DBLP-QuAD: A Question Answering Dataset over the DBLP Scholarly Knowledge Graph

Debayan **Banerjee**[1], Sushil **Awale**[1], Ricardo **Usbeck**[1] and Chris **Biemann**[1]

[1]*Universität Hamburg, Hamburg, Germany*

Abstract

In this work we create a question answering dataset over the DBLP scholarly knowledge graph (KG). DBLP is an on-line reference for bibliographic information on major computer science publications that indexes over 4.4 million publications published by more than 2.2 million authors. Our dataset consists of 10,000 question answer pairs with the corresponding SPARQL queries which can be executed over the DBLP KG to fetch the correct answer. DBLP-QuAD is the largest scholarly question answering dataset.

Keywords
Question Answering Scholarly Knowledge Graph DBLP Dataset

1. Introduction

Over the past decade, knowledge graphs (KG) such as Freebase [1], DBpedia [2], and Wikidata[3] have emerged as important repositories of general information. They store facts about the world in the linked data architecture, commonly in the format of <subject predicate object> triples. These triples can also be visualised as node-edge-node molecules of a graph structure. Much interest has been generated in finding ways to retrieve information from these KGs. Question Answering over Knowledge Graphs (KGQA) is one of the techniques used to achieve this goal. In KGQA, the focus is generally on translating a natural language question to a formal logical form. This task has, in the past, been achieved by rule-based systems [4]. More recently, neural network and machine learning based methods have gained popularity [5].

A scholarly KG is a specific class of KGs that contains bibliographic information. Some well known scholarly KGs are the Microsoft Academic Graph[1], OpenAlex[2], ORKG[3] and DBLP[4]. DBLP caters specifically to the bibliography of computer science, and as a result, it is smaller in size than other scholarly KGs. We decided to build our KGQA dataset over DBLP due to its focused domain and manageable size so that we could concentrate on adding complexity to the composition of the KGQA dataset itself.

Datasets are important, especially for ML-based systems, because such systems often have to be trained on a sample of data before they can be used on a similar test set. To this end, several

[1]https://www.microsoft.com/en-us/research/project/microsoft-academic-graph/
[2]http://openalex.org/
[3]https://orkg.org/
[4]https://dblp.org/

KGQA datasets exist [6]. However, not all datasets contain a mapping of natural language questions to the logical form (e.g. SPARQL, λ-calculus, S-expression). Some simply contain the question and the eventual answer. Such datasets can not be used to train models in the task of semantic parsing.

In this work, we present a KGQA dataset called DBLP-QuAD, which consists of 10,000 questions with corresponding SPARQL queries. The question formation process begins with human-written templates, and later, we machine-generate more questions from these templates. DBLP-QuAD consists of a variety of simple and complex questions and also tests the compositional generalisation of the models. DBLP-QuAD is the largest scholarly KGQA dataset being made available to the public[5].

2. Related Work

ORKG-QA benchmark [7] is the first scholarly KGQA dataset grounded to ORKG. The dataset was prepared using the ORKG API and focuses on the content of academic publications structured in comparison tables. The dataset is relatively small in size with only 100 question-answer pairs covering only 100 research publications.

Several other QA datasets exist, both for IR-based QA [8, 9] and KGQA [10, 11] approaches. Several different approaches have been deployed to generate the KGQA datasets. These approaches range from manual to machine generation. However, most datasets lie in between and use a combination of manual and automated process.

A clear separation can be created between datasets that contain logical forms and those that do not. Datasets that do not require logical forms can be crowd-sourced and such datasets are generally large in size. Crowd sourcing is generally not possible for annotating logical forms because this task requires high domain expertise and it is not easy to find such experts on crowd sourcing platforms. We focus on datasets that contain logical forms.

Free917 and QALD [12, 13] datasets were created manually by domain experts, however, their sizes are relatively small (917 and 806 respectively).

WebQuestionsSP and ComplexWebQuestions [14, 15] are developed using exisiting datasets. WebQuestionsSP is a semantic parsing dataset developed by using questions from WebQuestions [16]. Yih et al. [14] developed a dialogue-like user interface which allowed five expert human annotators to annotate the data in stages.

ComplexWebQuestions is a collection of 34,689 complex question paired with answers and SPARQL queries grounded to Freebase KG. The dataset builds on WebQuestionsSP by sampling question-query pairs from the dataset and automatically generating questions and complex SPARQL queries with composition, conjunctions, superlatives, and comparatives functions. The machine generated questions are manually annotated to natural questions and validated by 200 AMT crowd workers.

The OVERNIGHT (ON) approach is a semantic parsing dataset generation framework introduced by Wang et al. [17]. In this approach, the question-logical form pairs are collected with a three step process. In the first step, the logical forms are generated from a KG. Secondly, the logical forms are converted automatically into canonical questions. These canonical questions

---

[5]https://doi.org/10.5281/zenodo.7643971

are grammatically incorrect but successfully carry the semantic meaning. Lastly, the canonical questions are converted into natural forms via crowdsourcing. Following are some of the datasets developed using this approach.

GraphQuestions [18] consists of 5,166 natural questions accompanied by two paraphrases of the original question, an answer, and a valid SPARQL query grounded against the Freebase KG. GraphQuestions uses a semi-automated three-step algorithm to generate the natural questions for the KG.

LC-QuAD 1.0 [10] is another semantic parsing dataset for the DBpedia KG. LC-QuAD 1.0 is relatively larger in size with 5,000 natural language English questions and corresponding SPARQL queries. The generation process starts with the set of manually created SPARQL query templates, a list of seed entities, and a whitelist of predicates. Using the list of seed entities, two-hop subgraphs from DBpedia are extracted. The SPARQL query templates consist of placeholders for both entities and predicates which are instantiated using triples from the subgraph. These SPARQL queries are then used to instantiate natural question templates which form the base for manual paraphrasing by humans.

LC-QuAD 2.0 [19] is the second iteration of LC-QuAD 1.0 with 30,000 questions, their paraphrases and their corresponding SPARQL queries compatible with both Wikidata and DBpedia KGs. Similar to LC-QuAD 1.0, in LC-QuAD 2.0 a sub-graph is generated using seed entities and a SPARQL query template is selected based on whitelist predicates. Then, the query template is instantiated using the sub-graph. Next, a template question is generated from the SPARQL query which is then verbalised and paraphrased by AMT crowd workers. LC-QuAD 2.0 has more questions and more variation compared to LC-QuAD 1.0 with paraphrases to the natural questions.

GrailQA [20] extends the approach in [18] to generate 64,331 question-S-expression pairs grounded to the Freebase Commons KG. Here, S-expression are linearized forms of graph queries. Query templates extracted from graph queries generated from the KG are used to generate canonical logical forms grounded to compatible entities. The canonical logic forms are then validated by a graduate student if they represent plausible user query or not. Next, another graduate student annotated the validated canonical logic form with a canonical question. Finally, 6,685 Amazon Mechanical Turk workers write five natural paraphrases for each canonical question which are further validated by multiple independent crowd workers.

KQA Pro [21] is a large collection of 117,000 complex questions paired with SPARQL queries for the Wikidata KG. KQA Pro dataset also follows the OVERNIGHT approach where firstly facts from the KG are extracted. Next, canonical questions are generated with corresponding SPARQL queries, ten answer choices and a golden answer. The canonical questions are then converted into natural language with paraphrases using crowd sourcing.

CFQ [22] (Compositional Freebase Questions) is a semantic parsing dataset developed completely using synthetic generation approaches that consists of simple natural language questions with corresponding SPARQL query against the Freebase KG. CFQ contains 239,357 English questions which are generated using hand-crafted grammar and inference rules with a corresponding logical form. Next, resolution rules are used to map the logical forms to SPARQL queries. The CFQ dataset was specifically designed to measure compositional generalization.

In this work, we loosely follow the OVERNIGHT approach to create a large scholarly KGQA dataset for the DBLP KG.
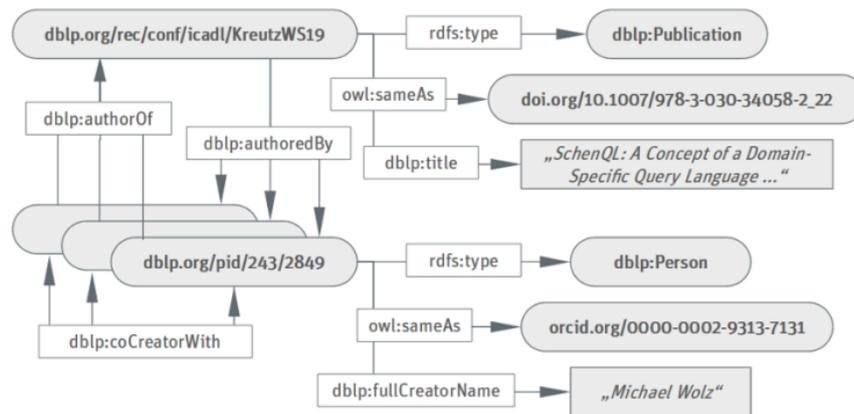
3. DBLP KG

Figure 1: Example of entries in the DBLP KG with its schema

DBLP, which used to stand for Data Bases and Logic Programming[6], was created in 1993 by Michael Ley at the University of Trier, Germany [23]. The service was originally designed as a bibliographic database for research papers and proceedings from the fields of database systems and logic programming. Over time, the service has grown in size and scope, and today includes bibliographic information on a wide range of topics within the field of computer science. The DBLP RDF data models a person-publication graph shown in Figure 1.

The DBLP KG contains two main entities: *Person* and *Publication*, where as other metadata such as journal and conferences, affiliation of authors are currently only string literals. Henceforth, we use the term *person* and *creator* interchangeably. At the time of its release, the RDF dump consisted of 2,941,316 person entities, 6,010,605 publication entities, and 252,573,199 RDF triples. DBLP currently does not provide a SPARQL endpoint but the RDF dump can be downloaded and a local SPARQL endpoint such as Virtuoso Server can be setup to run a SPARQL query against the DBLP KG.

The live RDF data model on the DBLP website follows the schema shown in Figure 1. However, the RDF snapshots available for download have the *coCreatorWith* and *authorOf* predicates missing. Although these predicates are missing, the *authoredBy* predicate can be used to derive the missing relations. DBLP-QuAD is based on the DBLP KG schema of the downloadable RDF graph.

4. Dataset Generation Framework

In this work, the aim is to generate a large variety of scholarly questions and corresponding SPARQL query pairs for the DBLP KG. Initially, a small set of templates T containing a SPARQL query template s_t and a few semantically equivalent natural language question templates Q_t
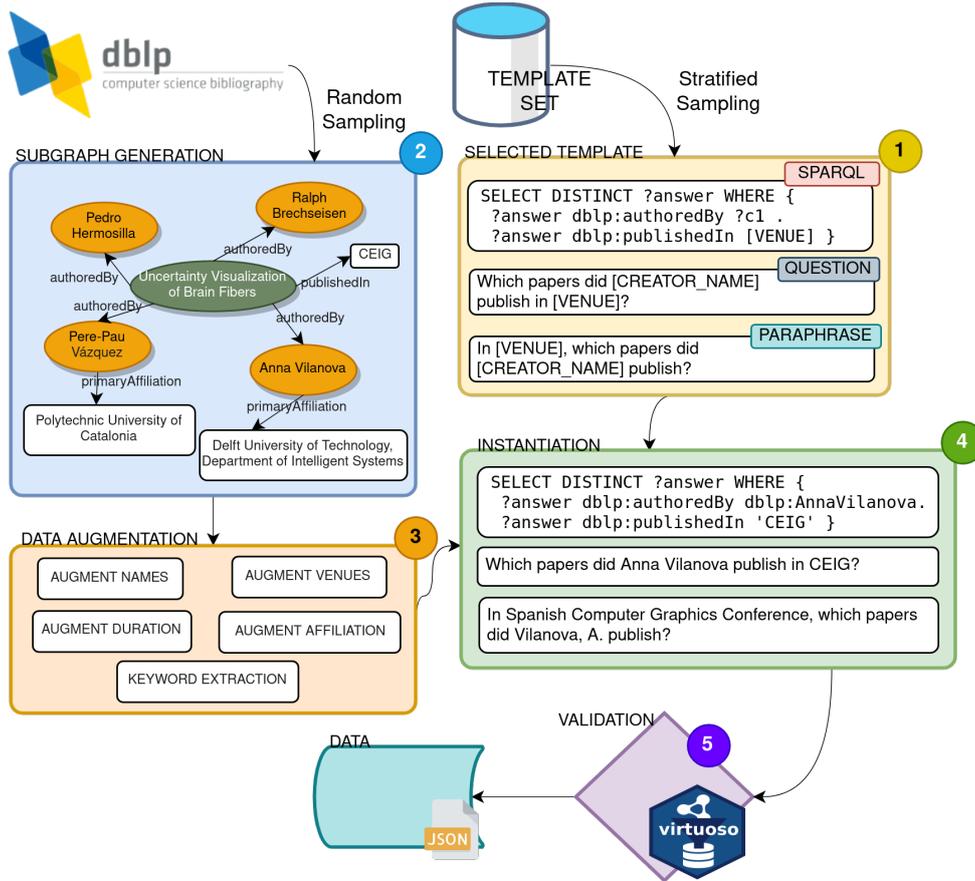

Figure 2: Motivating Example. The generation process starts with (1) selection of a template tuple followed by (2) subgraph generation. Then, literals in subgraph are (3) augmented before being used to (4) instantiate the selected template tuple. The generated data is (5) filtered based on if they produce answers or not.

are created. The questions and query templates are created such that they cover a wide range of scholarly metadata user information need while also being answerable using a SPARQL query against the DBLP KG. Next, we synthetically generate a large set of question-query pairs (q_i, s_i) suitable for training a neural network semantic parser.

The core methodology of the dataset generation framework encompasses instantiating the templates using literals of subgraphs sampled from the KG. Moreover, to capture different representations of the literal values from a human perspective, we randomly mix in different augmentations of these textual representations. The dataset generation workflow is shown in Figure 2.

4.1. Templates

The first step in the dataset generation process starts with the creation of a template set. After carefully analyzing the ontology of the DBLP KG, we manually wrote 98 pairs of valid SPARQL

query templates and a set of semantically equivalent natural language question templates. The template set was written by one author and verified for correctness by another author. The query and question templates consist of placeholder markers instead of URIs, entity surface forms or literals. For example, in Figure 2 (Section 1), the SPARQL query template includes the placeholders $?c1$ and $[VENUE]$ for DBLP person URI and venue literal respectively. Similarly, the question templates include placeholders $[CREATOR\_NAME]$ and $[VENUE]$ for creator name and venue literal respectively. The template set covers the two entities creator and publication, and additionally the foreign entity bibtex type. Additionally, they also cover the 11 different predicates of DBLP KG.

The template set consists of template tuples. A template tuple $t = (s_t, Q_t, E_t, P_t)$ is composed of a SPARQL query template s_t, a set of semantically equivalent natural language question templates Q_t, a set of entity placeholders E_t and a set of predicates P_t used in s_t. We also add a boolean indicating whether the query template is temporal or not and another boolean indicating whether to use or not use the template while generating $train$ dataset. Each template tuple contains between four and seven paraphrased question templates offering wide linguistic diversity. While most of the question templates use the *"Wh-"* question keyword, we also include instruction-style paraphrases.

We group the template tuples as creator-focused or publication-focused ϵ and further group them by query types δ. We have 10 different query types and they include Single Fact, Multiple Facts, Boolean, Negation, Double Negation, Double Intent, Union, Count, Superlative/Comparative, and Disambiguation. The question types are discussed in Section 4.6 with examples. The distribution of templates per entity and query type is shown in Table 1. During dataset generation, for each data instance we sample a template tuple from the template set using stratified sampling maintaining equal distribution of entity types and query types.

Query Type	Creator-focused	Publication-focused	Total
Single Fact	5	5	10
Multiple Facts	7	7	14
Boolean	6	6	12
Negation	4	4	8
Double Negation	4	4	8
Double Intent	5	4	9
Union	4	4	8
Count	6	5	11
Superlative/Comparative	6	6	12
Disambiguation	3	3	6
Total	50	48	98

Table 1
Total number of template tuples per query type grouped by entity type

4.2. Subgraph generation

The second part of the dataset generation framework is subgraph generation. Given a graph $G = (V, E)$ where V are the vertices, and E are edges, we draw a subgraph $g = (v, e)$ where

$v \subset V, e \subset E$. For the DBLP KG, V are the creator and publication entity URIs or literals, and the E are the predicates of the entities.

The subgraph generation process starts with random sampling of a publication entity v_i from the DBLP KG. We only draw from the set of publication entities as the RDF snapshot available for download has $authorOf$ and $coCreatorWith$ predicates missing for creator entity. As such, a subgraph centered on a creator entity would not have end vertices that can be expanded further. With the sampled publication entity v_i, we iterate through all the predicates e to extract creator entities v' as well as the literal values. We further, expand the creator entities and extract their literal values to form a two-hop subgraph $g = (v, e)$ as shown in Figure 2 (Section 2).

4.3. Template Instantiation

Using the generated subgraph and the sampled template tuple, the template tuple is instantiated with entity URIs and literal values from the subgraph. In the instantiation process, a placeholder marker in a string is replaced by the corresponding text representation.

For the SPARQL query template s_t, we instantiate the creator/publication placeholder markers with DBLP creator/publication entity URIs or literal values for affiliation and conference or journals to create a valid SPARQL query s that returns answers when run against the DBLP KG SPARQL endpoint.

In case of natural language question templates, we randomly sample two from the set of question templates $q_t^1, q_t^2 \in Q_T$, and instantiate each using only the literal values from the subgraph to form one main natural language question q^1 and one natural language question paraphrase q^2. In natural language, humans can write the literal strings in various forms. Hence to introduce this linguistic variation, we randomly mix in alternate string representations of these literal values in both natural language questions. The data augmentation process allows us to add heuristically manipulated alternate literal representations to the natural questions. A example of an instantiated template is shown in Figure 2 (Section 3).

4.4. Data Augmentation

For the template instantiation process, we perform simple string manipulations to generate alternate literal representations. Then, we randomly select between the original literal representation and the alternate representation to instantiate the natural language questions. For each literal type, we apply different string manipulation techniques which we describe below.

Names: For names we generate four different alternatives involving switching parts of names or keeping only initials of the names. Consider the name *John William Smith* for which we produce *Smith, John William*, *J. William Smith*, *John W. Smith*, and *Smith, J. William*.

Venues: Venues can be represented using either its short form or its full form. For example, *ECIR* or *European Conference on Information Retrieval*. In DBLP venues are stored in its short form. We use a selected list of conference and journals[7] containing the short form and its equivalent full form to get the full venue names.

Duration: About 20% of the templates contain temporal queries, and some of them require dummy numbers to represent duration. For example, the question *"In the last five years, which*

---

papers did Mante S. Nieuwland publish?" uses the dummy value *five*. We randomly select between the numerical representation and the textual representation for the dummy duration value.

Affiliation: In natural language questions, only the institution name is widely used to refer to the affiliation of an author. However, the DBLP KG uses the full address of an institution including city and country name. Hence, using RegeEx we extract the institution names and randomly select between the institution name and the full institution address in the instantiation process.

Keywords: For disambiguation queries, we do not use the full title of a publication but rather a part of it by extracting keywords. For this purpose, we use SpaCy's Matcher API[8] to extract noun phrases from the title.

Algorithm 1: Dataset Generation Process

GenerateDataset (T, x, N, G)

    inputs : template set T; dataset set to generate x; size of dataset to generate N; KG
            to sample subgraphs from G;

    output : dataset D;

    $D \leftarrow \emptyset$;

    $n \leftarrow (N/|\epsilon|)/|\delta|$;

    foreach $e \in \epsilon$ **do**

        foreach $s \in \delta$ **do**

            $i \leftarrow 0$;

            $T_{es} \leftarrow T[e][s]$;

            if $x == train$ **then**

                $T_{es} \leftarrow Filter(T_{es}, test\_only == True)$

            while $i < n$ **do**

                $g_1, g_2 \leftarrow SampleSubgraph(G, 2)$;

                $t_i \leftarrow random.sample(T_{es})$;

                $d_i \leftarrow Instantiate(t_i, g_1, g_2, x)$;

                $answer \leftarrow Query(d_i)$;

                if $answer$ **then**

                    $D \leftarrow d_i$;

                    $i \leftarrow i + 1$;

    return D

4.5. Dataset Generation

For each data instance d_i, we sample 2 subgraphs (*SampleSubgraph(G,2)*) and instantiate a template tuple t_i (*Instantiate(t_i, g_1, g_2, x)*). We sample 2 subgraphs as some template tuples require to be instantiated with two publication titles. Each data instance $d_i = (s_i, q_i^1, q_i^2, E_i, P_i, y, z)$ comprises of a valid SPARQL query s_i, one main natural language question q_i^1, one semantically

---

equivalent paraphrase of the main question q_i^2, a list of entities E_i used in s_i, a list of predicates P_i used in s_i, a Boolean indicating whether the SPARQL query is temporal or not y, and another Boolean informing whether the SPARQL query is found only in *valid* and *test* sets z. We generate an equal number n of questions for each entity group ϵ equally divided for each query type δ.

To foster a focus on generalization ability, we manually marked 20 template tuples to withhold during generation of the *train* set. However, we use all the template tuples in the generation of *valid* and *test* sets. Furthermore, we also withhold 2 question templates when generating *train* questions but use all question templates when generating *valid* and *test* sets. This controlled generation process allows us to withhold some entity classes, predicates and paraphrases from *train* set. Our aim with this control is to create a scholarly KGQA dataset that facilitates development of KGQA models that adhere to *i.i.d*, *compositional*, and *zero-shot* [20] generalization.

Further, we validate each data instance d_i by running the SPARQL query s_i against the DBLP KG via a Virtuoso SPARQL endpoint[9]. We filter out data instances for which the SPARQL query is invalid or generates a blank response. A SPARQL query may generate a blank response if the generated subgraphs have missing literal values. In the DBLP KG, some of the entities have missing literals for predicates such as *primaryAffiliation*, *orcid*, *wikidata*, and so on. Additionally, we also store the answers produced by the SPARQL query against the DBLP KG formatted according to *https://www.w3.org/TR/sparql11-results-json/*. The dataset generation process is summarized in Algorithm 1.

4.6. Types of Questions

The dataset is composed of the following question types. The examples shown here are hand-picked from the dataset.

- **Single fact**: These questions can be answered using a single fact. For example, "What year was 'SIRA: SNR-Aware Intra-Frame Rate Adaptation' published?"
- **Multiple facts**: These questions require connecting two or more facts to answer. For example, "In SIGCSE, which paper written by Darina Dicheva with Dichev, Christo was published?"
- **Boolean**: These questions answer where a given fact is true or false. We can also add negation keywords to negate the questions. For example, "Does Szeider, Stefan have an ORCID?"
- **Negation**: These questions require to negate the answer to the Boolean questions. For example, "Did M. Hachani not publish in ICCP?"
- **Double negation**: These questions require to negate the Boolean question answers twice which results. For example, "Wasn't the paper 'Multi-Task Feature Selection on Multiple Networks via Maximum Flows' not published in 2014?"
- **Count**: These questions pertain to the count of occurrence of facts. For example, "Count the authors of 'Optimal Symmetry Breaking for Graph Problems' who have Carnegie Mellon University as their primary affiliation."

---

[9]https://docs.openlinksw.com/virtuoso/whatisvirtuoso/

- **Superlative/Comparative**: Superlative questions ask about the maximum and minimum for a subject and comparative questions compare values between two subjects. We group both types under one group. For example, "Who has published the most papers among the authors of 'k-Pareto optimality for many-objective genetic optimization'?"
- **Union** questions cover a single intent but for multiple subjects at the same time. For example, "List all the papers that Pitas, Konstantinos published in ICML and ISCAS."
- **Double intent** questions poses two user intentions, usually about the same subject. For example, "In which venue was the paper 'Interactive Knowledge Distillation for image classification' published and when?"
- **Disambiguation** questions requires identifying the correct subject in the question. For example, "Which author with the name Li published the paper about Buck power converters?"

5. Dataset Statistics

DBLP-QuAD consists of 10,000 unique question-query pairs grouped into *train*, *valid* and *test* sets with a ratio of *7:1:2*. The dataset covers 13,348 creators and publications, and 11 predicates of the DBLP KG. For each query type in Table 1, the dataset includes 1,000 question-query pairs each of which is equally divided as creator-focused or publication-focused. Additionally, among the questions in DBLP-QuAD, 2,350 are temporal questions.

Linguistic Diversity. In DBLP-QuAD, a natural language question has an average word length of 17.32 words and an average character length of 114.1 characters. Similarly, a SPARQL query has an average vocab length of 12.65 and an average character length of 249.48 characters. Between the natural language question paraphrases, the average Jaccard similarity for unigram and bigram are 0.62 and 0.47 (with standard deviations of 0.22 and 0.24) respectively. The average Levenshtein edit distance between them is 32.99 (with standard deviation of 23.12). We believe the metrics signify a decent level of linguistic diversity.

Entity Linking. DBLP-QuAD also presents challenging entity linking with data augmentation performed on literals during the generation process. The augmented literals present more realistic and natural representation of the entity surface forms and literals compared to the entries in the KG.

Generalization. In the *valid* set 18.9% and in the *test* set 19.3% of instances were generated using the withheld templates. Hence, these SPARQL query templates and natural language question templates are unique to the *valid* and *test* sets. Table 2 shows the percent of questions with different levels of generalization in the *valid* and *test* sets of the dataset.

Dataset	I.I.D	Compositional	Zero-shot
Valid	82.8%	13.6%	3.6%
Test	81.2%	15.1%	3.8%

Table 2
Percent of questions with different levels of generalization in the *valid* and *test* sets of DBLP-QuAD

6. Semantic Parsing Baseline

To lay the foundation for future work on DBLP-QuAD, we also release baselines using the recent work by Banerjee et al. [24], where a pre-trained T5 model is fine-tuned [25] on the LC-QuAD 2.0 dataset.

Following Banerjee et al. [24], we assume the entities and the relations are linked, and only focus on query building. We formulate the source as shown in Figure 3, where for each natural language question a prefix "**parse text to SPARQL query:**" is added. The source string is further concatenated with entity URIs and relation schema URIs separated by a special token $[SEP]$. The target text is the corresponding SPARQL query which is padded with the tokens $< s >< /s >$. We also make use of the sentinel tokens provided by T5 to represent the DBLP prefixes e.g. *<extra_id_1>* denotes the prefix *https://dblp.org/pid/*, SPARQL vocabulary and symbols. This step helps the *T5-tokenizer* to correctly fragment the target text during inference.
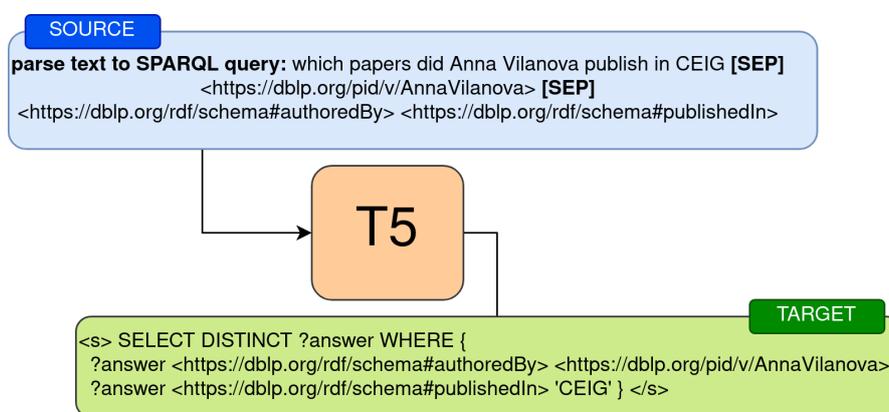

Figure 3: Representation of source and target text used to fine-tune the T5 model

We fine-tune *T5-Base* and *T5-Small* on DBLP-QuAD train set with a learning rate of *1e-4* for 5 epochs with an input as well as output text length of 512 and batch size of 4.

6.1. Experiment Results

We report the performance of the baseline model on the DBLP-QuAD test set. Firstly, we report on the exact-match between the gold and the generated SPARQL query. For the exact-match accuracy we compare the generated and the gold query token by token after removing whitespaces. Next, for each SPARQL query on the test set, we run both the gold and and the query generated by the T5 baseline models using Virtuoso SPARQL endpoint to fetch answers from the DBLP KG. Based on the answers collected, we report on the F1 score. The results are reported on Table 3.

7. Limitations

One of the drawbacks of our dataset generation framework is that natural questions are synthetically generated. (CFQ [22] has a similar limitation.) Although the question templates were human-written, only two people (authors of the paper) worked on the creation of the question

Evaluation metrics	T5-Small	T5-Base
Exact-match Accuracy	0.638	0.813
F1 Score	0.721	0.868

Table 3
Evaluation results of fine-tuned T5 to DBLP-QuAD

templates and was not crowd sourced from a group of researchers. Additionally, the questions are generated by drawing data from a KG. Hence, the questions may not perfectly reflect the distribution of user information need. However, the machine-generation process allows for programmatic configuration of the questions, setting question characteristics, and controlling dataset size. We utilize the advantage by programmatically augmenting text representations and generating a large scholarly KGQA with complex SPARQL queries.

Second, in generating *valid* and *test* sets, we utilize additional 19 template tuples which account for about 20% of the template set. Therefore, the syntactic structure for 80% of the generated data in *valid* and *test* would already be seen in the train set resulting in test leakage. However, to limit the leakage on 80% of the data, we withhold 2 question templates in generating the $train$ set. Moreover, the data augmentation steps carried out would also add challenges in the $valid$ and $test$ sets.

Another shortcoming of DBLP-QuAD is that the paper titles do not perfectly reflect user behavior. When a user asks a question, they do not type in the full paper title and also some papers are popularly known by a different short name. For example, the papers "Language Models are Few-shot Learners" and "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding" are also known as "GPT-3" and "BERT" respectively. This is a challenging entity linking problem which requires further investigation. Despite the shortcomings, we feel the large scholarly KGQA dataset would ignite more research interest in scholarly KGQA.

8. Conclusion

In this work, we presented a new KGQA dataset called DBLP-QuAD. The dataset is the largest scholarly KGQA dataset with corresponding SPARQL queries. The dataset contains a wide variety of questions and query types and we present the data generation framework and baseline results. We hope this dataset proves to be a valuable resource for the community.

As future work, we would like to build a robust question answering system for scholarly data using this dataset.

9. Acknowledgements

References

[1] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, J. Taylor, Freebase: A Collaboratively Created Graph Database for Structuring Human Knowledge, in: Proceedings of the 2008 ACM SIGMOD international conference on Management of data, AcM, 2008, pp. 1247–1250.

[2] J. Lehmann, R. Isele, M. Jakob, A. Jentzsch, D. Kontokostas, P. N. Mendes, S. Hellmann, M. Morsey, P. Van Kleef, S. Auer, et al., DBpedia – A Large-Scale, Multilingual Knowledge Base Extracted from Wikipedia, Semantic Web (2015).

[3] Vrandečić, Denny and Krötzsch, Markus, Wikidata: A Free Collaborative Knowledge Base, Communications of the ACM (2014).

[4] M. Dubey, S. Dasgupta, A. Sharma, K. Höffner, J. Lehmann, AskNow: A Framework for Natural Language Query Formalization in SPARQL, in: H. Sack, E. Blomqvist, M. d'Aquin, C. Ghidini, S. P. Ponzetto, C. Lange (Eds.), The Semantic Web. Latest Advances and New Domains, Springer International Publishing, Cham, 2016, pp. 300–316.

[5] N. Chakraborty, D. Lukovnikov, G. Maheshwari, P. Trivedi, J. Lehmann, A. Fischer, Introduction to Neural Network based Approaches for Question Answering over Knowledge Graphs, 2019. URL: https://arxiv.org/abs/1907.09361. doi:10.48550/ARXIV.1907.09361.

[6] A. Perevalov, X. Yan, L. Kovriguina, L. Jiang, A. Both, R. Usbeck, Knowledge Graph Question Answering Leaderboard: A Community Resource to Prevent a Replication Crisis, in: Proceedings of the Thirteenth Language Resources and Evaluation Conference, European Language Resources Association, Marseille, France, 2022, pp. 2998–3007. URL: https://aclanthology.org/2022.lrec-1.321.

[7] M. Y. Jaradeh, M. Stocker, S. Auer, Question answering on scholarly knowledge graphs, in: International Conference on Theory and Practice of Digital Libraries, Springer, 2020, pp. 19–32.

[8] P. Rajpurkar, R. Jia, P. Liang, Know what you don't know: Unanswerable questions for SQuAD, arXiv preprint arXiv:1806.03822 (2018).

[9] T. Kwiatkowski, J. Palomaki, O. Redfield, M. Collins, A. Parikh, C. Alberti, D. Epstein, I. Polosukhin, J. Devlin, K. Lee, et al., Natural questions: a benchmark for question answering research, Transactions of the Association for Computational Linguistics 7 (2019) 453–466.

[10] P. Trivedi, G. Maheshwari, M. Dubey, J. Lehmann, LC-QuAD: A Corpus for Complex Question Answering over Knowledge Graphs, in: C. d'Amato, M. Fernandez, V. Tamma, F. Lecue, P. Cudré-Mauroux, J. Sequeda, C. Lange, J. Heflin (Eds.), The Semantic Web – ISWC 2017, volume 10588, Springer International Publishing, Cham, 2017, pp. 210–218. doi:10.1007/978-3-319-68204-4_22.

[11] P. Sen, A. F. Aji, A. Saffari, Mintaka: A Complex, Natural, and Multilingual Dataset for End-to-End Question Answering, arXiv preprint arXiv:2210.01613 (2022).

[12] Q. Cai, A. Yates, Large-scale semantic parsing via schema matching and lexicon extension, in: Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), 2013, pp. 423–433.

[13] R. Usbeck, A.-C. N. Ngomo, B. Haarmann, A. Krithara, M. Röder, G. Napolitano, 7th Open Challenge on Question Answering over Linked Data (QALD-7), in: M. Dragoni, M. Solanki, E. Blomqvist (Eds.), Semantic Web Challenges, volume 769, Springer International Publish-

ing, Cham, 2017, pp. 59–69. doi:`10.1007/978-3-319-69146-6_6`.

[14] W.-t. Yih, M. Richardson, C. Meek, M.-W. Chang, J. Suh, The Value of Semantic Parse Labeling for Knowledge Base Question Answering, in: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), Association for Computational Linguistics, Berlin, Germany, 2016, pp. 201–206. doi:`10.18653/v1/P16-2033`.

[15] A. Talmor, J. Berant, The Web as a Knowledge-base for Answering Complex Questions, 2018. `arXiv:1803.06643`.

[16] J. Berant, A. Chou, R. Frostig, P. Liang, Semantic Parsing on Freebase from Question-Answer Pairs, in: Proceedings of the 2013 conference on empirical methods in natural language processing, 2013, pp. 1533–1544.

[17] Y. Wang, J. Berant, P. Liang, Building a semantic parser overnight, in: Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), 2015, pp. 1332–1342.

[18] Y. Su, H. Sun, B. Sadler, M. Srivatsa, I. Gur, Z. Yan, X. Yan, On Generating Characteristic-rich Question Sets for QA Evaluation, in: Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Austin, Texas, 2016, pp. 562–572. doi:`10.18653/v1/D16-1054`.

[19] M. Dubey, D. Banerjee, A. Abdelkawi, J. Lehmann, LC-QuAD 2.0: A Large Dataset for Complex Question Answering over Wikidata and DBpedia, in: C. Ghidini, O. Hartig, M. Maleshkova, V. Svátek, I. Cruz, A. Hogan, J. Song, M. Lefrançois, F. Gandon (Eds.), The Semantic Web – ISWC 2019, volume 11779, Springer International Publishing, Cham, 2019, pp. 69–78. doi:`10.1007/978-3-030-30796-7_5`.

[20] Y. Gu, S. Kase, M. Vanni, B. Sadler, P. Liang, X. Yan, Y. Su, Beyond I.I.D.: Three Levels of Generalization for Question Answering on Knowledge Bases, in: Proceedings of the Web Conference 2021, ACM, Ljubljana Slovenia, 2021, pp. 3477–3488. doi:`10.1145/3442381.3449992`.

[21] S. Cao, J. Shi, L. Pan, L. Nie, Y. Xiang, L. Hou, J. Li, B. He, H. Zhang, KQA pro: A dataset with explicit compositional programs for complex question answering over knowledge base, in: Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics, Association for Computational Linguistics, Dublin, Ireland, 2022, pp. 6101–6119. doi:`10.18653/v1/2022.AssociationforComputationalLinguistics-long.422`.

[22] D. Keysers, N. Schärli, N. Scales, H. Buisman, D. Furrer, S. Kashubin, N. Momchev, D. Sinopalnikov, L. Stafiniak, T. Tihon, D. Tsarkov, X. Wang, M. van Zee, O. Bousquet, Measuring Compositional Generalization: A Comprehensive Method on Realistic Data, 2020. `arXiv:1912.09713`.

[23] M. Ley, The DBLP Computer Science Bibliography: Evolution, Research Issues, Perspectives, in: G. Goos, J. Hartmanis, J. van Leeuwen, A. H. F. Laender, A. L. Oliveira (Eds.), String Processing and Information Retrieval, volume 2476, Springer Berlin Heidelberg, Berlin, Heidelberg, 2002, pp. 1–10. doi:`10.1007/3-540-45735-6_1`.

[24] D. Banerjee, P. A. Nair, J. N. Kaur, R. Usbeck, C. Biemann, Modern Baselines for SPARQL Semantic Parsing, in: Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2022, pp. 2260–2265. doi:`10.1145/`

3477495.3531841. arXiv:2204.12793.

[25] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, P. J. Liu, Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer, J. Mach. Learn. Res. 21 (2020) 1–67.