Question Answering on Scholarly Knowledge Graphs

Mohamad Yaser Jaradeh[1(✉)], Markus Stocker[2], and Sören Auer[1,2]

[1] L3S Research Center, Leibniz University of Hannover, Hanover, Germany
jaradeh@l3s.de
[2] TIB Leibniz Information Centre for Science and Technology, Hanover, Germany
{markus.stocker,auer}@tib.eu

Abstract. Answering questions on scholarly knowledge comprising text and other artifacts is a vital part of any research life cycle. Querying scholarly knowledge and retrieving suitable answers is currently hardly possible due to the following primary reason: machine inactionable, ambiguous and unstructured content in publications. We present JarvisQA, a BERT based system to answer questions on tabular views of scholarly knowledge graphs. Such tables can be found in a variety of shapes in the scholarly literature (e.g., surveys, comparisons or results). Our system can retrieve direct answers to a variety of different questions asked on tabular data in articles. Furthermore, we present a preliminary dataset of related tables and a corresponding set of natural language questions. This dataset is used as a benchmark for our system and can be reused by others. Additionally, JarvisQA is evaluated on two datasets against other baselines and shows an improvement of two to three folds in performance compared to related methods.

Keywords: Digital Libraries · Information retrieval · Question Answering · Semantic web · Semantic search · Scholarly knowledge

1 Introduction

Question Answering (QA) systems, such as Apple's Siri, Amazon's Alexa, or Google Now, answer questions by mining the answers from unstructured text corpora or open domain Knowledge Graphs (KG) [14]. The direct applicability of these approaches to specialized domains such as scholarly knowledge is questionable. On the one hand, no extensive knowledge graph for scholarly knowledge exists that can be employed in a question answering system. On the other hand, scholarly knowledge is represented mainly as unstructured raw text in articles (in proceedings or journals) [3]. In unstructured artifacts, knowledge is not machine actionable, hardly processable, ambiguous [4], and particularly also not FAIR [32]. Still, amid unstructured information some semi-structured information exists, in particular in tabular representations (e.g., survey tables, literature overviews, and paper comparisons). The task of QA on tabular data has challenges [18], shared with other types of question answering systems. We propose

M. Hall et al. (Eds.): TPDL 2020, LNCS 12246, pp. 19–32, 2020.
https://doi.org/10.1007/978-3-030-54956-5_2

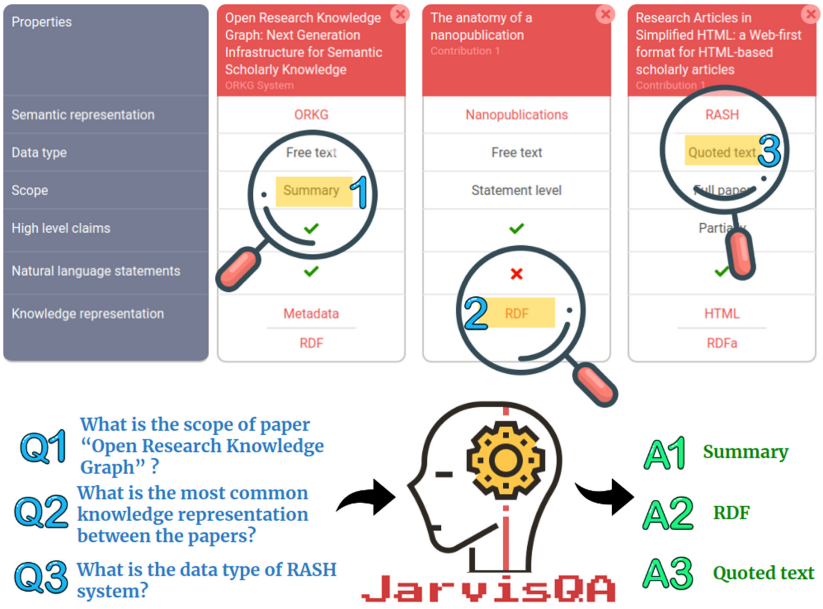

Fig. 1. Motivating Example. `JarvisQA` takes as input a table of semi-structured information and tries to answer questions. Three types of questions are depicted here. (**Q1**) Answer is directly correlated with the question. (**Q2**) Aggregation of information from candidate results. (**Q3**) Answer relates to another cell in the table.

a method to perform QA specifically on scholarly knowledge graphs representing tabular data. Moreover, we create a benchmark of tabular data retrieved from a scholarly knowledge graph and a set of related questions. This benchmark is collected using the Open Research Knowledge Graph (ORKG) [12].

The remainder of this article is structured as follows. Section 1 motivates the work with an example. Section 2 presents related work, which is supplemented by an analysis of the strengths and weaknesses of existing systems in the context of digital libraries. Section 3 describes the proposed approach. Section 4 presents the implementation and evaluation. Section 5 discusses results and future work. Finally, Sect. 6 concludes the paper.

Motivating Example. The research community has proposed many QA systems, but to the best of our knowledge none focus on scholarly knowledge. Leveraging the ORKG [12] and its structured scholarly knowledge, we propose a QA system specifically designed for this domain. Figure 1 illustrates a tabular comparison view[1] of structured scholarly contribution descriptions. Additionally, three questions related to the content of the comparison table are shown. The answers are implicitly or explicitly provided in the cells of the table. `JarvisQA`

[1] https://www.orkg.org/orkg/comparison/R8618.

can answer different types of questions. For Q1, the answer has a direct correlation with the question. For Q2, the system should first find the "knowledge representations" in the table and then find the most common value. For Q3, the answer is conditional upon finding another piece of information in the table first (i.e., JarvisQA has to find "RASH" in the table first), and then narrow its search to that column (or that paper) to find the correct answer.

We tackle the following research questions:

- **RQ1:** *Can a QA system retrieve answers from tabular representations of scholarly knowledge?*
- **RQ2:** *What type of questions can be posed on tabular scholarly knowledge?*

2 Related Work

Question answering is an important research problem frequently tackled by research communities in different variations, applications, and directions.

In open domain question answering, various systems and techniques have been proposed that rely on different forms of background knowledge. Pipeline-based systems, such as OpenQA [20], present a modular framework using standardized components for creating QA systems on structured knowledge graphs (e.g., DBpedia [1]). Frankenstein [28] creates the most suitable QA pipeline out of community created components based on the natural language input question. QAnswer [8] is a multilingual QA system that queries different linked open data datasets to fetch correct answers. Diefenbach et al. [7] discussed and compared other QA-over-KG systems (e.g., gAnswer [38], DEANNA [34], and SINA [27]) within the context of QALD "Question Answering over Linked Data" challanges [19].

Other types of QA systems rely on the raw unstructured text to produce the answers. Many of these systems are end-to-end systems that employ machine learning to mine the text and retrieve the answers. Deep learning models (e.g., Transformers) are trained and fine-tuned on certain QA datasets to find the answers from within the text. ALBERT [17] is a descendent of BERT [6] deep learning model. At the time of writing, ALBERT holds the third top position in answering the questions of SQuAD [24]. Such techniques model the linguistic knowledge from textual details and discard all the clutter in the text [37]. Other similar approaches include SG-Net [36], which uses syntax rules to guide the machine comprehension encoder-transformer models.

Tabular QA systems are also diverse and tackle the task with different techniques. TF-IDF [25] is used to extract features from the tables and the question, and to match them. Other models such as semantic parsers are used by Kwiatkowski et al. [16] and Krishnamurthy and Kollar [15]. Cheng et al. [5] propose a neural semantic parser that uses predicate-argument structures to convert natural language text into intermediate structured representations, which are then mapped to different target domains (e.g., SQL).

Another category of table QA systems is neural systems. TableQA [30] uses end-to-end memory networks to find a suitable cell in the table to choose.

Wang et al. [31] propose to use a directional self-attention network to find candidate tables and then use BiGRUs to score the answers. Other table oriented QA systems include HILDB [9] that converts natural language into SQL.

In the plethora of systems that the community has developed over the past decade, no system addresses the scholarly information domain, specifically. We propose a system to fill this gap and address the issues of QA on scholarly tabular data in the context of digital libraries (specifically with the ORKG[2]).

Though a variety of QA techniques exist, Digital Libraries (DL) primarily rely on standard information retrieval techniques [26]. We briefly analyze and show when and how QA techniques can be used to improve information retrieval and search capabilities in the context of DLs. Since DLs have different needs [11,26]; QA systems can improve information retrieval availability [2]. We argue that, Knowledge Graph based QA systems (or KG-QA) can work nicely within a DL context (i.e., aggregate information, list candidate answers). Nevertheless, the majority of the existing scholarly KGs (such as MAG [29], OC [23]) focus on metadata (e.g., authors, venues, and citations), not the scholarly knowledge content.

Another category of QA systems works on raw text, an important approach for DLs. However, such systems are not fine-tuned on scholarly data; rather, they are designed for open domain data. Furthermore, many of the end-to-end neural models have a built-in limitation [35] (i.e., model capacity) due to the architecture type, and as such cannot be used out of the box. Some systems circumvent the problem of capacity (i.e., the inability to feed the model large amounts of text) by having a component of indexing (e.g., inverted index, concept and entity recognition) that can narrow down the amount of text that the system needs to process as the context for questions.

3 Approach

We propose a system, called `JarvisQA`, that answers Natural Language (NL) questions on tabular views of scholarly knowledge graphs, specifically tabular views comprising research contribution information from scientific articles.

3.1 Data and Questions Collection

In order to evaluate our QA system we create the ORKG-QA benchmark, collected using the ORKG. The ORKG provides structured comparisons [21] of research contributions obtained from papers. The ORKG-QA benchmark comprises a dataset that integrates 13 tables, covering information spanning more than 100 academic publications. The data is collected through the ORKG API and the featured set of tables[3], which can be exported in CSV format.

Additionally, we created a set of questions that cover various types of information and facts that can be retrieved from those tables. The benchmark consists

[2] https://orkg.org/.
[3] https://www.orkg.org/orkg/featured-comparisons.

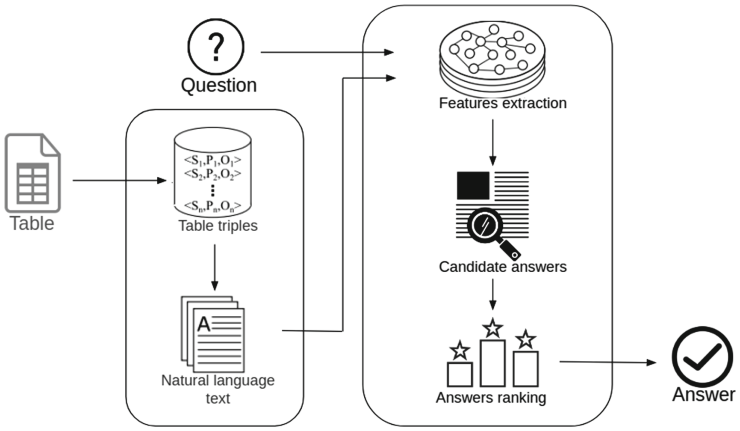

Fig. 2. System Architecture. JarvisQA was designed with modularity in mind. The system has two main components. (a) **Table 2Text (T2T)** component, which in turn has two functionalities: (1) to break the table into a set of triples $< s, p, o >$ and (2) to compile the triples into an NL sentence. Component (b) is the **engine of the QA system**, where an NL QA (BERT based) system is employed to answer the input question using the text, by extracting features, finding candidate answers, and ranking them.

of 80 questions in English. The questions cover a variety of question types that can be asked in the context of tables in the scholarly literature. These types of questions include aggregation questions (e.g., min, average and most common), ask questions (i.e., true, false), answer listing questions, and questions that rely on combining information. In the ORKG-QA dataset[4], 39% are normal questions addressing individual cells in tables, 20% are aggregation questions, 11% are questions for which the answer relates to other parts of the table, and the rest are questions of different types (i.e., listings, ask queries, empty answers).

We also use the TabMCQ [13] QA dataset, specifically questions on the *regents* tables. TabMCQ was derived from multiple choice questions of 4th grade science exams and contains 39 tables and 3 745 related questions. While TabMCQ is not a scholarly dataset, but is to the best of our knowledge the closest one available. Since TabMCQ has only multiple-choice questions, we adapted the questions with only the correct choice.

3.2 JarvisQA System Architecture

`JarvisQA` is designed with modularity in mind. Hence, the core QA components are replaceable with newer or more fine-tuned versions. Figure 2 depicts the architecture in more detail. Since we used a natural language QA system, we need a pre-processing step that transforms the table information into the textual description (representing only the information contained in the table not the

[4] https://doi.org/10.25835/0038751.

entire raw text of the article). With the output of the "Table2Text" step and the input question, the NL QA system can reason over the question with the provided context (textual table description) and attempts to answer the question. We now discuss the individual components of the architecture in more detail.

Table 1. Sample of an input table. The table is a part of the one shown in the motivating example.[7] Below, the representation in triples and as text is displayed.

Title	Semantic representation	Data type	Scope	High level claims
Paper 1 [12]	ORKG	Free text	Summary	Yes
Paper 2 [10]	Nanopublications	Free text	Statement level	Yes
Paper 3 [22]	RASH	Quoted text	Full paper	Partially
Triples	<Paper1, hasSemanticRepresentation, ORKG> <Paper1, hasDataType, FreeText> <Paper1, hasScope, Summary> ...			
Text	Paper 1's semantic representation is "ORKG", its data type is "Free Text", and its scope is "Summary" ...			

Table2Text (T2T) Converter. Although `JarvisQA` operates on tabular data, the core QA engine processes textual contexts. To that end, tables have to be converted into coherent text snippets that represent the entirety of the information presented in the table. T2T component splits tables into its entries and converts entries into triples. Table 1 illustrates a sample table containing some information about three publications, along with their triples and textual representations compiled by the T2T component. Furthermore, the T2T component enriches the textual description with aggregated information (i.e., max value of certain rows, most common value used within some columns). This enables the system to answer aggregation-type questions such as "Which system has the maximum accuracy?" and "What is the most common method used among the papers?".

QA Core Engine. This component is the primary building block of `JarvisQA`. It is where reasoning over questions happens. The component uses a pre-trained natural language QA model. The model is a deep transformer, fine tuned on the SQuADv2 dataset to perform the QA task. The component is replaceable with any other similar transformer model (of different sizes and architectures). Our base implementation uses a fine tuned version of a BERT model and we evaluate our model using different model sizes and architectures. The model parameters are set: *maximum sequence length* to 512, *document stride* to 128, *top k answers* to 10, *maximum answer length* to 15, and the *maximum question length* to 64. As illustrated in Fig. 2, the QA engine extracts sets of features from the questions and the text (i.e., embeddings), then it finds a set of candidate answers and ranks them by confidence score. The benefits of such architecture

Table 2. Evaluation metrics used to experimentally benchmark JarvisQA against other baselines.

Metric	Definition
Global Precision	Ratio between correct answers retrieved in the top ranked position and the total number of questions
Global Recall	Ratio between the number of questions answered correctly at any position (here till the 10th retrieved answer) and the total number of questions
F1-Score	Harmonic mean of global precision and global recall
Execution Time	Elapsed time between asking a question and returning the answer
Inv. Time	$1 - \frac{average\ execution\ time\ for\ baseline}{maximum\ execution\ time\ for\ all\ systems}$
In-Memory Size	The total memory size used by system
Inv. Memory	$1 - \frac{memory\ size\ of\ baseline}{maximum\ memory\ size\ among\ all\ systems}$
Precision@K	Cumulative precision at position K
Recall@K	Ratio of correctly answered questions in the top K position and total number of questions
F1-Score@K	Harmonic mean of precision and recall at position K

are the flexibility in model choice, multilingualism, and reusability. Different transformer models can replace ours to support other languages, other datasets, and potentially other features. To accomplish these objectives, the system is built using the Transformers framework [33].

4 Experimental Study

We empirically study the behavior of `JarvisQA` in the context of scholarly tables against different baselines. The experimental setup consists of metrics and baselines. Table 2 lists the evaluation metrics for the performance measurements of the systems. Since a QA system can produce multiple answers and the correct answer can be any of the retrieved answers we use a metric that takes the position of the answer into account.
As baselines we use the following two methods for answer generation:

- *Random*: the answer is selected from all choices randomly.
- *Lucene*[8]: is a platform for indexing, retrieving unstructured information, and used as a search engine. We index the triple-generated sentences by Lucene. For each question, the top answer produced by Lucene is regarded as the final answer.

[8] https://lucene.apache.org/.

Table 3. JarvisQA performance on the ORKG-QA benchmark dataset of tabular data. The evaluation metrics are precision, recall, and F1-score at k position. JarvisQA is compared against two baselines on the overall dataset and specific question types. The symbol (-) indicates that the performance metric showed no difference than the reported value for higher K values. The results suggest that JarvisQA outperforms the baselines by 2–3 folds.

Questions type	Baseline	Precision @K				Recall @K				F1-Score @K			
		#1	#3	#5	#10	#1	#3	#5	#10	#1	#3	#5	#10
All	Random	0.02	0.06	0.08	0.16	0.02	0.07	0.09	0.18	0.02	0.06	0.08	0.17
All	Lucene	0.09	0.19	0.20	0.25	0.09	0.18	0.19	0.24	0.09	0.18	0.19	0.24
Normal	JarvisQA	0.41	0.47	0.55	0.61	0.41	0.47	0.53	0.61	0.41	0.47	0.54	0.61
Aggregation	JarvisQA	0.45	-	-	-	0.45	-	-	-	0.45	-	-	-
Related	JarvisQA	0.50	0.50	1.00	1.00	0.50	0.50	1.00	1.00	0.50	0.500	1.00	1.00
Similar	JarvisQA	0.11	0.25	0.67	-	0.11	0.25	0.67	-	0.11	0.25	0.67	-
All	JarvisQA	0.34	0.38	0.46	**0.47**	0.35	0.38	0.46	**0.48**	0.34	0.38	0.45	**0.47**

The evaluation was performed on an Ubuntu 18.04 machine with 128 GB RAM and a 12 core Xeon processor. The implementation is mostly based on HuggingFace Transformers[9], and is written in Python 3.7. The evaluation results for precision, recall, and F1-score are reproducible while other metrics such as time and memory depend on the evaluation system hardware. However, the ratio of the difference between the baselines should be similar or at least show a similar trend. The code to reproduce the evaluation results and the presented results are available online.[10]

Experiment 1 - JarvisQA Performance on the ORKG-QA Benchmark. In order to evaluate the performance of `JarvisQA`, we run the system and other baselines on the ORKG-QA dataset at various k values (k denotes the position of the correct answer among all retrieved answers). For this experiment we evaluate $k \in \{1, 3, 5, 10\}$. Moreover, the experiment was conducted on a specific subset of questions (based on types) to show the performance of the system for certain categories of questions. The tested question categories are: *Normal*: normal questions about a specific cell in the table with a direct answer; *Aggregation*: questions about aggregation tasks on top of the table; *Related*: questions that require retrieving the answer from another cell in the table; *Similar*: questions that address the table using similar properties (e.g., synonyms). Table 3 shows the performance of the baselines and our system on the ORKG-QA benchmark. The results show that `JarvisQA` performs better by 2–3 folds against Lucene, and Random baselines respectively.

[9] https://github.com/huggingface/transformers.
[10] https://doi.org/10.5281/zenodo.3738666.

Experiment 2 - Different Models of QA and Their Performance. We evaluate different types of QA models simultaneously to show the difference in performance metrics, execution time, and resource usage. Table 4 illustrates the difference in performance on the ORKG-QA benchmark dataset for different classes of questions and the overall dataset. `JarvisQA`'s QA engine employs the `BERT L/U/S2` model due to its execution time and overall higher accuracy at higher positions.

Table 4. Performance comparison of different deep learning models on the task of question answering with different model sizes and architectures using the ORKG-QA benchmark dataset. The results suggest that different models perform differently on various question types, and generally the bigger the model the better it performs. For each question type, the best results are highlighted.

	Questions type	Precision @K				Recall @K				F1-Score @K			
		#1	#3	#5	#10	#1	#3	#5	#10	#1	#3	#5	#10
BERT L/U/S1	Normal	0.35	0.49	0.53	**0.68**	0.34	0.47	0.51	**0.67**	0.34	0.48	0.52	**0.67**
	Aggregation	0.39	0.39	0.45	-	0.39	0.39	0.45	-	0.39	0.39	0.45	-
	Related	0.50	0.64	0.64	0.80	0.50	0.64	0.64	0.80	0.50	0.64	0.64	0.80
	Similar	0.11	0.25	**0.67**	-	0.11	0.25	**0.67**	-	0.11	0.25	**0.67**	-
	All	0.31	0.38	0.44	**0.50**	0.31	0.38	0.43	**0.49**	0.3	0.38	0.43	**0.50**
BERT L/C/S1	Normal	0.31	0.44	0.45	-	0.31	0.43	0.45	-	0.31	0.43	0.45	-
	Aggregation	0.27	0.39	0.39	0.45	0.29	0.39	0.39	0.45	0.27	0.39	0.39	0.45
	Related	0.65	**1.00**	-	-	0.70	**1.00**	-	-	0.67	**1.00**	-	-
	Similar	0.11	0.11	0.25	0.43	0.11	0.11	0.25	0.43	0.11	0.11	0.25	0.43
	All	0.27	0.35	0.37	0.39	0.29	0.37	0.39	0.41	0.27	0.36	0.37	0.40
BERT L/U/S2	Normal	0.41	0.47	0.55	0.61	0.41	0.47	0.54	0.61	0.41	0.47	0.54	0.61
	Aggregation	0.45	-	-	-	0.45	-	-	-	0.45	-	-	-
	Related	0.50	0.50	**1.00**	-	0.50	0.50	**1.00**	-	0.50	0.50	**1.00**	-
	Similar	0.11	0.25	**0.67**	-	0.11	0.25	**0.67**	-	0.11	0.25	**0.67**	-
	All	0.35	0.38	0.46	0.48	0.35	0.38	0.46	0.48	0.34	0.38	0.46	0.48
Distil BERT B/U/S1	Normal	0.14	0.27	0.36	0.46	0.16	0.29	0.36	0.46	0.15	0.27	0.35	0.45
	Aggregation	0.22	0.39	-	-	0.25	0.41	-	-	0.24	0.39	-	-
	Related	0.31	0.50	0.64	-	0.31	0.50	0.64	-	0.31	0.50	0.64	-
	Similar	0.00	-	-	-	0.00	-	-	-	0.00	-	-	-
	All	0.16	0.23	0.28	0.33	0.17	0.26	0.29	0.35	0.16	0.24	0.28	0.33
ALBERT XL/S2	Normal	0.34	0.47	0.51	-	0.34	0.47	0.51	-	0.34	0.47	0.51	-
	Aggregation	0.45	0.45	**0.52**	-	0.45	0.45	**0.52**	-	0.45	0.45	**0.52**	-
	Related	**1.00**	-	-	-	**1.00**	-	-	-	**1.00**	-	-	-
	Similar	0.43	0.43	**0.67**	-	0.43	0.43	**0.67**	-	0.43	0.43	**0.67**	-
	All	0.36	0.42	0.46	-	0.37	0.43	0.47	-	0.36	0.42	0.46	-

B = Base; L = Large; XL = X-Large; C = Cased; U = Uncased; S1 = Finetuned on SQuAD1; S2 = Finetuned on SQuAD2

Experiment 3 - Trade-Offs Between Different Performance Metrics. We illustrate trade-offs between different dimensions of performance metrics for the `JarvisQA` approach compared to the baselines. We choose global precision, global recall, F1-score, in-memory size, and execution time as five different dimensions. Figure 3 depicts the performance metrics trade-offs between our system and other baselines. `JarvisQA` achieves higher precision and recall while consuming considerably more time and memory than the other baselines.

Experiment 4 - Performance on TabMCQ. We also show the performance of our system on the TabMCQ dataset against the ORKG-QA dataset. We see the same trend in both datasets, that `JarvisQA` outperforms the baselines by many folds. TabMCQ is not directly related to scholarly knowledge. However, it shows that `JarvisQA` can generalize to related data and can answer questions about it. Table 5 presents the results of this experiment.

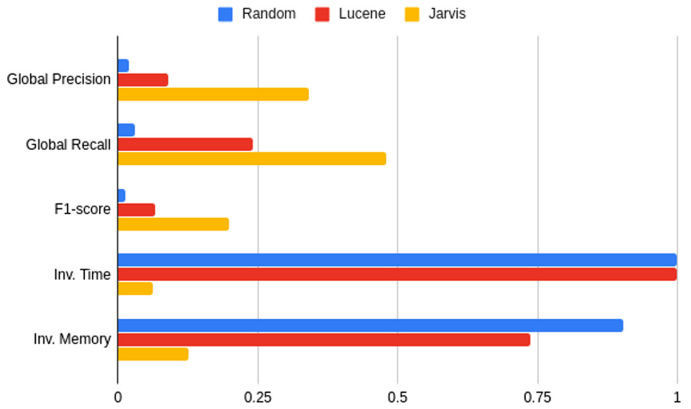

Fig. 3. Performance of the JarvisQA system. JarvisQA and the baselines are compared in terms of Global Precision, Global Recall, Global F1-Score, Inv.Time, Inv.Memory; higher values are better. JarvisQA improves Precision, Recall, and F1-Score by up to three times at the cost of execution time and memory consumption.

Table 5. Performance comparison using the two datasets TabMCQ and ORKG-QA against JarvisQA and the baselines. The results suggest that JarvisQA outperforms the baselines by substantially on both datasets. Best results are highlighted for both datasets.

System	Dataset	Precision @K				Recall @K				F1-Score @K			
		#1	#3	#5	#10	#1	#3	#5	#10	#1	#3	#5	#10
Random	TabMCQ	0.006	0.010	0.020	0.030	0.010	0.020	0.030	0.040	0.007	0.010	0.024	0.030
	ORKG	0.020	0.060	0.080	0.160	0.020	0.070	0.090	0.180	0.020	0.060	0.080	0.017
Lucene	TabMCQ	0.004	0.018	0.027	0.036	0.006	0.017	0.026	0.037	0.005	0.016	0.024	0.033
	ORKG	0.090	0.190	0.200	0.250	0.090	0.180	0.190	0.240	0.090	0.180	0.190	0.240
Jarvis	TabMCQ	0.060	0.090	0.100	**0.110**	0.070	0.090	0.110	**0.120**	0.060	0.080	0.100	**0.110**
	ORKG	0.340	0.380	0.460	**0.470**	0.350	0.380	0.460	**0.480**	0.340	0.380	0.450	**0.470**

5 Discussion and Future Work

The main objective of JarvisQA is to serve as a system that allows users to ask natural language questions on tablar views of scholarly knowledge. As such, the system addresses only a small part of the scholarly information corpus.

We performed several experimental evaluations to benchmark the performance of JarvisQA against other baselines using two different QA datasets. Different datasets showed different results based on the types of questions and the nature of the scholarly data encoded in the tables. Based on these extensive experiments, we conclude that usual information retrieval techniques used in search engines are failing to find specific answers for questions posed by a user. JarvisQA outperforms the other baselines in terms of precision, recall, and F1-score measure at the cost of higher execution time and memory requirements. Moreover, our system cannot yet answer all types of questions (e.g., non-answerable questions and listing questions).

Since JarvisQA utilizes a BERT based QA component, different components can perform differently, depending on the use case and scenario. Our system struggles with answers spanning across multiple cells of the table, and also in answering true/false questions. Furthermore, the answers are limited to information in the table (extractive method), since tables are not supplemented with further background information to improve the answers.

As indicated, the system can still be significantly improved. Future work will focus on improving answer selection techniques, and supporting more types of questions. Additionally, we will improve and enlarge the ORKG-QA dataset to become a better benchmark with more tables (content) and questions. JarvisQA currently selects the answer only from a single table, but use cases might require the combination of multiple tables or the identification of target table automatically (i.e., the system selects the table containing the correct answer from a pool of tables). Moreover, in the context of digital libraries, we want to integrate the system into the ORKG infrastructure so it can be used on live data directly.

6 Conclusion

Retrieving answers from scientific literature is a complicated task. Manually answering questions on scholarly data is cumbersome, time consuming. Thus, an automatic method of answering questions posed on scientific content is needed. JarvisQA is a question answering system addressing scholarly data that is encoded in tables or sub-graphs representing table content. It can answer several types of questions on table content. Furthermore, our ORKG-QA benchmark is a starting point to collaborate on adding more data to better train, evaluate, and test QA systems designed for tabular views of scholarly knowledge. To conclude, JarvisQA addresses several open questions in current information retrieval in the scholarly communication domain, and contributes towards improved information retrieval on scholarly knowledge. t can help researchers, librarians, and ordinary users to inquire for answers with higher accuracy than traditional information retrieval methods.

Acknowledgments. This work was co-funded by the European Research Council for the project ScienceGRAPH (Grant agreement ID: 819536) and the TIB Leibniz Information Centre for Science and Technology. The authors would like to thank our colleagues Kheir Eddine Farfar, Manuel Prinz, and especially Allard Oelen and Vitalis Wiens for their valuable input and comments.

References

1. Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., Ives, Z.: DBpedia: a nucleus for a web of open data. In: Aberer, K. (ed.) ASWC/ISWC -2007. LNCS, vol. 4825, pp. 722–735. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-76298-0_52
2. Bloehdorn, S.: Ontology-based question answering for digital libraries. In: Kovács, L., Fuhr, N., Meghini, C. (eds.) ECDL 2007. LNCS, vol. 4675, pp. 14–25. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-74851-9_2
3. Bornmann, L., Mutz, R.: Growth rates of modern science: a bibliometric analysis based on the number of publications and cited references. J. Assoc. Inf. Sci. Technol. **66**(11), 2215–2222 (2015). https://doi.org/10.1002/asi.23329
4. Bosman, J., et al.: The scholarly commons - principles and practices to guide research communication. https://doi.org/10.31219/OSF.IO/6C2XT
5. Cheng, J., Reddy, S., Saraswat, V., Lapata, M.: Learning structured natural language representations for semantic parsing. In: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 44–55. Association for Computational Linguistics, Stroudsburg (2017). https://doi.org/10.18653/v1/P17-1005
6. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: pre-training of deep bidirectional transformers for language understanding. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 4171–418. Association for Computational Linguistics, Stroudsburg (2019). https://doi.org/10.18653/v1/N19-1423
7. Diefenbach, D., Lopez, V., Singh, K., Maret, P.: Core techniques of question answering systems over knowledge bases: a survey. Knowl. Inf. Syst. **55**(3), 529–569 (2017). https://doi.org/10.1007/s10115-017-1100-y
8. Diefenbach, D., Lully, V., Migliatti, P.H., Singh, K., Qawasmeh, O., Maret, P.: QAnswer: a question answering prototype bridging the gap between a considerable part of the LOD cloud and end-users. In: The Web Conference 2019 - Proceedings of the World Wide Web Conference, WWW 2019, pp. 3507–3510. Association for Computing Machinery, Inc., May 2019. https://doi.org/10.1145/3308558.3314124
9. Dua, M., Kumar, S., Virk, Z.S.: Hindi language graphical user interface to database management system. In: Proceedings - 2013 12th International Conference on Machine Learning and Applications, ICMLA 2013, vol. 2, pp. 555–559. IEEE Computer Society (2013). https://doi.org/10.1109/ICMLA.2013.176
10. Groth, P., Gibson, A., Velterop, J.: The anatomy of a nanopublication. Inf. Serv. Use **30**(1–2), 51–56 (2010). https://doi.org/10.3233/ISU-2010-0613
11. Hersh, W.R.: Information Retrieval and Digital Libraries. In: Chen, H., Fuller, S.S., Friedman, C., Hersh, W. (eds.) Medical Informatics, Integrated Series in Information Systems, pp. 237–275. Springer, Boston (2005). https://doi.org/10.1007/0-387-25739-X_9

12. Jaradeh, M.Y., et al.: Open research knowledge graph: next generation infrastructure for semantic scholarly knowledge. Marina Del K-CAP**19** (2019). https://doi.org/10.1145/3360901.3364435
13. Jauhar, S.K., Turney, P., Hovy, E.: TabMCQ: a dataset of general knowledge tables and multiple-choice questions, February 2016. http://arxiv.org/abs/1602.03960
14. Karki, B., et al.: Question answering via web extracted tables and pipelined models, March 2019. http://arxiv.org/abs/1903.07113
15. Krishnamurthy, J., Kollar, T.: Jointly learning to parse and perceive: connecting natural language to the physical world. Trans. Assoc. Comput. Linguist. **1**, 193–206 (2013). https://doi.org/10.1162/tacl_a_00220
16. Kwiatkowski, T., Choi, E., Artzi, Y., Zettlemoyer, L.: Scaling semantic parsers with on-the-fly ontology matching. Technical report. www.wiktionary.com
17. Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P., Soricut, R.: Albert: a lite Bert for self-supervised learning of language representations, September 2019. http://arxiv.org/abs/1909.11942
18. Lin, J.: The web as a resource for question answering: perspectives and challenges. In: LREC. Las Palmas (2002). https://www.aclweb.org/anthology/L02-1085/
19. Lopez, V., Unger, C., Cimiano, P., Motta, E.: Evaluating question answering over linked data. J. Web Semant. **21**, 3–13 (2013). https://doi.org/10.1016/j.websem.2013.05.006
20. Marx, E., Usbeck, R., Ngomo, A.C.N., Höffner, K., Lehmann, J., Auer, S.: Towards an open question answering architecture. In: ACM International Conference Proceeding Series, vol. 2014-September, pp. 57–60. Association for Computing Machinery, September 2014. https://doi.org/10.1145/2660517.2660519
21. Oelen, A., Jaradeh, M.Y., Stocker, M., Auer, S.: Generate fair literature surveys with scholarly knowledge graphs. In: JCDL 2020: The 20th ACM/IEEE Joint Conference on Digital Libraries (2020). https://doi.org/10.1145/3383583.3398520
22. Peroni, S., et al.: Research articles in simplified HTML: a web-first format for HTML-based scholarly articles. PeerJ Comput. Sci. **2017**(10) (2017). https://doi.org/10.7717/peerj-cs.132
23. Peroni, S., Shotton, D.: Opencitations, an infrastructure organization for open scholarship. Quant. Sci. Stud. **1**(1), 1–17 (2020). https://doi.org/10.1162/qss_a_00023
24. Rajpurkar, P., Zhang, J., Lopyrev, K., Liang, P.: Squad: 100,000+ questions for machine comprehension of text. In: EMNLP 2016 - Proceedings Conference on Empirical Methods in Natural Language Processing, pp. 2383–2392. Association for Computational Linguistics (ACL) (2016). https://doi.org/10.18653/v1/d16-1264
25. Ramos, J.: Using TF-IDF to determine word relevance in document queries. Technical report
26. Schatz, B.R.: Information retrieval in digital libraries: bringing search to the net. Science **275**(5298), 327–334 (1997). https://doi.org/10.1126/science.275.5298.327
27. Shekarpour, S., Marx, E., Ngonga Ngomo, A.C., Auer, S.: Sina: semantic interpretation of user queries for question answering on interlinked data. J. Web Semant. **30**, 39–51 (2015). https://doi.org/10.1016/j.websem.2014.06.002
28. Singh, K., et al.: Why reinvent the wheel: let's build question answering systems together. In: WWW 2018: Proceedings of the 2018 World Wide Web Conference, pp. 1247–1256. Association for Computing Machinery (ACM) (2018). https://doi.org/10.1145/3178876.3186023

29. Sinha, A., et al.: An overview of Microsoft Academic Service (MAS) and applications. In: WWW 2015 Companion - Proceedings of the 24th International Conference on World Wide Web, pp. 243–246. Association for Computing Machinery Inc., New York, May 2015. https://doi.org/10.1145/2740908.2742839
30. Vakulenko, S., Savenkov, V.: Tableqa: Question answering on tabular data, May 2017.http://arxiv.org/abs/1705.06504
31. Wang, H., Zhang, X., Ma, S., Sun, X., Wang, H., Wang, M.: A neural question answering model based on semi-structured tables. Technical report
32. Wilkinson, M.D., et al.: The FAIR guiding principles for scientific data management and stewardship. Sci. Data **3**(1), 1–9 (2016). https://doi.org/10.1038/sdata.2016.18
33. Wolf, T., et al.: Huggingface's transformers: state-of-the-art natural language processing, October 2019. http://arxiv.org/abs/1910.03771
34. Yahya, M., Berberich, K., Elbassuoni, S., Ramanath, M., Tresp, V., Weikum, G.: Natural language questions for the web of data. Technical report (2012)
35. Yin, J., Jiang, X., Lu, Z., Shang, L., Li, H., Li, X.: Neural generative question answering. In: IJCAI International Joint Conference on Artificial Intelligence 2016-January, pp. 2972–2978 , December 2015. http://arxiv.org/abs/1512.01337
36. Zhang, Z., Wu, Y., Zhou, J., Duan, S., Zhao, H., Wang, R.: SG-Net: syntax-guided machine reading comprehension, August 2019. http://arxiv.org/abs/1908.05147
37. Zinsser, W.: On Writing Well, 30th Anniversary Edition: An Informal Guide to Writing Nonfiction. HarperCollins (2012)
38. Zou, L., Huang, R., Wang, H., Yu, J.X., He, W., Zhao, D.: Natural language question answering over RDF - a graph data driven approach. In: Proceedings of the ACM SIGMOD International Conference on Management of Data, pp. 313–324. Association for Computing Machinery (2014). https://doi.org/10.1145/2588555.2610525