Hamburg Universität

MIN Faculty

Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

Master Thesis

A Question Answering Dataset over the DBLP Scholarly Knowledge Graph

Submitted by:

Sushil Awale

Matr.-Nr.: 7374333

Course of study: M.Sc. Intelligent Adapative Systems

First Reviewer:

Prof. Dr. Chris Biemann

Second Reviewer:

Prof. Dr. Ricardo Usbeck

Supervisor:

Debayan Banerjee

Submission Date:

21.02.2023

Acknowledgment

I would like to express my deepest gratitude to my supervisor Debayan Banerjee for his unwavering support and guidance throughout the completion of this thesis. I am grateful for his expertise, patience, and encouragement during the research process.

I would also like to thank Prof. Dr. Chris Biemann for his valuable feedback and suggestions on my work. I am also grateful to Prof. Dr. Ricardo Usbeck for his insightful comments and suggestions that helped me improve the quality of my work.

I am also thankful for the support and discussions I had with my talented fellow students from the IAS course. In particular, special thanks to Diana Rueda for reading the first draft of this thesis and providing meaningful feedback. Also, a special thanks to my long time friend Bikalpa Baniya for his suggestions and feedback on my writing.

I am grateful to my family and friends for their love, encouragement, and support throughout my studies. I would especially like to thank my partner Lilly Seiffert for her unwavering support during the ups and downs of this journey. Her patience, encouragement, and understanding helped me to stay motivated and focused.

Thank you all for your contributions to this work.

Sushil Awale

Abstract

Knowledge Graph Question Answering (KGQA) is one of the popular approaches for retrieving information from knowledge graphs. Development of KGQA systems for querying public KGs such as Wikidata, DBpedia, and Freebase has been an active area of research in the past decade. These KGQA systems are capable of answering questions on general information and world facts that can be found in these KGs. One of the limitations of KGQA systems is that they are tied to the KGs they were developed for. Hence, the systems are incapable of answering out-of-domain questions. In addition to the KGs, another building block for KGQA systems are KGQA datasets. The publicly available KGQA datasets have also been developed for the above mentioned KGs.

A useful application for KGQA is in the scholarly domain. However, only a few scholarly KGQA datasets are publicly available. With the recent release of the RDF data graph of DBLP, a computer science bibliographic database, development of a KGQA dataset for the DBLP KG is feasible. In this work, a KGQA dataset, called DBLP-QuAD, is developed for the scholarly DBLP KG. The dataset consists of 10,000 question-SPARQL query pairs distributed among 10 different simple and complex query types. DBLP-QuAD encloses challenges for KGQA systems by including augmented entity and literal surface forms, and compositional and zero-shot questions in the test sets.

In addition, this thesis also develops and evaluates a semantic parsing baseline on DBLP-QuAD. An evaluation of the T5 model, the current SOTA model on semantic parsing, on DBLP-QuAD shows its shortcomings on compositional and zero-shot questions.

The main contribution of this thesis, is DBLP-QuAD, the largest and the first scholarly KGQA dataset for the DBLP KG. DBLP-QuAD introduces challenging entity linking problems, and invites research on generalization ability of large language models.

Keywords: *Knowledge Graph Question Answering, Semantic Parsing, Scholarly Knowledge Graph*

Contents

List of Figures

List of Tables

Listings

List of Abbreviations

AMT - Amazon Mechanical Turk

API - Application Programming Interface

BM25 - Best Match 25

DBLP - Data Bases and Logic Programming

HTML - HyperText Markup Language

I.I.D - Independent and Identically Distributed

IR - Information Retrieval

KB - Knowledge Base

KG - Knowledge Graph

KGQA - Knowledge Graph Question Answering

LLM - Large Language Model

MRC - Machine Reading Comprehension

NLP - Natural Language Processing

ORKG - Open Research Knowledge Graph

QA - Question Answering

QuAD - Question Answering Dataset

RDF - Resource Description Framework

SPARQL - SPARQL Protocol and RDF Query Language

SQL - Structured Query Language

TF-IDF - Term Frequency - Inverse Document Frequency

XML - Markup Language

1 Introduction

The internet is a large repository of information, with approximately billions of web pages and documents containing information on a diverse set of topics. The information is often found in an unstructured form, which can be challenging for individuals to find precise and accurate information quickly. A subset of this information is also found in an organized and structured format called knowledge graphs (KGs). A KG represents information as entities and forming relationships between them, creating a graph structure that is programmatically easy to query and search. In the last decade, several KGs such as Freebase [Bollacker et al., 2008], Wikidata [Vrandecic and Krötzsch, 2014], and DBpedia [Auer et al., 2007] have surfaced as repositories of world facts and other general information. The structured format simplifies retrieval of information for machines but is yet difficult for humans. Much interest has been garnered in simplifying this retrieval technique for layman users. Question Answering over Knowledge Graph (KGQA) is one of the approaches used to achieve this goal.

In KGQA, the general idea is to transform natural questions into a logical form. The main motivation in this approach is to add a layer of abstraction over the logical forms by allowing users to formulate their queries in natural language. In the past, rule-based systems were used for this task [Dubey et al., 2016]. In recent years, neural and machine learning approaches have been widely used [Chakraborty et al., 2021]. The neural and machine learning models are trained in a supervised fashion and require KGQA datasets i.e. pairs of natural questions and corresponding logical forms.

One key limitation of KGQA is that the system is tied to the KG it is built for. Hence, a KGQA system designed for Wikidata [Vrandecic and Krötzsch, 2014] can not be used for DBpedia [Auer et al., 2007], for example. This is the main reason why when Freebase [Bollacker et al., 2008] was made defunct in 2016, many KGQA datasets developed for Freebase had to be migrated to other KGs such as Wikidata and DBpedia. Hence, a specific KGQA dataset targeted towards a specific KG is required to develop KGQA systems for a specialized domain. One such domain is the scholarly field. KGQA can be a useful tool for researchers and students to access bibliographic information easily and efficiently. Scholarly KGQA would handle questions such as “What are the papers written by Author X?”, “In which venue was the paper Y published in?”, and so on. For the development of a scholarly KGQA systems two main components are required.

The first key component in building scholarly KGQA system is a scholarly KG. The scholarly KG is a specific class of KG that contain bibliographic information. A few scholarly KG exist such as Microsoft Academic Graph[1], OpenAlex [Priem et al., 2022], ORKG [Jaradeh et al., 2019], and DBLP

[1]https://www.microsoft.com/en-us/research/project/microsoft-academic-graph/

[Ley, 2002]. DBLP is a scholarly KG containing bibliographic information about the computer science discipline. In this thesis, DBLP is chosen as the scholarly KG.

The second key resource required to build scholarly KGQA system is a scholarly KGQA dataset. To the best of my knowledge, only one scholarly KGQA dataset is publicly available (ORKG-QA benchmark [Jaradeh et al., 2020]). However, it only contains 100 questions, making it a small dataset. For LLM research, a large sized dataset is recommended. Hence, this thesis focuses on developing a large scholarly KGQA dataset for the DBLP KG and a scholarly KGQA baseline model with the dataset.

1.1 Motivation

The main focus of the thesis is the development of a KGQA dataset for the scholarly DBLP KG. KGQA is an active area of research [Chakraborty et al., 2021] and several KGQA dataset exist [Perevalov et al., 2022b]. However, KGQA systems are tightly coupled with the KGQA dataset used for its development. These systems are constrained by the facts present in the KG for answering the questions, and thus are incapable to answer out-of-domain questions.

Most KGQA datasets that exist were developed for KGs such as Wikidata [Sen et al., 2022] and DBpedia [Dubey et al., 2019] which contain world facts and general information and hence incapable to answer scholarly questions. With the motivation to contribute to scholarly KGQA research, this thesis aims to develop a scholarly KGQA dataset called DBLP-QuAD for the DBLP scholarly KG.

1.2 Approach

For the thesis, two main tasks will be carried out. The first task is the development of a dataset generation framework. The task involves developing a machine generation process that utilizes the facts in the DBLP KG and forms question-query pairs representative of user information need. Prior machine dataset generation framework already exist such as the OVERNIGHT approach [Wang et al., 2015], which is used in the development of KGQA datasets such as LC-QuAD [Trivedi et al., 2017; Dubey et al., 2019] and GraphQuestions [Su et al., 2016], and will be used as an inspiration to design this dataset generation framework. Using the framework, DBLP-QuAD, a semantic parsing dataset, will be created for the DBLP KG.

The chosen scholarly KG for this thesis is the DBLP KG. The scholarly KG was chosen due to its manageable size and focused domain which allows for control over the composition of questions by type and complexity to be included in DBLP-QuAD. Another factor in choosing this scholarly KG is its easy availability as RDF graph[2].

The second task is developing a baseline semantic parsing model for DBLP-QuAD. Current state-of-the-art semantic parsing models fine-tune pre-trained models such as T5 [Raffel et al., 2020]. Baner-

[2]`https://blog.dblp.org/2022/03/02/dblp-in-rdf/`

jee et al. [2022] shows that the T5 model performs the best for the LC-QuAD dataset [Dubey et al., 2019]. In this work, following Banerjee et al. [2022], T5 is fine-tuned on DBLP-QuAD and its performance is analysed.

1.3 Research Questions

The main research questions explored in this thesis are listed below. The questions stem from the discussion above.

- **RQ1:** Can a suitable KGQA dataset be built with an automated dataset generation framework?
- **RQ2:** How do the current SOTA KGQA models fare against the developed dataset?

1.4 Thesis Outline

This thesis begins with necessary background information and overview of related research. Chapter 1 of the thesis provides the some background and outlines the objective of this research work. Chapter 2 discusses the relevant theoretical concepts in reference to the thesis. Chapter 3 presents an overview of existing research on the development and scope of KGQA datasets.

The following chapters then detail the methodology and experiments carried out in the thesis. Chapter 4 delineates the dataset generation framework and describes each component and processes in detail. Chapter 5 describes the development of a semantic parsing baseline model for the generated dataset and reports on its performance.

Finally the last chapters summarize the research work and discuss its shortcomings. Chapter 6 describes the limitations of the KGQA dataset generation framework and the shortcomings of DBLP-QuAD. Chapter 7 provides a concluding remark. The source code of the thesis can be found at `https://github.com/awalesushil/DBLP-QuAD` and the DBLP-QuAD can be downloaded from `https://doi.org/10.5281/zenodo.7643971` or using Huggingface's datasets library[3]. The DBLP RDF graph used in this work can be found at `https://doi.org/10.5281/zenodo.7638511`

[3] `https://huggingface.co/datasets/awalesushil/DBLP-QuAD`

2 Background

This chapter describes the foundational concepts for the task of QA and development of a KGQA dataset. Section 2.1 describes what question answering task entails and discuss the different approaches to the task. Section 2.2 dives deeper into knowledge-based question answering and the various approaches within. Section 2.3 discussess different aspects of a KGQA dataset. Section 2.4 discusses the DBLP database and compares it with other scholarly KGs.

2.1 Question Answering

Question Answering is the task of learning to predict answers to natural language questions. In this NLP task, given a question, "How tall is Mount Everest?" the objective is to provide the correct answer (*here, 8,848 meters*). Question answering systems serve the purpose of fulfilling human information needs [Jurafsky and Martin, 2022]. The users convey their information needs using natural language questions and the system finds the answers to the questions from a data source. As such, QA can be considered an advanced form of information retrieval [Cao et al., 2010].

Question answering systems can be developed using several paradigms, among which two major paradigms are information-retrieval-based (IR-based) and knowledge-based. IR-based QA systems, also known as open domain QA, learn to retrieve relevant documents and then use machine-reading comprehension techniques to extract or generate answers from unstructured text documents [Wang et al., 2017; Yu et al., 2018]. On the other hand, knowledge-based systems rely on a large, structured knowledge base to find the answers [Banerjee et al., 2022; Saxena et al., 2021]. Recently, a new paradigm has evolved, called closed-domain QA, where a pretrained language model is queried directly to answer the questions [Roberts et al., 2020]. We briefly discuss these different paradigms in Sections 2.1.1, 2.1.2, and 2.1.3.

In question answering systems, questions can be various types including long-form questions (questions starting with "why" that require long explanation answers) and community question answering such as Quora or Stack Overflow [Jurafsky and Martin, 2022]. Moreover, question answering systems can also be multi-modal. Visual question answering is an active area of research [Wu et al., 2017]. In NLP most question answering systems, however, focus only on factoid-questions, questions that can be answered with facts [Jurafsky and Martin, 2022]. "What is the capital of Germany?", "How many countries are in the European Union?" are some examples of factoid questions. In this thesis as well, only factoid questions in the computer science scholarly domain are considered.

Question Types

Factoid questions can be further classified into various types based on intent of the question. Some of these question types are discussed below.

- **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 Fact**: 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 whether 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 inverting a boolean question two times. 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.”
- **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?”

2.1.1 Information-retrieval-based Question Answering

Information-retrieval-based QA systems use techniques from information retrieval to search through large collections of documents or the web to find relevant information and generate answers. IR-based systems use a two-stage **retrieve and read** model shown in Figure 2.1.

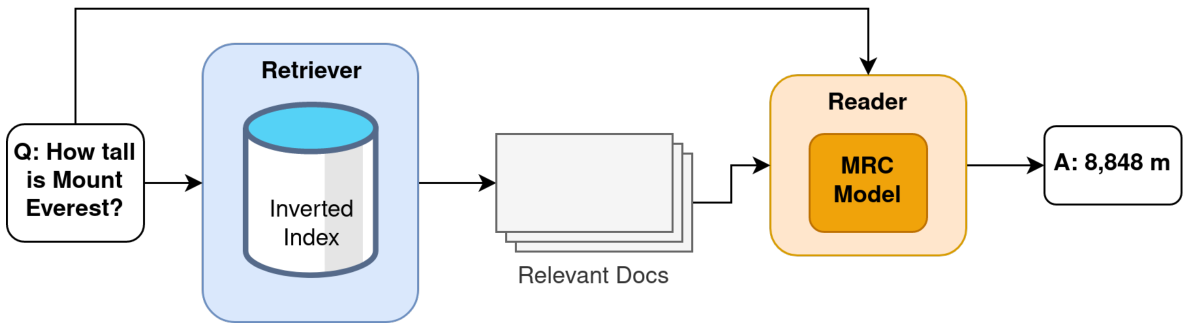


Figure 2.1: Schematic representation of IR-based QA process

First stage - Retrieval: In the first stage, the system retrieves relevant text passages P from a document collection D. A prominent approach to store these documents is in an inverted index. An inverted index stores a mapping between a term and a list of documents that contain the term [Jurafsky and Martin, 2022]. An inverted index makes the retrieval process efficient, and allows the storage of useful metadata such as term frequency, document frequency, and frequency of a term in a document. Using these metadata, retrieval algorithms such as TF-IDF [Jones, 2004] or BM25 [Robertson and Jones, 1976] can be used to retrieve the relevant documents. However, these classic algorithms do not incorporate semantics, hence dense vector retrieval approaches have also been used. In dense vector retrieval, the documents are mapped into an n-dimensional vector space where conceptually similar documents have minimum distance between each other [Deerwester et al., 1990]. Then, a similarity measure such as cosine similarity is used to rank and retrieve the relevant documents.

Second stage - Read: In the second stage, machine reading comprehension models are used to extract answers from the relevant passages P. Given a question q and a passage p, an MRC model reads the passage and extracts the answer a, usually a span of text from the passage. MRC models are trained with triples of question, passage, and answer (q, p, a).

Stanford Question Answering Dataset (SQuAD) [Rajpurkar et al., 2018] is a popular IR-based QA dataset that comprises of passages from Wikipedia with questions and spans of answers from the passage.

One of the main limitations of IR-based QA is that they are unable to answer complex questions that require finding answers in more than one document.

2.1.2 Knowledge-based Question Answering

In knowledge-based question answering, the facts are stored in a structured data store such as a relational database or a knowledge base and the objective is to retrieve the relevant facts/answers from the data store. In comparison to IR-base approaches, knowledge-base QA system are better equipped to handle complex questions i.e. questions that require multiple facts and relationships to find the answers.

In knowledge-based QA, two paradigms used are graph-traversal-based approach and semantic parsing. In graph-traversal-based approach, the knowledge is modeled as a graph, with entities as nodes and relations as edges between nodes. Then the entities and relations in the question are detected and mapped to the KG to answer the question. In semantic parsing (discussed in Section 2.2.2), the natural question is translated into a logical form in the form of λ-calculus, SQL or SPARQL which can be run against the knowledge-base to fetch the answer. In both approaches, entity and relation linking are performed (see Section 2.2.2). [Jurafsky and Martin, 2022]

In knowledge-based question answering, however, the answering capability is constrained by the quality and completeness of the knowledge base.

2.1.3 QA using Language Models

Question Answering using language models, also known as closed-domain QA, is an alternative approach to QA where a pretrained language model such as T5 [Raffel et al., 2020] or GPT-3 [Brown et al., 2020] is queried to find answers from information stored in the model parameters. Roberts et al. fine-tunes the T5 language model to question answering task by training it to output the answer text in the decoder given a question. The recently publicly available ChatGPT[1] is another language model trained to have a dialogue-style conversation capable of answering not just factoid questions but also long-form (why) questions.

Although these LLMs perform competitively, they do not provide any concrete source for their answers and lack interpretability. [Jurafsky and Martin, 2022]

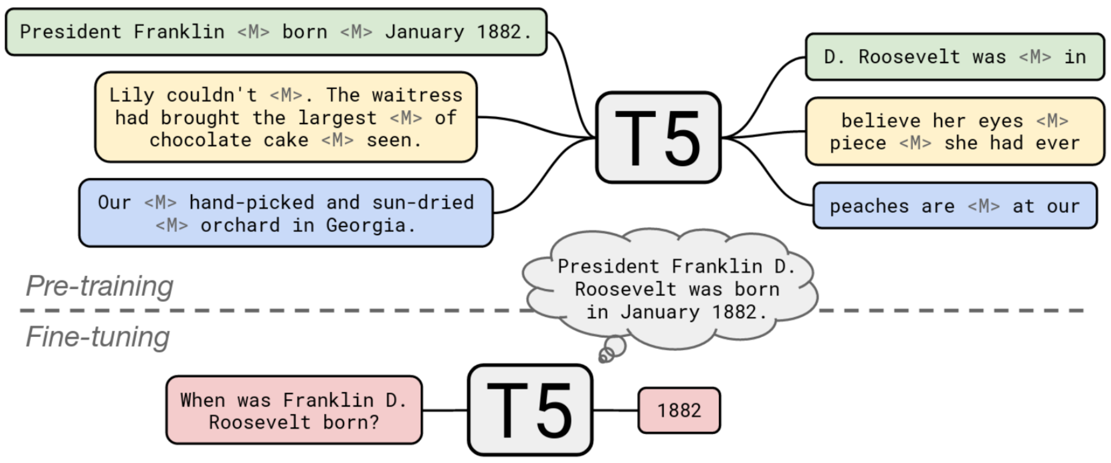

President Franklin <M> born <M> January 1882.
Lily couldn't <M>. The waitress had brought the largest <M> of chocolate cake <M> seen.
Our <M> hand-picked and sun-dried <M> orchard in Georgia.
T5
D. Roosevelt was <M> in
believe her eyes <M> piece <M> she had ever
peaches are <M> at our
Pre-training
Fine-tuning
President Franklin D. Roosevelt was born in January 1882.
When was Franklin D. Roosevelt born?
T5
1882

Figure 2.2: T5 a pre-trained LLM fine-tuned to answer questions without additional context forcing it answer questions using the "knowledge" stored in its parameters during pre-training. Figure from Roberts et al. [2020]

[1]`https://openai.com/blog/chatgpt/`

2.2 Knowledge-based Question Answering

2.2.1 Knowledge Graph

A knowledge graph or a knowledge base is a multi-graph representing a network of real-world entities and the relationships between them. The entities are represented as nodes and the relationship between the entities are represented as edges. A KG is denoted as $KG = (V, E)$, where V is a set of vertices, and E is a set of edges expressed as RDF triples[2]. Here, $E \subseteq (\mathcal{E} \cup \mathcal{C}) \times \mathcal{R} \times (\mathcal{E} \cup \mathcal{C} \cup \mathcal{L})$, where $\mathcal{E}$ is the set of entities, $\mathcal{C}$ is the set of classes, $\mathcal{R}$ is the set of relations and $\mathcal{L}$ is the set of non-unique literal values. A KG has two components, the T-Box and the A-Box [Su et al., 2016]. T-Box, also known as the schema or ontology $\mathcal{O} \subseteq \mathcal{C} \times \mathcal{R} \times \mathcal{C}$ defines the classes and relations used in the KG. A-Box is the set of facts $\mathcal{M} \subseteq \mathcal{E} \times \mathcal{R} \times (\mathcal{E} \cup \mathcal{C} \cup \mathcal{L})$ [Gu et al., 2021].

In a KG, each entity and relations are identified with a IRI[3] (Internationalized Resource Identifier). IRIs, with an RDF graph, uniquely identify an entity or a relation and are enclosed in "<" and ">" and may contain numeric escape sequences. For example, <*https://dblp.org/pid/v/AnnaVilanova*> is an IRI for the Creator entity denoting the creator Anna Vilanova in the DBLP KG.

Some popular examples of public KGs are Wikidata [Vrandecic and Krötzsch, 2014], DBpedia [Auer et al., 2007], and Freebase [Bollacker et al., 2008], and an example of an industrial KG is Google KG[4]. These KGs contain general information and world facts. KGs can also contain only information on a specialized domain such as bibliographic information, medical domain, movies, and so on.

Scholarly Knowledge Graph

A scholarly KG is a specific class of KGs that contains bibliographic information. The scholarly KGs stores information on researchers, research publications, and conferences and journals. Some well-known scholarly KGs are the Microsoft Academic Graph[5], OpenAlex [Priem et al., 2022], ORKG [Jaradeh et al., 2019] and DBLP [Ley, 2002] (discussed in Section 2.4).

2.2.2 Semantic Parsing

Semantic parsing is defined as the task of mapping a natural language question into a logical form to produce an answer [Jurafsky and Martin, 2022]. The logical form can be λ-calculus [Berant et al., 2013], SQL [Zelle and Mooney, 1996], or SPARQL, and the data store against which the logical form is executed can be relational or graph data store. This task has, in the past, been achieved by rule-based systems [Dubey et al., 2016]. More recently, neural network and machine learning based methods have gained popularity [Chakraborty et al., 2021]

The task of semantic parsing, typically comprises of the following steps [Chakraborty et al., 2021].

[2]https://www.w3.org/TR/n-triples/
[3]https://www.w3.org/TR/rdf11-concepts/#dfn-iri
[4]https://blog.google/products/search/introducing-knowledge-graph-things-not/
[5]https://www.microsoft.com/en-us/research/project/microsoft-academic-graph/

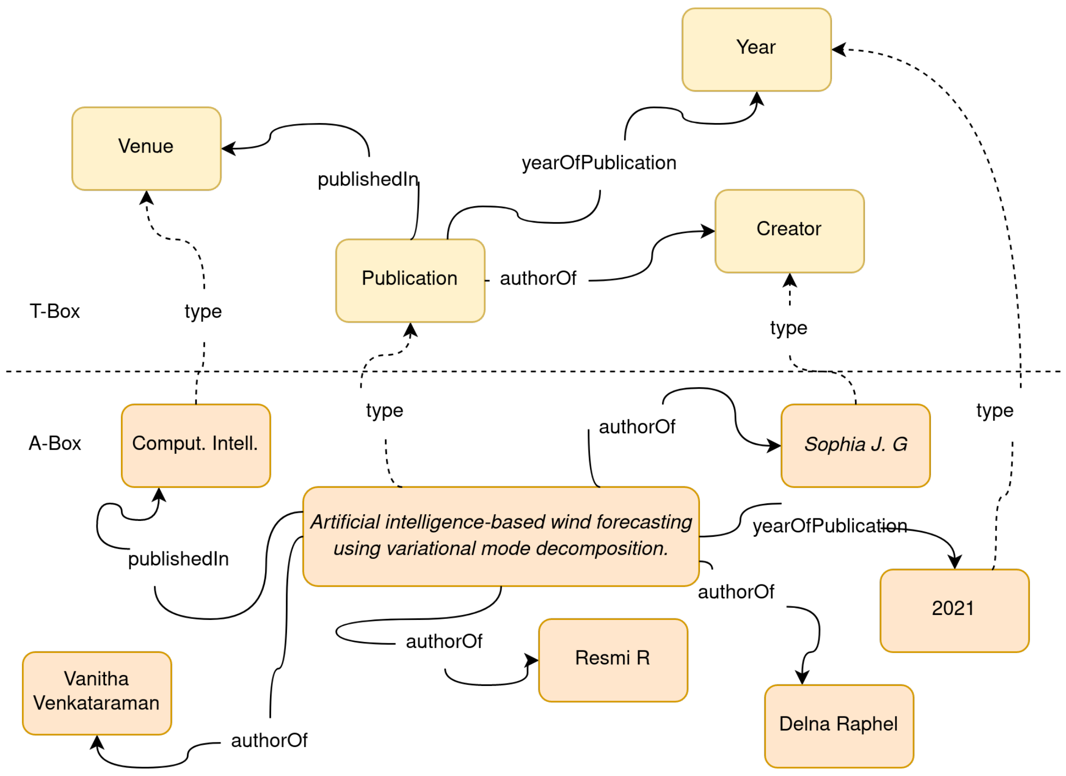


Figure 2.3: Representation of a knowledge graph
The figure shows an example of a scholarly knowledge graph with its components: T-box and A-box

I. Entity and Relation Linking

Entity linking is the task of associating a mention in text with the representation of some real-world entity in a KG [Ji and Grishman, 2011]. In general, entity linking is a two step process. The first step is to identify the entity span in a text, followed by, if required, disambiguation of the entity or to link the entity to a KG. Relation linking is the same process but linking relation spans to predicates in a KG. However, unlike entities that are represented by distinguishing noun phrases, relations are expressed by noun and verb phrase patterns that use less distinguishing words. The entity and relation spans are mapped to the entity and relation IRIs respectively.

Most modern KGQA systems externalize the task of entity linking by employing a standalone entity linking system like EARL [Dubey et al., 2018] for DBpedia KG.

II. Identifying Special Operators

In addition to the entities and relations, semantic parsing also includes the task of identifying additional operators and special functions. Based on the question, different operators/functions need to be identified. For example, questions that ask about frequency need to be mapped to the *COUNT* operator for SPARQL. Other operator/functions may include aggregate functions. However, compared to the entity and relation sets, the set of operators/functions is often fixed and small.

III. Query Building

With the identification of entities, relations and operators/functions, the next step involves arranging the components to form a valid logical form that is semantically equivalent to the corresponding natural language question.

Typically, neural-network-based approaches are end-to-end and performs all these sub-tasks in a single process. Although, the entity and relation linking are often performed separately. The neural-network-based models are prediction models that are fitted on a dataset, and trained as demanded by the architecture. Chakraborty et al. [2021] categorise the semantic parsing models into three categories, namely: (i) classification, (ii) ranking, and (iii) translation.

In classification-based approaches [Mohammed et al., 2018; Petrochuk and Zettlemoyer, 2018], given a natural language question, text-classification methods are used to predict the different parts of a target formal query. The models often assume a fixed structure for the formal query and as such are better suited for only single fact questions.

Ranking-based methods [Bordes et al., 2014; Yin et al., 2018], on the other hand, do not make the assumption that all fixed queries follow a fixed structure. These methods employ some searching mechanism to retrieve a smaller set of candidate formal queries for a given natural language question, and then use neural-network-based ranking models to select the top matching candidate.

In translation-based methods [Guo et al., 2018; He and Golub, 2016], semantic parsing is setup as a translation problem where the task is to translate the natural language question into a semantically equivalent logical form. In this method, neural sequence-to-sequence models learns to generate a sequence of tokens to form a valid logical form.

2.2.3 Logical forms

SPARQL

SPARQL[6] is one of the logical forms that can be used to query data stored in RDF format. SPARQL which stands for SPARQL Protocol and RDF Query Language is used to retrieve and manipulate data stored in RDF format. A SPARQL query, shown in Listing 2.1, comprises of triple patterns. The triple patterns are similar to RDF triples however the pattern may contain a variable (starts with ? symbol). When running a SPARQL query against a RDF graph, the triple patterns are matched with a subgraph of the RDF graph. The match occurs when the RDF triples from the subgraph may be substituted for the variables and the result is RDF graph equivalent to the subgraph.

```
SELECT ?books ?year WHERE {
    ?books authoredBy 'JKRowling' . ?books publishedIn ?year .
}
```

Listing 2.1: An example of a SPARQL query

[6]https://www.w3.org/TR/sparql11-query/

The SPARQL query in Listing 2.1 consists of two parts: the *SELECT* clause describes the variables to appear in the result and the *WHERE* clause specifies the triple pattern to match against the RDF graph. SPARQL query may also contain additional operations such as *ORDER BY* to order the results, *LIMIT* to limit the number of rows in result, *FILTER* to filter the results, and more. Additionally, SPARQL query may also use the *ASK* clause to determine if a triple pattern exists in the RDF graph or not.

SPARQL query are also equipped with aggregate functions that allow advanced operations on the RDF graph such as:

- *COUNT:* Count the number of rows of the results
- *SUM, AVG:* Sum or average of an element of the results
- *MIN, MAX:* Minimum or maximum of an element of the results

[See `https://www.wikidata.org/wiki/Wikidata:SPARQL_tutorial` for more]

2.2.4 Evaluation

For semantic parsing evaluation, the evaluation metrics are drawn from the domain of information retrieval. A popular evaluation metric is F1 score, which is calculated based on the answers retrieved by running the gold query and the query produced by the semantic parsing model.

F1 score

The F1 score is calculated as follows:

$$precision = \frac{|\text{relevant answers} \cap \text{retrieved answers}|}{|\text{retrieved answers}|} \tag{2.1}$$

$$recall = \frac{|\text{relevant answers} \cap \text{retrieved answers}|}{|\text{relevant answers}|} \tag{2.2}$$

$$f1 = 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}} \tag{2.3}$$

where, *relevant answers* are the answers present in the gold dataset and the *retrieved answers* are the all answers produced by a query. The final F1 score is computed by averaging the F1 score for each query in the test dataset. [Jurafsky and Martin, 2022]

Exact string match accuracy

Another straightforward way to evaluate semantic parsing models are by comparing the gold and the generated queries character by character after removing white spaces. Here, as well, the average accuracy over all queries is reported.

2.3 KGQA Datasets

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 KGQA datasets exist [Perevalov et al., 2022b]. The datasets themselves have various defining characteristics that relate to size, questions type, answer type, knowledge graph, generalization, etc. A summary of various KGQA datasets is presented in Table 3.1.

Naturalness. The naturalness of a KGQA dataset comes at various levels from synthetically generated questions to natural elicited questions.

Answer/Logical form. KGQA datasets can have either question-answer pairs or question-logical form pairs. End-to-end KGQA datasets have question-answer pairs and can be used to solve both KGQA and end-to-end question answering. On the other hand, semantic parsing KGQA datasets have question-logical form pairs, and are particularly used to solve semantic parsing problems. The logical form can be λ-calculus, SQL, or SPARQL.

Size. Larger the size of a KGQA dataset, the greater the variation of the natural question representation. Moreover, large KGQA datasets are suited to train large neural network models.

Question types. KGQA datasets can have either simple questions or complex questions or both. The various question types are discussed in Section 2.1.

Paraphrases. KGQA system should be robust to paraphrases. [Su et al., 2016] Hence, a KGQA dataset should also have paraphrases for the natural questions.

Multilingual. Most KGQA datasets are monolingual with only English questions. However, some dataset also include natural questions in multiple languages. **QALD** [Usbeck et al., 2017] and **MinTaka** [Sen et al., 2022] are two such examples.

Knowledge graph. Most KGQA datasets are grounded to one Knowledge Graph. However, there are some exceptions such as LC-QuAD 2.0, which is grounded to both Wikidata and DBpedia. [Dubey et al., 2019]

Generalization. Another important aspect of KGQA datasets is generalization. Most of the KGQA datasets adhere to i.i.d. (independent and identically distributed) generalization but fail to follow compositional, and zero-shot generalization. The three-level of generalization, discussed in [Gu et al., 2022], are explained below.

Let S be a set of schema items that contains relations R, classes C and operators Y in a logical form. Let, $S_{train} \subseteq S$ be a set of schema items in the training split Q_{train} of the dataset and S_q be for a question q in the test split Q_{test}. For each question q each level of generalization is defined as follows:

- **I.I.D. generalization:** Here, the test questions follow the training distribution hence all relations, classes and logical form constructs have been seen in the training split except for the actual entities and literals.

$$\forall q \in Q_{test}, S_q \subset S_{train}$$

- **Compositional generalization:** Here, the relations and classes have been observed in the training split but not the specific composition or the specific logical form.

$$\forall q \in Q_{test}, S_q \subset S_{train}; q \not\subset Q_{train}$$

- **Zero-shot generalization:** Here, for a given question, either one of the relations, classes or logical form operators are not seen in the training split.

$$\forall q \in Q_{test} : \exists s \in S_q, s \in S \mid S_{train}$$

2.4 DBLP - Data Bases and Logic Programming

DBLP is a computer science bibliography service that provides bibliographic information on major computer science journals and proceedings. It is currently maintained by Schloss Dagstuhl[7] and is one of the most comprehensive online bibliographies in the computer science field. DBLP is freely available to the public both as an online service[8] and downloadable XML and RDF data dumps. Researchers, students, and professionals use it as a tool to search for research works in the field and to access bibliographic details of these research works. As of February 22, 2022, DBLP had over six million computer science publications from more than 7,600 conferences and journals credited to more than 2.9 million researchers[9].

2.4.1 DBLP as a Computer Science Bibliography Service

DBLP, which stands for Data Bases and Logic Programming, was created in 1993 by Michael Ley at the University of Trier in Germany [Ley, 2002]. 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.

[7]`https://dagstuhl.de`
[8]`https://dblp.org/`
[9]`https://blog.dblp.org/2022/02/22/6-million-publications/`

In its early years, DBLP started as a small collection of HTML files with hyperlinking each author to a page that enumerates all the publications for the author. Later as the number of records grew, it was migrated to standard XML format with assigned unique IDs to each publication as shown in Listing 2.2. The XML records were available as a service through the MG information retrieval system with a substring matching search function.

```
<article key="journals/eswa/FangWTY23" mdate="2022−12−05">
    <author orcid="0000−0002−1340−6937">Haichuan Fang</author>
    <author orcid="0000−0002−5033−5576">Youwei Wang</author>
    <author>Zhen Tian</author>
    <author orcid="0000−0001−7027−8313">Yangdong Ye</author>
    <title>
      Learning knowledge graph embedding with a dual−attention embedding
      network.
    </title>
    <pages>118806</pages>
    <year>2023</year>
    <volume>212</volume>
    <journal>Expert Syst. Appl.</journal>
    <ee>https://doi.org/10.1016/j.eswa.2022.118806</ee>
    <url>db/journals/eswa/eswa212.html#FangWTY23</url>
</article>
```

Listing 2.2: An XML record example from DBLP

Today, the DBLP service is a free publicly accessible online service through `https://dblp.org`. It provides alphabetically sorted lists of research publications, major computer science journals and conferences, and authors. The metadata information about these conferences and journals, research publications, and authors can be downloaded in BibTex, RDF N-Triples, RDF Turtle, RDF/XML, and XML formats. Moreover, the website also provides a text-based search interface to search for these listings online.

2.4.2 DBLP as a scholarly Knowledge Graph

In March of 2022, DBLP released an RDF version of its XML files[10] under CC0 1.0 Public Domain Dedication license[11]. The RDF data is available both as an API and persistent downloadable monthly RDF dump. The RDF data are updated daily and are in sync with the DBLP XML files. The DBLP RDF data models a person-publication graph shown in Figure 2.4

As seen in the Figure 2.4, 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

[10] `https://blog.dblp.org/2022/03/02/dblp-in-rdf/`
[11] `https://creativecommons.org/publicdomain/zero/1.0/`

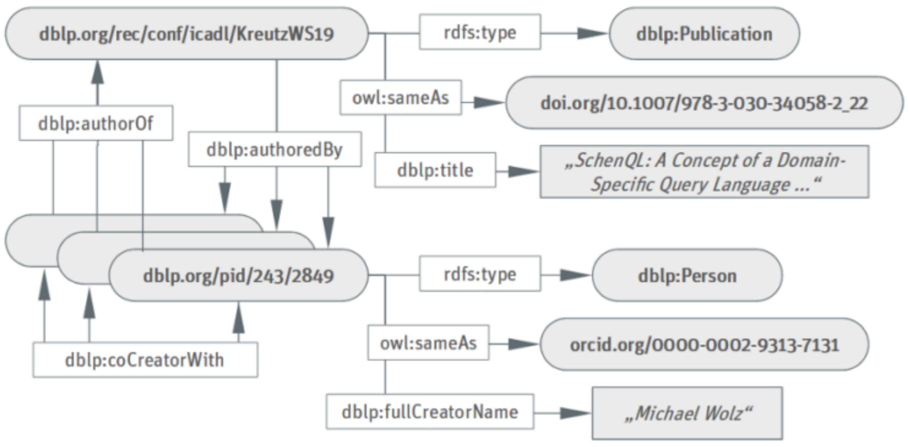


Figure 2.4: DBLP RDF schema

literals. 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 Virtuso Server can be setup to run SPARQL query against the DBLP KG.

3 Related Work

Several QA datasets exist, both for IR-based QA [Rajpurkar et al., 2018; Kwiatkowski et al., 2019] and KGQA [Trivedi et al., 2017; Sen et al., 2022] approaches. However, most of these datasets cover general knowledge topics. The main reason for this are the large publicly available knowledge sources such as Wikipedia[1] for IR-based QA, and knowledge bases such as Freebase [Bollacker et al., 2008], Wikidata [Vrandecic and Krötzsch, 2014], and DBpedia [Auer et al., 2007] for KGQA.

In the following sections, we discuss on scholarly and the other KGQA datasets, and their development process.

3.1 Scholarly KGQA

ORKG-QA benchmark [Jaradeh et al., 2020] 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.

ORKG-QA benchmark is designed to tackle QA on tabular data and includes question types that can be asked in the context of tables in the scholarly literature. The benchmark includes questions about the content, methodology and processes discussed in a research publication. “Does paper ‘Disambiguating authors in citations on the web and authorship correlations’ employ cosine similarity?” is an example from the dataset.

ORKG-QA benchmark does not include questions on bibliographic metadata which is the focus of DBLP-QuAD. Further, ORKG-QA is relatively small in size with only 100 questions in English and covers only 100 research publications.

To the best of our knowledge, ORKG-QA is the only scholarly KGQA available to the public. Most of the other KGQA datasets, includes questions on general information and world facts.

3.2 KGQA Dataset Generation Process

Several approaches have been deployed to generate the KGQA datasets on general information. These approaches range from manual to machine generation. However, most datasets lie in between and use a combination of manual and automated process. We group the KGQA datasets based on the main process used for the generation and discuss these processes in detail.

[1]`https://wikipedia.com`

3.2.1 Manual

In this generation process, a few domain experts manually create questions-logical form pairs.

Free917 [Cai and Yates, 2013] is a semantic parsing dataset developed on Freebase Commons a subset of Freebase. It consists of 917 natural questions taken from 81 domains and corresponding logical forms in λ-calculus [Church, 1932]. The questions pertain to domains such as film and business. The dataset was developed manually by two native English speakers and was used to test the semantic parser developed by the authors. The dataset only covered a limited number of domains compared to the vast knowledge and scope of Freebase, and it was difficult to scale up as experts were required to annotate the logical forms.

QALD (Question Answering over Linked Data) is a popular KGQA benchmark that has been running every year for the past decade. QALD consists of manually written natural questions and their corresponding SPARQL queries. The natural questions have multiple translations in multiple languages written by native speakers. The latest QALD benchmark is the QALD-9-plus [Perevalov et al., 2022a] which consists of around 500 questions in eight different languages for the Wikidata. Although the quality of questions in these datasets have been high, the size of the dataset in this benchmark has always been small.

The process yields a high quality of KGQA dataset but it is difficult to scale to large KGs. However, for KG smaller in size and KG focusing on niche domains, manual generation can be a good approach.

3.2.2 Third party sources

Some KGQA datasets, collect questions from third-party sources such as a commercial search engine or suggestion APIs. The collected questions are from real-world users and hence are in natural form.

WebQuestions [Berant et al., 2013] is another QA dataset developed to match the scale and scope of Freebase. The dataset consists of 5,810 natural questions and corresponding answers grounded to Freebase. The questions in the dataset were obtained using the Google Suggest API covering more domains than Free917. Questions beginning with *Wh*-word and containing exactly one entity were collected. One million questions were visited and randomly 100,000 questions were selected and submitted to Amazon Mechanical Turk. Then, the AMT workers answered these questions using Freebase page of the questions' entity, resulting in 5,810 question-answer pairs.

ComplexQuestions [Bao et al., 2014] is collection of natural questions collected using three month query log from a practical search engine. The dataset also includes complex questions from other KGQA datasets WebQuestions [Berant et al., 2013] and [Yin et al., 2015] amounting to a total of 2,100 natural questions with answers grounded to Freebase KG. The collected questions were filtered to remove questions containing pronouns and word length lower than seven and higher than 20 as short queries are seldom complex, and long queries are usually difficult to answer. Further, only 10 percent of questions were sampled and the entities were detected using an entity linker. Questions without any detected entities were further filtered out. Finally, answers from Freebase were added to each questions.

3.2.3 Existing datasets

A few KGQA dataset were generated by using other existing datasets. Mostly, the KGQA dataset are migrated from one KG, often discontinued KG such as Freebase, to another live and maintained KG such as DBpedia and Wikidata. Some KGQA datasets only target a certain domain or type of questions (e.g. temporal) and hence, draw a subset of questions from an existing dataset.

WebQuestionsSP [Yih et al., 2016] is a semantic parsing dataset developed by using questions from WebQuestions [Berant et al., 2013]. The dataset provides SPARQL queries for 4,737 questions from WebQuestions executable against the Freebase KG. 1,073 of the original questions in WebQuestions were skipped as these questions were ambiguous or without clear intent and hence not answerable using SPARQL.

Yih et al. [2016] developed a dialogue-like user interface which allowed the five expert human annotators to annotate the data in stages. In the first stage, given a question, entities were marked using an entity linking system and allowed the user to select a *topic* entity (the main entity the question is about). In the second stage, the annotators were instructed to select an appropriate Freebase predicate that linked the *topic* entity and the answer. Then, similar to Yih et al. [2016] the corresponding SPARQL query is anchored to the *topic* entity and the SPARQL query is generated such that it best represents the relation between the *topic* entity and the answer entity.

SimpleDBpediaQA [Azmy et al., 2018] is created by mapping entities and predicates in SimpleQuestions [Bordes et al., 2015] from the defunct KG Freebase to DBpedia. The dataset comprises of 43,086 questions and corresponding answers. Contrary to the name, the dataset also includes complex questions as required during the migration process. To create this dataset, the entities were first mapped from Freebase to DBpedia by running a SPARQL query on DBpedia as the two KGs are linked through the predicate `http://www.w3.org/2002/07/owl#sameAs`. Next, the paths between the topic entity and the answer entity were enumerated by running another SPARQL query over DBpedia, which resulted in both single and multiple predicates. In this process, the direction of the predicates was also stored as it is important for the QA task. Finally, not all predicates are valid so a rule-based filter was created manually annotated by human annotators.

SimpleQuestionsWikidata [Diefenbach et al., 2017], similar to SimpleDBpediaQA [Azmy et al., 2018], is created by mapping entities and predicates from Freebase to Wikidata. The entities are mapped automatically using the mapping process in Tanon et al. [2016], and the predicates are mapped using handmade mapping. During the mapping process, not all entities and predicates have a direct mapping from Freebase to Wikidata, hence some information is lost. These questions were segregated into a separate dataset. The questions that were answerable amounted to 21,957 questions, which were accompanied by the triples and SPARQL queries.

TempQuestions [Jia et al., 2018] is a collection of 1,271 temporal questions paired with answers linked to Freebase. Time-relation questions from Free917 [Cai and Yates, 2013], WebQuestions [Berant et al., 2013] and ComplexQuestions [Bao et al., 2014] were manually selected to form this dataset. Jia et al. use a combination of existing taggers, dictionaries, and lexico-syntactic patterns to automatically detect explicit and implicit temporal expressions and ordinal questions in the question string of the three datasets. Further, the extracted questions and the existing answers were manually verified to form the final dataset.

ComplexWebQuestions [Talmor and Berant, 2018] is a collection of 34,689 complex question paired with answers and SPARQL queries grounded to Freebase KG. The dataset builds on WebQuestionsSP [Yih et al., 2016] 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.

MCWQ [Cui et al., 2022] (Multilingual Compositional Wikidata Questions) is a multilingual semantic parsing dataset generated by migrating CFQ [Keysers et al., 2020] from Freebase to Wikidata. The MCWQ dataset contains 124,187 question-query pairs with translation of natural questions in three other languages Hebrew, Kannada and Chinese generated using Google Cloud Translate[2]. For the migration process, first the Freebase predicates are mapped to Wikidata. Secondly, following [Keysers et al., 2020], the entities are mapped to Wikidata by executing the SPARQL queries by replacing the entity placeholders in the questions.

3.2.4 OVERNIGHT approach

The OVERNIGHT (ON) approach is a semantic parsing dataset generation framework introduced by [Wang et al., 2015]. 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 are grammatically incorrect but successfully carry the semantic meaning. Lastly, the canonical questions are converted into natural forms via crowdsourcing. The following datasets use this or a variation of this approach.

SimpleQuestions [Bordes et al., 2015] is a large collection of human annotated simple questions grounded to the Freebase KG. The dataset consists of 108,442 English questions written by English-speaking human annotators with a corresponding fact from the Freebase knowledge graph. The knowledge base used was a smaller subset of the Freebase (FB2M), which contained two million entities and five thousand predicates. First, the facts where the $(subject, predicate)$ occurrences crossed the threshold of 10 were filtered out. Then, a sample of these selected facts were present to the human annotators along with hyperlinks to `freebase.com` to provide context while framing a question. The annotators were instructed to phrase a question involving the subject and the predicate of the fact with the answers being the object.

GraphQuestions [Su et al., 2016] 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.

[2]`https://cloud.google.com/translate`

In the first step, a query template is generated from the KG ontology. For this, a random class node is selected and then grown iteratively by adding adjoining nodes and edges. Additionally, for each query at most one function is included. These functions include counting (`count`), superlative (`max, min, argmax, argmin`), and comparative ($>$, $<$, $\leq$, $\geq$). Second, the query template is grounded with compatible entities and literals to generate graph queries. Further, redundant graph query components and graph queries encapsulating rare information were filtered out. In the final stage, the remaining graph queries were used to generate canonical questions. This process was screened and verified by graduate students. Finally 160 AMT workers paraphrased the canonical questions to produce natural questions, and further two additional paraphrases for each question were added.

LC-QuAD 1.0 [Trivedi et al., 2017] is another semantic parsing dataset for the DBpedia KG. LC-QuAD 1.0 is larger in size compared to QALD with 5,000 natural language English questions and corresponding SPARQL queries. In this work, the dataset generation process is inverted and automated. The 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, 2-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. The human paraphrases are reviewed by an independent reviewer to ensure a higher quality of data.

LC-QuAD 2.0 [Dubey et al., 2019] 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. The questions and paraphrases are validated by crowd workers to maintain quality. LC-QuAD 2.0 has more questions and more variation as compared to LC-QuAD 1.0. Additionally, paraphrases to the natural questions are also included.

GrailQA [Gu et al., 2021] extends the approach in [Su et al., 2016] to generate 64,331 question-S-expression pairs grounded to the Freebase Commons KG. Here, S-expression are linearized forms of graph queries. Following Su et al., 2016, 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 AMT workers write five natural paraphrases for each canonical question, which are further validated by multiple independent crowd workers. Additionally, appropriate surface forms of the entities are also selected by the crowd workers.

GrailQA is also designed to support three-level of generalization *i.i.d*, *zero-shot*, and *compositional*. The dataset is split into train/valid/test sets containing 70%/10%/20% of the data in which validation and test sets include questions from held-out domains not covered in training (*zero-shot*), questions with canonical logic forms not covered in training (*compositional*), and the remaining questions are randomly sampled from training *i.i.d.* The questions covering *compositional* and *i.i.d* have the involved schema items covered in training.

KQA Pro [Cao et al., 2022] 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. The dataset maintains a high level of linguistic variety of paraphrases by filtering out paraphrases with small edit distance with the canonical question.

3.2.5 Synthetic generation

Lately, some KGQA datasets have been fully developed synthetically without any human participation at all. These synthetic generation processes allow more control during the generation process and can be scaled to any arbitrary size.

MetaQA [Zhang et al., 2018] is a collection of more than 400 thousand synthetically generated questions in text and audio format for WikiMovies KG. The authors manually created a small set of question templates based on triples from the WikiMovies KG. Then based on the triples, 21 question types for 2-hop triples and 15 question types for 3-hop triples were created. Next, a number of questions were synthetically generated for each question types. The dataset also includes the simple questions from the WikiMovies KGQA dataset. Furthermore, the questions were translated to French and back to English using a neural translation model to generate syntactically varied paraphrases. Additionally, the text questions were transcribed using Google text-to-speech service to provide the audio version of the dataset.

CFQ [Keysers et al., 2020] (Compositional Freebase Questions) is a large collection of synthetically generated semantic parsing dataset consisting of simple natural language questions with corresponding SPARQL query against the Freebase KG. Additionally, CFQ also includes text answers and an intermediate logical form. 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.

CRONQuestions [Saxena et al., 2021] is a large collection of 410 thousand temporal questions automatically generated using a subset of Wikidata KG. This subset of Wikidata contains all facts with temporal annotations introduced by Lacroix et al. [2020]. The resulting temporal KG consists of 323 thousand facts, 125 thousand entities and 203 relations.

The dataset generation process starts with the selection of five most frequent relations from the temporal KG. Using these relations 30 unique seed templates were created which were then paraphrased by human annotators to create 246 unique templates. Next, a monolingual paraphraser was used to generate 654 templates using the 246 templates. These 654 templates were then verified by annotators and then used to produce 410,000 unique question-answer pairs by automatically replacing the entities with entity aliases from Wikidata.

3.2.6 Crowdsourcing

Mintaka [Sen et al., 2022] is a complex, natural, and multilingual dataset developed for building end-to-end question answering systems. Mintaka comprises of 20,000 natural questions in English elicited by crowd workers and answers grounded to Wikidata. Additionally, these English questions have translations in Arabic, French, German, Hindi, Italian, Japanese, Portuguese, and Spanish translated by professionals.

In Mintaka, questions were naturally elicited from AMT workers. First, the AMT workers elicited five complex questions and the answers for a topic category. Second, the entities in the answers were identified and then linked to Wikidata. Finally, AMT workers also marked the entities in the question text and grounded these entities to the knowledge graph.

Although using crowd workers to elicit questions and answers results in a natural high quality dataset, the approach is not adaptable for a niche domain such as scholarly question answering where the crowd workers may not be familiar with research paper authors and other bibliographic metadata.

A summary of the discussed KGQA datasets is given in Table 3.1.

In this work, the dataset generation process followed in LC-QuADs and CRONQuestions are loosely followed to create a large scholarly KGQA dataset for the DBLP KG.

Dataset	KG	LF	Size	PP	QG
Free917	FB	λ	917	×	M
QALD	DBp	SP	558	×	M
WebQuestions	FB	ANS	5.8K	×	GS
SimpleQuestions	FB	T	108K	×	ON
ComplexQuestions	FB	ANS	2.1K	×	SE
WebQuestionsSP	FB	SP	4.7K	×	DS
GraphQuestions	FB	ANS, SP	5.1K	✓	ON
TempQuestions	FB	ANS	1.2K	×	DS
ComplexWebQuestions	FB	ANS, SP	34K	×	DS
CFQ	FB	ANS, SP	239K	×	MG
GrailQA	FB	S-EXP	64K	✓	ON
LC-QuAD 1.0	DBp	SP	5K	×	ON
SimpleDBpediaQA	DBp	ANS	43K	×	DS
LC-QuAD 2.0	DBp, WD	SP	30K	✓	ON
SimpleQuestionsWikidata	WD	SP	21K	×	DS
MCWQ	WD	SP	124K	×	DS
KGA Pro	WD	SP	117K	✓	ON
CRONQuestions	WD	ANS	410K	×	MG
Mintaka	WD	ANS	20K	×	CS
MetaQA	WM	ANS	400K	✓	MG

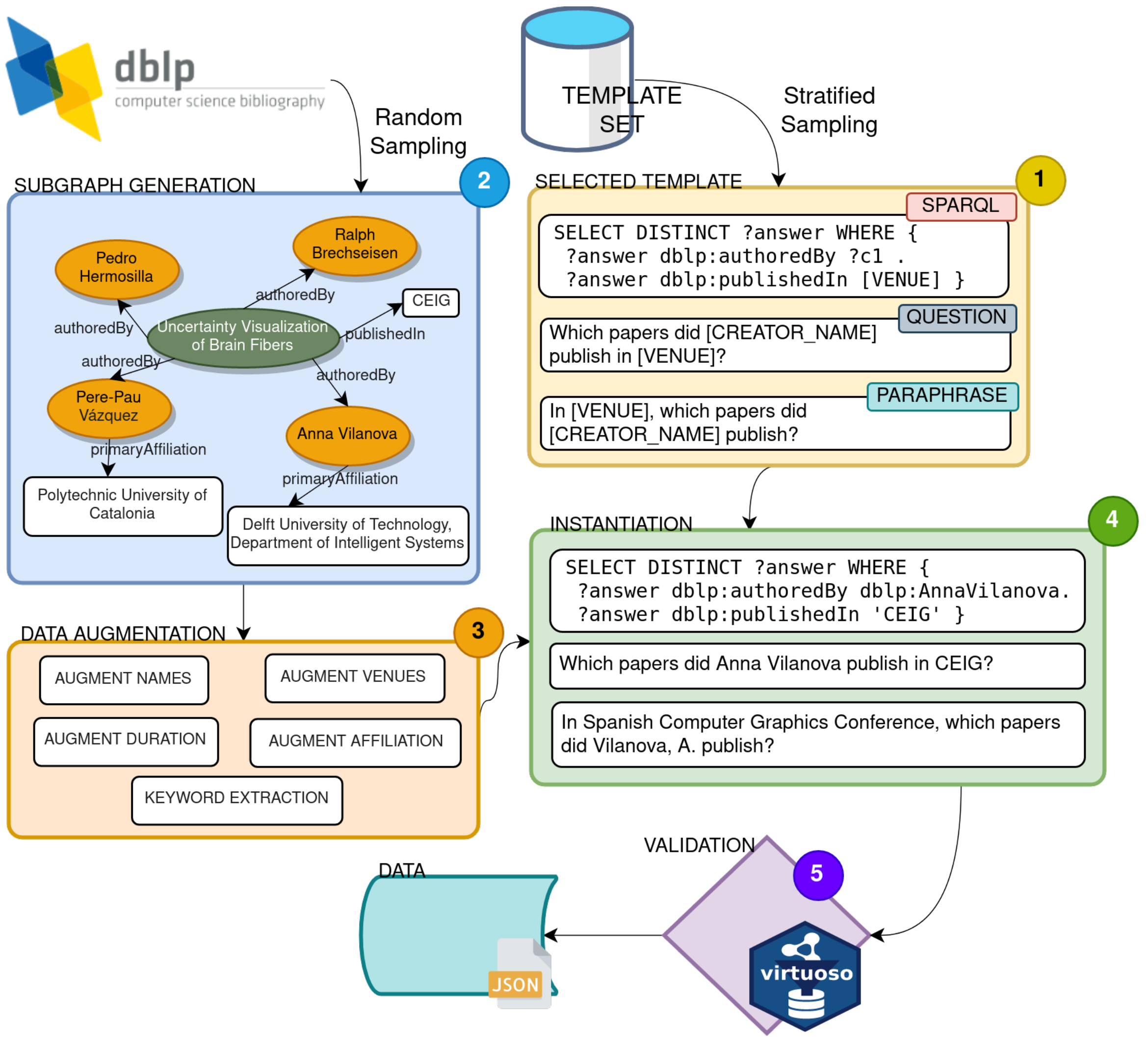
Table 3.1: Summary of the KGQA datasets
Knowledge graph (KG): FB - Freeebase, DBp - DBPedia, WD - Wikidata, WM - Wikimovies
Logical Form (LF): λ - λ-calculus, SP - SPARQL, ANS - Text Answers, T - Triples
Size: K - thousand, M - million; **Paraphrases (PP):** present (✓), absent (×)
Question Generation (QG): M - Manual, GS - Google Suggest API, SE - Search Engine, DS - existing datasets, ON - OVERNIGHT approach, MG - Machine Generation, CS- Crowdsourcing

4 Dataset Generation Framework

This chapter describes the dataset generation framework developed to generate DBLP-QuAD. Each design decision, and the implementation of the dataset generation framework are detailed in this section. Sections 4.1, 4.2, 4.3, 4.4, 4.5, 4.6 describe the processes involved in the dataset generation framework. Section 4.7 describes the characteristics of the generated data.

The main objective in this work 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 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, a large set of question-query pairs (q_i, s_i) suitable for training a neural network semantic parser is generated synthetically.

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, different augmentations of these textual representations are randomly mixed in. The dataset generation workflow is shown in Figure 4.1.



```
SELECT DISTINCT ?answer WHERE {
 ?answer dblp:authoredBy ?c1 .
 ?answer dblp:publishedIn [VENUE] }
```

Which papers did [CREATOR_NAME] publish in [VENUE]?

In [VENUE], which papers did [CREATOR_NAME] publish?

```
SELECT DISTINCT ?answer WHERE {
 ?answer dblp:authoredBy dblp:AnnaVilanova.
 ?answer dblp:publishedIn 'CEIG' }
```

Which papers did Anna Vilanova publish in CEIG?

In Spanish Computer Graphics Conference, which papers did Vilanova, A. publish?

Figure 4.1: Motivating Example of Dataset Generation Framework
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 are (5) filtered based on if they produce answers or not.

4.1 Template Set

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, 98 pairs of valid SPARQL query templates and a set of semantically equivalent natural language question templates are created. The template set was created such that it best represents the user information need while still being answerable using SPARQl query over the DBLP KG. The live RDF data model on the DBLP website follows the schema shown in Figure 2.4. However, the downloaded RDF snapshots have the *coCreatorWith* and *authorOf* predicates missing. Although these predicates are missing, the *authoredBy* predicate can be used to derive the missing relations. The template set is based on the DBLP KG schema of the downloaded RDF graph. The template set covers the two entities creator and publication, and additionally the foreign entity bibtex type. Additionally, the set also covers 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. A tuple also contains 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. The query and question templates consist of placeholder markers instead of IRIs, entity surface forms or literals.

SPARQL query template

The SPARQL query uses full DBLP IRIs for predicates. In the Listing 4.1, the SPARQL query has the DBLP IRI `<https://dblp.org/rdf/schema#authoredBy>`. Further, *SELECT*, *DISTINCT*, and *WHERE* are SPARQL keywords, *?c1* and *?c2* are placeholders for the IRI of DBLP publication entity and *?answer* is a SPARQL variable to represent the answer, which can be a DBLP entity IRI or a literal. The list of entity placeholders and the relations used in the query template are further also saved separately. The full list of SPARQL functions and placeholders used are shown in Appendix 9.

Natural language question templates

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, instruction-style paraphrases are also included. An example list of paraphrases used can be seen in the Listing 4.1. The paraphrases use a placeholder string enclosed by square brackets ([]). In the above example, the placeholder strings *[CREATOR_NAME]* and *[OTHER_CREATOR_NAME]* are used for names of the research paper authors.

```
{
    "id": "TC12",
    "query": {
        "sparql": "SELECT DISTINCT ?answer WHERE {
                        ?answer <https://dblp.org/rdf/schema#authoredBy> ?c1 .
                        ?answer <https://dblp.org/rdf/schema#authoredBy> ?c2
                    }",
        "temporal": False
    },
    "question": {
        "strings": [
            "What are the papers written by [CREATOR_NAME] and
                [OTHER_CREATOR_NAME] together?",
            "What are the publications written by the authors
                [CREATOR_NAME] and [OTHER_CREATOR_NAME] in collaboration?",
            "Which papers did [CREATOR_NAME] and [OTHER_CREATOR_NAME]
                write together?",
            "Which papers did the authors [CREATOR_NAME] and
                [OTHER_CREATOR_NAME] co−write?",
            "Find the papers written by [CREATOR_NAME] and
                [OTHER_CREATOR_NAME] together."
        ],
        "entities": ["?c1", "?c2"],
        "relations": ["<https://dblp.org/rdf/schema#authoredBy>"]
    },
    "test_only": False
}
```

Listing 4.1: An example of a template tuple

Other fields

In addition to the query and paraphrases, the template tuples contain two more properties. First is a boolean indicating whether the SPARQL query template is a temporal query or not. Second is another boolean specifying whether the template tuple is to be held out or not. Certain template tuples are selected to be held out for generating *train* set. These templates are selected such that they can be used to test *compositional* and *zero-shot* generalization during KGQA model development. More details about the different SPARQL keywords, placeholders, and variables used can be found in Appendix 9.

The template tuples are grouped as creator-focused or publication-focused and further grouped under query type subgroups δ for each entity group ϵ. There are 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 2.1 with examples. The distribution of templates per entity and query type are shown in Table 4.1. During dataset generation, for each data instance a template tuple from the template set is sampled using stratified sampling maintaining equal distribution of entity types and query types.

Query Type	Creator	Publication	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 4.1: 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. The step is motivated by the OVERNIGHT approach [Wang et al., 2015] where a canonical question is formulated by drawing a subgraph from the KG. In this dataset generation framework, two subgraphs are sampled from DBLP KG and the literal values from the subgraphs are used to instantiate the template tuples.

Given a graph $G = (V, E)$ where V are the vertices, and E are edges, a subgraph $g = (v, e)$ is drawn where $v \subset V$, $e \subset E$. For the DBLP KG, V are the creator and publication entity IRIs or literal values, and the E are the predicates of the KG.

The subgraph generation process starts with random sampling of a publication entity v_i from the DBLP KG. Only the set of publication entities are drawn 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. Moreover, publication-centered graph can be traversed in both directions to gather the required data values.

With the sampled publication entity v_i, all the predicates e are iterated to extract creator entities v' as well as the literal values. Further, the creator entities are expanded and their literal values are extracted to form a two-hop subgraph $g = (v, e)$ as shown in Figure 4.1.

4.3 Template Instantiation

Using the generated subgraph and the sampled template tuple, the template tuple is instantiated with entity IRIs 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, the creator/publication placeholder markers are instantiated with DBLP creator/publication entity IRIs 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, two random samples are drawn from the set of question templates $q_t^1, q_t^2 \in Q_T$, and each are instantiated 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, alternate string representations of these literal values are randomly inserted in both natural language questions. The data augmentation process allows addition of heuristically manipulated alternate literal representations to the natural questions. A example of an instantiated template is shown in Figure 4.1 (Section 3). Algorithm 1 details the procedure used in template instantiation.

Algorithm 1: Template Instantiation Algorithm

```
Instantiate ($T, S, g$)
    inputs : sampled template $T$; sampled subgraphs $S$; data group to generate for $g$;
    output: filled template $T'$;
    $slots \leftarrow dict()$;
    $Q \leftarrow T.get(questions)$;
    if $g == train$ then
        $Q.pop(1)$;
        $Q.pop(2)$;
    $question, paraphrase \leftarrow Sample(Q, 2)$;
    $query \leftarrow T.get(query)$;
    foreach $placeholder, values \in slots$ do
        foreach $x \in [question, paraphrase, query]$ do
            $selected\_value \leftarrow random.sample(values)$;
            $x' \leftarrow x.replace(placeholder, selected\_value)$;
            $T' \leftarrow x'$;
    return $T'$
```

4.4 Data Augmentation

For the template instantiation process, simple string manipulations are performed to generate alternate literal representations. Then, between the original literal representation and the alternate representation, one is selected randomly to instantiate the natural language questions. For each literal type, different string manipulation techniques are applied, which are described below.

4.4.1 Names

For names four different alternatives are generated, which involved switching parts of names or keeping only initials of the names. Consider the name *John William Smith* as a running example.

- Switch first name and last name. For example, *Smith, John William*
- Only keep initials of first name. For example, *J. William Smith*
- Only keep initials of middle name. For example, *John W. Smith*
- Switch first name and last name and only keep initials of first name. For example, *Smith, J. William*

Further, only partial names are used to instantiate the creator names in disambiguation queries.

4.4.2 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. For this case, a selected list of conference and journals[1] containing the short form and its equivalent full form is used to get the full venue names.

4.4.3 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*. During the instantiation process, between the numerical representation and the textual representation, one is selected randomly for the dummy duration value.

4.4.4 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, the institution names are extracted and between the institution name and the full institution address, one is selected randomly in the instantiation process. For example, between *Universität Hamburg* and *Universität Hamburg, Hamburg, Germany*, one is selected.

4.4.5 Keywords

For disambiguation queries, instead of the full title of a publication only a subpart of it is used by extracting keywords from the title. For this purpose, SpaCy's Matcher API [2] is used to extract noun

[1] `http://portal.core.edu.au/conf-ranks/?search=&by=all&source=CORE2021&sort=atitle&page=1`
[2] `https://spacy.io/api/matcher/`

phrases from the title. For example, from the paper title "Real-time collision-free path planning and tracking control of a nonholonomic mobile robot using a biologically inspired approach.", the noun phrase "Path planning" is extracted.

4.5 Dataset Generation

For each data instance d_i, two subgraphs are generated (see Section 4.2) and are used to instantiate a template tuple t_i. Some template tuples require two different publication titles but since each subgraph only contains one publication entity, two subgraphs are generated to fulfill this requirement. 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. Listing 4.2 shows an instance from the generated dataset.

```
{
    "id": "Q0577",
    "query_type": "MULTI_FACT",
    "question": {
        "string": "What are the primary affiliations of the authors of the paper
        'Graphical Partitions and Graphical Relations'?"
    },
    "paraphrased_question": {
        "string": "List the primary affiliations of the authors of
        'Graphical Partitions and Graphical Relations'."
    },
    "query": {
        "sparql": "
        SELECT DISTINCT ?answer WHERE {
            <https://dblp.org/rec/journals/fuin/ShaheenS19> <https://dblp.org/rdf/schema#authoredBy> ?x .
            ?x <https://dblp.org/rdf/schema#primaryAffiliation> ?answer }"
    },
    "template_id": "TP11",
    "entities": ["<https://dblp.org/rec/journals/fuin/ShaheenS19>"],
    "relations": [
        "<https://dblp.org/rdf/schema#authoredBy>", "<https://dblp.org/rdf/schema#primaryAffiliation>"
    ],
    "temporal": false,
    "held_out": true
}
```

Listing 4.2: A sample of data from the generated dataset

To foster a focus on generalization ability, 20 template tuples are marked manually so as to not use them during generation of the *train* set. However, all the template tuples are used in the generation of *valid* and *test* sets. Furthermore, two natural language question templates are also withheld when generating *train* questions but all question templates are used when generating *valid* and *test* sets. This controlled generation process allows withholding some entity classes, predicates and paraphrases from *train* set. The main 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* [Gu et al., 2021] generalization. Algorithm 2 summarizes the entire dataset generation process.

Algorithm 2: 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 $D$
```

4.6 Query Validation

Further, each data instance d_i is validated by running the SPARQL query s_i against the DBLP KG via a Virtuso SPARQL endpoint[3]. The data instances in which the SPARQL query is invalid or generates a blank response are filtered out. 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. Listing 4.3 shows an instance from

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

the dataset that was filtered out as invalid as the subgraph had missing information for the relation *primaryAffiliation*.

```
{
    "id": "Q0104",
    "query_type": "SINGLE_FACT",
    "question": {
        "string": "R. C. de Lamare is primarily affiliated to which
                        institution?"
    },
    "paraphrased_question": {
        "string": "Mention the primary affiliation of Rodrigo C. de Lamare."
    },
    "query": {
        "sparql": "SELECT DISTINCT ?answer WHERE {
            <https://dblp.org/pid/98/6670>
            <https://dblp.org/rdf/schema#primaryAffiliation ?answer }"
    },
    "template_id": "TC02",
    "entities": [
        "<https://dblp.org/pid/98/6670>"
    ],
    "relations": [
        "<https://dblp.org/rdf/schema#primaryAffiliation>"
    ],
    "temporal": false,
    "held_out": false
}
```

Listing 4.3: A sample of data consiting of a failed SPARQL query

Additionally, the answers produced by running the SPARQL query against the DBLP KG are also stored. The answers are formatted according to `https://www.w3.org/TR/sparql11-results-json/` and an example is shown in Listing 4.4.

```
{
    "id": "Q0577",
    "answer": {
        "head": {
            "link": [],
            "vars": [
                "answer"
            ]
        },
        "results": {
            "distinct": false,
            "ordered": true,
            "bindings": [
                {
                    "answer": {
                        "type": "literal",
                        "value": "University of Leeds, School of Computing, UK"
                    }
                }
            ]
        }
    }
}
```

Listing 4.4: A sample of answer stored in DBLP-QuAD

4.7 Dataset statistics

In this section, various meta statistics of DBLP-QuAD is presented. 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 count of data instances each dataset group is shown in Table 4.2.

	Train	Valid	Test
# Question-Query pairs	7,000	1,000	2,000

Table 4.2: Distribution of data instances by dataset groups

The dataset covers 13,379 creators and publications, and 11 predicates of the DBLP KG. The distribution of queries by relations is shown in Table 4.3. For each query type in Table 4.1, the dataset includes 1,000 question-query pairs each which is equally divided as creator-focused or publication-focused. Additionally, DBLP-QuAD comprises of 2,350 temporal questions.

Relations	#Questions
<https://dblp.org/rdf/schema#authoredBy>	8159
<https://dblp.org/rdf/schema#publishedIn>	3313
<https://dblp.org/rdf/schema#yearOfPublication>	2934
<https://dblp.org/rdf/schema#primaryAffiliation>	616
<https://dblp.org/rdf/schema#title>	462
<https://dblp.org/rdf/schema#numberOfCreators>	355
<https://dblp.org/rdf/schema#webpage>	100
<https://dblp.org/rdf/schema#orcid>	58
<https://dblp.org/rdf/schema#wikidata>	47
<http://purl.org/dc/terms/bibtexType>	35
<https://dblp.org/rdf/schema#bibtexType>	18

Table 4.3: Count of questions using different DBLP relations

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 (shown in Figure 4.2 and 4.3). The average Levenshtein edit distance between them is 32.99 (with standard deviation of 23.12) (shown in Figure 4.4). Table 4.4 summarizes the statistics of the dataset.

Characteristic	Mean	S.D.	Min	Max
Question Word Count	17.32	6.53	6	58
Paraphrase Word Count	17.31	6.52	4	58
Query Vocab Count	12.66	4.61	6	29
Question Character Count	114.12	50.84	25	444
Paraphrase Character Count	113.97	50.61	30	445
Query Character Count	249.48	101.87	76	677
Edit distance	32.99	23.12	16	302
Jaccard similarity (unigram)	0.62	0.22	0	1
Jaccard similarity (bigram)	0.47	0.23	0	1

Table 4.4: General statistics of the DBLP-QuAD Dataset

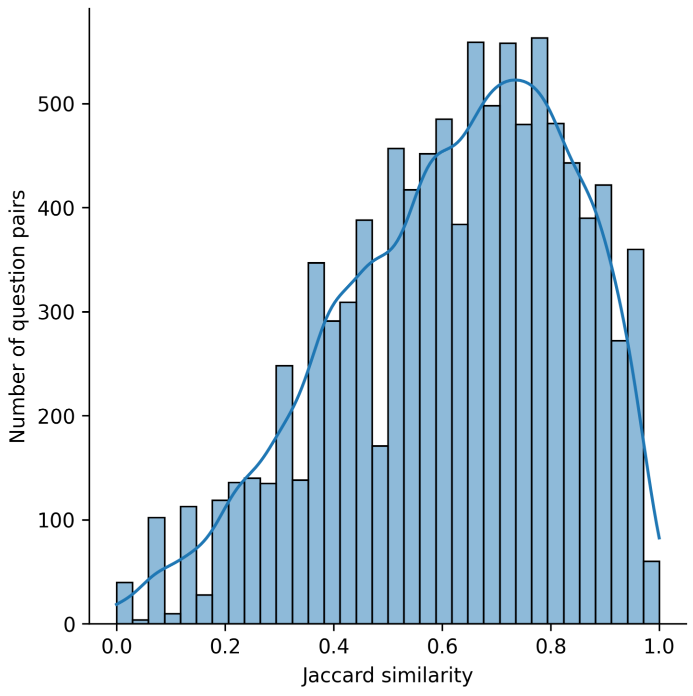


Figure 4.2: Distribution of unigram Jaccard Similarity between the question and its paraphrase

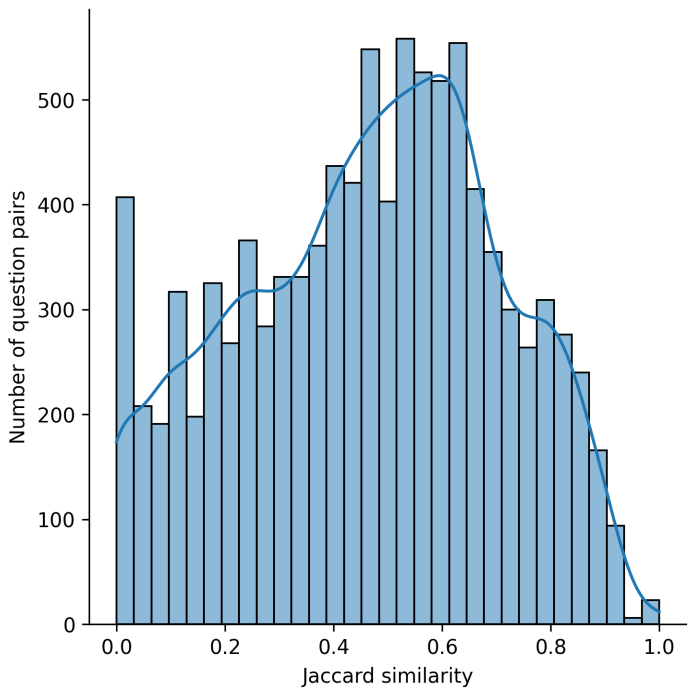


Figure 4.3: Distribution of bigram Jaccard Similarity between the question and its paraphrase

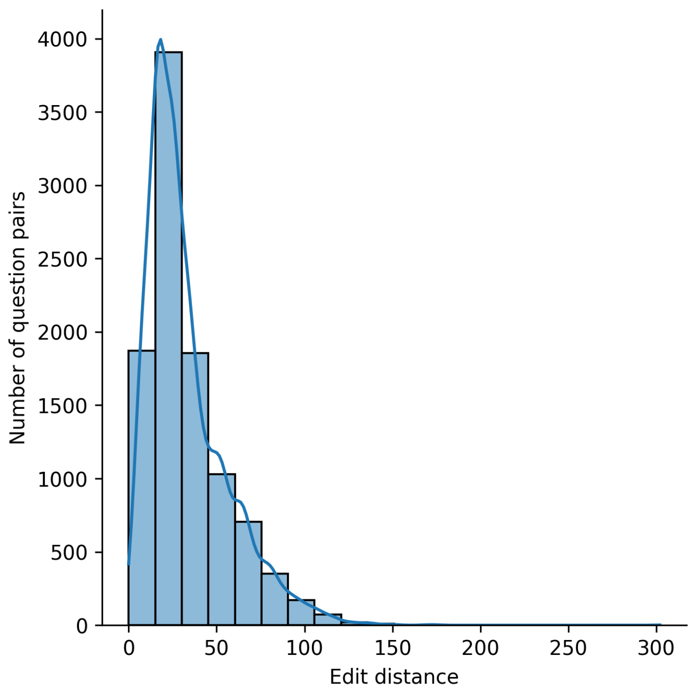


Figure 4.4: Distribution of Levenshtein Edit distance between the question and its paraphrase

Characteristic	DBLP-QuAD		CRONQuestions		LC-QuAD 2.0	
	Mean	S.D.	Mean	S.D.	Mean	S.D.
Edit distance	32.99	23.12	11.85	11.20	29.81	360
Jaccard similarity (unigram)	0.62	0.22	0.78	0.16	0.52	0.22
Jaccard similarity (bigram)	0.47	0.23	0.66	0.22	0.33	0.25

Table 4.5: Comparison of edit distance and Jaccard similarity between paraphrases of DBLP-QuAD, CRONQuestions and LC-QuAD 2.0

Table 4.5 shows the Edit distance and Jaccard similarity (unigram and bigram) between the paraphrases compared to other two KGQA datasets, LC-QuAD 2.0 [Dubey et al., 2019] and CRONQuestions [Jia et al., 2021].

LC-QuAD 2.0 has lower syntactic similarity between paraphrases than that of DBLP-QuAD. However, LC-QuAD 2.0 covers a wide range of topics than DBLP-QuAD and is larger in size with 20,000 more questions than DBLP-QuAD.

CRONQuestions also generates the data synthetically by replacing entity placeholders. For this analysis, two paraphrases were selected in random for comparison for each question in CRONQuestions. Despite CRONQuestions covering a wide range of topics and having a larger dataset size (410,000 questions), the syntactic similarity between paraphrases in DBLP-QuAD is quite low in comparison to that between paraphrases in CRONQuestions.

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. For example, augmented author and venue names, shortened institution names. (See Section 4.4)

Generalization

In the *valid* set 17.2% and in the *test* set 18.85% 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 4.6 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	81.15%	15.1%	3.75%

Table 4.6: Distribution of questions with different levels of generalization in the *valid* and *test* sets

5 Development of Semantic Parsing Baseline

Next, after the development of DBLP-QuAD, the performance of the current state-of-the-art semantic parsing model is evaluated on DBLP-QuAD. Banerjee et al. [2022] showed that the fine-tuning T5 performs well for the semantic parsing task where T5 even outperformed other PLMs such as BART. Following Banerjee et al. [2022], T5 [Raffel et al., 2020] is fine-tuned on DBLP-QuAD.

5.1 T5

T5 (Text-to-Text Transfer Transformer) [Raffel et al., 2020] is a large transformer [Vaswani et al., 2017] model trained on "Colossal Clean Crawled Corpus (C4)" dataset. T5 closely follows the original encoder-decoder transformer architecture [Vaswani et al., 2017] and is capable of performing several NLP tasks such as text summarization, question answering, machine translation, paraphrasing, sentiment analysis, etc. T5 model was trained with the objective of generating text conditioned on input text i.e. text-to-text format. During training, each input text is paired with a task-specific prompt. Some examples are shown in Figure 5.1. The pre-trained T5 model is publicly available[1] in five different sizes: *T5-Small* (60 million parameters), *T5-Base* (220 million parameters), *T5-Large* (770 million parameters), *T5-3B* (3 billion parameters), *T5-11B* (11 billion parameters).

[1]`https://huggingface.co/docs/transformers/model_doc/t5`

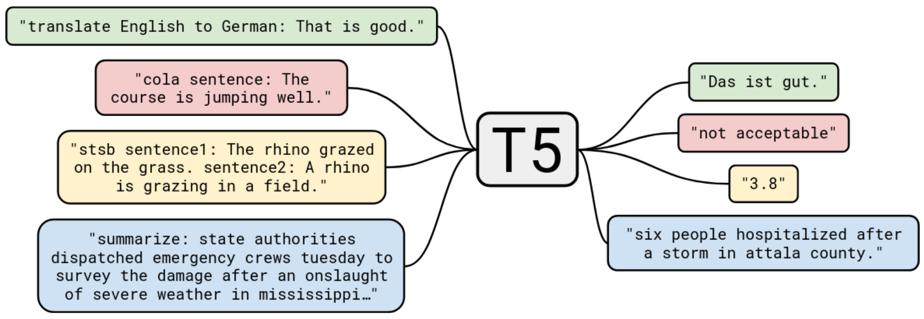

"translate English to German: That is good."
"cola sentence: The course is jumping well."
"stsb sentence1: The rhino grazed on the grass. sentence2: A rhino is grazing in a field."
"summarize: state authorities dispatched emergency crews tuesday to survey the damage after an onslaught of severe weather in mississippi…"
T5
"Das ist gut."
"not acceptable"
"3.8"
"six people hospitalized after a storm in attala county."

Figure 5.1: Diagram showing input-output format used in training T5
Figure from [Raffel et al., 2020]

5.2 Fine-tuning T5

The following section details the settings and configurations selected in developing the baseline model.

5.2.1 Data preparation

For each data instance d_i, the natural language question is denoted as $\mathbf{Q} = [w_1, w_2, ..., w_n]$, the entity IRIs as $\mathbf{E} = [e_1, e_2, ..., e_n]$, and the relation IRIs as $\mathbf{R} = [r_1, r_2, ..., r_n]$. Unlike in Banerjee et al. [2022], the entity or relation surface forms in the source text are omitted. Further, the source string is concatenated with the prefix “parse text to SPARQL query:” to form the final source text as below:

$$\text{parse text to SPARQL query: } w_1 w_2 ... w_n [SEP] e_1 e_2 ... e_n [SEP] r_1 r_2 ... r_n$$

where the words are separated by space and the different input groups by the special token *[SEP]*.

The target text is composed of SPARQL vocabulary along with tokens from **Q**, **E** and **R**. Here, the T5 model has to perform a form of "copying" to place the tokens from **Q**, **E** and **R** in the target text. Finally, the target text is wrapped with the tokens $< s >< /s >$.

Additionally, the sentinel tokens provided by T5 are used 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. Without this step, the models produce zero accuracy.

An example of the data instance used to fine-tune the T5 model is shown in Figure 5.2.

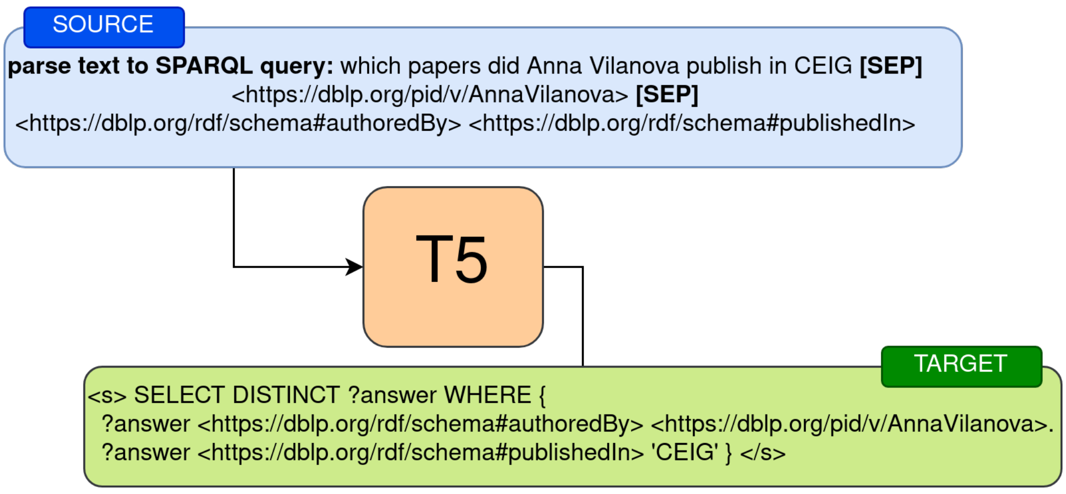

SOURCE
parse text to SPARQL query: which papers did Anna Vilanova publish in CEIG **[SEP]** <https://dblp.org/pid/v/AnnaVilanova> **[SEP]** <https://dblp.org/rdf/schema#authoredBy> <https://dblp.org/rdf/schema#publishedIn>
T5
TARGET
<s> SELECT DISTINCT ?answer WHERE {
?answer <https://dblp.org/rdf/schema#authoredBy> <https://dblp.org/pid/v/AnnaVilanova>.
?answer <https://dblp.org/rdf/schema#publishedIn> 'CEIG' } </s>

Figure 5.2: Representation of source and target text used to fine-tune the T5 model

5.2.2 Experiments

For fine-tuning T5 on DBLP-QuAD, the smallest two T5 models, *T5-Small* and *T5-Base* are selected. The models are fine-tuned on the *train* set with a learning rate of *1e-4* for 5 epochs with the source as well as target text length of 512, and batch size of 4 for both training and validation steps. During training, the models are optimized with Adam optimizer. For this purpose, the Python libraries PyTorch [Paszke et al., 2019] and HuggingFace[2] are used and the models are fine-tuned on an Nvidia RTX A6000 GPU.

5.3 Evaluation

5.3.1 Experiment Results

The performance of the baseline model on the DBLP-QuAD test set is reported. Firstly, the exact-match between the gold and the generated SPARQL query is computed. The exact-match accuracy is calculated between the generated and the gold query by comparing them token by token after removing white spaces. Next, for each SPARQL query on the test set, both the gold and the generated query are run against the DBLP KG. The KG is hosted as a service on Virtuoso SPARQL endpoint. The F1 score is reported by matching the answers collected by running the gold and generated query. The results are reported in Table 5.1.

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

Table 5.1: Evaluation results of fine-tuned T5 to DBLP-QuAD

Table 5.1 shows that *T5-Base*, the larger of the two, outperforms on both metrics. The reported results establish a baseline score on DBLP-QuAD. The baseline model is further evaluated and analyzed in the following section.

5.3.2 Error Analysis

The following error analysis details the performance of the T5-model on different types of query types. Table 5.2 shows the performance of the T5-variants on the different query types as well as the performance between the *T5-Small* and *T5-Base* for each of the query types. Table 5.3 compares the performance between temporal and non-temporal queries. Table 5.4 shows the generalization ability of the T5 model on DBLP-QuAD.

[2]huggingface.co/

Errors based on query type

Table 5.2 shows the larger model T5-Base outperforms the smaller model T5-Small on every query type. For the query type, NEGATION the difference in F1 score between the two variants is more than 0.30. While most query types have the difference around 0.20, some query types such as SINGLE_FACT and DOUBLE_INTENT have the lowest difference.

Query	F1 Scores	
Type	T5-Small	T5-Base
SINGLE_FACT	0.800	0.840
MULTI_FACT	0.797	0.915
DOUBLE_INTENT	0.730	0.780
BOOLEAN	0.730	0.835
NEGATION	0.530	0.935
DOUBLE_NEGATION	0.720	0.895
UNION	0.579	0.812
DISAMBIGUATION	0.820	0.905
COUNT	0.785	0.850
SUPERLATIVE+COMPARATIVE	0.715	0.915

Table 5.2: F1 scores of T5-Small and T5-Base by query types

T5-Small struggles mostly with NEGATION queries where by the model mostly generates non-negative queries. Between NEGATION and BOOLEAN queries, the natural questions are mainly differentiated by the "not" or equivalent string. The NEGATION queries have an additional **FILTER NOT EXISTS** part which is not present in BOOLEAN queries. In the following example, T5-Small generates the SPARQL query where by it misses the **FILTER NOT EXISTS** part present in the gold SPARQL query. T5-Base makes less of these mistakes.

```
Question: Has Jun Liu not published in IEEE Trans. Aerosp. Electron. Syst.?
Gold: ASK {
    ?x <https://dblp.org/rdf/schema#authoredBy> <https://dblp.org/pid/95/3736−4> .
    ?x <https://dblp.org/rdf/schema#publishedIn> ’IEEE Trans. Aerosp. Electron. Syst.’
    FILTER NOT EXISTS {
        ?x <https://dblp.org/rdf/schema#authoredBy> <https://dblp.org/pid/95/3736−4> .
        ?x <https://dblp.org/rdf/schema#publishedIn> ’IEEE Trans. Aerosp. Electron. Syst.’ } }
Generated: ASK {
    ?x <https://dblp.org/rdf/schema#authoredBy> <https://dblp.org/pid/95/3736−4> .
    ?x <https://dblp.org/rdf/schema#publishedIn> ’IEEE Trans. Aerosp. Electron. Syst.’ }
```

Among the different query types, both variants of T5 struggle with DOUBLE_INTENT queries. Figures 5.3 and 5.4 show the shortcoming is largely due to the held out compositional queries where both variants perform very poorly. The following example of a held out compositional query in the DOUBLE_INTENT category generated by T5-Base shows that it misses a nuanced detail in the question resulting in generating wrong query. The generated query is a valid one but the question required that the author in question “Rolf Ernst” needed to be filtered out from the result set which is missed by the model.

```
Question: With which other authors has Rolf Ernst co−authored papers and where are they affiliated?
Gold: SELECT DISTINCT ?firstanswer ?secondanswer WHERE {
    ?x <https://dblp.org/rdf/schema#authoredBy> <https://dblp.org/pid/e/RolfErnst> .
    ?x <https://dblp.org/rdf/schema#authoredBy> ?firstanswer
    FILTER (?firstanswer != <https://dblp.org/pid/e/RolfErnst>) .
    ?firstanswer <https://dblp.org/rdf/schema#primaryAffiliation> ?secondanswer }
Generated: SELECT DISTINCT ?firstanswer ?secondanswer WHERE {
    ?x <https://dblp.org/rdf/schema#authoredBy> <https://dblp.org/pid/e/RolfErnst> .
    ?x <https://dblp.org/rdf/schema#primaryAffiliation> ?secondanswer }
```

The T5 models also under performs on UNION queries. UNION queries use two separate entities but the following example shows T5-Base used the same entity twice.

```
Question: Mention the number of authors of ’Remote Sensing of Soil Moisture Using the Propagation of
    Loran−C Navigation Signal’ and ’The benefits of word embeddings features for active learning in clinical
    information extraction’.
Gold: SELECT ?answer WHERE { {
    <https://dblp.org/rec/journals/lgrs/FengA15> <https://dblp.org/rdf/schema#numberOfCreators> ?answer
    } UNION {
    <https://dblp.org/rec/journals/lgrs/FengA15> <https://dblp.org/rdf/schema#numberOfCreators> ?answer
    } }
Generated: SELECT DISTINCT ?answer WHERE { {
    <https://dblp.org/rec/journals/lgrs/FengA15> <https://dblp.org/rdf/schema#numberOfCreators> ?answer
    } UNION {
    <https://dblp.org/rec/journals/corr/KholghiVSZN16> <https://dblp.org/rdf/schema#numberOfCreators>
    ?answer } }
```

Furthermore, T5 also confuses between MUTLI_FACT and UNION queries where two entities are used. In the following example from a MULTI_FACT query, an intersection operation is required instead T5-Base performs a union.

```
Question: Which papers did the authors Oezguer Simsek and Marcus Buckmann co−write?
Gold: SELECT DISTINCT ?answer WHERE {
    ?answer <https://dblp.org/rdf/schema#authoredBy> <https://dblp.org/pid/64/5184> .
    ?answer <https://dblp.org/rdf/schema#authoredBy> <https://dblp.org/pid/176/0265> }
Generated: SELECT DISTINCT ?answer WHERE {
    ?answer <https://dblp.org/rdf/schema#authoredBy> <https://dblp.org/pid/64/5184>
    } UNION {
    ?answer <https://dblp.org/rdf/schema#authoredBy> <https://dblp.org/pid/176/0265> } }
```

T5 models makes a similar mix up between COUNT and SUPERLATIVE+COMPARATIVE queries. In the following example of SUPERLATIVE+COMPARATIVE query, the question asks for the author (expected answer is a tuple (HTTPS://DBLP.ORG/PID/E/PWEKLUND, 1988) but instead the generated query only produces the year the first paper of the author was published (i.e. 1988). The second information is important to answer this question but further inference is required.

```
Question: Name the author who published their first paper among the authors of
    'Concept Lattices for Information Visualization: Can Novices Read Line−Diagrams?'.
Gold: SELECT (GROUP_CONCAT (?answer; separator=', ') AS ?answer) ?z WHERE {
    SELECT DISTINCT ?answer ?z WHERE {
        <https://dblp.org/rec/conf/icfca/EklundDB04> <https://dblp.org/rdf/schema#authoredBy> ?answer .
        ?y <https://dblp.org/rdf/schema#authoredBy> ?answer .
        ?y <https://dblp.org/rdf/schema#yearOfPublication> ?z }
        GROUP BY ?z } ORDER BY ASC (?z) LIMIT 1
Generated: SELECT DISTINCT ?answer MIN (xsd:integer (?y)) AS ?y WHERE {
    <https://dblp.org/rec/conf/icfca/EklundDB04> <https://dblp.org/rdf/schema#authoredBy> ?answer .
    ?x <https://dblp.org/rdf/schema#authoredBy> ?answer .
    ?x <https://dblp.org/rdf/schema#yearOfPublication> ?y }
```

Error based on temporal queries

Query	F1 Scores	
Type	T5-Small	T5-Base
Temporal	0.710	0.837
Non-temporal	0.724	0.878

Table 5.3: F1 scores of T5-Small and T5-Base on temporal and non-temporal queries

Table 5.3 shows that the T5 models fair slightly worse with temporal queries compared to non-temporal queries. Since the temporal queries are spread out across the query types and the levels of generalization, no distinguishing pattern could be found between temporal and non-temporal query errors.

On the other hand, T5 performs well on translating text representation of duration to its numeric representation required in the SPARQL query. The following example shows evidence for this.

```
Question: List the papers published by Gao, X. in the last eight years.
Gold: SELECT DISTINCT ?answer WHERE {
    ?answer <https://dblp.org/rdf/schema#authoredBy> <https://dblp.org/pid/13/3109> .
    ?answer <https://dblp.org/rdf/schema#yearOfPublication> ?y FILTER (?y > YEAR (NOW ())−8) }
Generated: SELECT DISTINCT ?answer WHERE {
    ?answer <https://dblp.org/rdf/schema#authoredBy> <https://dblp.org/pid/13/3109> .
    ?answer <https://dblp.org/rdf/schema#yearOfPublication> ?y FILTER (?y > YEAR (NOW ())−8) }
```

Errors based on levels of generalization

The T5 models perform quite poorly on compositional queries and even worse on zero-shot queries. Table 5.4 shows the F1 scores of the two T5 models on three levels of generalization.

Generalization	F1 Scores	
Levels	T5-Small	T5-Base
I.I.D.	0.808	0.955
Compositional	0.425	0.561
Zero-shot	0.078	0.286

Table 5.4: F1 scores of T5-Small and T5-Base on three levels of generalization

Among the held out relations for zero-shot, one relation is <HTTP://PURL.ORG/DC/TERMS/BIBTEXTYPE>. The T5-Base model, in the following example, generates a query using the relation <HTTPS://DBLP.ORG/RDF/SCHEMA#PRIMARYAFFILIATION> instead. Additionally, it also misses the entity IRI.

```
Question: Is the paper 'Deep Compression: Compressing Deep Neural Network with Pruning,
    Trained Quantization and Huffman Coding' categorised as bibtex type Inproceedings?
Gold: ASK {
    <https://dblp.org/rec/journals/corr/HanMD15> <http://purl.org/dc/terms/bibtexType>
    <http://purl.org/net/nknouf/ns/bibtex#Inproceedings> }
Generated: ASK {
    <https://dblp.org/rec/journals/corr/HanMD15> <https://dblp.org/rdf/schema#primaryAffiliation>
    'bibtexTypeInproceedings' }
```

On other instances, although the model gets the relation correct, it generates an incomplete query as shown below.

```
Question: What is the ORCID of Johann, Patricia?
Gold: SELECT DISTINCT ?answer WHERE {
    <https://dblp.org/pid/29/826> <https://dblp.org/rdf/schema#orcid> ?answer }
Generated: SELECT DISTINCT ?answer WHERE {
    <https://dblp.org/pid/29/826> <https://dblp.org/rdf/schema#orcid> }
```

The following Figures 5.3 and 5.4 further show how the zero-shot and compositional queries affect the F1 scores on different query types. The models perform generalize well on I.I.D. level however struggle on other two levels. The blank white cells in the figures represent that there were no queries for the corresponding query types for the given level of generalization.

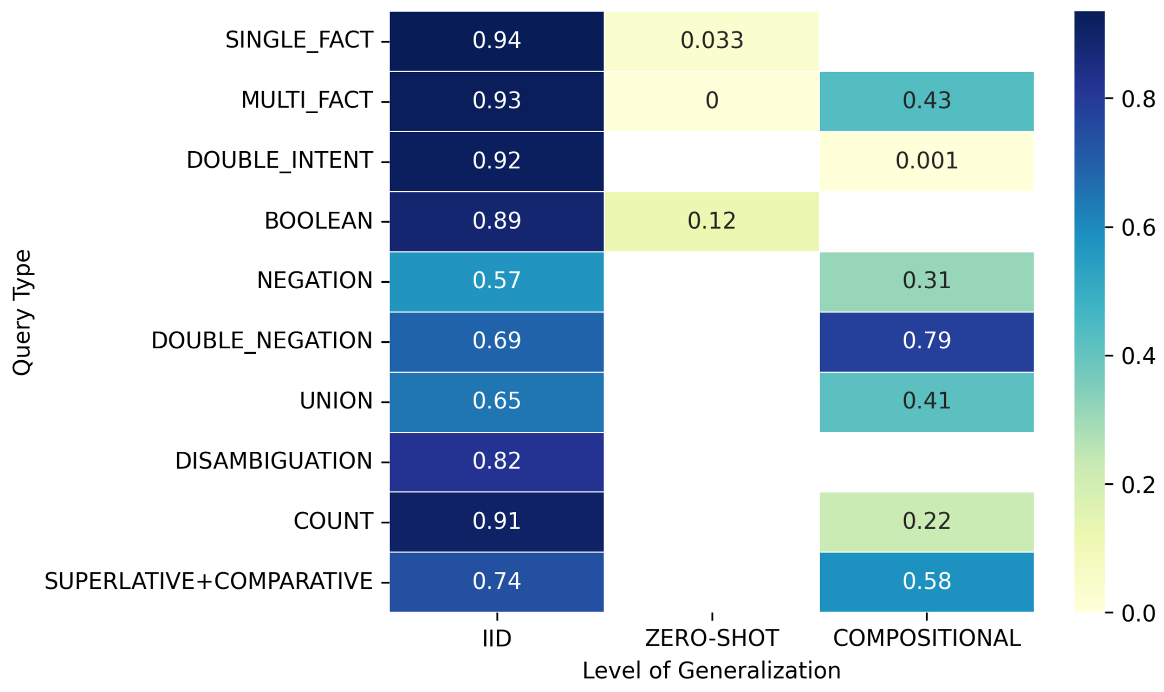


Figure 5.3: F1 score heatmap by query type and levels of generalization for T5-Small

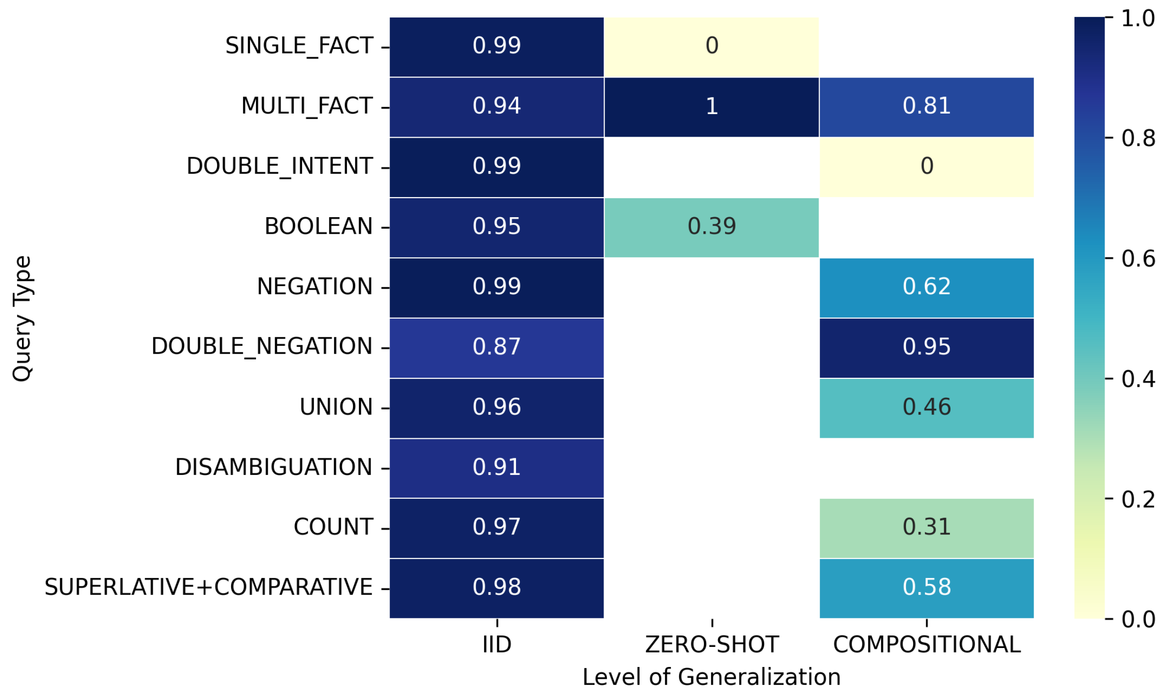


Figure 5.4: F1 score heatmap by query type and levels of generalization for T5-Base

Errors on venues and affiliations

The T5 models also under performed on queries that required to use an alternate textual representation for venues and affiliations. In the following example, T5-Base directly copies the full venue name "IEEE Global Telecommunications Conference" in the query when it was required to translate it to its short form "GLOBECOM"

```
Question: What publications did Brahim B. publish in IEEE Global Telecommunications Conference?
Gold: SELECT DISTINCT ?answer WHERE {
    ?answer <https://dblp.org/rdf/schema#authoredBy> <https://dblp.org/pid/b/BrahimBensaou> .
    ?answer <https://dblp.org/rdf/schema#publishedIn> 'GLOBECOM' }
Generated: SELECT DISTINCT ?answer WHERE {
    ?answer <https://dblp.org/rdf/schema#authoredBy> <https://dblp.org/pid/b/BrahimBensaou> .
    ?answer <https://dblp.org/rdf/schema#publishedIn> 'IEEE Global Telecommunications Conference' }
```

In the following example, T5-Base rightly adds in the country name for a affiliation as required but factually it's incorrect as "Concordia University" is in Canada and not Japan.

```
Question: Is Concordia University the primary affiliation of the authors of
    'AOP Extension for Security Testing of Programs'?
Gold: ASK {
    <https://dblp.org/rec/conf/ccece/BelblidiaDHY06> <https://dblp.org/rdf/schema#authoredBy> ?x .
    ?x <https://dblp.org/rdf/schema#primaryAffiliation> 'Concordia University, Montreal, Canada' }
Generated: ASK {
    <https://dblp.org/rec/conf/ccece/BelblidiaDHY06> <https://dblp.org/rdf/schema#authoredBy> ?x .
    ?y <https://dblp.org/rdf/schema#primaryAffiliation> 'Concordia University, Japan' }
```

Other errors

The T5 models also made some other nuanced errors such as generating incomplete or wrong entity IRIs by a few characters, generating wrong sort order (e.g. DESC for ASC and vice-versa), generating wrong symbols (e.g. { for <), applying COUNT operation on wrong variable, etc.

6 Discussion

6.1 Dataset

Revisiting the research questions discussed in Section 1.3, the first research question is as follows:

- **RQ1:** Can a suitable KGQA dataset be built with an automated dataset generation framework?

To investigate this question, a dataset generation framework was designed and a scholarly KGQA dataset, named DBLP-QuAD, was developed for the DBLP KG [Ley, 2002]. The developed KGQA dataset was then used to build a semantic parsing model by fine-tuning T5 [Raffel et al., 2020]. The evaluation carried out on the baseline model shows that building a scholarly KGQA system using DBLP-QuAD is feasible. The performance of the baseline model on DBLP-QuAD was discussed in Section 5.3.2. There are, however, limitations and shortcomings of DBLP-QuAD, which are discussed below.

User Information Need

One of the drawbacks of the dataset generation framework is that natural questions are machine-generated. (CFQ [Keysers et al., 2020] and CRONQuestions [Saxena et al., 2021] have a similar limitation.) The question templates were human-written and were not crowd sourced from a group of researchers. Additionally, the entities and literals for the questions were generated by drawing from the 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. Utilizing this advantage, DBLP-QuAD programmatically augments text representations in the questions, and generates 10,000 question-query pairs.

Test Leakage

In generating *valid* and *test* sets, the additional 19 template tuples which account for about 20% of the template set were also used. However, 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. Hence, to limit the leakage on 80% of the data, two question templates were withheld in generating the *train* set. Moreover, the text augmentation steps were carried out for entity and literal surface forms which would make entity linking challenging for the *valid* and *test* sets.

Publication Titles

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.

No Answers

When generating the dataset, data instances with SPARQL queries that did not produce any answers were filtered out. Although these queries produced blank responses from the KG, they may represent user information need. Hence, these data instances should be preserved in the dataset. One approach to implement this idea, similar to SQuAD 2.0 [Rajpurkar et al., 2018], is to have a blank response as a valid response.

6.2 Baseline Model

The second research question, stated below, was investigated by developing a semantic parsing baseline model using the current SOTA approach. The pre-trained T5 [Raffel et al., 2020] transformer [Vaswani et al., 2017] model was fine-tuned on DBLP-QuAD. The fine-tuning was carried out following Banerjee et al. [2022], which achieved SOTA results on the LC-QuAD 2.0 dataset [Dubey et al., 2019].

- **RQ2:** How do the current SOTA KGQA model fare against the developed dataset?

The pre-trained T5 model performs well on most of the questions of DBLP-QuAD (Section 5.3). T5-Base, the larger of the two models used in the evaluation, fares better on DBLP-QuAD. However, the error analysis carried out in Section 5.3.2, shows that T5 struggles with certain query types. Often these errors were attributed to nuanced errors such as missing minute details in the questions, which resulted in generating an incorrect SPARQL query although a valid one. The T5 models also struggled with text augmentations carried out on entity and literal surface forms. In particular, T5 failed to generate valid text representations of venues and affiliations. In addition, the results reported in this work assumed that entity linking and relation linking models were 100% accurate. Without this assumption, the performance would degrade further.

The evaluation of the baseline model on DBLP-QuAD shows that large pre-trained language models perform quite well for most of the question types. However, the dataset also showed some shortcomings of T5, especially for zero-shot and compositional generalization questions (shown in Table 5.4). The shortcomings largely impacted the results of the model on different query types (shown in Figures 5.3 and 5.4). Recently, there has been an increased focus on generalization capability of KGQA systems [Gu et al., 2021; Keysers et al., 2020; Cui et al., 2022; Jiang and Usbeck, 2022], and in the same spirit, DBLP-QuAD also highlights the generalization shortcomings of PLMs in KGQA.

7 Conclusion

KGQA has largely focused on question answering in the domain of general information and world facts. The major reason for this are the two important resources required for KGQA are easily available for this domain i.e. large public Knowledge Graphs such as Wikidata [Vrandecic and Krötzsch, 2014] and DBpedia [Auer et al., 2007], and their corresponding KGQA datasets [Dubey et al., 2019; Usbeck et al., 2017; Sen et al., 2022]. However, with the recent release of RDF data graph for DBLP scholarly KG, an accompanying KGQA dataset would foster the research on scholarly KGQA. With this motivation, this thesis focused on developing a scholarly KGQA dataset for the DBLP KG called DBLP-QuAD. Moreover, with the recent focus on three levels of generalization in KGQA [Gu et al., 2021], DBLP-QuAD also encompasses this challenge.

7.1 Contributions

The main contribution of this thesis work is the scholarly KGQA dataset for the DBLP KG. For this, a dataset development framework was designed to automatically generate arbitrary number of natural language question and SPARQL query pairs. The framework uses manually written template sets that include templates for both natural language question and SPARQL query. With sub-graphs drawn from the DBLP KG, the framework generates the data instances by instantiating the question and SPARQL queries. Additionally, the entity and literal surface forms were augmented to capture the syntactic variations used by humans.

Next, in this work, a semantic parsing baseline model was also developed using the current state-of-the-art semantic parsing model T5 [Banerjee et al., 2022]. The baseline scores on DBLP-QuAD was established on a thorough analysis of the model's performance was carried out highlighting the limitations of the T5 model and the challenges posed by DBLP-QuAD.

7.2 Future Work

DBLP-QuAD is the first scholarly KGQA dataset focusing on bibliographic information in the scholarly domain, and is the largest scholarly KGQA dataset. The other scholarly KGQA dataset, ORKG-QA benchmark [Jaradeh et al., 2020] contains only 100 questions and focuses on the content of research publications. DBLP-QuAD, given its size, is suitable to train large neural networks as was demonstrated in this thesis work and is a good starting point in developing scholarly KGQA systems.

In future works, new KGQA systems being developed can use DBLP-QuAD to make their systems capable of handling scholarly metadata questions.

DBLP-QuAD, as discussed in this thesis, still has space for improvements, especially in representing user information need and using user-friendly publication titles in the questions. This is an interesting direction that can be tackled on in the future iterations of DBLP-QuAD. Bringing in human input at certain stages of the dataset generation process should definitely improve the quality of questions in DBLP-QuAD.

8 Bibliography

S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. G. Ives. DBpedia: A Nucleus for a Web of Open Data. In *The Semantic Web, 6th International Semantic Web Conference, 2nd Asian Semantic Web Conference, ISWC 2007 + ASWC 2007*, volume 4825 of *Lecture Notes in Computer Science*, pages 722–735, Busan, Korea, 2007. Springer. doi: 10.1007/978-3-540-76298-0_52.

M. Azmy, P. Shi, J. Lin, and I. Ilyas. Farewell Freebase: Migrating the SimpleQuestions dataset to DBpedia. In *Proceedings of the 27th International Conference on Computational Linguistics, COLING*, pages 2093–2103, Santa Fe, New Mexico, USA, Aug. 2018. Association for Computational Linguistics.

D. Banerjee, P. A. Nair, J. N. Kaur, R. Usbeck, and C. Biemann. Modern baselines for SPARQL semantic parsing. In *SIGIR '22: The 45th International Association for Computing Machinery SIGIR Conference on Research and Development in Information Retrieval*, pages 2260–2265, Madrid, Spain, 2022. Association for Computing Machinery. doi: 10.1145/3477495.3531841.

J. Bao, N. Duan, M. Zhou, and T. Zhao. Knowledge-Based Question Answering as Machine Translation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, Association for Computational Linguistics*, pages 967–976, Baltimore, MD, USA, 2014. Association for Computational Linguistics. doi: 10.3115/v1/p14-1091.

J. Berant, A. Chou, R. Frostig, and P. Liang. Semantic Parsing on Freebase from Question-Answer Pairs. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, EMNLP*, pages 1533–1544, Seattle, Washington, USA, 2013. Association for Computational Linguistics.

K. D. Bollacker, C. Evans, P. K. Paritosh, T. Sturge, and J. Taylor. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the Association for Computing Machinery SIGMOD International Conference on Management of Data, SIGMOD*, pages 1247–1250, Vancouver, BC, Canada, 2008. Association for Computing Machinery. doi: 10.1145/1376616.1376746.

A. Bordes, S. Chopra, and J. Weston. Question Answering with Subgraph Embeddings. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP*, pages 615–620, Doha, Qatar, 2014. Association for Computational Linguistics. doi: 10.3115/v1/d14-1067.

A. Bordes, N. Usunier, S. Chopra, and J. Weston. Large-scale Simple Question Answering with Memory Networks. *CoRR*, abs/1506.02075, 2015.

T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei. Language Models are Few-Shot Learners. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020*, Online, 2020.

Q. Cai and 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, Association for Computational Linguistics*, pages 423–433, Sofia, Bulgaria, 2013. Association for Computer Linguistics.

S. Cao, J. Shi, L. Pan, L. Nie, Y. Xiang, L. Hou, J. Li, B. He, and 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*, pages 6101–6119, Dublin, Ireland, 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.AssociationforComputationalLinguistics-long.422.

Y. Cao, J. J. Cimino, J. W. Ely, and H. Yu. Automatically extracting information needs from complex clinical questions. *J. Biomed. Informatics*, 43(6):962–971, 2010. doi: 10.1016/j.jbi.2010.07.007.

N. Chakraborty, D. Lukovnikov, G. Maheshwari, P. Trivedi, J. Lehmann, and A. Fischer. Introduction to neural network-based question answering over knowledge graphs. *WIREs Data Mining Knowl. Discov.*, 11(3), 2021. doi: 10.1002/widm.1389.

A. Church. A set of postulates for the foundation of logic. *Annals of mathematics*, pages 346–366, 1932.

R. Cui, R. Aralikatte, H. C. Lent, and D. Hershcovich. Compositional Generalization in Multilingual Semantic Parsing over Wikidata. *Trans. Assoc. Comput. Linguistics*, 10:937–955, 2022.

S. C. Deerwester, S. T. Dumais, T. K. Landauer, G. W. Furnas, and R. A. Harshman. Indexing by Latent Semantic Analysis. *J. Am. Soc. Inf. Sci.*, 41(6):391–407, 1990. doi: 10.1002/(SICI)1097-4571(199009)41:6\<391::AID-ASI1\>3.0.CO;2-9.

D. Diefenbach, T. P. Tanon, K. D. Singh, and P. Maret. Question Answering Benchmarks for Wikidata. In *Proceedings of the ISWC 2017 Posters & Demonstrations and Industry Tracks co-located with 16th International Semantic Web Conference (ISWC 2017)*, volume 1963 of *CEUR Workshop Proceedings*, Vienna, Austria, 2017. CEUR-WS.org.

M. Dubey, S. Dasgupta, A. Sharma, K. Höffner, and J. Lehmann. AskNow: A Framework for Natural Language Query Formalization in SPARQL. In *The Semantic Web. Latest Advances and New Domains*, pages 300–316, Heraklion, Crete, Greece, 2016. Springer International Publishing.

M. Dubey, D. Banerjee, D. Chaudhuri, and J. Lehmann. EARL: joint entity and relation linking for question answering over knowledge graphs. In *The Semantic Web - ISWC 2018 - 17th International Semantic Web Conference*, volume 11136 of *Lecture Notes in Computer Science*, pages 108–126, Monterey, CA, USA, 2018. Springer. doi: 10.1007/978-3-030-00671-6_7.

M. Dubey, D. Banerjee, A. Abdelkawi, and J. Lehmann. Lc-quad 2.0: A large dataset for complex question answering over wikidata and dbpedia. In *The Semantic Web - ISWC 2019 - 18th International Semantic Web Conference*, volume 11779 of *Lecture Notes in Computer Science*, pages 69–78, Auckland, New Zealand, 2019. Springer. doi: 10.1007/978-3-030-30796-7_5.

Y. Gu, S. Kase, M. Vanni, B. M. Sadler, P. Liang, X. Yan, and Y. Su. Beyond I.I.D.: Three Levels of Generalization for Question Answering on Knowledge Bases. In *WWW '21: The Web Conference 2021*, pages 3477–3488, Virtual Event / Ljubljana, Slovenia, 2021. Association for Computing Machinery. doi: 10.1145/3442381.3449992.

Y. Gu, V. Pahuja, G. Cheng, and Y. Su. Knowledge Base Question Answering: A Semantic Parsing Perspective. In *4th Conference on Automated Knowledge Base Construction*, 2022.

D. Guo, D. Tang, N. Duan, M. Zhou, and J. Yin. Dialog-to-Action: Conversational Question Answering Over a Large-Scale Knowledge Base. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018*, pages 2946–2955, Montréal, Canada, 2018.

X. He and D. Golub. Character-Level Question Answering with Attention. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP*, pages 1598–1607, Austin, Texas, USA, 2016. Association for Computational Linguistics. doi: 10.18653/v1/d16-1166.

M. Y. Jaradeh, A. Oelen, K. E. Farfar, M. Prinz, J. D'Souza, G. Kismihók, M. Stocker, and S. Auer. Open Research Knowledge Graph: Next Generation Infrastructure for Semantic Scholarly Knowledge. In *Proceedings of the 10th International Conference on Knowledge Capture, K-CAP*, pages 243–246, Marina Del Rey, CA, USA, 2019. Association for Computing Machinery. doi: 10.1145/3360901.3364435.

M. Y. Jaradeh, M. Stocker, and S. Auer. Question Answering on Scholarly Knowledge Graphs. In *Digital Libraries for Open Knowledge - 24th International Conference on Theory and Practice of Digital Libraries, TPDL*, volume 12246 of *Lecture Notes in Computer Science*, pages 19–32, Lyon, France, 2020. Springer. doi: 10.1007/978-3-030-54956-5_2.

H. Ji and R. Grishman. Knowledge Base Population: Successful Approaches and Challenges. In *The 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1148–1158, Portland, Oregon, USA, 2011. Association for Computer Linguistics.

Z. Jia, A. Abujabal, R. S. Roy, J. Strötgen, and G. Weikum. TempQuestions: A Benchmark for Temporal Question Answering. In *Companion of the The Web Conference 2018 on The Web Conference 2018, WWW*, pages 1057–1062, Lyon , France, 2018. Association for Computing Machinery. doi: 10.1145/3184558.3191536.

Z. Jia, S. Pramanik, R. S. Roy, and G. Weikum. Complex Temporal Question Answering on Knowledge Graphs. In *CIKM '21: The 30th ACM International Conference on Information and Knowledge Management*, pages 792–802, Virtual Event, Queensland, Australia, 2021. Association for Computing Machinery. doi: 10.1145/3459637.3482416.

L. Jiang and R. Usbeck. Knowledge Graph Question Answering Datasets and Their Generalizability: Are They Enough for Future Research? In *SIGIR '22: The 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 3209–3218, Madrid, Spain, 2022. Association for Computing Machinery. doi: 10.1145/3477495.3531751.

K. S. Jones. A statistical interpretation of term specificity and its application in retrieval. *J. Documentation*, 60(5):493–502, 2004. doi: 10.1108/00220410410560573.

D. Jurafsky and J. Martin. *Speech & language processing*. Third Edition Draft, 3rd edition, 2022. `https://web.stanford.edu/~jurafsky/slp3/23.pdf`(Visited 2023-01-03).

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, and O. Bousquet. Measuring Compositional Generalization: A Comprehensive Method on Realistic Data. In *8th International Conference on Learning Representations, ICLR*, Addis Ababa, Ethiopia, 2020. OpenReview.net.

T. Kwiatkowski, J. Palomaki, O. Redfield, M. Collins, A. P. Parikh, C. Alberti, D. Epstein, I. Polosukhin, J. Devlin, K. Lee, K. Toutanova, L. Jones, M. Kelcey, M. Chang, A. M. Dai, J. Uszkoreit, Q. Le, and S. Petrov. Natural Questions: a Benchmark for Question Answering Research. *Trans. Assoc. Comput. Linguistics*, 7:452–466, 2019. doi: 10.1162/tAssociationforComputationalLinguistics_a_00276.

T. Lacroix, G. Obozinski, and N. Usunier. Tensor Decompositions for Temporal Knowledge Base Completion. In *8th International Conference on Learning Representations, ICLR*, Addis Ababa, Ethiopia, 2020. OpenReview.net.

M. Ley. The DBLP computer science bibliography: Evolution, research issues, perspectives. In *String Processing and Information Retrieval, 9th International Symposium, SPIRE*, volume 2476 of

Lecture Notes in Computer Science, pages 1–10, Lisbon, Portugal, 2002. Springer. doi: 10.1007/3-540-45735-6_1.

S. Mohammed, P. Shi, and J. Lin. Strong Baselines for Simple Question Answering over Knowledge Graphs with and without Neural Networks. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAAssociation for Computational Linguistics-HLT*, pages 291–296, New Orleans, Louisiana, USA, 2018. Association for Computational Linguistics. doi: 10.18653/v1/n18-2047.

A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köpf, E. Z. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019*, pages 8024–8035, Vancouver, BC, Canada, 2019.

A. Perevalov, D. Diefenbach, R. Usbeck, and A. Both. QALD-9-plus: A Multilingual Dataset for Question Answering over DBpedia and Wikidata Translated by Native Speakers. In *16th IEEE International Conference on Semantic Computing, ICSC*, pages 229–234, Laguna Hills, CA, USA, 2022a. IEEE. doi: 10.1109/ICSC52841.2022.00045.

A. Perevalov, X. Yan, L. Kovriguina, L. Jiang, A. Both, and 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, LREC*, pages 2998–3007, Marseille, France, 2022b. European Language Resources Association.

M. Petrochuk and L. Zettlemoyer. SimpleQuestions Nearly Solved: A New Upperbound and Baseline Approach. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 554–558, Brussels, Belgium, 2018. Association for Computational Linguistics. doi: 10.18653/v1/d18-1051.

J. Priem, H. A. Piwowar, and R. Orr. OpenAlex: A fully-open index of scholarly works, authors, venues, institutions, and concepts. In *26th International Conference on Science, Technology and Innovation Indicators (STI 2022)*, volume abs/2205.01833, Granada, Spain, Sept. 2022. Zenodo. doi: 10.5281/zenodo.6936227.

C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *J. Mach. Learn. Res.*, 21:140:1–140:67, 2020.

P. Rajpurkar, R. Jia, and P. Liang. Know What You Don't Know: Unanswerable Questions for SQuAD. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL*, pages 784–789, Melbourne, Australia, 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-2124.

A. Roberts, C. Raffel, and N. Shazeer. How Much Knowledge Can You Pack Into the Parameters of a Language Model? In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP*, pages 5418–5426, Online, 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.437.

S. E. Robertson and K. S. Jones. Relevance weighting of search terms. *J. Am. Soc. Inf. Sci.*, 27(3): 129–146, 1976. doi: 10.1002/asi.4630270302.

A. Saxena, S. Chakrabarti, and P. P. Talukdar. Question Answering Over Temporal Knowledge Graphs. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP*, pages 6663–6676, Online, 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.AssociationforComputationalLinguistics-long.520.

P. Sen, A. F. Aji, and A. Saffari. Mintaka: A Complex, Natural, and Multilingual Dataset for End-to-End Question Answering. In *Proceedings of the 29th International Conference on Computational Linguistics, COLING*, pages 1604–1619, Gyeongju, Republic of Korea, 2022. International Committee on Computational Linguistics.

Y. Su, H. Sun, B. M. Sadler, M. Srivatsa, I. Gur, Z. Yan, and X. Yan. On Generating Characteristic-rich Question Sets for QA Evaluation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP*, pages 562–572, Austin, Texas, 2016. Association for Computational Linguistics. doi: 10.18653/v1/d16-1054.

A. Talmor and J. Berant. The web as a knowledge-base for answering complex questions. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAAssociation for Computational Linguistics-HLT*, pages 641–651, New Orleans, Louisiana, USA, 2018. Association for Computational Linguistics. doi: 10.18653/v1/n18-1059.

T. P. Tanon, D. Vrandecic, S. Schaffert, T. Steiner, and L. Pintscher. From Freebase to Wikidata: The Great Migration. In *Proceedings of the 25th International Conference on World Wide Web, WWW*, pages 1419–1428, Montreal, Canada, 2016. Association for Computing Machinery. doi: 10.1145/2872427.2874809.

P. Trivedi, G. Maheshwari, M. Dubey, and J. Lehmann. LC-QuAD: A Corpus for Complex Question Answering over Knowledge Graphs. In *The Semantic Web - ISWC 2017 - 16th International Semantic Web Conference*, volume 10588 of *Lecture Notes in Computer Science*, pages 210–218, Vienna, Austria, 2017. Springer. doi: 10.1007/978-3-319-68204-4_22.

R. Usbeck, A. N. Ngomo, B. Haarmann, A. Krithara, M. Röder, and G. Napolitano. 7th Open Challenge on Question Answering over Linked Data (QALD-7). In *Semantic Web Challenges - 4th SemWebEval Challenge at ESWC*, volume 769 of *Communications in Computer and Information Science*, pages 59–69, Portoroz, Slovenia, 2017. Springer. doi: 10.1007/978-3-319-69146-6_6.

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is All you Need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems*, pages 5998–6008, Long Beach, CA, USA, 2017.

D. Vrandecic and M. Krötzsch. Wikidata: a free collaborative knowledgebase. *Commun. Association for Computing Machinery*, 57(10):78–85, 2014. doi: 10.1145/2629489.

W. Wang, N. Yang, F. Wei, B. Chang, and M. Zhou. Gated Self-Matching Networks for Reading Comprehension and Question Answering. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL*, pages 189–198, Vancouver, Canada, 2017. Association for Computational Linguistics. doi: 10.18653/v1/P17-1018.

Y. Wang, J. Berant, and 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 of the Asian Federation of Natural Language Processing, ACL*, pages 1332–1342, Beijing, China, 2015. Association for Computer Linguistics. doi: 10.3115/v1/p15-1129.

Q. Wu, D. Teney, P. Wang, C. Shen, A. R. Dick, and A. van den Hengel. Visual question answering: A survey of methods and datasets. *Comput. Vis. Image Underst.*, 163:21–40, 2017. doi: 10.1016/j.cviu.2017.05.001.

W. Yih, M. Richardson, C. Meek, M. Chang, and 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, Association for Computational Linguistics*, Berlin, Germany, 2016. Association for Computer Linguistics. doi: 10.18653/v1/p16-2033.

P. Yin, N. Duan, B. Kao, J. Bao, and M. Zhou. Answering Questions with Complex Semantic Constraints on Open Knowledge Bases. In *Proceedings of the 24th ACM International Conference on Information and Knowledge Management, CIKM*, pages 1301–1310, Melbourne, VIC, Australia, 2015. Association for Computing Machinery. doi: 10.1145/2806416.2806542.

P. Yin, C. Zhou, J. He, and G. Neubig. StructVAE: Tree-structured Latent Variable Models for Semi-supervised Semantic Parsing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, pages 754–765, Melbourne, Australia, 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-1070.

A. W. Yu, D. Dohan, M. Luong, R. Zhao, K. Chen, M. Norouzi, and Q. V. Le. QANet: Combining Local Convolution with Global Self-Attention for Reading Comprehension. In *6th International Conference on Learning Representations, ICLR*, Vancouver, BC, Canada, 2018. OpenReview.net.

J. M. Zelle and R. J. Mooney. Learning to Parse Database Queries Using Inductive Logic Programming. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence and Eighth Innovative Applications of Artificial Intelligence Conference, AAAI 96, IAAI 96*, pages 1050–1055, Portland, Oregon, USA, 1996. AAAI Press / The MIT Press.

Y. Zhang, H. Dai, Z. Kozareva, A. J. Smola, and L. Song. Variational Reasoning for Question Answering With Knowledge Graph. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18)*, pages 6069–6076, New Orleans, Louisiana, USA, 2018. AAAI Press.

9 Appendix

SPARQL variables

The SPARQL variables used in the queries are listed below:

- `?x`: variable for an intermediate result
- `?y`: variable for an intermediate result
- `?z`: variable for an intermediate result
- `?t`: variable for an intermediate result
- `?answer`: variable for final result
- `?firstanswer`: variable for first result in double intent queries
- `?secondanswer`: variable second final result
- `?count`: variable for count of the final result

Placeholders

The placeholders used in the SPARQL queries are listed below:

- `?p1`: placeholder for IRI of a publication
- `?p2`: placeholder for IRI of another publication
- `?c1`: placeholder for IRI of a creator
- `?c2`: placeholder for IRI of another creator
- `?b`: placeholder for IRI of a bibtextype

The placeholders used in the paraphrases are listed below:

- `[TITLE]`: placeholder for title of a publication
- `[OTHER_TITLE]`: placeholder for title of another publication

- `[CREATOR_NAME]`: placeholder for full name of a creator
- `[OTHER_CREATOR_NAME]`: placeholder for full name of another creator
- `[PARTIAL_CREATOR_NAME]`: placeholder for partial name of a creator
- `[AFFILIATION]`: placeholder for affiliation of a creator
- `[VENUE]`: placeholder for venue of a publication
- `[OTHER_VENUE]`: placeholder venue of another publication
- `[YEAR]`: placeholder for year of a publication
- `[TYPE]`: placeholder for bibtextype of a publication
- `[DURATION]`: placeholder for duration in years
- `[KEYWORD]`: placeholder for a keyword generated from the title of a publication

Eidesstattliche Erklärung

Hiermit versichere ich an Eides statt, dass ich die vorliegende Arbeit im Masterstudiengang Intelligent Adaptive Systems selbstständig verfasst und keine anderen als die angegebenen Hilfsmittel – insbesondere keine im Quellenverzeichnis nicht benannten Internet-Quellen – benutzt habe. Alle Stellen, die wörtlich oder sinngemäß aus Veröffentlichungen entnommen wurden, sind als solche kenntlich gemacht. Ich versichere weiterhin, dass ich die Arbeit vorher nicht in einem anderen Prüfungsverfahren eingereicht habe und die eingereichte schriftliche Fassung der elektronischen Abgabe entspricht.

Hamburg, den

21.02.2023 [redacted]

Datum Unterschrift

Veröffentlichung

Ich stimme der Einstellung der Arbeit in die Bibliothek des Fachbereichs Informatik zu.

Hamburg, den

21.02.2023 [redacted]

Datum Unterschrift