Question Answering on Statistical Linked Data

Konrad Höffner and Jens Lehmann

University of Leipzig, Institute of Computer Science, AKSW Group
Augustusplatz 10, D-04109 Leipzig, Germany
E-mail: {hoeffner,lehmann}@informatik.uni-leipzig.de

Abstract. Statistical data is immensely valuable as it influences decisions in health care, policy and finance, among others. With the growth of the open data movement, more and more statistical data is becoming freely available. Consuming multidimensional numerical data requires experts and specialized tools, however, which is alleviated by the RDF Data Cube vocabulary which provides standardized access and context. Semantic Question Answering systems provide intuitive access to RDF data by transforming natural language queries into formal queries on RDF knowledge bases. Existing approaches, however, perform poorly on data cubes because of their specific structure. Our contribution is to design and implement the first Question Answering system on statistical Linked Data, built upon a previously published algorithm outline. Additionally, we kick-start this new research sub-field by creating a benchmark which will be added to the QALD challenge in 2016. The evaluation of our system on the benchmark shows the general feasibility of the approach, but also highlights shortcomings and challenges which we discuss in detail in order to provide a starting point for future work in the field.

1 Introduction

Statistical data is immensely valuable as it influences decisions in health care, policy and finance, among others. The general public is putting a high value [16] on open access to this information, which coincides with the open data movement and which has lead to an increased availability of statistical government data in initiatives like OpenSpending[1] and World Bank Open Data[2].

Consuming multidimensional numerical data requires specialized tools, however, which is alleviated by the RDF Data Cube vocabulary which provides standardized access and context. A data cube (also *OLAP cube* or *hypercube*) is a multidimensional array of values, which can be used to hold statistical data. Each element of the array, that is each cell of the cube, is called an observation, which is identified by the values of the dimensions for this observation, and which holds one or more measurements. Optionally, there are attributes which

[1] `http://openspending.org/`
[2] `http://data.worldbank.org/`

provide further context to the whole dataset or to single observations, such as the currency of an amount of money. The RDF Data Cube Vocabulary [1] allows expressing data cubes in RDF. Statistical Linked Data is available from projects such as Eurostat—Linked Data[3] and LinkedSpending [11] (a conversion of OpenSpending). More generally, from 294 032 542 RDF entities processed by LODStats[4] in March 2015, 47 302 049 have at least one triple with the DataCube class `qb:Observation` in object position which amounts to 16.08 % of all RDF entities.

Formal query languages, like SPARQL, are the standard way of querying RDF data but they require precise knowledge of the formal language and the vocabulary and are thus only suitable for experts. Semantic Question Answering systems provide a more intuitive access to Linked Data by accepting natural language queries which are then transformed into formal ones. Existing systems perform poorly on statistical data however (see Section 4), because of the different structure of the underlying data. Our contribution is to design and implement the first Question Answering system on statistical Linked Data, built upon a previously published algorithm outline. Additionally, we kick-start this new research sub-field by creating a benchmark, based on a question corpus compiled in previous work, which will be added to the QALD challenge in 2016. The evaluation of our system on the benchmark shows the general feasibility of the approach, but also highlights shortcomings and challenges which we discuss in detail in order to provide a starting point for future work in the field. Combined with speech recognition technology, this has the potential to simplify access of complex multi-dimensional data even on small mobile devices.

One of the reasons why standard Question Answering methods cannot be successfully applied is that answers need to be derived from statistical observations, which can have many dimensions and whose values are meaningless without the proper context and further processing, such as aggregate functions. For example, in traditional SQA, users typically ask about entities with certain properties, e.g. "Who is the wife of Barack Obama?". On statistical data, on the other hand, users typically ask about measurement values such as budgets for a certain purpose or about entities with certain values or value ranges, such as "Which were the top 10 funded research institutions in Europe in 2013?". This motivated previous work [10], where we compiled statistical user questions as a statistical question corpus used to analyse commonly used phrases and the information they represent.

In this article, we make the following core contributions:

- To our knowledge, with Tree-Based Cube Question Answering (TCQA) we designed the first Question Answering algorithm for statistical RDF data.
- An intelligent and flexible recursive parse tree matching technique which is particular suitable for typical questions asked against statistical datasets.
- We provide a benchmark consisting of 100 questions significantly extending a survey presented in previous work.

[3] `http://eurostat.linked-statistics.org/`
[4] `http://lodstats.aksw.org`

– We provide first results and a discussion of challenges to open up statistical RDF Question Answering as a new research field.

Section 2 shows existing SQA approaches and their workflow and differentiates, which of their parts can be reused and which ones have to be adapted for statistical data. Section 3 explains requirements, design decisions and structure of the TCQA algorithm. Section 4 evaluates and discusses the performance of the TCQA algorithm on a benchmark created out of the question corpus. Section 5 concludes with our plan to extend the corpus and implement as well as evaluate the algorithm presented here.

2 Related Work

To the best of our knowledge, only our own previous work addresses Question Answering on statistical Linked Data.[5] Question Answering on Linked Data however is an active area of research with numerous publications, of which we can only list a small selection. There are several different benchmark competitions which help evaluate and compare the multitude of QA approaches such as the general QALD [5] challenges or the specialized BioASQ [22] competition for biomedical data. Question Answering and search in general is even more established, as shown by specialized annual conferences, such as the Text REtrieval Conference (TREC) and the Conference and Labs of the Evaluation Forum (CLEF).

Wolfram—Alpha is a natural language interface that queries several structured sources using the computational platform *Mathematica* [24]. It can answer questions such as "What is the fifty-second smallest country by GDP per capita?", "What is the average country population?" or "What was the population in Germany in 1950?". However it does not support Linked Data and neither the source code nor the algorithm is published.

The *Clinical Narrative Temporal Relation Ontology (CNTRO)* is used in a system [21] that incorporates the time dimension in answering clinical questions. The ontology is based on Allen's Interval Based Temporal Logic [2] but it represents time points as well. The framework includes a reasoner that can infer additional time information, for example based on the transitivity of the *before* and *after* relations. The time dimension is used to identify the direction of possible causality between different events.

A hybrid geo-spatial approach [25] incorporates the spatial dimension by enriching semantic data with spatial relationships such as crossing, inclusion and nearness.

Intui2 [7] includes an algorithm that, similar to our TCQA, generates query templates recursively. Intui2 uses DBpedia and is based on *synfragments*, which are minimal parse subtrees of a question that refer to an RDF triple or RDF

[5] Seemingly similar, but technically different, some approaches, such as the massively parallel IBM Watson [15], use statistical methods instead of statistical data to addresses problems such as named-entity recognition and word-sense disambiguation.

query. The synfragments are combined in an order based on syntactic and semantic characteristics in order to create the final query.

Other approaches for querying Linked Data include facetted browsing approaches such as Broccoli [4] and Facete [20], which allow intuitive navigation from a certain starting resource of list of resources using property values.

There are a number of surveys on Question Answering. [6] present a data-driven problem analysis of Question Answering on the Geobase dataset. The authors identify eleven challenges that Question Answering has to solve. They are called: question types, language "light"[6], lexical ambiguities, syntactic ambiguities, scope ambiguities, spatial prepositions, adjective modifiers and superlatives, aggregation, comparison and negation operators, non-compositionality, and out of scope[7]. [12] give a detailed analysis of the system of the participants of the first and second QALD challenge. The authors give a short introduction to all systems which participated in this challenge and finally describe the occurring problems and solution strategies. A wide overview for Question Answering systems in the context of the Semantic Web is presented in [13]. After defining the goals and dimensions of Question Answering and presenting some related and historic work of Question Answering, the authors concentrate on ontology-based Question Answering. Before concluding the survey, the authors summarize the achievements of Question Answering so far and the challenges that are still open. Another related survey from 2012, [8], gives a broader overview of the challenges involved in constructing effective query mechanisms for Web-scale data. The authors analyze different approaches, such as PowerAqua and Treo, for five different challenges: usability, query expressivity, vocabulary-level semantic matching, entity recognition and improvement of semantic tractability. The same is done for architectural elements such as user interaction and interfaces and the impact on these challenges is reported. BioASQ [22] is a benchmark challenge which ran until October 2014 and consists of semantic indexing as well as an Semantic Question Answering part on biomedical data. In the Semantic Question Answering part, systems are expected to be hybrids, returning matching triples as well as text snippets but partial evaluation (text or triples only) is possible as well. Similar to [13], [3] gives a broad overview of domain specific Question Answering systems for biomedicine.

3 TCQA Algorithm

Question Answering on generic RDF data presents different types of complexity than QA on RDF Data Cubes. The former task presents *complex relationships* between entities but the number of relationships expressed in a single query and thus the structure of the query is usually low. This motivates approaches that focus on bridging the gap between properties and relationship references, such

[6] semantically weak constructions

[7] cannot be answered as the information required is not contained in the knowledge base

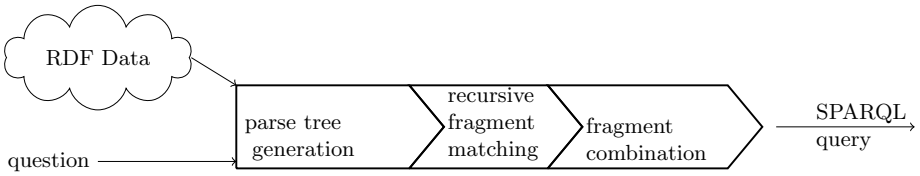

Fig. 1. The TCQA pipeline.

as the BOA framework [9]. Additionally, predefined query patterns are effective in covering a wide range of query types.

Queries that can be answered on RDF Data Cubes necessarily represent the simpler relationships that are possible in the RDF Data Cube Model. However, RDCs contain a large number of component properties (dimensions, measures and attributes), see Figure 3. Thus, queries usually have a *complex structure* with references to each component property that is to be restricted.

In order to be able to handle those complex structures, the TCQA algorithm (see Figure 1) is designed in a recursive manner, similar to Intui2 [7].

3.1 Initialization and Preprocessing

First, the index and the data cube cache are filled using the given knowledge base. Next, the question is preprocessed using the Detectors, which match specific keyphrases and are used to match constructs such as aggregates, intervals ("more than 100000 $") and top/bottom-n ("the 5 highest amounts"). Explicit references to aggregates are detected using a manually created mapping based on the question corpus. In the example (Figure 2), there is no explicit aggregate reference, so that the aggregate detector defaults to the sum aggregate. Then, a parse tree is generated using a syntactic tree parse. This tree is then traversed for node matching in pre-order (root-left-right), so that longer phrases have precedence before shorter ones. For example, the phrase "United States of America" can be fully matched to the country or partially matched to the continent, but the full match is assumed to have a higher probability of being the intended reference.

3.2 Node Matching

Each node gets assigned an empty Template Fragment and the phrase which is the part of the question with that node as its root. The phrase is then checked by the scorer of each component property, which depends on the property range, generating a Match Result $m = (N, V)$, as defined in Table 1.

Each component property, that is dimension, attribute or measure, of the data cube has an associated Scorer, which returns a set of possible references to one if its labels or values along with a score between 0 and 1. For example, the word "2007" is treated as a label (of the property) when applied to the

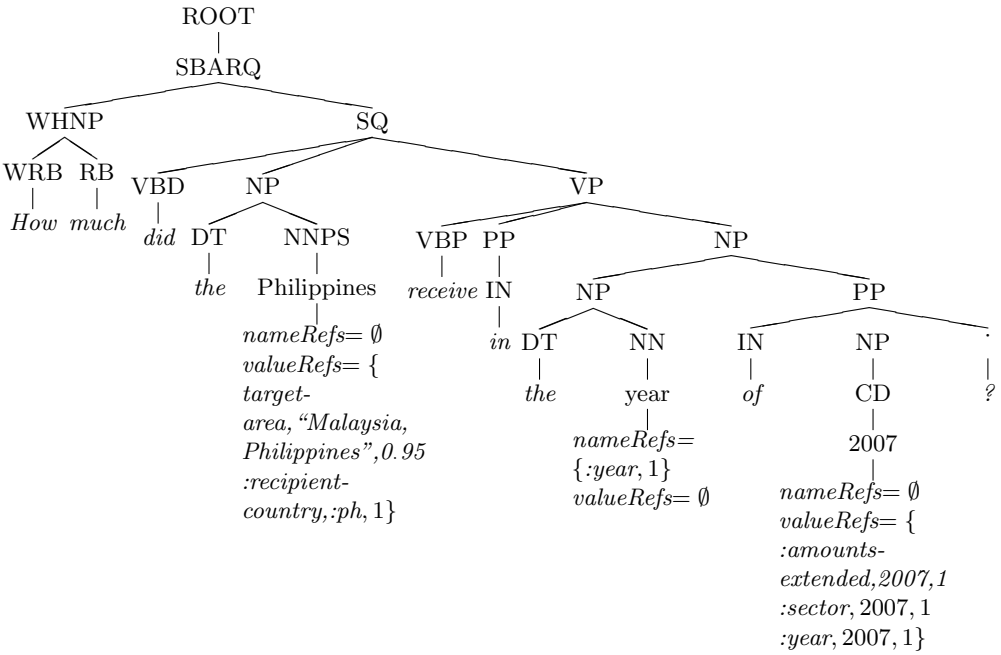

Fig. 2. Parse tree for the question "How much did the Philippines receive in the year of 2007?" along with simplified potential component property references and values.

N	scored component property (name) references	$N \subseteq P \times [0,1]$
V	scored comp. property value references	$V \subseteq P \times L \cup U \times [0,1]$
P	component properties	
L	literals	
U	uris	

Table 1. Definition of a Match Result

Scorer of an object property, as a time interval[8] when applied to a property with range `xsd:date` and as a single year when applied to `xsd:year`. Temporal intervals in the question match enclosing intervals in observations but not the other way around. For example, the phrase "transactions in 2014" matches the observation date of 2014-02-03, but "transactions on 2014-02-03" does not match observations with the year 2014.

For labels and string values, we combine two Apache Lucene indexes with: (1) a standard vector space model with tokenization and stemming to handle word transpositions and different word forms and (2) Levenshtein-Automatas[9] [17] with a maximum edit distance of 2 to handle typing errors. As the indexes

[8] inspired by Allen's Interval Based Temporal Logic [2]
[9] Apache Lucene FuzzyQuery

handle different cases, a candidate gets included in a Template Fragment when found by any single index. The vector space model of the standard Lucene index is more suitable for longer documents, however, and shows some unfavourable properties when used for matching of RDF resources, such as returning hits for partial matches with incomparable score values. We partially mitigate this by discarding all matches with more than twice the length of a phrase.

3.3 Template Fragment Combination

If a node cannot be matched or it has an unmatched phrase, its children are matched and then combined. If a node is matched, its children are skipped as the phrase they represent is a subphrase from the matched phrase. If a node has children with non-empty results from the node matching, those are combined into its own Template Fragment. The results of each matching on a subtree is both a set of scored component properties[10] and a set of scored component property values[11]. While property values also include the property itself, using them immediately is problematic because, contrary to Question Answering on non-numerical data, there is a high amount of ambiguity in statistical values, even for exact matches. In the example in Figure 2, the value "2007" matches several component properties with the maximum score as it is for each of them an exact match. Because of this, those references are not directly included in the restrictions of the Template Fragments. Template Fragments are filters on the cells of the cube, in this case a subtype of restriction: the value restriction besides interval and other restrictions. They are defined is a tuple (N, V, R), with N and V defined as in Table 1 and $R \subseteq P \times 2^V$ is a set of restrictions. The combination function $c(m_1, m_2)$ creates a Template Fragment out of two Match Results in the following way: (1) N and V are unions of N and V from m_1 and m_2 minus property references and values in the restriction and (2) R is the combination of property reference and fitting property value between both match results with highest score product. That is, for each component property, a pair of the highest scored name and value reference to that property is searched. If it exists, and both references are not from the same phrase, a new restriction is formed, that restricts that property to a referenced value. In the running example in Figure 2, the name reference (:year,1) is combined with the value reference (:year,2007,1) to form the value restriction (:year,2007). Both match results are then removed from their fragments. After this process has been done for each property, the leftover references are added to the nodes own fragment. Combining in this order has the advantage of combining close elements in the tree first which are assumed to have a higher chance of being related.

In the parse tree, some references are not represented exactly by a phrase of a subtree, but by parts from different subtrees. This impedes the combination and recognition of those references. We salvage some of these cases by joining the non-matched parts of a phrase of a node and creating a pseudo-node out of the joined text.

[10] such as (:year,1) in Figure 2

[11] such as (target-area,"Malaysia, Philippines",0.95) in Figure 2

Measure Property	:aid-amount
Restrictions	(:year,2007),(:recipient-country,:ph)
Aggregate	sum

SPARQL Query	`select SUM(xsd:decimal(?v1))`
	`{`
	`  ?obs a qb:Observation.`
	`  ?obs :recipient-country :ph.`
	`  ?obs qb:dataSet :finland-aid.`
	`  ?obs :refYear ?v0.`
	`  filter(year(?v0)=2007).`
	`  ?obs :finland-aid-amount ?v1.`
	`}`

Table 2. Query Template and SPARQL query resulting from the combination in the root step of Figure 2.

3.4 SPARQL query generation

When this recursive process reaches the root node, the Template Fragment that results from the successive combination up to that point is transformed into a template, whereby all leftover value references whose property has not been referenced yet over a certain score threshold are transformed into additional restrictions. All other name and value references are discarded and the restrictions as well as the aggreagate, if available, are used to construct a SPARQL query which forms the output of the process. If the resulting template does not include a measure, the default measure of the dataset is used. In the example in Figure 2, (:recipient-country,:ph) is such a leftover value and no measure is given, which results in the template and SPARQL query shown in Table 2.

The algorithm implementation is publicly available under an open license at `https://github.com/AKSW/cubeqa`.

4 Evaluation

4.1 Research Questions

The goal of the evaluation was to obtain answers to the following research questions:

Q1: Is TCQA powerful enough to be practically useful on challenging statistical questions?

Q2: Is there a tendency towards either high precision or recall?

Q3: How do other QA systems perform when applied to statistical data?

Q4: What types of errors occur? How frequently are they? What are the reasons for those?

4.2 Experimental Setup / Benchmark

question word	expected answer type	f
what	any	28
how much	quantity (uncountable)	26
which	equivalent to "what"	17
how many	quantity (countable)	13
when	time point or interval	10
is, do, ...	"yes" or "no"	4
where	location or purpose	1
none (statement)	any	1
total		100

Table 3. Frequency of question words in the benchmark.

As there was no existing benchmark for this task, we created a benchmark based on the statistical question corpus compiled in previous work [10], in which we collected questions from users. We used this basic corpus and significantly extended it to 100 questions. While keeping a similar structure and question word distribution (see Table 3), we adapted it to one specific dataset, which details foreign aid from Finland over several years to different countries.[12] The dataset contains 4 dimensions, 16 measures, 11 attributes, 13 750 observations and 467 367 triples. To stimulate future research and more elaborate approaches, the benchmark contains difficult queries and provides several challenges. The challenges include implied aggregations, intervals, implied or differently referenced measures and numerical values which are contained in several component properties, which are supported by the TCQA algorithm, as well as other challenges, such as questions requiring SPARQL subqueries, which are not supported.

Precision p and recall r are defined as follows, where G is the set of resources given by the gold standard of the benchmark and O is the output of the algorithm:

$$p = \frac{|G| \cap |O|}{|O|} \qquad r = \frac{|G| \cap |O|}{|G|}$$

Results Of the 100 questions, 18 contain constructs that are not supported by TCQA: 6 contained unions which result from logical operators, 8 require subqueries and 4 are affirmative questions. Of the remaining 82 questions, TCQA achieves an average precision of 0.286, an average recall of 0.195 and a macro f-score of 0.232, see Table 4. Without manually created stopwords, the F-score

[12] `http://linkedspending.aksw.org/resource/properties/?r=http://linkedspending.aksw.org/instance/finland-aid`

deceases by 0.015 and the removal of the default measure further reduces this by 0.129. This shows that minimal manual adaption of the algorithm to a dataset is necessary—primarily the selection of a default measure, which is often only implicitly mentioned. More time-intensive customizations are not required to set up the algorithm.

Algorithm	Average Precision	Average Recall	Macro F-Score
TCQA, default measure, custom stopwords	0.286	0.195	0.232
TCQA, default measure, no custom stopwords	0.267	0.183	0.217
TCQA, no default measure, no custom stopwords	0.109	0.073	0.088
SINA	0.000	0.000	0.000
TBSL	0.000	0.000	0.000
OpenQA SINA/TBSL Combinator	0.000	0.000	0.000

Table 4. Benchmark performance of different TCQA configurations and the OpenQA framework, indicated by average precision, average recall and macro f-score (harmonic mean of average precision and average recall). All values are rounded to 3 decimal places.

To evaluate the performance of existing approaches on statistical RDF data, we used the OpenQA [14] framework, which runs and evaluates different algorithms. However both available algorithms, SINA [18] and TBSL [23] as well as combinations of their components, could not answer a single question of the benchmark. This is mainly due to the complexity of the questions in the benchmark and the previously described different nature of the questions. We believe that despite the low F-score value, TCQA will be a strong baseline in this new research field. In the future work section, we outline how we believe further improvements can be made and want to achieve community wide engagement on those advances via QALD.

Running the 100 benchmark questions took 46 seconds (averaged over three different runs with very low standard deviation) on an Intel Core i5-3230M CPU @ 2.60GHz with 8 GB RAM which hosted both the SPARQL endpoint and the system implementation (the combined RAM usage was less than 600 MB).

4.3 Discussion

Table 5 categorizes the different errors that prevented TCQA from returning a correct result to a benchmark question. The most common error is the structural error, where a feature needed to successfully answer a question is either not

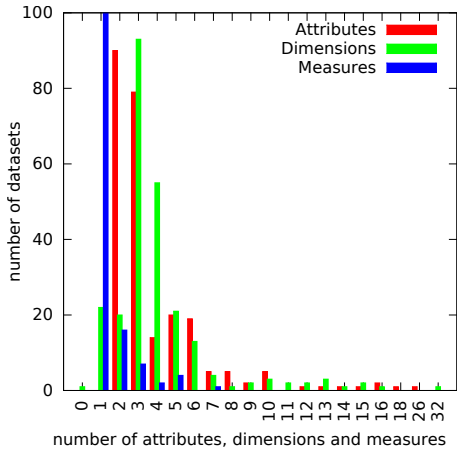

Fig. 3. Histogram of measures, attributes and dimensions of the LinkedSpending datasets, version 0.1. 217 datasets have exactly one measure (clipped bar). Originally published for the LinkedSpending project [11].

supported or not correctly applied, for example "Which sectors have Paraguay and the Central African Republic in common?", where the concept of having something in common has to be recognized, and "Which countries received more than they were committed?" where two entities have to be compared.

The second most common cause is ambiguity, which mainly results from the high number of similar resources or equal numbers in the observation values. Common approaches are to associate query templates with a score, so that the user chooses amongst the top n (used by the TBSL algorithm [23]) and to prefer candidate combinations that maximize textual and semantic relatedness between the candidates [19]. Those approaches are not directly applicable to TCQA, however, because of the locality based combination step. Instead, TCQA relys on references consisting of a name reference as well as a value reference, as in "the year 2008", where the name-value pair with the maximal score product of the name reference and the value reference is chosen, see Section 3. In case such a two-part reference does not occur, it is alleviated by giving temporal dimensions priority to others, so that for example "2008" gets mapped to the year, if it exists, rather than the a more improbable measurement value. In the question "How much money Embassy of Finland contribute to Egyptian projects?", the parser does not isolate the phrase "Embassy of Finland", so that instead "Finland" gets found before the phrase gets constructed during the combination of left-over fragments.

The distance between a surface form in a question and the labels of the concept, to which the surface form refers, is called the lexical gap. It is caused, among others, by different capitalization, typing errors, word transpositions ("extended amount", "amount extended") and different word forms ("committed", "com-

error cause	occurrences	description
structure	32	query structure not appropriately recognized
ambiguity	19	wrong choice amongst the index results
lexicalgap	11	correct resource not in index results
subquery	8	query requires subqueries, which are not supported
union	6	query requires unions, which are not supported
affirmative	4	yes/no question, which is not supported
recalculation	3	query requires application of function on result
no error	17	
total errors	83	

Table 5. Categorization of errors from the different benchmark questions (at most one error per question), including the categories automatically excluded before the evaluation.

mitments"). All of those, except typing errors, occur in the benchmark. As these mentioned causes occur in document retrieval and Web search as well, full text indexes have been developed that robustly handle those problems. We adopt two Apache Lucene indexes as described in Section 3. The previous example also shows the limitations of the hybrid index to overcome the lexical gap, as "Egyptian" is not mapped to "Egypt" because the stemmer transform it and the edit distance is too large for the fuzzy index as well. Sometimes a concept is implicitly required but there is no explicit reference at all, for example in "How much does Uruguay receive", benchmark question 13[13]. Implicit references are part of future work (see Section 5) and include aggregates, as most benchmark questions (58 of 100) require summation of all selected values. Thus, summation is the aggregation method selected as default. Another issue with the lexical gap is that a measurement can be referenced using a quantity reference ("amount"), a unit ("How many dollars are given"), or a the type ("aid"), of which only the first one guarantees a match. TCQA matches the range of a property as well as a its label. As this problem mainly occurs on measures, and our query model requires the use of a least one measure, our fallback is the specification of a default measure for each dataset. Figure 3 shows that most datasets of the LinkedSpending dataset only have one measure, where the fallback cannot misidentify. The RDF Data Cube vocabulary provides `sdmx-attribute:unitMeasure` to specify units of measurement, but it does not support multiple measures so that the fallback has the same effect. In case of future vocabulary specification updates, we plan to integrate measurement units into our approach.

Queries requiring subqueries and unions as well as affirmative questions are the next common error causes which are not supported yet by TCQA.

[13] `https://github.com/AKSW/cubeqa/blob/master/benchmark/finland-aid.xml`

Three questions require recalculation of its results, for example "How much aid is received by Zambia on a single day?" where the queriable yearly aid has to be divided by the average numbers of days in a year.

4.4 Research Question Summary

A brief summary of the initial research questions is as follows:

Q1: TCQA is not yet sufficiently powerful to be applied on challenging questions over statistical data, but we believe it will be a strong baseline for future research.

Q2: There is a tendency towards higher precision than recall, which however can be adjusted via the NLP components in the pipeline.

Q3: Other systems, two of which we included in our evaluation, are not close to being able to answer even one of the questions. This result is unsurprising: From a separate (yet unpublished) systematic survey of all QA systems published from 2011 onwards, it is evident that none of the current systems is suitable for this task.

Q4: The most common cause for problems is the incorrect detection of the query structure, followed by ambiguity, the lexical gap and various unsupported language features.

5 Conclusions and Future Work

To the best of our knowledge, we provide the first approach for Question Answering over statistical Linked Data. We described the underlying algorithm, created a benchmark based on real data and provided results and a discussion on the specifics of the field. In future work, we aim to advance along the following lines:

1. Integrating the Benchmark with the QALD Benchmark Challenge: In order to stimulate and measure progress in this new research sub-field, the benchmark developed for this work has already been approved for inclusion in the established QALD Question Answering benchmark.

2. Integration into Generic Question Answering Algorithms: Generic Question Answering algorithms can provide necessary world knowledge that is not available in the data cube, for example to determine which countries are part of Europe in the question "Which European countries receive the most development aid?". We will investigate how to seamlessly integrate statistical QA techniques into generic QA algorithms, such as into the OpenQA [14] framework.

3. Cross Dataset Queries and Detailed User Study: While the 100-question corpus we used for his work allows to draw basic conclusions about typical questions and has the same size as other benchmarks in QALD, we plan to launch another questionnaire with a significantly larger user base, in particular to unveil further less frequent patterns in statistical queries and reduce the size of the confidence interval for measurements against the corpus. Moreover, we plan to extend our algorithm to cover the case of queries against a set of datasets (in contrast to a single dataset), for which we envision two possible scenarios: 1.) The QA algorithm needs to decide which dataset is most relevant for the question and run the query against it. 2.) The QA algorithm needs to be capable to produce queries against multiple datasets, i.e. joining statistical information on the fly. This is particularily important in this field as statistical information of similar type is frequently obtained from several independant sources with overlapping but not identical schema use, e.g. different municipalities or countries publishing E-Government information.

4. Improvements of the TCQA algorithm We identified the following approaches for improvement of the TCQA algorithm

- Implement selection filters as logical formula of restrictions instead of flat sets, including negations and unions.
- Support SPARQL subqueries to answer queries with nested information dependencies.
- Detect implied references and implied aggregation.
- Add inference over the time dimension based on CNTRO [21].
- Support languages other than English by integrating a language detection component as well as fitting parsers and indexes.
- Incorporate measurement units if RDF Data Cube vocabulary adds support for them for multiple measures. For elaborate phrase patterns, like "How many people live in" for "population", there are pattern libraries like BOA [9] which need to be adapted to statistical data by retraining on a comprehensive statistical question corpus.

Overall, we believe to have opened an interesting field of research and practise, which will increase in importance due to the rise of both the volume of statistical data and the usage of Question Answering approaches in everyday life.

References

1. The RDF Data Cube Vocabulary. Technical report, W3C, 2013.
2. J. F. Allen. Maintaining knowledge about temporal intervals. *Communications of the ACM*, 26(11):832–843, 1983.
3. S. Athenikos and H. Han. Biomedical Question Answering: A survey. *Computer methods and programs in biomedicine*, 99(1):1–24, 2010.
4. H. Bast, F. Bäurle, B. Buchhold, and E. Haussmann. Broccoli: Semantic full-text search at your fingertips. *arXiv preprint arXiv:1207.2615*, 2012.
5. P. Cimiano, V. Lopez, C. Unger, E. Cabrio, A.-C. Ngonga Ngomo, and S. Walter. Multilingual Question Answering over linked data (QALD-3): Lab overview. In *CLEF*, pages 321–332, 2013.
6. P. Cimiano and M. Minock. Natural language interfaces: What is the problem?—a data-driven quantitative analysis. In *Natural Language Processing and Information Systems, 15th International Conference on Applications of Natural Language to Information Systems, NLDB 2010*, volume 6177 of *Lecture Notes in Computer Science*, pages 192–206, Berlin Heidelberg, Germany, 2010. Springer.
7. C. Dima. Intui2: A prototype system for Question Answering over Linked Data. In P. Forner, R. Navigli, and D. Tufis, editors, *Question Answering over Linked Data (QALD-3), CLEF 2013 Evaluation Labs and Workshop, Online Working Notes*, 2013.
8. A. Freitas, E. Curry, J. Oliveira, and S. O'Riain. Querying heterogeneous datasets on the Linked Data Web: Challenges, approaches, and trends. *IEEE Internet Computing*, 16(1):24–33, Jan 2012.
9. D. Gerber and A.-C. N. Ngomo. Extracting multilingual natural-language patterns for RDF predicates. In *Knowledge Engineering and Knowledge Management*, pages 87–96. Springer, 2012.
10. K. Höffner and J. Lehmann. Towards Question Answering on statistical linked data. In *Proceedings of the 10th International Conference on Semantic Systems*, SEM '14, pages 61–64. ACM, 2014.
11. K. Höffner, M. Martin, and J. Lehmann. LinkedSpending: OpenSpending becomes Linked Open Data. *Semantic Web Journal*, 2015.
12. V. Lopez, C. Unger, P. Cimiano, and E. Motta. Evaluating Question Answering over Linked Data. *Web Semantics*, 21:3–13, 2013.
13. V. Lopez, V. Uren, M. Sabou, and E. Motta. Is Question Answering fit for the Semantic Web?: A survey. *Semantic Web Journal*, 2(2):125–155, April 2011.
14. E. Marx, R. Usbeck, A.-C. Ngomo Ngonga, K. Höffner, J. Lehmann, and S. Auer. Towards an open Question Answering architecture. In *SEMANTiCS 2014*, 2014.
15. J. W. Murdock, A. Kalyanpur, C. Welty, J. Fan, D. A. Ferrucci, D. Gondek, L. Zhang, and H. Kanayama. Typing candidate answers using type coercion. *IBM Journal of Research and Development*, 56(3.4):7–1, 2012.
16. S. J. Piotrowski and G. G. Van Ryzin. Citizen Attitudes Toward Transparency in Local Government. *The American Review of Public Administration*, 37(3):306–323, 2007.
17. K. U. Schulz and S. Mihov. Fast string correction with Levenshtein automata. *International Journal on Document Analysis and Recognition*, 5(1):67–85, 2002.
18. S. Shekarpour, A.-C. N. Ngomo, and S. Auer. Question Answering on interlinked data. In D. Schwabe, V. A. F. Almeida, H. Glaser, R. A. Baeza-Yates, and S. B. Moon, editors, *WWW*, pages 1145–1156. International World Wide Web Conferences Steering Committee / ACM, 2013.

19. S. Shekarpour, A.-C. Ngonga Ngomo, and S. Auer. Query segmentation and resource disambiguation leveraging background knowledge. In *Proceedings of WoLE Workshop*, 2012.
20. C. Stadler, M. Martin, and S. Auer. Exploring the web of spatial data with Facete. In *Proc. of WWW*, pages 175–178. Int. WWW Conf. Steering Committee, 2014.
21. C. Tao, H. R. Solbrig, D. K. Sharma, W.-Q. Wei, G. K. Savova, and C. G. Chute. Time-oriented Question Answering from clinical narratives using Semantic-Web techniques. In *The Semantic Web–ISWC 2010*, pages 241–256. Springer, Berlin Heidelberg, Germany, 2010.
22. G. Tsatsaronis, M. Schroeder, G. Paliouras, Y. Almirantis, I. Androutsopoulos, E. Gaussier, P. Gallinari, T. Artieres, M. R. Alvers, M. Zschunke, et al. BioASQ: A challenge on large-scale biomedical semantic indexing and Question Answering. In *2012 AAAI Fall Symposium Series*, 2012.
23. C. Unger, L. Bühmann, J. Lehmann, A.-C. Ngonga Ngomo, D. Gerber, and P. Cimiano. Template-based Question Answering over RDF data. In *Proceedings of the 21st international conference on World Wide Web*, pages 639–648, 2012.
24. S. Wolfram. The Mathematica book. *Cambridge University Press and Wolfram Research, Inc., New York, NY, USA and*, 100:61820–7237, 2000.
25. E. M. G. Younis, C. B. Jones, V. Tanasescu, and A. I. Abdelmoty. Hybrid geo-spatial query methods on the Semantic Web with a spatially-enhanced index of DBpedia. In *GIScience*, pages 340–353, 2012.