What is in the KGQA Benchmark Datasets? Survey on Challenges in Datasets for Question Answering on Knowledge Graphs

Nadine Steinmetz¹ · Kai-Uwe Sattler¹

Received: 11 April 2020 / Revised: 26 February 2021 / Accepted: 5 May 2021 / Published online: 1 June 2021
© The Author(s) 2021

Abstract

Question Answering based on Knowledge Graphs (KGQA) still faces difficult challenges when transforming natural language (NL) to SPARQL queries. Simple questions only referring to one triple are answerable by most QA systems, but more complex questions requiring complex queries containing subqueries or several functions are still a tough challenge within this field of research. Evaluation results of QA systems therefore also might depend on the benchmark dataset the system has been tested on. For the purpose to give an overview and reveal specific characteristics, we examined currently available KGQA datasets regarding several challenging aspects. This paper presents a detailed look into the datasets and compares them in terms of challenges a KGQA system is facing.

Keywords Question answering on knowledge graphs · Dataset analysis · Natural language transformation · Pattern recognition

1 Introduction

Question answering (QA) aims at answering questions formulated in natural language on data sources and, therefore, combines methods from natural language processing (NLP), linguistics, database processing, and information retrieval. Though early research activities have been already conducted in the sixties, QA has received a great attention again over the last few years. Main reasons are significant progress in speech recognition and NLP but also the public availability of knowledge bases as a primary data source to answer questions from general domains.

In general, applications that transform natural language questions to formal queries on structured data can be summarized as the class of Natural Language Interfaces to Databases (NLIDB). Approaches based on semantic knowledge bases, such as RDF knowledge graphs—which we reference as Question Answering on Knowledge Graphs (KGQA) in the following—are a very promising idea because they can rely on large knowledge datasets such as DBpedia and also simplify tasks such as mapping and disambiguation. Thus,

numerous approaches and systems have been proposed in the past and also various datasets and challenges have been published. One of the most prominent examples for the QA on DBpedia community is the QALD workshop series¹. For each edition of the challenge, new datasets for training and testing have been published. Over the years, the datasets have increased in terms of amount of questions and of course they have been adapted to the current DBpedia versions at that time. In addition, further datasets have been created and published for the purpose to evaluate KGQA systems that transform NL to DBpedia-based SPARQL queries.

However, the multitude of datasets makes it difficult for researchers to choose the right dataset. Thus, we present in this work a comparative survey of available datasets for KGQA. The intention of this survey is two-fold:

- provide QA researchers with an overview of existing datasets, their structure and characteristics, and
- show the specific challenges KGQA systems require to overcome.

We performed an extensive analysis of 26 different datasets: the training and test datasets of all QALD challenges (18 datasets in total), LC-QuAD 1.0, and SimpleDBpedia-

✉ Nadine Steinmetz
nadine.steinmetz@tu-ilmenau.de

Kai-Uwe Sattler
kus@tu-ilmenau.de

¹ Technische Universität Ilmenau, Ilmenau, Germany

¹ <http://qald.aksw.org/>.

aQA. These datasets are all based on DBpedia of 2016² (the latest pure DBpedia versions without migrated Wikidata information). In addition, we analyzed the WebQuestions and SimpleQuestions datasets to provide a comparison in terms of linguistic characteristics.

We analyzed and compare these datasets in view of the following challenges to KGQA systems.

- ambiguity,
- lexical gap,
- complex queries,
- templates,
- ontology types, and
- answer types.

For all aspects, we introduce the characteristic of the aspect, describe our analysis methods and measures, as well as discuss our findings.

The remainder of this paper is organized as follows: Sect. 2 introduces some related work mainly surveys on existing (KG)QA systems. We introduce the datasets and some general information about them in Sect. 3. Our analysis results are presented in Sect. 4, and we conclude our work in Sect. 5.

2 Related Work

In this survey, the research field of interest is Question Answering (QA). In specific, we focus on the transformation of natural language (NL) to SPARQL queries which we refer to as Question Answering over knowledge graphs (KG QA). Since Semantic Web technologies enable knowledge to be represented as RDF triples in triple stores, the access to this structured knowledge via NL has become an interesting research field. The first challenge on Question Answering over Linked Data (QALD) has been organized in 2011 in co-location with the Extended Semantic Web Conference (ESWC)³. The 9th and latest edition took place in 2018 in co-location with the International Semantic Web Conference (ISWC). For all nine editions of the challenge, the organizers provided datasets in terms of training and test data. These datasets are among the datasets under observation for this survey and are described more in detail in Sect. 3.

The first survey (to the best of our knowledge) explicitly referring to KGQA and comparing KGQA systems has been published by Höffner et al. [6]. The authors present an overview of 62 different KGQA systems. The comparison is accomplished based on several challenges the authors have

identified: ambiguity, complexity of queries, and the lexical gap amongst others. For our survey, we adopted the challenges listed by the authors as analysis aspects of the datasets. We also added a few more aspects. Section 4 presents more details on the challenges we chose for our analysis processes.

Bouziane et al. [4] published their work on a survey of QA systems. The authors compare 31 different (KG)QA systems regarding specific characteristics, such as interfaces to databases, open domains, ontologies, and the focus on (web) documents. Beside a more or less detailed description of the systems, the authors present an overview of the quality of these systems in terms of success rate, respectively, correct answers.

Just recently, a survey on Natural Language Interfaces for databases (NLIDB) in general has been published by Affolter et al. [1]. The authors focus on QA systems in general, but not on KGQA systems specifically. They take KGQA systems into account when comparing them to other systems that transform natural language to SQL queries. Overall, the authors present an overview of 24 (KG)QA systems and evaluate them based on a set of 10 different questions.

The surveys described above focus on the overview and comparison of NLIDB systems in general or specifically KGQA systems. In contrast and supplementary, we focus on the datasets that are available to evaluate KGQA systems—specifically based on DBpedia. We analyzed several KGQA datasets and examined specific characteristics regarding the challenges researchers are facing when developing a KGQA system. For this study, we focused on datasets that provide questions to be answered via DBpedia (cf. Auer et al. [2]).

3 Benchmark Datasets

For the task of KGQA, a dataset should at least contain the following information:

- the NL question string,
- the SPARQL query that gives the relevant answers, and
- a specified SPARQL endpoint and affected graph.

For reasons of unavailability of the endpoint or an updated knowledge graph version, it is helpful to have the expected results provided in the dataset. Thus, researchers are able to reproduce the results retrieved on an outdated knowledge base when the SPARQL endpoint has been updated.

For this study, we analyzed the most popular KGQA datasets based on DBpedia (cf. Table 1 in Kacupaj et al. [7]):

- the datasets of the QALD challenge (train and test dataset, respectively, 18 datasets overall)
- LC-QuAD 1.0 (train and test dataset)

² either 2016-04 or 2016-10.

³ <http://qald.aksw.org>.

- SimpleDBpediaQA (train and test dataset)

Several other datasets have been published in terms of QA, respectively, NLI on knowledge bases other than triple stores containing RDF data. Due to missing SPARQL queries, these datasets cannot be compared to the datasets introduced above in all aspects. But, to provide a comparison regarding linguistic characteristics, we take into account the datasets WebQuestions and SimpleQuestions for the dataset analysis presented in this survey.

Overall, we analyzed 26 datasets. For our analysis process, we only utilized the English language questions—in case the dataset provides the questions in multiple languages. This means, all further analysis results and statistics refer to the English language parts of the datasets.

The benchmark datasets are described more in detail in the next sections. The analysis results are summarized in Sect. 4.

3.1 QALD

In recent years, the QALD challenge has become a well-established competition in terms of KGQA on DBpedia facts. By now, nine challenges have been organized since 2011. For each challenge, the organizers provided a training and a test dataset⁴. In early years, these datasets contained at least the NL question, SPARQL query and the relevant results. Later, keywords, answer type and the information about required aggregation functions, a different knowledge base other than DBpedia and hybrid question answering on RDF and free text has been added to the datasets. For the latest edition of the challenge, the datasets contain the following fields:

- answertype—values are one of Boolean, date, number, resource, string
- aggregation—true or false
- onlydbo—constitutes, if only DBpedia ontology properties are required for the SPARQL query; true or false
- hybrid—always set false for QALD 8 and 9 datasets
- question—each question is represented in different languages. At maximum 12 languages are available: de, ru, pt, en, hi_IN, fa, it, pt_BR, fr, ro, es, nl. Cf. Table 1 for more details which languages are available for the datasets.
- query—the SPARQL query
- answers—the result of the query provided as result bindings

The datasets for challenges 1–5 have been provided in XML format and for challenges 6–9 in JSON format. The

datasets have been compiled manually or from query logs regarding the NL question.

Table 2 shows the results of the challenges QALD 8/9 and highlights the best-performing systems.

Overall, for the QALD challenges we have analyzed 18 different datasets, each containing between 41 and 408 questions. The datasets QALD 8 and 9 are the most recent and based on the latest DBpedia version. For the benefit of clearness, we include the analysis results only for the most recent datasets (8 and 9) and provide the results for QALD 1–7 in the “Appendix.”

3.2 LC-QuAD

In 2017, the LC-QuAD 1.0 dataset has been published. LC-QuAD 2.0 has been published in early 2019. Both datasets are structured in a test and a training dataset⁵. While LC-QuAD 1.0 provides SPARQL queries over pure DBpedia (version of 2016), LC-QuAD 2.0 provides SPARQL queries based on Wikidata and the Wikidata-migrated DBpedia version of 2018. As all other datasets of this survey utilize pure DBpedia as of 2016 and therefore in terms of comparability, we provide our analysis results for the LC-QuAD 1.0 dataset. The test dataset contains 1000 and the training dataset contains 4000 question-query pairs. The datasets are structured using the following fields for each record:

- _id, the record id
- corrected_question, the actual NL question
- intermediary_question, the NL question having surface forms of (named) entities enclosed with angle brackets
- sparql_query, SPARQL query based on the 04-2016 release of DBpedia
- sparql_template_id, one of 37 different SPARQL template ids applicable for the respective query

Both, the training and the test dataset contain 37 different types of SPARQL template IDs. Trivedi et al. [11] describe the LC-QuAD 1.0 dataset in detail. The creators of LC-QuAD 1.0 have published evaluation results of KGQA systems against their dataset. Table 3 shows these results. Details on the competing systems can be found on the authors’ website⁶.

3.3 SimpleDBpediaQA

The SimpleDBpediaQA (SDBQA) dataset has been introduced by Azmy et al. [3] as a derivative of the SimpleQuestions dataset. The authors created the new dataset using a mapping of Freebase to DBpedia and provided a subset of the

⁴ The datasets are available here: <https://github.com/ag-sc/QALD>.

⁵ The datasets are available here: <https://github.com/AskNowQA/LC-QuAD>.

⁶ cf. <http://lc-quad.sda.tech/lcquad1.0.html> as of January 2021.

Table 1 Available languages for datasets of QALD 3–9; for QALD 1 and 2 the questions are only provided in English language

	<i>de</i>	<i>ru</i>	<i>pt</i>	<i>en</i>	<i>hi_IN</i>	<i>fa</i>	<i>it</i>	<i>pt_BR</i>	<i>fr</i>	<i>ro</i>	<i>es</i>	<i>nl</i>
QALD 3 train	100	0	0	100	0	0	100	0	100	0	100	100
QALD 3 test	99	0	0	99	0	0	99	0	99	0	99	99
QALD 4 train	200	0	0	200	0	0	200	0	200	200	200	200
QALD 4 test	50	0	0	50	0	0	50	0	50	50	50	50
QALD 5 train	300	0	0	340	0	0	300	0	300	300	300	300
QALD 5 test	49	0	0	59	0	0	49	0	49	49	49	49
QALD 6 train	350	0	0	350	0	350	350	0	350	350	350	350
QALD 6 test	100	0	0	100	0	100	100	0	100	100	100	100
QALD 7 train	215	0	0	215	215	96	215	0	215	168	215	215
QALD 7 test	43	0	0	43	0	43	43	0	43	43	43	43
QALD 8 train	219	0	0	219	179	120	219	0	219	185	219	219
QALD 8 test	0	0	0	41	0	0	0	0	0	0	0	0
QALD 9 train	408	399	399	408	404	403	408	4	408	407	408	408
QALD 9 test	150	150	150	150	150	150	22	150	150	150	150	150

Table 2 Competing systems and results of QALD challenges 8 (Usbeck et al. [12]) and 9 (Usbeck et al. [13])

System	Macro Recall	Macro Precision	Macro F1	Average Time per Doc in ms
QALD 8				
QAKIS	0.0528	0.061	0.0563	15,414
gAnswer	0.3902	0.3862	0.388	1919
WDAqua-core0	0.4065	0.3912	0.3872	1725
QALD 9				
Elon	0.053	0.049	0.050	219
QASystem	0.116	0.097	0.098	1014
TeBaQA	0.134	0.129	0.130	2668
wdaqua-core1	0.267	0.261	0.250	661
gAnswer	0.327	0.293	0.298	3076

Bold indicates the best results for the respective dataset and measure

Table 3 Competing systems and results for LC-QuAD 1.0

System	Recall	Precision	F1
QAmP	0.50	0.25	0.33
WDAqua	0.38	0.22	0.28

Table 4 Comparison of original SimpleQuestions and derived SimpleDBpediaQA datasets

Dataset	Validation	Training	Test	Total
SimpleQuestions	75,910	10,845	21,687	108,442
SimpleDBpediaQA	30,186	4305	8595	43,086

original questions. Table 4 shows an overview of the original and the derived datasets. The dataset is formatted as JSON files in the following manner:

– ID

- Query—the actual NL question
- Subject—the DBpedia URI of the entity required in the SPARQL query
- FreebasePredicate—the URI of the Freebase property from the original SimpleQuestions dataset
- PredicateList—a list of formalized SPARQL query triples, containing the following keys:
 - Predicate—the DBpedia URI of the required property in the triple
 - Direction—forward or backward—states if the entity of the Subject field is used as subject (forward) or as object (backward) within the triple
 - Constraint—either null or an URI of a DBpedia ontology class

If the PredicateList field contains more than one object, the objects need to be joined in the SPARQL query via the

Fig. 1 Example of a question object from the SimpleDBpediaQA dataset and the resulting required SPARQL query

```
{
  "ID": "00323",
  "Query": "what is the name of a work done by gary paulsen",
  "Subject": "http://dbpedia.org/resource/Gary_Paulsen",
  "FreebasePredicate": "www.freebase.com/book/author/works_written",
  "PredicateList": [
    {
      "Predicate": "http://dbpedia.org/ontology/author",
      "Direction": "backward",
      "Constraint": "http://dbpedia.org/ontology/WrittenWork"
    },
    {
      "Predicate": "http://dbpedia.org/ontology/notableWork",
      "Direction": "forward",
      "Constraint": null
    }
  ]
},
```

```
PREFIX dbo: <http://dbpedia.org/ontology/>
PREFIX dbr: <http://dbpedia.org/resource/>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
SELECT ?x WHERE {
  { ?x dbo:author dbr:Gary_Paulsen .
    ?x rdf:type dbo:WrittenWork . }
  UNION
  { dbr:Gary_Paulsen dbo:notableWork ?x . }
}
```

UNION operator. Figure 1 shows a sample question object and the resulting SPARQL query.

To the best of our knowledge, no KGQA system has been evaluated on the SDBQA dataset, yet. Respectively, evaluation results of a performance on the dataset have not been published so far.

3.4 WebQuestions

WebQuestions consists of a test, a validation and training dataset. The dataset has been created based on Freebase facts. The dataset provides the answers to a question as triple facts, describing a subject-relationship-object as explanation for the answers. The datasets (provided in JSON format⁷) contain the following keys:

- url—the Freebase URI of the focus entity of the question
- targetValue—the list of answers for the questions
- utterance—the actual question

The training dataset contains 3778 records, and the test dataset contains 2032 records. State-of-the-art systems achieve an accuracy of 45.5% on the dataset (cf. Brown et al. [5]).

3.5 SimpleQuestions

The first version of the dataset has been published in 2015. This version has been used for the creation of the Sim-

pleDBpediaQA dataset, as described in 3.3. For our survey, we analyzed version 2.0 of the SimpleQuestions dataset⁸. Similar to WebQuestions, SimpleQuestions facts have been extracted from Freebase. The questions then have been created manually based on the extracted facts. The dataset is a tab-separated text file with four columns:

- the first three columns contain the subject, property and object of the fact triple grounded in the Freebase knowledge graph
- fourth column: the actual NL question

The training dataset contains 75,909 records, and the test dataset contains 21,686 records. The latest QA approach evaluated against the SimpleQuestions dataset achieves an accuracy of 78.1% (cf. Petrochuk and Zettlemoyer [8]).

4 Dataset Analysis

For the analysis of the 26 datasets, we took into account a set of aspects, respectively, challenges (KG)QA systems are facing when utilizing a dataset. Adopted from Höffner et al. [6], we examined the datasets regarding the following aspects:

- Ambiguity (Sect. 4.2),
- Lexical Gap (Sect. 4.3),
- Complex Queries (Sect. 4.4), and
- Templates (Sect. 4.5).

⁷ Available for download at: <https://worksheets.codalab.org/worksheets/0xba659fe363cb46e7a505c5b6a774dc8a>.

⁸ Available for download at: <https://github.com/davidgolub/SimpleQA/tree/master/datasets/SimpleQuestions>.

Table 5 Overview of all datasets, tr—training dataset, te—test dataset

Dataset	#Q	POS sequences	Normalized POS sequences	Graph terns	pat-	Min words	Max words	Average word count
QALD 8 tr	219	190	165	11		3	14	7
QALD 8 te	41	39	37	6		4	15	8
QALD 9 tr	408	334	297	16		3	16	7
QALD 9 te	150	132	127	14		3	15	8
LC-QuAD tr	4000	3668	3454	8		2	25	11
LC-QuAD te	1000	962	933	8		3	21	11
SDBQA tr	30,186	13,194	10,809	3		1	34	7
SDBQA te	8595	4773	3979	3		3	20	7
WebQu tr	3778	2227	1682	n/a		3	14	7
WebQu te	2032	1558	1528	n/a		3	15	7
SimpleQu tr	75,910	12,303	10,672	n/a		1	34	8
SimpleQu te	21,687	34,891	30,005	n/a		1	25	7

In addition, we analyzed the datasets for existing ontology types. The types of occurring named entities give a hint about the domain of the question. The results are shown in Sect. 4.6.

Another challenging aspect of KGQA systems is the identification of the question type which results in the definition of the answer type. Hereby, it is analyzed if the question asks for a date, or a resource, etc. We added an answer type analysis based on the given data to our survey and analyzed the datasets regarding 13 different types of answers. The results are shown in Sect. 4.7.

4.1 Overview

Table 5 gives an overview of general statistical parameters. The table includes the number of records contained in the dataset, the number of POS sequences, and normalized POS sequences retrieved from the NL questions⁹, the minimum, maximum and average number of words in the NL questions. For the analysis, we processed the provided information of the datasets, such as the question, the SPARQL query as well as additional information if available. Also, we performed several NLP algorithms on the NL question, such as Part-of-Speech (POS) tagging and Named Entity Linking (NEL). For POS tagging, we utilized the Stanford POS tagger¹⁰ with the model *english-left3words-dist3sim*. The table gives a short overview of all analyzed datasets. We provide further statistics, the description of our analysis processes and the result discussion in detail in the following sections.

⁹ With POS sequence, we refer to the extracted POS tags in the same order as they occur in the NL question. The purpose of this analysis is described in detail in Sect. 4.5.

¹⁰ <https://nlp.stanford.edu/software/tagger.shtml>.

4.2 Ambiguity

Main findings:

- Datasets contain surface forms with high amount of entity candidates.
- Many required named entities are not the most popular (respectively easy to be disambiguated) within the candidate list of the surface form.

4.2.1 Topic Definition

The more ambiguous the question in a dataset, the harder it is to retrieve the correct answer. For our analysis, we examined several aspects of ambiguity:

- How many named entities are mentioned in the NL question, minimum/ maximum per question? The more entities to be disambiguated, the harder is the query generation.
- How many entity candidates can be retrieved from the underlying knowledge base¹¹ for each mention? (which means, how ambiguous is the respective surface form?)
- Is the most popular candidate (in terms of the indegree of Wikipedia links) the correct one required for the SPARQL query? The disambiguation process is easier in case the most popular candidate is the relevant one, which means, how hard is it to disambiguate the surface forms.

¹¹ in our case, the knowledge base is DBpedia of version 2016–10 resp. 2016–04.

Table 6 We considered these POS tags for the identification of named entities

Tag	Description	Example
JJ	adjective	total
N	noun	capital
IN	preposition or subordinating conjunction	of
DT	determiner	the

4.2.2 Analysis Description

As for the datasets, we do not have information which textual parts of the NL question refer to which part in the given SPARQL query (if any). Especially for the description of relationships, respectively, reference of ontology properties, this is a difficult task for a KGQA system.

For the analysis of the dataset, we therefore took a detailed look at the surface forms¹², the entity candidates and the respective SPARQL query. For each question, we took into account only specific POS tags (cf. Table 6) to identify the mentioned named entities and considered the following POS sequences:

- JJ N IN N
- N IN N
- JJ N
- N IN DT N
- N

These, POS sequences have been detected for Steinmetz [10] from the most common POS sequences of DBpedia labels.

Here, N refers to any noun which can be a singular or mass noun (NN), a plural noun (NNS), singular proper noun (NNP), or a plural proper noun (NNPS). Each POS sequence might be followed by more nouns which are also taken into account as part of the mentioned entity. For each identified sequence in the question, we retrieve potential entity candidates from DBpedia. For the dictionary, disambiguation and redirect labels are utilized. Then, this extracted list of entity candidates for the complete question is compared to the entities contained in the provided SPARQL query¹³.

In addition, we utilized the Falcon 2.0 API to identify surface forms and entity candidates in the questions. The API has been introduced by Sakor et al. [9] and proposes to identify entities and relations within short texts or questions

over Wikidata and DBpedia. For the analysis, we requested the API with the following parameters:

- db=1, for DBpedia entities
- k=500, for the top 500 entity candidates for an identified surface form¹⁴

Tables 7 and 8 show the result of our analysis. For each approach to identify the entities in the NL question, we detect the following measures, as shown in the table:

- number of surface forms that reference entities—how many entities can be detected compared to the number of required entities for SPARQL query? (Entities NL, Entities SPARQL)
- maximum number of entity candidates per surface form—how hard is it to disambiguate the entities? (max Candidates)
- the number of named entities (identified in the SPARQL query) that are the most popular in terms of indegree within all candidates for a surface form (Most popular)

As there is no DBpedia-based SPARQL query provided in WebQuestions and SimpleQuestions, the information about entities in SPARQL queries and if the most popular entity candidate is the correct one cannot be examined and is marked as *not applicable* (n/a) in the tables.

4.2.3 Result Discussion

Our analysis shows, that ambiguity is a serious challenge for QA systems throughout all datasets. There are mentions of named entities having more than 100 entity candidates w.r.t. DBpedia.

For our approach, the most ambiguous term within all QALD datasets is *Lincoln*, having 215 entity candidates¹⁵. The most ambiguous term for all datasets is contained in the SimpleQuestions datasets: *pilot* with 479 entity candidates¹⁶.

In general, the Falcon 2.0 API provides far less entity candidates per surface form. The phrase with the highest amount of entity candidates is *Jacob and Abraham* with 44 entity candidates. It is contained in the LC-QuAD train dataset.

We also analyzed, how hard the disambiguation process for the detected entities would be. For this, we detected, if

¹² The surface form of an entity is the textual reference of the entity as it appears in the NL text.

¹³ By entities, we refer to instances of classes, not the classes themselves. That means, for the SPARQL query we simply count the resources that start with <http://dbpedia.org/resource/>.

¹⁴ We chose this limit, because for our approach the maximum number of candidates is as high as 479. The analysis results show that the maximum number of candidates is far less than 500 for the Falcon 2.0 API.

¹⁵ Originating from the question *Who was the wife of U.S. president Lincoln?*.

¹⁶ Originating from the question *who is the director of pilot?.*, amongst others.

Table 7 Analysis results of ambiguity aspects, tr—training dataset, te—test dataset

Dataset	#Q	Entities SPARQL	Our Approach			Falcon API		
			Entities NL	Most popular		Entities NL	Most popular	
QALD 8 tr	219	243	241	162	67%	246	174	70%
QALD 8 te	41	41	38	19	50%	42	18	43%
QALD 9 tr	408	419	421	285	68%	444	294	66%
QALD 9 te	150	152	149	84	56%	168	87	50%
LC-QuAD tr	4000	5275	5891	3279	56%	6029	2596	43%
LC-QuAD te	1000	1346	1491	838	56%	1533	650	42%
SDBQA tr	30,186	30,186	34,383	16,509	48%	23,273	7169	31%
SDBQA te	8595	8595	9894	4687	47%	6617	2045	31%
WebQu tr	3778	n/a	4999	n/a		3198	n/a	
WebQu te	2032	n/a	2665	n/a		1734	n/a	
SimpleQu tr	75,910	n/a	85,477	n/a		55,869	n/a	
SimpleQu te	21,687	n/a	24,411	n/a		15,926	n/a	

Table 8 Analysis results of ambiguity aspects, tr—training dataset, te—test dataset

Dataset	#Q	Our Approach		Falcon API	
		max Candidates	max Entities NL	max Candidates	max Entities NL
QALD 8 tr	219	215	3	43	3
QALD 8 te	41	129	4	21	2
QALD 9 tr	408	215	4	43	4
QALD 9 te	150	156	4	21	2
LC-QuAD tr	4000	249	9	44	6
LC-QuAD te	1000	267	7	39	6
SDBQA tr	30,186	461	8	37	6
SDBQA te	8595	461	7	39	4
WebQu tr	3778	304	5	21	3
WebQu te	2032	461	5	21	3
SimpleQu tr	75,910	479	14	43	6
SimpleQu te	21,687	432	11	39	4

the required entity is the most popular among the list of candidates for the respective surface form. A disambiguation or ranking process can be considered more simple, if the NL questions always mention very popular entities with the respective surface forms.

However, our analysis shows that in many cases the relevant entity is not the most popular among the candidates of the list. The SDBQA datasets seem to be very hard to disambiguate, as we detected the lowest amount of entities that are the most popular for both datasets and both entity detection approaches. For the Falcon 2.0 API, the required entity is the most popular in only 31% of the cases. According to our analysis, the QALD 8 train dataset requires the least elaborate disambiguation process as up to 70% (for the Falcon 2.0 API, 67% for our approach) of the required entities are the most popular among the candidates.

Overall this means, a QA system must be able to disambiguate the mentioned entities—either using an answer

ranking or according to the given context. Or, the system provides queries for all (or a subset of) relevant entities and provides the results to the user to receive feedback which entity and result is the demanded one.

4.3 Lexical Gap

Main findings:

- High amount of named entities and relations are hard to be identified within the NL questions.
- There is a significant difference between datasets regarding severity of lexical gap.

then compared this list to the list of the extracted relations by the Falcon 2.0 API from the NL question. We counted the properties that were not found by the API in proportion to the number of properties required for the SPARQL query.

Table 10 shows the amount of extracted properties from the SPARQL query (Properties SPARQL), and the results of the Falcon API for the extraction of relations (Relations Not Found). The number depicts the total number of properties not found by the API. The percentage depicts the proportion of relations not found compared to the number of required properties as extracted from the SPARQL query.

4.3.3 Result Discussion

Regarding the analysis of the identification of the required entities in the NL question, the amount of entities not found is remarkably high. By our approach, a minimum of 25% of the entities contained in the SPARQL queries of the QALD datasets could not be found. For the QALD 8 test dataset, the percentage is as high as 46%. For instance, the question *What was the university of the rugby player who coached the Stanford rugby teams during 1906–1917?* requires the entity `dbr:1906-17_Stanford_rugby_teams`. For this, different parts of the question (and also numbers besides nouns) must be combined to find the label for this entity.

In comparison, the Spotlight API achieved even a lower rate of correct entities detected for most datasets (respectively, a higher percentage of entities not identified correctly from the NL question). The Spotlight API only returns the most likely entity for each identified surface form according to the given context. But, with the questions the context is meager and a disambiguation is apparently not successful in many cases. This experiment shows that the disambiguation process should not be considered before creating the SPARQL queries during the QA pipeline. A sample question where the API fails is *Does the Toyota Verossa have the front engine design platform?*. The required entities here are `dbr:Toyota_Verossa` and `dbr:Front-engine_design`. The API only detects the first one.

The Falcon 2.0 API performs similar or slightly better than our approach on the QALD datasets. The results for the LC-QuAD datasets are very good—only 9% of the entities are not among the candidates extracted by the API. In contrast, the API performs worse than our approach on the SDBQA datasets—the amount of entities that could not be identified is as high as 40%. A sample question where the Falcon 2.0 API fails to identify the relevant entities is *Which computer scientist won an oscar?*. Here, the required entities are `dbr:Computer_Science` and `dbr:Academy_Award`.

As Table 10 shows, the correct identification of DBpedia ontology properties is even harder than the entity identification.

The amount of properties not detected by the Falcon 2.0 API is remarkably high for all datasets, but especially for the SDBQA datasets with 73%. Mostly, this results from the fact that DBpedia facts and subgraphs are modeled along the ontology and not directly as expressed in natural language. For instance, the question *Give me English actors starring in Lovesick*. requires the properties `dbo:country` and `dbo:birthPlace` to create the English heritage of the requested actors. Obviously, these relations cannot be deduced from the NL alone. But, the API also fails to detect the property `dbo:knownFor` within the the question *What is Elon Musk famous for?*.

We provide our analysis results for properties not identified by the Falcon 2.0 API as JSON dataset¹⁹. Future mapping processes to identity alternative labels for DBpedia ontology properties might benefit from this dataset.

Our analyses show that the datasets contain a high number of questions where the correct entities and properties required for the SPARQL query cannot be detected by all of the approaches considered for our analyses. Furthermore, this means that for many questions the correct SPARQL query cannot be created using the correct entities which results in incorrect answers.

Apparently, the lexical gap is a significant challenge not only for mapping of relationship descriptions to ontology properties, but even for the identification of the correct entities mentioned in the NL question. But obviously, there are significant differences between the datasets.

4.4 Complex Queries

Main findings:

- Only the QALD datasets require SPARQL operators other than UNION. LC-QuAD datasets do not contain any SPARQL operator.
- Entity detection approaches often identify more entities in the NL question than required, especially for the LC-QuAD datasets.
- Except for QALD 8 test and the SDBQA datasets, all datasets contain questions where the required SPARQL query does not contain any named entity.

¹⁹ <http://doi.org/10.5281/zenodo.4782639>.

Table 10 Lexical gap for relation mentions in natural language and properties occurring in the SPARQL query, tr—training dataset, te—test dataset

Dataset	#Q	Properties SPARQL	Relations not found – Falcon API	
QALD 8 tr	219	265	139	52%
QALD 8 te	41	33	18	55%
QALD 9 tr	408	276	157	57%
QALD 9 te	150	101	61	60%
LC-QuAD tr	4000	6197	3423	55%
LC-QuAD te	1000	1080	568	53%
SDBQA tr	30,186	28,380	20,708	73%
SDBQA te	8595	8044	5895	73%

Table 11 Overview of SPARQL operators contained in the provided queries in the datasets – #Q denotes the number of queries overall, UN—UNION, OPT—OPTIONAL, HAV—HAVING, GRO—GROUP BY, FIL—FILTER, ORD—ORDER, LIM—LIMIT, OFF—OFFSET, tr—training dataset, te—test dataset

	#Q	ASK	UN	OPT	HAV	GRO	FIL	ORD	LIM	OFF
QALD 8 tr	219	34	2	1	0	8	9	23	23	18
QALD 8 te	41	0	0	0	0	1	1	3	8	3
QALD 9 tr	408	37	29	1	3	19	32	36	39	24
QALD 9 te	150	4	15	2	2	7	16	10	10	5
LC-QuAD tr	4000	285	0	0	0	0	0	0	0	0
LC-QuAD te	100	83	0	0	0	0	0	0	0	0
SDBQA tr	30,186	0	6370	0	0	0	0	0	0	0
SDBQA te	8595	0	1748	0	0	0	0	0	0	0

Table 12 Overview of maximum/minimum number of entities in the SPARQL queries and the maximum number of triples, tr—training dataset, te—test dataset

	max entities	min entities	max #triples	median #triples	average #triples
QALD 8 tr	3	0	5	1	2
QALD 8 te	1	1	5	1	1
QALD 9 tr	3	0	5	2	2
QALD 9 te	3	0	4	2	2
LC-QuAD tr	2	1	3	2	2
LC-QuAD te	2	1	3	2	2
SDBQA tr	1	1	2	1	1
SDBQA te	1	1	2	1	1

4.4.1 Topic Definition

The expressiveness of semantic knowledge bases is based on the rather simple data structure having facts stored as triples and the effective approach of using these graph patterns in the SPARQL query to access the knowledge. However, SPARQL supports several operators which might lead to rather complex queries. Obviously, more complex queries result from complex questions and are certainly a challenge for developers of KGQA systems.

4.4.2 Analysis Description

For our analysis, we examined the datasets (that provide a SPARQL query) on the existence of the following query operators: FILTER, OFFSET, LIMIT, ORDER, GROUP, UNION, OPTIONAL, Subquery, HAVING, ASK type.

Detailed information how often each operator occurs in each dataset is given in Table 11. As none of the datasets contain SPARQL queries with subqueries, we left that out in the table.

As another parameter for complexity, we also counted the maximum/minimum number of entities extracted from the SPARQL query and the maximum/average/median number of triples in the SPARQL query. The results are also shown in Table 12.

A further essential process step within KGQA systems is the identification of the correct focus in the NL question. The challenge here is to examine which part of the question is the subject of interest and how it relates to the rest of the question. For template-based KGQA systems, the graph patterns of the SPARQL query are constructed around this focus. Sequence-to-sequence systems can benefit from a preceding focus identification, as the trained model might make use of

Table 13 Overview of the number of entities identified within the NL question compared to entities extracted from the SPARQL query; tr—training dataset, te—test dataset

	#Q	Our Approach			Falcon 2.0 API		
		More in SPARQL	More in NL	Equal	More in SPARQL	More in NL	Equal
QALD 8 tr	219	24	25	170	14	21	184
QALD 8 te	41	10	5	26	7	8	26
QALD 9 tr	408	43	46	319	29	72	307
QALD 9 te	150	22	21	107	15	45	90
LC-QuAD tr	4000	467	832	2701	118	747	3135
LC-QuAD te	1000	121	207	672	40	191	769
SDBQA tr	30,186	5959	7748	16,479	8725	1734	19,727
SDBQA te	8595	1703	2252	4640	2472	484	5639

Fig. 2 Sample question that does not require a named entity within the SPARQL query

Which actors have the last name "Affleck"?	<pre> PREFIX dbo: <http://dbpedia.org/ontology/> PREFIX foaf: <http://xmlns.com/foaf/0.1/> SELECT ?x WHERE { ?x a dbo:Actor . ?x foaf:surname 'Affleck'@en . }</pre>
--	--

a preprocessed input question where the focus is tagged. In some cases, there are more than one focus to be identified in the question. Mostly, the focus(es) are represented by a named entity in the question which results in a resource URI in the SPARQL query.

To examine this aspect of complexity, we analyzed and compared the number of named entities in the NL question and the SPARQL query. If the NL question contains more than the SPARQL query, the process of focus identification is an essential step. If the numbers of entities in the NL question and the SPARQL query are equal, this might be a hint, that all entities found in the natural language can be adopted for the SPARQL query. If there are more entities in the SPARQL query than in the NL question, an analysis process might be required to deduce the additional entities from the focus(es) and relationships extracted from the linguistics of the question.

Table 13 shows the results for all datasets and contrasts the results for our approach and the results of Falcon 2.0 API. The table contains the information if more named entities have been found in the SPARQL query (More in SPARQL) compared to the NL question or in the NL question (More in NL) compared to the SPARQL query, or if the number of identified named entities are equal in the SPARQL query and the NL question.

4.4.3 Result Discussion

Our analysis shows, that for all datasets the test datasets reflect the complexity of the training datasets or they contain even less complex queries, because sometimes operators are not present in the test dataset although they occurred in the

training dataset. The HAVING and OFFSET operator is utilized only rarely. None of the SPARQL operators is present in any query of the LC-QuAD datasets. Only, the QALD datasets contain all operators to some extent. The SDBQA datasets naturally do not contain ASK queries or any other SPARQL operators other than UNION queries. The UNION queries are only utilized to model the SPARQL query with alternative properties—as described in Sect. 3.3.

For almost all datasets, the minimum number of named entities contained in the SPARQL queries is zero. An example for a question resulting in zero named entities in the SPARQL query is: *Which actors have the last name "Affleck"?*. Here, the query only asks for a specific type of entities that contain the string "Affleck" as object for a property `lastName` (i.e., `foaf:lastName`). Figure 2 shows this sample question and the resulting SPARQL query without a named entity in it.

Regarding maximum/minimum number of entities, the dataset QALD 8 test stands out among the others. All SPARQL queries comprise exactly one entity²⁰ which could be a hint that this dataset is a bit easier to process in terms of evaluation results than the others. This assumption can be confirmed by the analysis results shown in Table 11. QALD 8 test comprises only a few SPARQL operators and no ASK question. On the other hand, our analysis process was not able to find a relatively high number of entities (between 17 and 19 out of 41), as shown in Table 9.

In most cases, the number of entities detected in the NL question is higher than the number of entities extracted from

²⁰ While the occurrence of exactly one entity per query is the natural characteristic of the SDBQA datasets.

the SPARQL query—as shown in Table 13. For our approach, this results from the detection process itself rather aiming at high recall than high precision. As described in Sect. 4.3, we detected entities in the NL question according to several POS sequences—all patterns include at least one noun. This procedure extracts all (combined) nouns from the question, even though it is not relevant as entity for the query. But, the Falcon 2.0 API also extracts in many cases too many entities, especially for the LC-QuAD datasets.

An example for a question having more named entities in the NL than in the SPARQL query is: *How many gold medals did Michael Phelps win at the 2008 Olympics?*. Here, both our algorithm and the Falcon 2.0 API detect *Michael Phelps* and *2008 Olympics* as named entities, but the SPARQL query only asks for the gold medalist `dbr:Michael_Phelps` and filters the respective events for the strings “2008” and “Olympics”:

```
SELECT COUNT(?sub) as ?c WHERE
{ ?sub http://dbo:goldMedalist dbr:Michael_Phelps .
FILTER(contains(str(?sub), "2008")
&& contains(str(?sub), "Olympics")) }
```

Nevertheless, the number of questions where there are more entities in the SPARQL query is also reasonable high for all datasets. In these cases, the additional entities must be deduced from the linguistics of the question or along the edges of the knowledge graph. An example for such a case that often occurs is the mistreatment of an apparent type information using a property and a resource in the SPARQL query. For instance, in many cases a type constraint is expressed in the way *Which [ontology class name] was [...]*?. So, for these cases the phrase following the word *which* must be used to identify the correct ontology class from the KG. But in some cases—specifically for the DBpedia—such class membership is modeled using a property. For instance, the question *Which professional surfers were born in Australia?* might ask for instances of the class `dbo:Surfer`. But, the given SPARQL query in the dataset models the fact using the property `dbo:occupation` and the resource `dbr:Surfer`:

```
SELECT DISTINCT ?uri WHERE
{ ?uri dbo:occupation dbr:Surfer ;
  dbo:birthPlace dbr:Australia }
```

This example shows, that an obvious class membership can also be modeled as relationship between entities. This circumstance must be taken into account, when transforming NL questions to SPARQL (for DBpedia).

4.5 Templates

Main findings:

- Datasets represent a high number of different POS sequences: 56,844 out of 171,487 different questions.
- We identified 22 different graph patterns within the SPARQL queries, but only a few (between 5-7) are frequently required.

4.5.1 Topic Definition

As described in [6], template-based approaches try to identify patterns within the natural language and transform them to SPARQL query templates. The relevant parts of the templates are then mapped to the underlying knowledge base, and the complete query is created. Most approaches use linguistic and syntactic parsers to identify similar natural language patterns that lead to the same SPARQL query template. For the analysis of the datasets regarding templates, we followed the assumption that the amount of different patterns is limited. Of course, natural language can be very expressive (also depending on the language), but in terms of KGQA, we assumed that a SPARQL query template can only be deduced from a limited number of NL patterns. Therefore, we extracted the POS sequences of the NL questions and performed a normalization step.

Furthermore, templates can also be found regarding the SPARQL query. The query represents a subgraph of the complete knowledge graph. Depending on the different options how the subjects and objects of the triples are connected, different graph patterns are depicted. Therefore, we analyzed the SPARQL queries of the datasets in order to detect the amount of different graph patterns.

4.5.2 Analysis Description

We retrieved the Part-of-Speech (POS) patterns for all questions of all datasets. Which means, we annotated a question with POS tags—utilizing the Stanford POS tagger—and retrieved the patterns by only using the tags in the order they occur in the question. Furthermore, we processed the POS sequences in terms of normalization. After the identification of named entities in the NL question²¹, we replaced all POS tags that belong to this entity with the placeholder `RESOURCE`. Consecutive `RESOURCE` occurrences

²¹ For this step, we utilized our approach as described in Sect. 4.2, as the other two external APIs do not provide information about the identification process of the entities or the exact position within the NL question.

Fig. 3 Overview of all Graph Patterns extracted from the datasets

are replaced by only one RESOURCE. In that way, the two questions (initially having different POS sequences):

- When was Harry Potter born? (POS sequence: WRB VBD NNP NNP VBN), and
- When was Beyoncé born? (POS sequence: WRB VBD NNP VBN)

are linked to the same normalized POS sequence: WRB VBD RESOURCE VBN. After this normalization step, we counted the occurrences of the patterns in the datasets again. The numbers for the extracted (normalized) sequences are shown in Table 5.

In addition to the POS sequences of the NL questions, we also analyzed what type of subgraphs require to be constructed for the SPARQL queries. Therefore, we extracted the graph patterns and counted the occurrence of the patterns per dataset. For the extraction of the graphs, we considered the following principles:

- We removed GROUP, ORDER, LIMIT, OFFSET, HAVING and FILTER restrictions. These operators do not affect the subgraph.

- As OPTIONAL triples are not necessarily required to answer a question, we also removed these clauses.
- SPARQL queries containing UNION clauses are disaggregated to all relevant graphs. As all graphs might contribute to answer the question, all graphs are assigned as graph pattern for this question.

After extraction of all graphs from the queries, we analyzed the set of graphs for isomorphism and counted the occurrence of the graph patterns for each dataset.

4.5.3 Result Discussion

As shown in Table 5, our assumption (of a limited number of POS sequences compared to different questions) is certainly rebutted by the numbers of different sequences we found within the datasets. Altogether, we found 56,844 different POS sequences (out of 171,487 different questions for all datasets) for the questions in English language. Noteworthy is the fact, that not one of these sequences occurs in all datasets.

The sequence with the most occurrences of 1,612 is WP VBZ DT NN IN NNP NNP. An example question for

that sequence is *Who is the owner of Universal Studios?*. But it only occurs in 4 of the 26 datasets. The most frequent sequences in terms of different datasets are WP VBD NNP²², WP VBZ DT NN IN NNP²³, and WP VBZ DT NN IN NNP NNP²⁴. They all occur in 21 of the 26 datasets.

Utilizing the normalization step, the overall number of sequences is reduced to 50,455. But still, there is no normalized POS sequence that occurs in all datasets. The most frequent normalized sequence with 2601 occurrences is WP VBZ DT NN IN RESOURCE which also originates from questions like *Who is the owner of Universal Studios?* or *What is the revenue of IBM?*. This sequence only occurs in 4 of the 26 datasets. The most frequent sequence in terms of different datasets is WP VBD RESOURCE²⁵. This sequence occurs in 24 of the 26 datasets.

Obviously, the number of different POS sequences that must be taken into account might be limited, but on a very high level.

Overall, we identified 22 different graph patterns for QALD 1–9, LC-QuAD and SDBQA. The patterns are shown in Fig. 3.

In addition, we analyzed by how many different normalized POS sequences the graph patterns are represented within each dataset. The results for this analysis are shown in Table 14.

As already detected for the aspect of complex queries, the SPARQL query graphs require at most 5 triples. These few graph patterns are represented by two different subgraphs—graph IDs 15 and 22 in Fig. 3. These patterns are only contained in the QALD 8 (both) and QALD 9 train datasets. All other datasets contain 4 triples at most.

Within the QALD datasets, only 5 different graphs (graph IDs 1–5) are remarkably present, while the other graph patterns are only used sparsely or not at all. The LC-QuAD datasets contain 7 different graph patterns that are mainly used for the queries. Only one further pattern is used a few times. That means, LC-QuAD only utilizes 8 different patterns with 3 triples at most.

4.6 Ontology Types

Main findings:

- In general, DBpedia ontology classes are mostly too general to deduce a domain focus of the dataset.
- SDBQA contains many entities assigned to DBpedia ontology classes that might hint at the entertainment domain.
- For the other datasets, a domain cannot be allocated.

4.6.1 Topic Definition

The difficulty to identify the correct formal query for a given NL question is also dependent on the specific domain of the question. For some domains (technical), terms are nearly unique. That means, the disambiguation task can be omitted and the mapping of surface forms to properties, classes and entities is straightforward. For other domains, these tasks might be much more difficult which hinders the overall task of question answering. Therefore, we analyzed the datasets for the ontology classes that inherit from the entities used in the SPARQL queries. These ontology classes give a hint about the domain of the question. For instance, the SPARQL query contains an entity of class *dbo:Athlete*²⁶ the question is most likely from the sports domain.

4.6.2 Analysis Description

For the analysis, we extracted the entities of the given SPARQL queries and retrieved the respective ontology classes via `rdf:type` information of the DBpedia knowledge graph. We took into account all assigned classes along the class hierarchy of the DBpedia ontology. Table 15 shows the top 10 DBpedia ontology classes and their frequency belonging to named entities of the SPARQL queries distributed over all datasets. The table also lists the top 5 classes for each dataset group (train and test dataset together) separately.

4.6.3 Result Discussion

DBpedia does not provide specific domain information for the resources, and most of the ontology classes are too general to hint a certain domain for the question. Therefore, a domain assignment of the datasets cannot be performed based on these results.

²² occurs 103 times overall, sample question: *Who created Goofy?*.

²³ occurs 265 times overall, sample question: *What is the revenue of IBM?*.

²⁴ occurs 1,220 times overall, sample question: *Who is the owner of Universal Studios?*.

²⁵ For questions like *Who developed DBpedia?*.

²⁶ <http://dbpedia.org/ontology/Athlete>.

Table 14 Occurrence of Graph patterns in the datasets—amount of different normalized POS sequences per graph pattern

#Q ID	<i>QALD8te</i>	<i>QALD8tr</i>	<i>QALD9te</i>	<i>QALD9tr</i>	<i>LC – QuADte</i>	<i>LC – QuADtr</i>	<i>SDBQAtr</i>	<i>SDBQAte</i>
	41	219	150	408	1000	4000	8595	30,186
1	31	98	55	152	234	770	8751	3167
2	0	19	10	27	191	820	0	0
3	2	31	38	86	194	708	2692	1042
4	1	24	15	31	79	295	116	50
5	1	1	10	9	66	268	0	0
6	0	0	1	2	0	0	0	0
7	1	0	3	4	0	0	0	0
8	0	0	1	1	0	0	0	0
9	0	3	0	4	0	0	0	0
10	0	0	0	0	0	0	0	0
11	0	1	0	4	0	0	0	0
12	0	0	1	1	0	0	0	0
13	0	2	1	2	156	537	0	0
14	0	1	1	1	21	109	0	0
15	0	1	0	1	0	0	0	0
16	0	1	1	3	0	0	0	0
17	0	0	0	0	2	13	0	0
18	0	0	0	0	0	0	0	0
19	0	0	0	0	0	0	0	0
20	0	0	3	1	0	0	0	0
21	0	0	1	0	0	0	0	0
22	1	0	0	0	0	0	0	0

Table 15 List of the top 10 DBpedia ontology classes found as types of the occurring named entities in the SPARQL queries over all datasets, and the top 5 classes for QALD 1–9, both LC-QuAD and both SDBQA datasets

DBpedia class	Frequency
Top 10 over all datasets	
http://dbpedia.org/ontology/Person	3718
http://dbpedia.org/ontology/MusicalArtist	3431
http://dbpedia.org/ontology/Album	2552
http://dbpedia.org/ontology/Film	2015
http://dbpedia.org/ontology/Band	1922
http://dbpedia.org/ontology/Settlement	1914
http://dbpedia.org/ontology/MusicGenre	1674
http://dbpedia.org/ontology/City	1628
http://dbpedia.org/ontology/SoccerPlayer	1292
http://dbpedia.org/ontology/Country	1128
Top 5 QALD 8+9	
http://dbpedia.org/ontology/Country	232
http://dbpedia.org/ontology/Person	194
http://dbpedia.org/ontology/OfficeHolder	90
http://dbpedia.org/ontology/Company	52

Table 15 continued

DBpedia class	Frequency
http://dbpedia.org/ontology/City	50
Top 5 LC-QuAD	
http://dbpedia.org/ontology/Person	347
http://dbpedia.org/ontology/Company	205
http://dbpedia.org/ontology/Settlement	175
http://dbpedia.org/ontology/OfficeHolder	169
http://dbpedia.org/ontology/MusicalArtist	162
Top 5 SDBQA	
http://dbpedia.org/ontology/MusicalArtist	3245
http://dbpedia.org/ontology/Person	3066
http://dbpedia.org/ontology/Album	2501
http://dbpedia.org/ontology/Film	1876
http://dbpedia.org/ontology/Band	1842

However, Table 15 lists for the SDBQA datasets 4 of 5 ontology classes (Film, Band, MusicalArtist, and Album) that might hint that the contained questions are mostly from the entertainment domain.

Table 16 Overview of the answer types in the different datasets; tr—training dataset, te—test dataset; date—date, b—Boolean, s—string, nc—number count, np—number property, rlt—resource list typed, rlut—resource list untyped, rt—resource typed, rut—resource untyped, un—unknown

	#Q	d	b	s	nc	np	rlt	rlut	rt	rut	un
QALD 8 tr	219	9	34	15	7	8	15	30	29	62	10
QALD 8 te	41	4	0	16	1	0	0	4	0	7	9
QALD 9 tr	408	16	37	37	14	11	68	67	35	95	28
QALD 9 te	150	11	4	16	8	6	21	21	4	25	34
LC-QuAD tr	4000	4	285	292	283	0	283	1163	407	1168	115
LC-QuAD te	1000	1	83	69	61	0	73	268	100	327	18
SDBQA tr	30,186	0	0	189	0	0	4105	11,048	604	7709	6531
SDBQA te	8595	0	0	54	0	0	1151	3084	174	2282	1850

4.7 Answer Types

Main findings:

- Answers and answer types for SDBQA datasets cannot be retrieved for a large amount of questions.
- For LC-QuAD the number answer type is only given by COUNT operator – not via property with number as range.
- For most datasets, the resource answer type mostly does not require a type assignment in the SPARQL query.

- number count—a number resulting from a COUNT operator in the SPARQL query
- number property—a number resulting from a property in the SPARQL query
- resource list typed—a list or resources with a specific type
- resource list untyped—a list of resources without specific type
- resource typed—one resource with a specific type
- resource untyped—one resource without specific type
- unknown—the answer type could not be detected

The QALD challenge provides a hint about the answer type in the datasets, but only for the latest editions. Also, the provided answer types are more general than the types we included for our survey. Therefore, we performed an analysis regarding answer types for all KGQA datasets. Some datasets provide the answers for each question as part of the dataset. In this case, we analyzed the answer type according to the provided answers. For some datasets, the answers are not provided: both LC-QuAD datasets, the SDBQA datasets, and some test datasets of the QALD challenge. In this case, we used the SPARQL query to retrieve the answers the respective DBpedia version. If we could not retrieve the answers, we further analyzed the question:

4.7.1 Topic Definition

Recently, a challenge on answer type prediction has been published as part of the International Semantic Web Conference 2020 (ISWC)²⁷. The task of this challenge is to predict the answer type of the question according to the structure of the NL question. For instance, the question *Who is the heaviest player of the Chicago Bulls?* requires the answer to be of type `dbo:BasketballPlayer` or the question *How many employees does IBM have?* requires the answer to be of type `xsd:integer`.

4.7.2 Analysis Description

For the analysis of the datasets regarding answer types, we defined 10 different types:

- date
- Boolean—resulting from an ASK question
- string—asking for string objects, such as last names or nick names

- the questions starts with *When*—the answer types is set to *date*
- the query starts with ASK—the answer type is set to *boolean*
- the query contains a COUNT operator for the only variable—the answer type is set to *number count*

If none of these analysis steps results on a proper answer type, the type is set to *unknown*. This applies for many of the LC-QuAD questions, because no results could be retrieved. Table 16 shows the results of our analysis. The table contains the

²⁷ <https://smart-task.github.io/#>.

overall numbers of the occurrences of the answer types we pre-defined.

4.7.3 Result Discussion

The most obvious observation is the high number of unknown answer types for both LC-QuAD datasets. This results from missing results in the datasets and the missing response answers for the SPARQL queries when fetching the answers on the DBpedia graph of 2016-10. Overall, we had to set the answer type to unknown for a remarkably high amount of questions for all datasets. This means, that for these questions the answers are not available and the question cannot be answered—either because of missing facts in the knowledge graph or faulty SPARQL queries. However, a KGQA system would try to generate a query for these questions and retrieve answers. The system would fail for these cases.

We provide the results of our answer type analysis as a separate dataset. The dataset contains the id from the original dataset, the name of the source dataset, the question string and the detected answer type as a JSON file for each dataset file²⁸.

5 Discussion & Summary

The analysis presented in this paper gives a thorough overview of the KGQA evaluation datasets currently available. We examined 22 datasets that provide NL language questions (some of them in multiple languages) and a respective SPARQL query. Additionally, four further datasets containing a reasonable number of interesting questions have been taken into account for comparison issues. Based on several aspects, we examined essential characteristics of the datasets to be able to compare them. The performed experiments reveal the requirements that KGQA systems need to fulfill regarding SPARQL functions, disambiguating surface forms, or detecting the correct answer type. Therefore, the survey provides researchers with extensive information which specific challenges are contained in the datasets (amongst others):

- The required entities are often hard to be identified, because of very ambiguous surface forms and tough disambiguation processes.
- Also, the lexical gap is remarkably high for entity and relations names.
- The datasets differ in severity of complex queries in terms of SPARQL operators required for the SPARQL query.

- We identified 22 different graph patterns within the datasets, but only a few are required frequently.

In terms of comparability, researchers need a dataset that provides realistic questions, the SPARQL query and answers according to a current SPARQL endpoint.

Unfortunately, the QALD datasets for editions 1–7 are—in general—outdated regarding the DBpedia version compared to the version currently available at the public DBpedia SPARQL endpoint²⁹. However, the DBpedia versions are simply outdated, because in newer versions facts are missing or properties are replaced. The general approach, how facts are modeled, is maintained throughout the versions of the knowledge graph. Therefore, even the outdated datasets are a useful source for sample questions and complex queries. The LC-QuAD 1.0 datasets provide a reasonable amount of records, but we identified two problems:

- compared to the QALD datasets, LC-QuAD 1.0 does not contain any SPARQL queries with additional options, such as UNION, OPTIONAL, HAVING, etc., and
- a large amount of the SPARQL queries (referencing DBpedia 2016–10) do not provide any result on the respective SPARQL endpoint³⁰.

SDBQA is the dataset with the highest amount of questions. But similar to the LC-QuAD 1.0 datasets, it does not contain any further SPARQL options except for the UNION operator. And likewise, the dataset contains a high amount of questions containing properties from the GOLD ontology, which is not contained in the DBpedia datasets of 2016-10 (anymore).

Our results show that actually there are differences between the datasets. While the datasets of QALD datasets overall are fairly similar and only individual datasets stand out, the differences to the LC-QuAD and SDBQA datasets are significant. However, the WebQuestions and Simple-Questions datasets show similar structure and characteristics as the questions of the KGQA datasets. Altogether, the four QA datasets contain over 26.000 questions and might serve as a good source for further examinations of questions and their structure often asked on the internet.

With our work, we aim at a detailed insight in KGQA datasets available for evaluation. We provide the results of our answer type analysis and the property detection fails as separate datasets for download and further examination.

²⁸ The dataset can be downloaded here: <https://doi.org/10.5281/zenodo.4782653>.

²⁹ As of January 2021.

³⁰ As of January 2021.

Overall, we examined 26 different datasets based on several challenging aspects and provided statistical numbers on ambiguity, complexity, templates, the lexical gap, ontology types, and answer types. Although the datasets show significant differences for several aspects, none of the datasets stands out in terms of a low or high difficulty level when all aspects are considered altogether. Nevertheless, our analysis results exemplify the characteristics of each dataset in detail. In this way, developers of KGQA systems are able to choose a certain training dataset when they want to further focus on a specific challenging aspect. Overall, our analysis results show that (KG)QA is a sophisticated, but interesting research field which deals with the diversity of natural language and the expressiveness of SPARQL queries.

Funding Open Access funding enabled and organized by Projekt DEAL.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

A Appendix

See Tables 17, 18, 19, 20, 21, 22, 23, 24, 25, and 26

Table 17 Overview of all datasets, tr—training dataset, te—test dataset

Dataset	# <i>Q</i>	POS sequences	Normalized POS sequences	Graph terms	pat-	Min words	Max words	Average word count
QALD 1 tr	50	47	47	7		3	14	7
QALD 1 te	50	47	47	9		3	12	8
QALD 2 tr	100	95	96	10		3	15	8
QALD 2 te	99	94	88	10		3	14	8
QALD 3 tr	100	95	95	7		3	15	8
QALD 3 te	99	89	84	9		3	14	8
QALD 4 tr	200	173	162	12		3	15	8
QALD 4 te	50	49	48	6		3	16	8
QALD 5 tr	340	297	276	13		3	18	8
QALD 5 te	59	58	55	10		4	18	8
QALD 6 tr	350	299	270	15		3	16	8
QALD 6 te	100	93	90	7		3	15	7
QALD 7 tr	215	191	163	11		3	14	7
QALD 7 te	43	41	38	8		3	13	7

Table 18 Analysis results of ambiguity aspects, tr—training dataset, te—test dataset

Dataset	#Q	Entities SPARQL	Our Approach		Falcon API	
			Entities NL	Most popular	Entities NL	Most popular
QALD 1 tr	50	14	47	11	54	10
QALD 1 te	50	37	94	28	60	27
QALD 2 tr	100	79	101	63	118	57
QALD 2 te	99	102	112	73	124	81
QALD 3 tr	100	79	99	61	116	55
QALD 3 te	99	102	111	75	121	81
QALD 4 tr	200	185	208	140	238	138
QALD 4 te	50	41	46	41	58	22
QALD 5 tr	340	274	357	195	421	189
QALD 5 te	59	55	65	42	81	43
QALD 6 tr	350	328	356	236	415	230
QALD 6 te	100	100	95	59	115	75
QALD 7 tr	215	239	242	163	251	180
QALD 7 te	43	49	42	23	47	22

Table 19 Analysis results of ambiguity aspects, tr—training dataset, te—test dataset

Dataset	#Q	Our Approach		Falcon API	
		max Candidates	max Entities NL	max Candidates	max Entities NL
QALD 1 tr	50	147	2	20	3
QALD 1 te	50	132	2	21	2
QALD 2 tr	100	215	2	21	2
QALD 2 te	99	132	3	39	4
QALD 3 tr	100	215	2	21	2
QALD 3 te	99	132	3	39	3
QALD 4 tr	200	147	3	39	3
QALD 4 te	50	147	2	43	4
QALD 5 tr	340	215	5	43	4
QALD 5 te	59	141	3	20	4
QALD 6 tr	350	215	4	43	4
QALD 6 te	100	141	3	20	3
QALD 7 tr	215	215	4	43	3
QALD 7 te	43	156	2	35	2

Table 20 Lexical gap of entity mentions in natural language and entities occurring in the SPARQL query, tr—training dataset, te—test dataset

Dataset	#Q	Entities SPARQL	Entities not found – our approach		Entities not found – Spot- light		Entities not found – Fal- con API	
QALD 1 tr	50	14	3	21%	3	21%	3	22%
QALD 1 te	50	37	6	16%	8	21%	4	11%
QALD 2 tr	100	79	13	16%	16	20%	9	12%
QALD 2 te	99	102	27	26%	28	27%	12	12%
QALD 3 tr	100	79	15	19%	16	20%	11	15%
QALD 3 te	99	102	26	25%	28	27%	11	11%
QALD 4 tr	200	185	39	21%	46	25%	25	14%
QALD 4 te	50	41	12	29%	15	37%	11	27%
QALD 5 tr	340	274	61	22%	75	27%	47	18%
QALD 5 te	59	55	13	24%	13	24%	8	15%
QALD 6 tr	350	328	71	22%	85	26%	53	17%
QALD 6 te	100	100	27	27%	34	34%	17	17%
QALD 7 tr	215	239	54	23%	69	29%	26	11%
QALD 7 te	43	49	24	49%	25	51%	22	45%

Table 21 Lexical gap for relation mentions in natural language and properties occurring in the SPARQL query, tr—training dataset, te—test dataset

Dataset	#Q	Properties SPARQL	Relations not found – Falcon API	
QALD 1 tr	50	16	13	81%
QALD 1 te	50	39	28	72%
QALD 2 tr	100	52	30	58%
QALD 2 te	99	64	40	63%
QALD 3 tr	93	53	30	57%
QALD 3 te	95	65	40	62%
QALD 4 tr	188	125	74	59%
QALD 4 te	48	36	26	72%
QALD 5 tr	325	192	121	63%
QALD 5 te	58	34	20	58%
QALD 6 tr	335	211	128	61%
QALD 6 te	95	73	42	58%
QALD 7 tr	215	147	57	39%
QALD 7 te	43	23	16	70%

Table 22 Overview of SPARQL functions contained in the provided queries in the datasets—#Q denotes the number of queries overall, UN—UNION, OPT—OPTIONAL, HAV—HAVING, GRO—GROUP BY, FIL—FILTER, ORD—ORDER, LIM—LIMIT, OFF—OFFSET, tr—training dataset, te—test dataset

	#Q	ASK	UN	OPT	HAV	GRO	FIL	ORD	LIM	OFF
QALD 1 tr	50	2	8	36	0	0	41	1	1	0
QALD 1 te	50	4	4	26	0	0	33	3	3	0
QALD 2 tr	100	8	10	67	2	2	75	4	4	0
QALD 2 te	99	8	9	69	0	0	72	6	6	5
QALD 3 tr	100	8	12	1	2	2	16	4	4	0
QALD 3 te	99	8	9	0	0	0	11	6	6	5
QALD 4 tr	200	17	21	1	2	2	26	10	10	10
QALD 4 te	50	4	1	0	0	0	3	7	7	5
QALD 5 tr	340	22	27	1	2	2	27	22	22	20
QALD 5 te	59	3	4	0	0	0	2	6	6	6
QALD 6 tr	350	27	33	1	2	21	28	28	28	26
QALD 6 te	100	3	3	0	1	1	4	6	6	6
QALD 7 tr	215	29	3	1	0	7	10	19	19	17
QALD 7 te	43	7	1	0	0	3	3	6	6	3

Table 23 Overview of maximum/minimum number of entities in the SPARQL queries and the maximum number of triples, tr—training dataset, te—test dataset

	max Entities	min entities	max #triples	median #triples	average #triples
QALD 1 tr	2	0	4	2	2
QALD 1 te	2	0	4	2	2
QALD 2 tr	3	0	4	2	2
QALD 2 te	3	0	4	2	2
QALD 3 tr	6	0	3	2	2
QALD 3 te	3	0	4	2	2
QALD 4 tr	4	0	4	2	2
QALD 4 te	3	0	4	2	2
QALD 5 tr	4	0	4	2	2
QALD 5 te	3	0	5	2	2
QALD 6 tr	4	0	5	2	2
QALD 6 te	2	0	3	1	1
QALD 7 tr	3	0	5	1	2
QALD 7 te	2	0	4	1	2

Table 24 Overview of the number of entities identified within the NL question compared to entities extracted from the SPARQL query; tr—training dataset, te—test dataset

	#Q	Our Approach			Falcon 2.0 API		
		More in SPARQL	More in NL	Equal	More in SPARQL	More in NL	Equal
QALD 1 tr	50	1	30	19	0	36	14
QALD 1 te	50	2	16	32	1	20	29
QALD 2 tr	100	5	26	69	2	32	66
QALD 2 te	99	7	17	75	4	21	74
QALD 3 tr	93	6	25	69	3	32	65
QALD 3 te	95	7	16	76	4	20	75
QALD 4 tr	188	15	41	144	10	52	138
QALD 4 te	48	4	8	38	5	16	29
QALD 5 tr	325	27	89	224	16	115	209
QALD 5 te	58	4	11	44	5	20	34
QALD 6 tr	335	31	61	258	22	90	238
QALD 6 te	95	14	9	77	6	19	75
QALD 7 tr	215	21	26	168	14	24	177
QALD 7 te	43	11	5	27	9	7	27

Table 25 Occurrence of Graph patterns in the datasets—amount of different POS sequence per graph pattern

#Q	QALD 1 te	QALD 1 tr	QALD 2 te	QALD 2 tr	QALD 3 te	QALD 3 tr	QALD 4 te	QALD 4 tr	QALD 5 te	QALD 5 tr	QALD 6 te	QALD 6 tr	QALD 7 te	QALD 7 tr
ID	50	50	99	100	99	100	50	200	59	340	100	350	43	215
1	19	8	48	39	42	37	19	70	15	100	53	111	23	92
2	3	11	5	8	5	8	5	12	7	19	6	25	7	13
3	13	21	23	27	22	30	13	48	10	78	16	86	7	31
4	1	1	8	3	8	5	6	13	5	24	12	29	1	28
5	1	3	4	3	4	4	2	8	3	12	2	14	1	3
6	0	0	0	0	0	0	0	2	0	2	0	2	0	0
7	0	0	2	0	2	0	0	3	2	3	0	6	0	1
8	0	0	1	1	1	0	0	1	0	1	0	1	0	1
9	0	0	3	0	3	0	0	3	2	2	0	4	0	4
10	0	0	1	0	1	0	0	1	0	0	0	0	0	0
11	2	0	0	4	0	4	0	3	0	3	1	4	1	0
12	0	0	0	1	0	1	0	1	0	1	0	1	0	0
13	4	0	1	1	0	0	0	0	1	1	1	1	1	1
14	0	0	0	0	0	0	0	0	1	0	0	1	0	1
15	0	0	0	0	0	0	0	0	1	0	0	1	0	1
16	1	1	0	0	0	0	2	0	0	3	0	3	1	0
17	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	1	0	0	1	0	0	0	0	0	0	0	0	0	0
19	0	1	0	0	0	0	0	0	0	0	0	0	0	0
20	0	0	0	0	0	0	0	0	0	0	0	0	0	0
21	0	0	0	0	0	0	0	0	0	0	0	0	0	0
22	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 26 Overview of the answer types in the different datasets; tr—training dataset, te—test dataset; date—date, b—Boolean, s—string, nc—number count, np—number property, rlt—resource list typed,

rlut—resource list untyped, rt—resource typed, rut—resource untyped, un—unknown

	# <i>Q</i>	<i>d</i>	<i>b</i>	<i>s</i>	<i>nc</i>	<i>np</i>	<i>rlt</i>	<i>rlut</i>	<i>rt</i>	<i>rut</i>	<i>un</i>
QALD 1 tr	50	0	3	1	0	0	10	8	0	1	27
QALD 1 te	50	0	4	4	0	1	5	9	0	1	26
QALD 2 tr	100	2	8	5	0	1	18	22	0	2	42
QALD 2 te	99	2	8	7	0	1	13	28	0	0	40
QALD 3 tr	100	4	8	1	3	2	16	11	4	11	40
QALD 3 te	99	4	8	3	3	4	8	16	4	11	38
QALD 4 tr	200	8	17	7	6	7	36	27	9	24	59
QALD 4 te	50	2	4	3	3	2	4	5	4	5	18
QALD 5 tr	340	13	22	14	11	10	55	38	18	42	117
QALD 5 te	59	1	3	2	5	1	5	3	5	8	26
QALD 6 tr	350	14	27	15	16	11	61	42	21	50	93
QALD 6 te	100	6	3	4	3	4	10	15	10	27	18
QALD 7 tr	215	9	29	9	6	7	14	30	26	61	24
QALD 7 te	43	4	7	6	3	2	1	1	3	4	12

References

- Affolter K, Stockinger K, Bernstein A (2019) A comparative survey of recent natural language interfaces for databases. CoRR, abs/1906.08990,
- Auer S, Bizer C, Kobilarov G, Lehmann J, Cyganiak R, Ives Z (2007) Dbpedia: A nucleus for a web of open data. In: Aberer K, Choi K-S, Noy N, Allemang D, Lee K-II, Nixon L, Golbeck J, Mika P, Maynard D, Mizoguchi R, Schreiber G, and Cudré-Mauroux P (eds) The Semantic Web, pp 722–735, Berlin, Heidelberg, Springer Berlin Heidelberg. ISBN 978-3-540-76298-0
- Azmy M, Shi P, Lin J, Ilyas I (2018) Farewell freebase: Migrating the simplequestions dataset to dbpedia. In: Proceedings of the 27th international conference on computational linguistics, pp 2093–2103
- Bouziane A, Bouchiha D, Doumi N, Malki M (2015) Question answering systems: Survey and trends. Procedia Comput Sci, 73:366 – 375, 2015. ISSN 1877-0509. <https://doi.org/10.1016/j.procs.2015.12.005>. <http://www.sciencedirect.com/science/article/pii/S1877050915034663>. International Conference on Advanced Wireless Information and Communication Technologies (AWICT 2015)
- Brown TB, Mann B, Ryder N, Subbiah M, Kaplan J, Dhariwal P, Neelakantan A, Shyam P, Sastry G, Askell A, Agarwal S, Herbert-Voss A, Krueger G, Henighan T, Child R, Ramesh A, Ziegler DM, Wu J, Winter C, Hesse C, Chen M, Sigler E, Litwin M, Gray S, Chess B, Clark J, Berner C, McCandlish S, Radford A, Sutskever I, Amodei D (2020) Language models are few-shot learners
- Höffner K, Walter S, Marx E, Usbeck R, Lehmann J, Ngomo A-CN (2017) Survey on challenges of Question Answering in the Semantic Web. Semant Web J 8(6):895–920. <http://www.semantic-web-journal.net/system/files/swj1375.pdf>
- Kacupaj E, Zafar H, Lehmann J, Maleshkova M (2020) Vquanda: Verbalization question answering dataset. In: Harth A, Kiriene S, Ngomo A-CN, Paulheim H, Rula A, Gentile AL, Haase P, Cochez M (eds) The Semantic Web. Cham, Springer International Publishing, pp 531–547. ISBN 978-3-030-49461-2
- Petrochuk M, Zettlemoyer L (2018) SimpleQuestions nearly solved: A new upperbound and baseline approach. In: Proceedings of the 2018 conference on empirical methods in natural language processing, pp 554–558, Brussels, Belgium, October–November 2018. Association for Computational Linguistics. <https://doi.org/10.18653/v1/D18-1051>. <https://www.aclweb.org/anthology/D18-1051>
- Sakor A, Singh K, Patel A, Vidal M-E (2020) Falcon 2.0: An Entity and Relation Linking Tool over Wikidata, pp 3141–3148. Association for Computing Machinery, New York, NY, USA. ISBN 9781450368599. <https://doi.org/10.1145/3340531.3412777>
- Steinmetz N (2014) Context-aware semantic analysis of video metadata. Phd. thesis, Universität Potsdam
- Trivedi P, Maheshwari G, Dubey M, Lehmann J (2017) Lc-quad: A corpus for complex question answering over knowledge graphs. In: Proceedings of the 16th international semantic web conference (ISWC), pp 210–218, Springer
- Usbeck R, Ngomo A-CN, Conrads F, Röder M, Napolitano G (2018) 8th challenge on question answering over linked data (qald-8). In Choi et al., pp 51–57. <http://ceur-ws.org/Vol-2241/#paper-05>
- Usbeck R, Gusmita RH, Ngomo ACN, Saleem MM (2019) 9th challenge on question answering over linked data (qald-9). In Choi et al. pp 58–64. <http://ceur-ws.org/Vol-2241/#paper-06>

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.