Semantic Web -1 (2021) 1–17
DOI 10.3233/SW-210440
IOS Press

MQALD: Evaluating the impact of modifiers in question answering over knowledge graphs

Lucia Siciliani *, Pierpaolo Basile, Pasquale Lops and Giovanni Semeraro

Department of Computer Science, University of Bari Aldo Moro, Via E. Orabona 4, 70125, Bari, Italy
E-mails: lucia.siciliani@uniba.it, pierpaolo.basile@uniba.it, pasquale.lops@uniba.it, giovanni.semeraro@uniba.it

Editor: Harald Sack, FIZ Karlsruhe – Leibniz Institute for Information Infrastructure, Germany
Solicited reviews: Simon Gottschalk, L3S Research Center, Leibniz Universität Hannover, Germany; Ricardo Usbeck, Data Science Group, Paderborn University / Fraunhofer IAIS, Germany; Rony Hasan, Fraunhofer IAIS, Germany; two anonymous reviewers

Abstract. Question Answering (QA) over Knowledge Graphs (KG) aims to develop a system that is capable of answering users' questions using the information coming from one or multiple Knowledge Graphs, like DBpedia, Wikidata, and so on. Question Answering systems need to translate the user's question, written using natural language, into a query formulated through a specific data query language that is compliant with the underlying KG. This translation process is already non-trivial when trying to answer simple questions that involve a single triple pattern. It becomes even more troublesome when trying to cope with questions that require modifiers in the final query, i.e., aggregate functions, query forms, and so on. The attention over this last aspect is growing but has never been thoroughly addressed by the existing literature. Starting from the latest advances in this field, we want to further step in this direction. This work aims to provide a publicly available dataset designed for evaluating the performance of a QA system in translating articulated questions into a specific data query language. This dataset has also been used to evaluate three QA systems available at the state of the art.

Keywords: Question answering, knowledge graphs, dataset, benchmark, SPARQL modifiers

1. Introduction

Question Answering (QA) is a challenging task that allows users to acquire information from a data source through questions written using natural language. The user remains unaware of the underlying structure of the data source, which could be anything ranging from a collection of textual documents to a database. For this reason, QA systems have always represented a goal for researchers as well as industries since it allows the creation of a Natural Language Interface (NLI) [16,18]. NLIs can make accessible, especially to non-expert users, a huge amount of data that would have been undisclosed otherwise.

Several criteria can be used to categorize QA systems like the application domain (closed or open) or the kind of question that the system can handle (factoid, causal, and so on) [17]. Nevertheless, one of the most distinguishing aspects is the structure of the data source used to retrieve the answers. This feature deeply affects the QA system structure and the methods applied to retrieve the correct answer.

*Corresponding author. E-mail: lucia.siciliani@uniba.it.

The first QA systems were built as an attempt to create NLIs for databases [1] thus QA over structured data has been investigated since the late sixties. However, this research area remains a challenge, and the recent advances in the Semantic Web have brought more attention to this topic. QA system can exploit the information contained within Knowledge Graphs (e.g., DBpedia [2], or Wikidata [29]), to answer questions regarding several topics and also can ease the access to this great amount of data to non-expert users.

RDF[1] and SPARQL[2] represent the two standards chosen by the W3C for data description language and data query language for information resources of the Web. Expressing an information need using SPARQL to retrieve specific information is undoubtedly a nontrivial task, especially for users that do not know this kind of language. For this reason, QA systems have to fulfill the role of interpreters, which take charge of the translation of the users' information needs into SPARQL queries.

However, translating the natural language, which is inherently ambiguous, into a formal query using a data query language like SPARQL is not easy. The main problem that makes this process particularly difficult is recognized in the literature as *lexical gap*. The lexical gap indicates the syntactic distance between the natural language question and the correspondent SPARQL query. In order to bridge it, it is necessary to perform several Natural Language Processing tasks.

The first one is Entity Linking, i.e., find a match between portions of the question to entities within the KG. This process can be straightforward if there is a string matching between the question and the label of a KG resource: in the question *"Which films were produced by Steven Spielberg?"*, *Steven Spielberg* can easily be mapped to its resource, e.g. `wiki:`[3]`Q8877` in Wikidata or `dbr:`[4]`Steven_Spielberg` in DBpedia through the label of the resources. However, many factors can easily make this task nontrivial. For example, there could be a periphrasis like in the question: *"When was the President of the USA born?"*.

Another important issue is related to the ambiguity of the natural language. In the question *"Who directed Philadelphia?"*, the word *Philadelphia* could refer to both the film or the city. In such cases, it is necessary to exploit the context provided by the question to select the right resource.

Similar issues can be found when dealing with Relation Linking, which requires a map between segments of the question and relations in the KG. Here the lexical gap can be even wider since there can be many expressions that refer to a particular relation. Questions like *"Who is the wife of Barack Obama?"*, *"Who is the spouse of Barack Obama?"*, *"Who married Barack Obama?"* always refer to the same relation, which is `dbo:`[5]`spouse` in DBpedia and `wiki:Property:P26` in Wikidata.

These two issues can be combined in several ways. An interesting example is represented by requests like *"Give me all Dutch parties"*. Here, to find the answer, the phrase *Dutch parties* requires a combination of Entity Linking and Relation Extraction, and the relation is implicit. Moreover, not one but two relations need to be concatenated to obtain the final query since we are looking for something that is a political party (first relation), and it has to be based in the Netherlands (second relation).

These two problems, intertwined, represent a struggle for any QA system, and several techniques have been applied in the literature to overcome them. However, there is another kind of lexical gap that has to be considered. For example, to find an answer for questions like *"Which presidents were born in 1945?"*, mapping phrases of the question to the right resources in the KG is not sufficient: the question must be translated into a query that involves the proper SPARQL modifier. By SPARQL modifiers, we refer to all the constructs that alter the basic SPARQL structure made up of a `SELECT` clause followed by a `WHERE` clause that encloses a graph pattern composed of one or more triple patterns. In the previous example, the question *"Which presidents were born in 1945?"* has to be translated into a query containing the `FILTER` modifier like shown in Listings 1.

The SPARQL query language makes available several modifiers: query forms (like `ASK`), solution sequence modifiers (e.g. `ORDER BY`, `LIMIT`, `OFFSET`), functions (e.g. `FILTER`, `COUNT`, `SUM`, `AVG`, `NOW`, `YEAR`, `MONTH`).

[1] https://www.w3.org/TR/rdf11-concepts/
[2] https://www.w3.org/TR/rdf-sparql-query/
[3] https://www.wikidata.org/wiki/
[4] http://dbpedia.org/resource/
[5] http://dbpedia.org/ontology/

```
SELECT DISTINCT ?uri WHERE {
?uri a dbo:President.
?uri dbo:birthDate ?date.
FILTER regex(?date, ''^1945'') }
```

Listing 1. SPARQL query for *"Which presidents were born in 1945?"*

These modifiers can be used alone for simpler queries, but more frequently, they have to be combined to obtain a SPARQL query that is correct and allows to retrieve all the right answers.

The construction of queries containing modifiers still represents an issue that has been explicitly explored in literature only by few works [6,9,12,13,19,20], due to the complexity of the task and the lack of resources that can help researchers to develop and evaluate novel solutions. With this paper, we want to step forward in this direction and propose MQALD: a dataset created to allow KGQA systems to take on this challenge and overcome the issues related to this specific research area.

The paper is organized as follows: in Section 2 we describe the state-of-the-art datasets and evaluation metrics used in this research area; in Section 3 we introduce our MQALD dataset, explaining how it was created and examining the modifiers it contains; in Section 4 we introduce the QA systems chosen for the evaluation and describe how it was performed; in Section 5 we analyze the results obtained by the chosen systems, and finally, in Section 6, we draw the conclusion and propose some insights for future research.

2. Related work

The development of a unified dataset for evaluating KGQA systems started in 2011 with the first edition of the Question Answering over Linked Data (QALD)[6] challenge. Within each edition of the challenge, the organizers always proposed different tasks, the most important being QA over DBpedia. The first dataset proposed for this task was composed of 100 questions, equally split into a train and test set. After each edition, this set of questions was modified and further expanded, leading to 408 questions in the training set and 150 questions in the test set for the last edition of the challenge. QALD datasets contain questions that are manually compiled and curated to include different grades of complexity. Every dataset is composed of a set of English questions that (since QALD-3) is also translated into several languages, like German, Spanish, Italian, and French. Each question is associated with its answer (which is a set composed of one or more resources or literal values) and, most importantly, also the SPARQL query that translates the input question.

Besides the QALD dataset, other datasets have been proposed to benchmark QA systems over KGs.

WebQuestions and SimpleQuestion were both built for Freebase [4]. WebQuestions [3] contains 5,810 questions obtained by exploiting the Google Suggest API and the Amazon Mechanical Turk. Due to this generation process, questions within this dataset usually follow similar templates, and each of them revolves around a single entity of the KG. All questions are associated with an entity of Freebase that can be found in the question and the right answer.

SimpleQuestions [5] instead is the dataset with the highest number of questions, and it was created to provide a richer dataset. It consists of 108,442 questions that have been created in two steps: in the first one, a list of facts or triples were extracted directly from Freebase, then these triples were sent to human annotators who were in charge of translating such triples into natural language questions. Like in the QALD dataset, each question is associated with its SPARQL version. The dataset derives its name from the fact that all the questions can be answered by exploiting just one relation of the KG.

LC-QuAD [22] is based on Wikidata using templates that allow formulating questions that are translated into SPARQL queries that require 2-hop relations. These templates are then reviewed by human annotators that verify the correctness of the questions. The final dataset comprises 5,000 questions, with the corresponding SPARQL

[6] http://qald.aksw.org/

translations and the templates used for generating them. With LC-QuAD 2.0 [11] the dataset has been further extended and now contains 30,000 questions.

TempQuestion [15] is a dataset based on Freebase and obtained by collecting questions containing temporal expressions from already existing source datasets. ComplexWebQuestions [21] is also based on Freebase and requires reasoning over multiple web snippets. Finally Event-QA [7] is a novel dataset based on a custom KG called EventKG.[7]

As aforementioned, QALD represents a dataset and a challenge that makes available a benchmark for QA systems. Factoid QA inherited its evaluation metrics from classical evaluation systems, thus estimating the performance in terms of Precision, Recall, and F-measure. Since the seventh edition of the challenge, the GERBIL benchmark platform [26,27] was used to evaluate the performance of the participating systems. GERBIL, which is publicly available online,[8] embeds several QA systems and allows to evaluate different QA datasets, including those distributed during all the QALD challenges.

3. MQALD: Dataset creation

In this section, we introduce MQALD, a dataset that is composed only of questions containing modifiers. Two sections compose MQALD: the first one contains novel questions, and the second one is obtained by extracting questions that require modifiers from the QALD dataset. In the following, we will describe with more detail each portion of MQALD.

3.1. Generating novel questions

MQALD comprises 100 questions that have been manually created by human annotators based on DBpedia 2016-10. To create MQALD, we employed two annotators who are both familiar with the use of SPARQL and SQL-like Data Query Languages. Since the dataset aims to evaluate QA systems' capability to translate complex information needs into query with modifiers, we did not apply any constraint to the question's creation process. Annotators are free to choose question topics and modifiers. In this way, the results obtained by a system are more likely to depend on its ability to create questions with modifiers.

Each annotator was in charge of developing 50 questions and then review the other 50 questions created by the other annotator.

Each question is provided in four languages: English, Italian, French, and Spanish. Both annotators have Italian as their mother tongue and have a good knowledge of the English language. Each question was first written in English and then translated to Italian. For the other two languages, translations of the original English question were obtained through machine translation.[9]

The two annotators discussed any further discrepancy until they met an agreement.[10] Questions were built paying attention so that the modifiers occurring in each SPARQL query were:

- mandatory: the use of the modifier is indispensable to retrieve the right answer;
- KG independent: modifiers are not used to adapt the query to a particular structure of the underlying KG. An example of KG dependency is represented by the use of the `LIMIT` modifier to retrieve a specific resource or literal from the result set. An example of this behavior is given in Section 3.3.3.

Considering these two aspects is particularly important since, in this way, it is possible to guarantee that each modifier is connected to a specific semantics of the natural language question. Table 1 shows the distribution of modifiers among the 100 questions made by the annotators.

[7] http://eventkg.l3s.uni-hannover.de/

[8] http://gerbil-qa.aksw.org/gerbil/

[9] We use Google Translate.

[10] During the annotation process, the first annotator edited the 12% of questions of the second annotator, while the 18% of questions of the first annotator were rectified.

Table 1
Frequencies of each modifier within the novel questions available in MQALD

Modifier	#occurrences
LIMIT	25
ORDER BY	31
FILTER	41
ASK	20
UNION	10
OFFSET	6
COUNT	26
GROUP BY	15
HAVING	10
YEAR	3
NOW	0


```
{"id": 187,
"aggregation": true,
"hybrid": false,
"question": [{
"string": "Which is the fifth most populous country in the world?",
"keywords": "fifth, most populous, country",
"language": "en"},
...],
"answertype": "resource",
"onlydbo": true,
"query": {
"sparql": "PREFIX dbo: <http://dbpedia.org/ontology/> PREFIX rdf: <http://
   www.w3.org/1999/02/22-rdf-syntax-ns#> SELECT ?country WHERE { ?country
   rdf:type dbo:Country; dbo:populationTotal ?population } ORDER BY DESC(
   xsd:integer(?population)) LIMIT 1 OFFSET 4"
},
"answers": [
{ "head": {"vars": ["country"] },
"results": {
"bindings": [
{ "country": { "type": "uri", "value": "http://dbpedia.org/resource/China"
   }}
]}}],
"modifiers": ["ORDER BY", "LIMIT", "OFFSET"]}
```

Listing 2. JSON structure of a question within the dataset

The MQALD dataset is in JSON format and compliant with the QALD dataset structure: each question is serialized as a JSON object which is then stored in a JSON array that contains all the questions as shown in Listings 2. The only exceptions in the structure of each question is represented by the presence of a further JSON array, named `modifiers`, containing all the modifiers occurring in the SPARQL query (this array is empty if there are no modifiers).

Table 2
Number of questions included in each file

Dataset	#questions
QALD_train	462
QALD_train_MODS	154
QALD_train_NO_MODS	308
QALD_test	115
QALD_test_MODS	41
QALD_test_NO_MODS	74

3.2. Extracting queries with modifiers from QALD

To further extend our dataset, we also collect questions with modifiers available in the QALD dataset. As stated in Section 2, in each edition of QALD, the dataset is obtained as an extension of the previous edition dataset, although we noticed there was not a complete overlap. For this reason, we decided to merge the questions contained in the last three editions of the QALD challenge over DBpedia: QALD-7 [25], QALD-8 [24], and QALD-9 [23]. We decide not to include queries with modifiers from other datasets since they are not based on DBpedia or are based on a no-standard version of DBpedia.[11]

Considering that each QALD dataset is split in training and test, the merge of the three editions was done separately. Therefore two lists of questions were obtained: one containing all the training questions from QALD-9, 8, and 7, and the other one containing all the test questions from QALD-9, 8, and 7. Since training data could have been used to develop the systems, we excluded from the test set those questions that also appeared within the training data in any of the three editions. Thus, those questions were removed from the test questions of QALD-9, 8, and 7 and added to the list composed by the training set questions. We obtained the following files:

- **QALD_train**: contains the merge of all questions coming from the training sets of QALD-7, QALD-8, QALD-9;
- **QALD_train_MODS**: contains only the questions that require a modifier extracted from QALD_train;
- **QALD_train_NO_MODS**: contains only the questions that do not require a modifier extracted from QALD_train;
- **QALD_test**: contains the merge of all questions coming from the test sets of QALD-7, QALD-8, QALD-9;
- **QALD_test_MODS**: contains only the questions that require a modifier extracted from QALD_test;
- **QALD_test_NO_MODS**: contains only the questions that do not require a modifier extracted from QALD_test;

Table 2 shows the number of questions included in each file. The format of the JSON file remains the same as the one shown in Listings 2 with the addition of the attribute `qald-version` which indicates the QALD dataset of origin. Table 3 shows the distribution of modifiers within the training and test sets obtained by merging QALD-9, 8, and 7.

3.3. Modifiers within the dataset

In this section, we will describe in detail the modifiers that are contained in the dataset through real examples extracted from MQALD. The modifiers will be listed according to their frequency.

3.3.1. FILTER

The `FILTER` keyword is used to restrict the number of results that match the graph pattern specified just before the `FILTER` itself. The `FILTER` keyword precedes an expression that can be constituted by a wide variety of operators categorized according to the number of parameters involved.

[11] This is the case of LC-QuAD that uses a version of DBpedia, called DBpedia 2018, that merges DBpedia with information coming from Wikidata.

Table 3
Frequencies of each modifier obtained after merging the last three editions of the QALD dataset

Modifier	QALD train	QALD test
LIMIT	48	11
ORDER BY	45	9
FILTER	34	16
ASK	40	3
UNION	33	9
OFFSET	33	5
COUNT	29	7
GROUP BY	3	2
HAVING	3	1
YEAR	2	2
NOW	0	2

```
SELECT DISTINCT ?uri WHERE {
?uri a dbo:Cave;
dbp:entranceCount ?entrance
FILTER (?entrance>3)}
```

Listing 3. SPARQL query for *"Which caves have more than 3 entrances?"*

There are *unary operators* like "`!`" which represents the logical NOT and "`BOUND`" that checks if its argument is bounded to a value. These operators can be used alone or in conjunction to answer questions like *"Is Frank Herbert still alive?"*. In this case, the answer can be found only checking if the object of the relation `res:Frank_Herbert dbo:deathDate ?date` does not exist in the KG (i.e. `FILTER(!BOUND(?date))`).

We also have *binary operators* that comprise logical connectives AND "`&&`" and OR "`||`", as well as all the other test operators between two values (like `==`, `!=`, `>`, `<`, and so on). These are used to answer questions like *"Which caves have more than 3 entrances?"* creating a SPARQL query as shown by Listing 3.

Finally, there is also a single *ternary operator* which is `REGEX`, that checks if a certain string (first parameter) matches a regular expression (second parameter). The third parameter is an optional flag, which specifies a case-insensitive pattern match). This ternary operator can be used in questions like *"Is the wife of president Obama called Michelle?"* to check if the name *"Michelle"* is contained within the label of the resource `dbo:Michelle_Obama`.

3.3.2. ORDER BY

The second most common modifier is `ORDER BY`, which is used to alter the order of the solution sequence. It is important to notice that the dataset, the `ORDER BY` modifier is always used in conjunction with other modifiers except one question within the dataset. An example where `ORDER BY` appears by itself is *"What are the top selling luxury vehicle brands in Germany?"*, where the answer is represented by all the results of an ordered list.

3.3.3. LIMIT

The `LIMIT` modifier is usually used in combination with `ORDER BY` and `OFFSET` which allows answering questions containing superlatives, i.e. *"Who is the tallest player of the Atlanta Falcons?"* or *"For which label did Elvis record his first album?"*.

Another combination is composed of the previous one (`LIMIT`, `ORDER BY`, and `OFFSET`) with the addition of the `COUNT` aggregation function for questions like *"Which country has the most official languages?"* where the `ORDER BY` needs to be applied to the object of the `COUNT` function as illustrated by Listing 4.

The `LIMIT` modifier is also subject of controversial use within the QALD dataset. An example is represented by the question *"With how many countries Iran has borders?"* that translates into a SPARQL query with the `LIMIT`

```
SELECT DISTINCT ?uri WHERE {
?uri a dbo:Country;
dbp:officialLanguages ?language.}
ORDER BY DESC(COUNT(?language))
OFFSET 0 LIMIT 1
```

Listing 4. SPARQL query for *"Which country has the most official languages?"*

```
SELECT ?capital WHERE {
?uri dbo:capital ?capital;
dbo:populationDensity ?density}
ORDER BY DESC(?density)
LIMIT 2 OFFSET 2
```

Listing 5. SPARQL query for *"Which are the second and third capitals in the world with higher population density?"*

```
ASK WHERE {
res:Aristotle
dbo:influencedBy res:Socrates }
```

Listing 6. SPARQL query for *"Did Socrates influence Aristotle?"*

modifier set to 8. This is correct since by executing the query against DBpedia without this modifier, only the first 8 results represent the right answer (i.e. the resources corresponding to the countries Iran has borders with), while the following are just literals not relevant to the question. However, in this case, the usage of the `LIMIT` modifier is deeply connected to the structure of DBpedia.

For the novel questions introduced in MQALD we employed the *LIMIT* modifier to translate the semantics of superlatives or select elements in a specific position from an ordered list as shown in Listing 5.

3.3.4. ASK

Among all the possible SPARQL query forms besides `SELECT`, the only one present within the dataset is `ASK` which is reasonable since QA systems on KGs are usually factoid QA systems. The `ASK` query form is necessary to construct questions that require a Yes/No answer thus returning a Boolean value, like *"Did Socrates influence Aristotle?"* as shown in Listing 6.

3.3.5. COUNT

Besides from being used in combination with `ORDER BY` and `LIMIT`, the `COUNT` modifier is used alone, for example in questions that start with *"How many"*, like *"How many films did Leonardo DiCaprio star in?"*.

3.3.6. UNION

The `UNION` modifier is used to merge the results of two different patterns. Within the QALD dataset, it is usually employed to take into account more relations for obtaining the full set of answers as shown in Listing 7.

Despite being correct, the use of the `UNION` modifier in this query is associated with how the KG is structured, i.e., it is necessary to know that for retrieving all the companies that are in Munich, we need three relations (`dbo:location`, `dbo:headquarter`, and `dbo:locationCity`).

For this reason, in the questions we introduced in MQALD, through the help of human annotators, we employed the `UNION` modifier as a translator of the semantics of the logic OR like shown in Listings 8.

```
SELECT DISTINCT ?uri WHERE {
?uri a dbo:Company
{?uri dbo:location dbr:Munich}
UNION
{?uri dbo:headquarter dbr:Munich}
UNION
{?uri dbo:locationCity dbr:Munich}}
```

Listing 7. SPARQL query for *"Give me all companies in Munich"*

```
SELECT ?uri WHERE {
?uri rdf:type dbo:Band.
?uri dbo:genre dbr:Electronic_music.
{{?uri dbo:hometown dbr:France}
UNION
{?uri dbo:hometown dbr:Italy}}}
```

Listing 8. SPARQL query for *"Give me all the electronic music bands having hometown in France or in Italy"*

```
SELECT DISTINCT ?uri WHERE {
?uri a dbo:Mountain;
dbo:elevation ?elevation}
ORDER BY DESC(?elevation)
OFFSET 1 LIMIT 1
```

Listing 9. SPARQL query for *"What is the second highest mountain on Earth?"*

3.3.7. OFFSET

The `OFFSET` modifier is used to shift the start of the retrieved solutions after the specified number.

In the questions extracted from QALD, the majority of times, the queries include an `OFFSET` of 0, which has no effect on the results and therefore could also be neglected. However, there are some questions where there is an `OFFSET` of a different number that allows answering correctly to the input question but appears controversial. Let us consider the question *"How many foreigners speak German?"*. This question is translated into a SPARQL query where `OFFSET 1` is applied over the object of the relation `dbr:German_language dbp:speakers ?Ger_lang`. This relation has as object three literal values, namely: *"L2 speakers: 10–15 million"*, *"as a foreign language: 75–100 million"*, and *"million to 95 million"*, so actually the use of `OFFSET 1` allows to pick up the second literal which represents the right answer. Nevertheless, obtaining this SPARQL query without knowing how the KG is structured is infeasible, especially considering that in this case (but this frequently happens in DBpedia), the property `dbp:speakers` does not have any specification concerning its domain and range thus, the object of this property is not consistent (there can be strings, numbers or even resources). A similar scenario occurs for the question *"Where is the most deep point in the ocean?"* where a `OFFSET 13` is required to retrieve the answer.

The other existing case of non-zero `OFFSET` within the QALD questions ensues from the question *"What is the second highest mountain on Earth?"*, where the semantics of the question justifies the use of `OFFSET 1` as shown in Listing 9.

In our dataset, we always employ the `OFFSET` modifier with the aforementioned semantics. `OFFSET` is thus used in conjuction with `ORDER BY` and `LIMIT` to obtain a particular element of an ordered list.

```
SELECT DISTINCT ?uri WHERE {
?uri a dbo:Country. ?cave a dbo:Cave
{ ?cave dbo:location ?uri } UNION
{ ?cave dbo:location ?loc .
?loc dbo:country ?uri } }
GROUP BY ?uri
HAVING ( COUNT(?cave) > 10 )
```

Listing 10. SPARQL query for *"Which countries have more than ten caves?"*

```
SELECT DISTINCT ?uri WHERE {
?uri rdf:type dbo:Person;
dct:subject
dbc:Presidents_of_the_United_States;
dbo:activeYearsEndDate ?d
FILTER ((YEAR(NOW())-YEAR(?d))<=20)}
```

Listing 11. SPARQL query for *"Give me all American presidents of the last 20 years"*

3.3.8. GROUP BY – HAVING

The `GROUP BY` modifier is used to divide the result set into groups, which can then be used as input to an aggregation function like `COUNT`.

If the query requires to filter the result of aggregation, then it must be introduced with the modifier `HAVING` (instead of the keyword `FILTER`), as it happens for the question *"Which countries have more than ten caves?"* in Listing 10.

3.3.9. YEAR – NOW

SPARQL makes available several functions for dates and times, even though `YEAR` and `NOW` are the only two occurring in the dataset. The `YEAR` function returns the year part of the date given as an argument, while the `NOW` function does not require an argument and returns the current time (formatted according to the XSD dateTime format). These functions can be used to answer very peculiar questions, like *"Give me all American presidents of the last 20 years."*.

3.4. Dataset availability and usage statistics

The dataset can be downloaded from Zenodo[12] and was published under the MIT license. A description of the dataset is provided within the repository. The code used for creating the dataset and the evaluation is freely available on GitHub.[13] We published the first version of the dataset in May 2020. Since then, the dataset has collected 70 unique views and 26 unique downloads.[14]

4. Evaluation

We compare the performance of three state of the art QA systems for KGs over the different datasets we constructed. Our evaluation aims to examine how QA systems perform on questions that require one or more modifiers

[12] https://zenodo.org/record/4657496
[13] https://github.com/lsiciliani/MQALD
[14] Statistics computed on 1st April 2021.

within the final SPARQL query and how this result impacts the performance of the system itself. The QA systems chosen for the evaluation are the following:

- **gAnswer**[15]: the system proposed by Hu et al. [14] that won the QALD-9 challenge. The method used by this system consists of transforming the input question into a Semantic Query Graph, a particular graph where each node corresponds to a resource and each vertex to a relation of the underlying KG. Resources are found using DBpedia Lookup, while relations are extracted using a Multi-Channel Convolutional Neural Network. Next, the Semantic Query Graph is translated into a SPARQL question that is then used to obtain the final answer;
- **QAnswer**[16]: a QA system developed by Diefenbach et al. [8]. It represents the extension of the WDAqua-core0 system [10] that took part in the QALD-9 challenge gaining the second place. The approach behind QAnswer is based upon four steps: the first one consists of retrieving all the resources that can be linked to a question by considering its n-grams, the second phase uses these resources to create all the possible queries according to specific patterns in the third step these queries are then ranked according to several features and finally the query that obtained the highest score is executed, and the answer is returned to the user;
- **TeBaQA**[17]: created by Vollmers et al. [28], exploits a template-based approach. The SPARQL templates are generated upon the questions available for the QALD challenge and a classifier is trained to assign each question to a specific template. Given a question, the classifier assigns to it a template which is later filled to build the final SPARQL query. TeBaQA ranked third during the QALD-9.

We decided to implement an evaluation tool following the guidelines reported in the QALD-9 report [23]. In particular, the evaluation script computes for each query q a set of metrics, such as precision (P), recall (R), and F-measure according to the following equations:

$$P(q) = \frac{\#\textit{correct system answers for } q}{\#\textit{system answers for } q}$$

$$R(q) = \frac{\#\textit{correct system answers for } q}{\#\textit{gold standard answers for } q}$$

$$F(q) = \frac{2 \times P(q) \times R(q)}{P(q) + R(q)}$$

Moreover, there are additional information to take into account:

- If the golden answer-set is empty and the system retrieves an empty answer, precision, recall and F-measure are equal to 1.
- If the golden answer-set is empty but the system responds with any answer-set, precision, recall, and F-measure are equal to 0.
- If there is a golden answer but the QA system responds with an empty answer-set, it is assumed that the system could not respond. Then the precision, recall, and F-measure are equal to 0.
- In any other case, the standard precision, recall, and F-measure are computed.

In our paper, we consider the macro measures as the QALD challenge: we calculated precision, recall, and F-measure per question and averaged the values at the end. As adopted in QALD, for the final evaluation, the Macro F1 QALD metric is used. This metric uses the previously mentioned additional information with the following exception:

- If the golden answer-set is not empty but the QA system responds with an empty answer-set, it is assumed that the system determined that it cannot answer the question. Then the precision is set to 1 and the recall and F-measure to 0.

[15] http://ganswer.gstore-pku.com/api/qald.jsp
[16] http://qanswer-core1.univ-st-etienne.fr/api/gerbil
[17] http://139.18.2.39:8187/

Table 4
Results of the three systems over the ten datasets. The metrics used are Precision (P), Recall (R), F-Measure (F), and F1-QALD measure (F1-Q)

	gAnswer				QAnswer				TeBaQA			
	P	R	F	F1-Q	P	R	F	F1-Q	P	R	F	F1-Q
qald-9_test	.607	.316	.296	**.416**	.459	.222	.197	.299	.644	.141	.139	.231
mqald	.497	.037	.031	.069	.550	.083	.056	**.145**	.531	.039	.046	.072
mqald (only new queries)	.472	.032	.032	.060	.320	.093	.071	**.144**	.483	.036	.045	.068
mqald+qald_test_NO_MODS	.526	.173	.162	**.260**	.394	.161	.137	.228	.562	.104	.109	.176
qald_train_all	.577	.334	.313	**.423**	.529	.334	.302	.409	.602	.154	.144	.246
qald_train_MODS	.350	.054	.027	.094	.356	.223	.175	**.274**	.479	.051	.047	.092
qald_train_NO_MODS	.474	.466	.456	**.470**	.612	.392	.369	.478	.657	.199	.186	.306
qald_test_all	.590	.295	.284	**.393**	.465	.232	.203	.309	.634	.155	.156	.249
qald_test_MODS	.608	.049	.054	.091	.391	.139	.087	**.205**	.671	.045	.048	.085
qald_test_NO_MODS	.580	.431	.412	**.495**	.515	.269	.260	.353	.606	.215	.215	.317

5. Results

The results of the evaluation of the three mentioned QA systems, i.e. gAnswer, QAnswer, and TeBaQA are reported in Table 4.

Overall, an analysis of the performance of the systems in terms of F1-QALD measure for the datasets without modifiers (no_mods) and those containing all the questions (with and without modifiers) confirms the results reported by the QALD-9 challenge: gAnswer represents the best system at the state-of-the-art for this task, followed by QAnswer and TeBaQA.

However, this ranking changes when comparing the results containing only questions that require modifiers. In fact, in this case, QAnswer outperforms the results obtained by gAnswer. To obtain more insights regarding this aspect, we checked which questions that require modifiers are answered by each system within the test set, and the results are shown in Table 5 for questions contained in the QALD testset and Table 6 for the novel questions added in MQALD. By analyzing Precision, Recall, and F-measure calculated for each question and the set of answers returned by each system, we can have an insight into how they handle modifiers.

Out of the 41 questions that require modifiers extracted from the QALD datasets, the number of questions where the F-score is not equal to zero is 3 for gAnswer, 7 for QAnswer, and 3 for TeBaQA. Most of the questions answered by the three systems involve the `UNION` modifier. As stated in Section 3.3, within the QALD datasets, the `UNION` modifier is exclusively used to merge the results of different triple patterns rather than encapsulating a particular semantics and sometimes are even unnecessary due to the constant evolution of DBpedia. This is the example of the question *"Which software has been published by Mean Hamster Software?"* where the `UNION` modifier is not needed at all since the query `SELECT DISTINCT ?uri WHERE {?uri onto:publisher res:Mean_Hamster_Software}` is sufficient to retrieve all the answers. This could be due to a mistake or a lack of update of the dataset and allows gAnswer and QAnswer to obtain the right answer without using the modifier. Another case is represented instead by questions like *"Which countries are connected by the Rhine?"* where the `UNION` modifier is well used since it is necessary to retrieve the whole set of answers provided by the QALD. In this case, gAnswer and TeBaQA manage to return an answer but only the precision is equal to 1, meaning that not all the answers have been retrieved and `UNION` was not used.

The same happens for all the questions where the F-score is lower than 1. For example, QAnswer answers the question *"What is the longest river in China?"* by returning all the rivers that flow through China and, of course, the longest river is also included in this set. The use of the three modifiers (`LIMIT`, `OFFSET`, and `ORDER BY`) would have allowed to properly order the list by the rivers' length and pick the first element of the ordered list. With this result, we can deduce that modifiers were not used to retrieve the answer.

Also, for the question *"How many grand-children did Jacques Cousteau have?"* we can observe a misleading behavior since the number of grand-children of Jacques Cousteau actually coincides with his children's number. Thus QAnswer could answer correctly by just considering the property `dbp:children`. It is important to notice that

Table 5
List of the questions answered by each system over the qald_test_MODS dataset

gAnswer					
id	Question	Modifiers	P	R	F
7	Which software has been published by Mean Hamster Software?	UNION	1	1	1
19	Give me all cars that are produced in Germany.	UNION	.920	.191	.316
24	Which countries are connected by the Rhine?	UNION	1	.833	.909

QAnswer					
id	Question	Modifiers	P	R	F
1	What is the highest mountain in Germany?	LIMIT, ORDER BY	.001	1	.002
5	Which airports are located in California, USA?	UNION	1	.351	.520
7	Which software has been published by Mean Hamster Software?	UNION	1	1	1
8	How many grand-children did Jacques Cousteau have?	COUNT	1	1	1
12	What is the longest river in China?	LIMIT, OFFSET, ORDER BY	.016	1	.032
15	Is Pamela Anderson a vegan?	ASK	1	1	1
31	Which politicians were married to a German?	UNION	.063	.091	.074

TeBaQA					
id	Question	Modifiers	P	R	F
6	Which countries in the European Union adopted the Euro?	UNION	.5	.022	.043
14	How many awards has Bertrand Russell?	COUNT	1	1	1
24	Which countries are connected by the Rhine?	UNION	1	.833	.909

the property `dbp:children` does not exist in the current version of DBpedia: we have the property `dbo:child` which has as object the resources of two of the four children of Jacques Cousteau. This implies that also this question should be updated properly to be compliant with the KG. Nevertheless, the property `dbp:children` is available in the 2016-10 version of DBpedia, and the object of this property is 4, which can be obtained without modifiers and erroneously used as the correct answer.

There is only one question that requires an `ASK` among all the others: *"Is Pamela Anderson a vegan?"*. Only QAnswer returns the right answer but using a SPARQL query that is completely unrelated to the question that just checks the existence of the resource `dbr:Pamela_Anderson`.

An exception is represented by the question *"How many awards has Bertrand Russell?"* which requires the `COUNT` modifier since this information is not available in any other way from DBpedia.

Regarding the novel questions included in MQALD (new queries), 4 answers were provided by gAnswer, 11 by QAnswer, and 8 by TeBaQA. The only two systems that manage to return a full answer, with Precision, Recall, and F-measure at 1, are QAnswer and TeBaQA. As stated in Section 3, MQALD questions are formulated such that modifiers are necessary to retrieve the correct set of answers; thus, by analyzing these scores, it is possible to identify which modifiers are covered by which system easily. QAnswer is able to manage the `ASK` and `COUNT` modifiers while TeBAQA is capable to handle the `COUNT` modifiers as well. Of course, if a system can handle a specific modifier, it does not mean that it will answer all the questions containing that modifier since the lexical gap still represents a major problem in the translation from natural language to SPARQL.

Table 6
List of the questions answered by each system over the new queries added in MQALD

gAnswer					
id	Question	Modifiers	P	R	F
164	Give me all Stephen King books for which the publication date is known.	FILTER	.488	.976	.651
189	Give me the list of who directed and produced Saving Private Ryan.	UNION	.600	.750	.667
226	What are the first publication date and the last publication date of Mazinger Z?	UNION	1	.500	.667
243	Which Elvis songs are part of the Elvis is Back album?	FILTER, ORDER BY	.106	1	.192

QAnswer					
id	Question	Modifiers	P	R	F
162	What is the lowest mountain in Germany?	LIMIT, ORDER BY	.001	1	.002
169	Is Leonardo Da Vinci the author of the Mona Lisa?	ASK	1	1	1
170	What are the first 10 works in alphabetical order made by Leonardo da Vinci?	LIMIT, ORDER BY	.286	1	.444
182	Which films of the Jurassic Park saga has Steven Spielberg directed?	FILTER	.667	1	.800
183	How many movies has George Lucas directed?	COUNT	1	1	1
189	Give me the list of who directed and produced Saving Private Ryan.	UNION	1	.250	.400
194	Is Barack Obama a politician?	ASK	1	1	1
220	Which films in which Catherine Zeta-Jones starred in have earned less than the initial budget?	FILTER, ORDER BY	.077	1	.143
226	What are the first publication date and the last publication date of Mazinger Z?	UNION	1	.5	.667
231	Who are the author and publisher of Don Quixote?	UNION	1	.5	.667
238	How many cartoons are produced by Walt Disney?	COUNT	1	1	1

TeBaQA					
id	Question	Modifiers	P	R	F
170	What are the first 10 works in alphabetical order made by Leonardo da Vinci?	LIMIT, ORDER BY	.250	.1	.143
174	Give me all the electronic music bands having hometown in France or in Italy.	UNION	.020	.043	.027
176	What are the birthplace and the death place of Elvis Presley?	UNION	1	.500	.667
186	What are the birth names of Tom Cruise and Whoopi Goldberg?	UNION	1	.500	.667
190	What are the capitals of Italy and France?	UNION	1	.500	.667
201	How many children does Barack Obama have?	COUNT	1	1	1
231	Who are the author and publisher of Don Quixote?	UNION	1	.500	.667
239	Give me the population density of Warsaw as well as its population total.	UNION	1	.500	.667

Overall, the results show how all the systems under analysis do not perform well at handling modifiers, just with a few exceptions. Nevertheless, the capability of QAnswer to handle ASK and COUNT modifiers and better bridge the lexical gap by at least returning partial answers granted it the best score among the three selected systems.

6. Conclusions and future works

In this paper, we have introduced MQALD, a dataset of questions that contain SPARQL modifiers. The dataset is composed of 100 new manually created queries plus 41 queries with modifiers of the last three editions of the QALD. These modifiers were then analyzed to better understand their functioning considering the SPARQL syntax and how they can reflect specific semantics of the natural language query. Through the evaluation of three systems available at the state-of-the-art, emerged that there is still much work that must be done to create a QA system capable of handling this kind of questions, not only from an architectural point of view but also in creating and updating the existing resources to allow these systems to be fairly compared.

This work aimed to give more insight about this specific issue related to Question Answering over Knowledge Graphs. This research area is fastly growing, and the availability of novel resources can significantly help develop novel solutions.

As future work, we plan to further encourage the research and development of new solutions to improve QA systems' performance over KG handling questions involving modifiers, which are very common in natural language. The first step towards this direction would be to revise the available resources properly, fix the datasets so that question/query pairs are compliant with the last version of DBpedia, and make sure that using a specific modifier is the only way to retrieve the right answers. Next, it would be beneficial for the whole community to expand this dataset further so that a sufficient number of questions properly represent all the modifiers available in SPARQL.

Acknowledgement

This work was supported by the PRIN 2009 project: "Modelli per la personalizzazione di percorsi formativi in un sistema di gestione dell'apprendimento".

References

[1] I. Androutsopoulos, G.D. Ritchie and P. Thanisch, Natural language interfaces to databases – an introduction, *Nat. Lang. Eng.* **1**(1) (1995), 29–81. doi:10.1017/S135132490000005X.

[2] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak and Z.G. Ives, DBpedia: A nucleus for a web of open data, in: *The Semantic Web, 6th International Semantic Web Conference, 2nd Asian Semantic Web Conference, ISWC 2007 + ASWC 2007*, Busan, Korea, November 11–15, 2007, K. Aberer, K. Choi, N.F. Noy, D. Allemang, K. Lee, L.J.B. Nixon, J. Golbeck, P. Mika, D. Maynard, R. Mizoguchi, G. Schreiber and P. Cudré-Mauroux, eds, Lecture Notes in Computer Science, Vol. 4825, Springer, 2007, pp. 722–735. doi:10.1007/978-3-540-76298-0_52.

[3] J. Berant, A. Chou, R. Frostig and P. Liang, 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, ACL, 2013, pp. 1533–1544, https://www.aclweb.org/anthology/D13-1160/.

[4] K.D. Bollacker, C. Evans, P. Paritosh, T. Sturge and J. Taylor, Freebase: A collaboratively created graph database for structuring human knowledge, in: *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2008*, Vancouver, BC, Canada, June 10–12, 2008, J.T. Wang, ed., ACM, 2008, pp. 1247–1250. doi:10.1145/1376616.1376746.

[5] A. Bordes, N. Usunier, S. Chopra and J. Weston, Large-scale simple question answering with memory networks, 2015, CoRR, abs/1506.02075.

[6] N. Chakraborty, D. Lukovnikov, G. Maheshwari, P. Trivedi, J. Lehmann and A. Fischer, Introduction to neural network based approaches for question answering over knowledge graphs, 2019, CoRR, abs/1907.09361.

[7] T.S. Costa, S. Gottschalk and E. Demidova, Event-QA: A dataset for event-centric question answering over knowledge graphs, in: *CIKM '20: Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, 2020, pp. 3157–3164. doi:10.1145/3340531.3412760.

[8] D. Diefenbach, A. Both, K. Singh and P. Maret, Towards a question answering system over the semantic web, *Semantic Web* **11**(3) (2020), 421–439. doi:10.3233/SW-190343.

[9] D. Diefenbach, V. López, K.D. Singh and P. Maret, Core techniques of question answering systems over knowledge bases: A survey, *Knowl. Inf. Syst.* **55**(3) (2018), 529–569. doi:10.1007/s10115-017-1100-y.

[10] D. Diefenbach, K.D. Singh and P. Maret, WDAqua-core0: A question answering component for the research community, in: *Semantic Web Challenges – 4th SemWebEval Challenge at ESWC 2017, Revised Selected Papers*, Portoroz, Slovenia, May 28–June 1, 2017, M. Dragoni, M. Solanki and E. Blomqvist, eds, Communications in Computer and Information Science, Vol. 769, Springer, 2017, pp. 84–89. doi:10.1007/978-3-319-69146-6_8.

[11] M. Dubey, D. Banerjee, A. Abdelkawi and J. Lehmann, LC-QuAD 2.0: A large dataset for complex question answering over wikidata and DBpedia, in: *The Semantic Web – ISWC 2019 – 18th International Semantic Web Conference, Proceedings, Part II*, Auckland, New Zealand, October 26–30, 2019, C. Ghidini, O. Hartig, M. Maleshkova, V. Svátek, I.F. Cruz, A. Hogan, J. Song, M. Lefrançois and F. Gandon, eds, Lecture Notes in Computer Science, Vol. 11779, Springer, 2019, pp. 69–78. doi:10.1007/978-3-030-30796-7_5.

[12] M. Dubey, S. Dasgupta, A. Sharma, K. Höffner and J. Lehmann, AskNow: A framework for natural language query formalization in SPARQL, in: *The Semantic Web. Latest Advances and New Domains – 13th International Conference, ESWC 2016, Proceedings*, Heraklion, Crete, Greece, May 29–June 2, 2016, H. Sack, E. Blomqvist, M. d'Aquin, C. Ghidini, S.P. Ponzetto and C. Lange, eds, Lecture Notes in Computer Science, Vol. 9678, Springer, 2016, pp. 300–316. doi:10.1007/978-3-319-34129-3_19.

[13] K. Höffner, S. Walter, E. Marx, R. Usbeck, J. Lehmann and A.N. Ngomo, Survey on challenges of question answering in the semantic web, *Semantic Web* **8**(6) (2017), 895–920. doi:10.3233/SW-160247.

[14] S. Hu, L. Zou and X. Zhang, A state-transition framework to answer complex questions over knowledge base, in: *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, Brussels, Belgium, October 31–November 4, 2018, E. Riloff, D. Chiang, J. Hockenmaier and J. Tsujii, eds, Association for Computational Linguistics, 2018, pp. 2098–2108. doi:10.18653/v1/D18-1234.

[15] Z. Jia, A. Abujabal, R.S. Roy, J. Strötgen and G. Weikum, TempQuestions: A benchmark for temporal question answering, in: *Companion of the Web Conference 2018 on the Web Conference 2018, WWW 2018*, Lyon, France, April 23–27, 2018, P. Champin, F. Gandon, M. Lalmas and P.G. Ipeirotis, eds, ACM, 2018, pp. 1057–1062. doi:10.1145/3184558.3191536.

[16] E. Kaufmann and A. Bernstein, How useful are natural language interfaces to the semantic web for casual end-users? in: *The Semantic Web, 6th International Semantic Web Conference, 2nd Asian Semantic Web Conference, ISWC 2007 + ASWC 2007*, Busan, Korea, November 11–15, 2007, K. Aberer, K. Choi, N.F. Noy, D. Allemang, K. Lee, L.J.B. Nixon, J. Golbeck, P. Mika, D. Maynard, R. Mizoguchi, G. Schreiber and P. Cudré-Mauroux, eds, Lecture Notes in Computer Science, Vol. 4825, Springer, 2007, pp. 281–294. doi:10.1007/978-3-540-76298-0_21.

[17] A. Mishra and S.K. Jain, A survey on question answering systems with classification, *J. King Saud Univ. Comput. Inf. Sci.* **28**(3) (2016), 345–361. doi:10.1016/j.jksuci.2014.10.007.

[18] F. Ozcan, A. Quamar, J. Sen, C. Lei and V. Efthymiou, State of the art and open challenges in natural language interfaces to data, in: *Proceedings of the 2020 International Conference on Management of Data, SIGMOD Conference 2020, Online Conference*, Portland, OR, USA, June 14–19, 2020, D. Maier, R. Pottinger, A. Doan, W. Tan, A. Alawini and H.Q. Ngo, eds, ACM, 2020, pp. 2629–2636. doi:10.1145/3318464.3383128.

[19] M. Saleem, S.N. Dastjerdi, R. Usbeck and A.N. Ngomo, Question answering over linked data: What is difficult to answer? What affects the F scores? in: *Joint Proceedings of BLINK2017: 2nd International Workshop on Benchmarking Linked Data and NLIWoD3: Natural Language Interfaces for the Web of Data Co-Located with 16th International Semantic Web Conference (ISWC 2017)*, Vienna, Austria, October 21st–22nd 2017, R. Usbeck, A.N. Ngomo, J. Kim, K. Choi, P. Cimiano, I. Fundulaki and A. Krithara, eds, CEUR Workshop Proceedings, Vol. 1932, CEUR-WS.org, 2017, http://ceur-ws.org/Vol-1932/paper-02.pdf.

[20] L. Siciliani, D. Diefenbach, P. Maret, P. Basile and P. Lops, Handling modifiers in question answering over knowledge graphs, in: *AI*IA 2019 – Advances in Artificial Intelligence – XVIIIth International Conference of the Italian Association for Artificial Intelligence, Proceedings*, Rende, Italy, November 19–22, 2019, M. Alviano, G. Greco and F. Scarcello, eds, Lecture Notes in Computer Science, Vol. 11946, Springer, 2019, pp. 210–222. doi:10.1007/978-3-030-35166-3_15.

[21] A. Talmor and J. Berant, The web as a knowledge-base for answering complex questions, in: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, Volume 1 (Long Papers)*, New Orleans, Louisiana, USA, June 1–6, 2018, M.A. Walker, H. Ji and A. Stent, eds, Association for Computational Linguistics, 2018, pp. 641–651. doi:10.18653/v1/n18-1059.

[22] P. Trivedi, G. Maheshwari, M. Dubey and J. Lehmann, LC-QuAD: A corpus for complex question answering over knowledge graphs, in: *The Semantic Web – ISWC 2017 – 16th International Semantic Web Conference, Proceedings, Part II*, Vienna, Austria, October 21–25, 2017, C. d'Amato, M. Fernández, V.A.M. Tamma, F. Lécué, P. Cudré-Mauroux, J.F. Sequeda, C. Lange and J. Heflin, eds, Lecture Notes in Computer Science, Vol. 10588, Springer, 2017, pp. 210–218. doi:10.1007/978-3-319-68204-4_22.

[23] R. Usbeck, R.H. Gusmita, A.N. Ngomo and M. Saleem, 9th challenge on question answering over linked data (QALD-9) (invited paper), in: *Joint Proceedings of the 4th Workshop on Semantic Deep Learning (SemDeep-4) and NLIWoD4: Natural Language Interfaces for the Web of Data (NLIWOD-4) and 9th Question Answering over Linked Data Challenge (QALD-9) Co-Located with 17th International Semantic Web Conference (ISWC 2018)*, Monterey, California, United States of America, October 8th–9th, 2018, K. Choi, L.E. Anke, T. Declerck, D. Gromann, J. Kim, A.N. Ngomo, M. Saleem and R. Usbeck, eds, CEUR Workshop Proceedings, Vol. 2241, CEUR-WS.org, 2018, pp. 58–64, http://ceur-ws.org/Vol-2241/paper-06.pdf.

[24] R. Usbeck, A.N. Ngomo, F. Conrads, M. Röder and G. Napolitano, 8th challenge on question answering over linked data (QALD-8) (invited paper), Vol. 2241, 2018, pp. 51–57, http://ceur-ws.org/Vol-2241/paper-05.pdf.

[25] R. Usbeck, A.N. Ngomo, B. Haarmann, A. Krithara, M. Röder and G. Napolitano, 7th open challenge on question answering over linked data (QALD-7), in: *Semantic Web Challenges – 4th SemWebEval Challenge at ESWC 2017, Revised Selected Papers*, Portoroz, Slovenia, May 28–June 1, 2017, M. Dragoni, M. Solanki and E. Blomqvist, eds, Communications in Computer and Information Science, Vol. 769, Springer, 2017, pp. 59–69. doi:10.1007/978-3-319-69146-6_6.

[26] R. Usbeck, M. Röder, M. Hoffmann, F. Conrads, J. Huthmann, A.N. Ngomo, C. Demmler and C. Unger, Benchmarking question answering systems, *Semantic Web* **10**(2) (2019), 293–304. doi:10.3233/SW-180312.

[27] R. Usbeck, M. Röder, A.N. Ngomo, C. Baron, A. Both, M. Brümmer, D. Ceccarelli, M. Cornolti, D. Cherix, B. Eickmann, P. Ferragina, C. Lemke, A. Moro, R. Navigli, F. Piccinno, G. Rizzo, H. Sack, R. Speck, R. Troncy, J. Waitelonis and L. Wesemann, GERBIL: General entity annotator benchmarking framework, in: *Proceedings of the 24th International Conference on World Wide Web, WWW 2015*, Florence, Italy, May 18–22, 2015, A. Gangemi, S. Leonardi and A. Panconesi, eds, ACM, 2015, pp. 1133–1143. doi:10.1145/2736277.2741626.

[28] D. Vollmers, R. Jalota, D. Moussallem, H. Topiwala, A.N. Ngomo and R. Usbeck, Knowledge graph question answering using graph-pattern isomorphism, 2021, CoRR, abs/2103.06752.

[29] D. Vrandecic and M. Krötzsch, Wikidata: A free collaborative knowledgebase, *Commun. ACM* **57**(10) (2014), 78–85. doi:10.1145/2629489.