Semantic Web Conference (ISWC 2015), Bethlehem, PA, USA, 11 October 2015 (2015)
12. Speck, R., Ngomo, A.N.: Ensemble learning of named entity recognition algorithms using multilayer perceptron for the multilingual web of data. In: Corcho, Ó., Janowicz, K., Rizzo, G., Tiddi, I., Garijo, D. (eds.) K-CAP 2017, pp. 26:1–26:4. ACM (2017)
13. Trivedi, P., Maheshwari, G., Dubey, M., Lehmann, J.: LC-QuAD: a corpus for complex question answering over knowledge graphs. In: d'Amato, C., et al. (eds.) ISWC 2017. LNCS, vol. 10588, pp. 210–218. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-68204-4_22
14. Unger, C., Ngomo, A.-C.N., Cabrio, E.: 6th open challenge on question answering over linked data (QALD-6). In: Sack, H., Dietze, S., Tordai, A., Lange, C. (eds.) SemWebEval 2016. CCIS, vol. 641, pp. 171–177. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46565-4_13
15. Usbeck, R., Ngomo, A.-C.N., Haarmann, B., Krithara, A., Röder, M., Napolitano, G.: 7th open challenge on question answering over linked data (QALD-7). In: Dragoni, M., Solanki, M., Blomqvist, E. (eds.) SemWebEval 2017. CCIS, vol. 769, pp. 59–69. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-69146-6_6
16. Usbeck, R., et al.: Benchmarking question answering systems. Semant. Web J. **10**(2), 293–304 (2018)
17. Voorhees, E.M., et al.: The TREC-8 question answering track report. TREC **99**, 77–82 (1999)
18. Yih, W., Richardson, M., Meek, C., Chang, M., Suh, J.: The value of semantic parse labeling for knowledge base question answering. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, Volume 2: Short Papers, Berlin, Germany, 7–12 August 2016 (2016)