WAGENINGEN
UNIVERSITY & RESEARCH

Question answering over knowledge graphs : A graph-driven approach

International Joint Conference on Web Intelligence and Intelligent Agent Technology, WI-IAT 2022

Aghaei, Sareh; Masoudi, Sepide; Chhetri, Tek Raj; Fensel, Anna

https://doi.org/10.1109/WI-IAT55865.2022.00050

This publication is made publicly available in the institutional repository of Wageningen University and Research, under the terms of article 25fa of the Dutch Copyright Act, also known as the Amendment Taverne.

Article 25fa states that the author of a short scientific work funded either wholly or partially by Dutch public funds is entitled to make that work publicly available for no consideration following a reasonable period of time after the work was first published, provided that clear reference is made to the source of the first publication of the work.

This publication is distributed using the principles as determined in the Association of Universities in the Netherlands (VSNU) 'Article 25fa implementation' project. According to these principles research outputs of researchers employed by Dutch Universities that comply with the legal requirements of Article 25fa of the Dutch Copyright Act are distributed online and free of cost or other barriers in institutional repositories. Research outputs are distributed six months after their first online publication in the original published version and with proper attribution to the source of the original publication.

You are permitted to download and use the publication for personal purposes. All rights remain with the author(s) and / or copyright owner(s) of this work. Any use of the publication or parts of it other than authorised under article 25fa of the Dutch Copyright act is prohibited. Wageningen University & Research and the author(s) of this publication shall not be held responsible or liable for any damages resulting from your (re)use of this publication.

For questions regarding the public availability of this publication please contact openaccess.library@wur.nl

2022 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT)

2022 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT) | 978-1-6654-9402-1/22/$31.00 ©2022 IEEE | DOI: 10.1109/WI-IAT55865.2022.00050

Question answering over knowledge graphs: a graph-driven approach

Sareh Aghaei
Department of Computer Science
University of Innsbruck
Innsbruck, Austria
sareh.aghaei@sti2.at

Sepide Masoudi
Faculty of Computer Engineering
Islamic Azad University
Najafabad, Iran
s.masoudi@sco.iaun.ac.ir

Tek Raj Chhetri
Department of Computer Science
University of Innsbruck
Innsbruck, Austria
tekraj.chhetri@sti2.at

Anna Fensel
Wageningen Data Competence Center Wageningen University and Research, Wageningen, The Netherlands
Consumption and Healthy Lifestyles Chair Group, Wageningen University & Research, Wageningen, The Netherlands
Semantic Technology Institute, University of Innsbruck, Innsbruck, Austria
anna.fensel@wur.nl

***Abstract*—With the growth of knowledge graphs (KGs), question answering systems make the KGs easily accessible for end-users. Question answering over KGs aims to provide crisp answers to natural language questions across facts stored in the KGs. This paper proposes a graph-driven approach to answer questions over a KG through four steps, including (1) knowledge subgraph construction, (2) question graph construction, (3) graph matching, and (4) query execution. Given an input question, a knowledge subgraph, which is likely to include the answer is extracted to reduce the KG's search space. A graph, named question graph, is built to represent the question's intention. Then, the question graph is matched over the knowledge subgraph to find a query graph corresponding to a SPARQL query. Finally, the corresponding SPARQL is executed to return the answers to the question. The performance of the proposed approach is empirically evaluated using the 6th Question Answering over Linked Data Challenge (QALD-6). Experimental results show that the proposed approach improves the performance compared to the-state-of-art in terms of recall, precision, and F1-score.**

***Index Terms*—Knowledge Graphs, Question Answering, Graph Matching, SPARQL**

I. Introduction

With the steady growth of web data, knowledge graphs (KGs) have been increasingly adopted to structure and store large-scale data on the web. A KG, presented in the Resource Description Framework (RDF) [1], often includes millions or billions of facts in the form of subject-predicate-object triples.

While SPARQL[1] is a standard way to access RDF KGs, end-users find it difficult due to the complexity of the SPARQL syntax and the RDF schema [2]–[4]. To tackle this drawback, question answering (QA) over KGs has emerged. A KG-based question answering system (KGQAS) targets to automatically answer natural language questions (NLQs) posed by end-users.

Parsing NLQs and obtaining answers present numerous challenges. Some of the key challenges are: (1) large search space in KGs; (2) mapping unstructured NLQs to structured queries without losing the intention of the NLQs; and (3) the linguistic gap between questions and KGs' vocabularies.

To address the mentioned challenges and improve KGQASs' performance, this paper proposes a graph-driven approach focusing on factoid[2] questions, both simple and complex.

Based on our objective, which is to improve the performance of QA over KG, our work seeks to answer the following key research questions: (1) how to extract a knowledge subgraph for a posed NLQ that reduces the size of the KG while covering the answer?; (2) how to bridge the gap between NLQs and SPARQL queries in a semantic parsing based KGQAS?; and (3) how to take advantage of the task of graph matching to match question graphs and knowledge subgraphs?.

Inline with our research questions, this paper makes the following contributions:

- We extend our previous research study to reduce the search space in KGs that introduces a new derivation of the personal page rank (PPR) algorithm, named Bi-Directed PPR (BiDPPR) [5].
- The syntactic structure of the question and a set of effective transformation rules are used to present question's intentions in graph structures.
- The task of graph matching between question graphs and knowledge subgraphs is reduced to node matching and edge matching.
- Extensive experiments are conducted to validate the performance of the proposed approach in terms of recall, precision and F1.

In our study, we conceptualize the task of QA over KGs to parse NLQs and then convert the questions into SPARQL queries. Even though QA over KGs remains a challenging

[1]https://www.w3.org/TR/rdf-sparql-query/

[2]The factoid questions are factual in nature and commonly start with a WH-word such as when, where, who, what, etc.

978-1-6654-9402-1/22/$31.00 ©2022 IEEE
DOI 10.1109/WI-IAT55865.2022.00050

Authorized licensed use limited to: Wageningen UR. Downloaded on December 07,2023 at 12:53:21 UTC from IEEE Xplore. Restrictions apply.

task, our approach outperforms state-of-the-art technologies, as demonstrated by our findings (see Section IV-C).

The remainder of the paper is organized as follows. Section II provides an overview of the related works. The proposed approach is described in Section III, and Section IV reports a detailed experimental evaluation, including a comparison against state-of-the-art solutions. Finally, Section V concludes the paper and gives directions for future research.

II. Related Works

In this section, we provide a review of the related works. First, we provide an overview of the different types of QA over KGs (see Section II-A). Thereafter, we discuss the related works, focusing on our study, which is graph-based QA approaches (see Section II-B).

A. Types of QA over KGs

1) *Template-based methods:* Template-based methods rely on predefined templates (a pseudo-query with some slots) to parse questions and provide logical forms [6]–[10]. The significant drawbacks of template-based systems are limited scalability and low flexibility of templates.
2) *Information retrieval-based methods:* These methods extract a subgraph with respect to a NLQ and treat the subgraph's nodes as candidate answers. Then, the semantic relevance of the question and candidate answers are computed according to the features extracted from the question and encoded candidate answers. Finally, the candidate with the highest matching score is selected as the final answer [11]. While these solutions can handle complex reasoning on a graph structure and fit into end-to-end training, the intermediate reasoning is less interpretable, and a considerable amount of training data is required [12]–[14].
3) *Semantic parsing-based methods:* This category aims to build a semantic parser that transforms an input NLQ into an intermediate form and then converts the intermediate form into a logical form. There are two mainstream branches to generate intermediate forms: (a) encoder-decoder-based techniques and (b) graph-based techniques [15]. The encoder-decoder techniques [16]–[18] utilise trees or high-level programming languages to represent questions' intentions which require a lot of annotated training examples [19]. The graph-based methods use graphs to bridge the gap between the questions and logical executable forms.

B. Graph-based QA approaches

The introduced system in [20], known as the graph answer (GAnswer), obtains two frameworks, namely node-first framework (NFF) and relation-first framework (RFF). In NFF, nodes of a given question are found and then the edges to connect nodes are identified to construct a graph. This framework finds candidates for each node and edge of the graph using two dictionaries. Once the candidate lists are provided, a button-up process is used to find approximate matches over the KG. Different from NFF, RFF begins with extracting semantic relations from a NLQ and then looks for the associated nodes based on heuristic rules. A user-guided graph similarity approach is presented in [21], named the graph alignment question answering (GAQA) framework. GAQA receives a question graph constructed by the end-user and builds a knowledge subgraph by filtering out unnecessary facts. An alignment-based method formulates the pairwise graph alignment with integer linear programming. The outcome of this alignment is converted to a structured query to retrieve the answer.

Since our approach does not rely on predefined templates, it is more robust in answering different questions than template-based techniques. Different from information retrieval-based techniques, our approach does not demand annotated training data and produces a more interpretable reasoning process by generating intermediate logic forms. Similar to the information retrieval-based approaches, we retrieve a knowledge subgraph for each NLQ. Moreover, our approach uses graph matching similar to GAnswer and GAQA. However, it is independent of users' assistance to arrive at query graphs and extracts smaller knowledge subgraphs for questions. Additionally, we use different similarity metrics to overcome ambiguities in the graph matching step.

III. The Proposed Approach

An overview of the proposed approach is presented in Figure 1, which consists of four major steps, including (1) knowledge subgraph construction, (2) question graph generation, (3) graph matching and (4) query execution. The first step targets to construct a knowledge subgraph for each question to avoid exploring the whole KG. To do this, we extend our previous research, BiDPPR [5], which enables us to find knowledge subgraphs through a bi-directed propagation process based on PPR. In the next step, we utilise the syntactic structure of the input NLQ and a set of predefined transformation rules to transform the question into an intermediate form named question graph. The third step matches the question graph to the knowledge subgraph to find the query graph. Finally, the query graph is converted to the corresponding SPARQL query to return the answers to the end-user.

Thus, in our work, a NLQ is converted into a question graph, a query graph is derived from the question graph, and the query graph is transformed into a SPARQL query, as shown in Figure 1.

A. Knowledge Subgraph Construction

The knowledge subgraph is constructed in three stages: (1) topic entity identification, (2) neighbourhood retrieval, and (3) knowledge subgraph retrieval.

1) Topic Entity Identification: The task of EL targets to map each entity mentioned in the questions to its corresponding entity in the KG.

Definition 1: A topic mention is a part of a given NLQ referring to an entity from a KG and shows the main focus of

Authorized licensed use limited to: Wageningen UR. Downloaded on December 07,2023 at 12:53:21 UTC from IEEE Xplore. Restrictions apply.

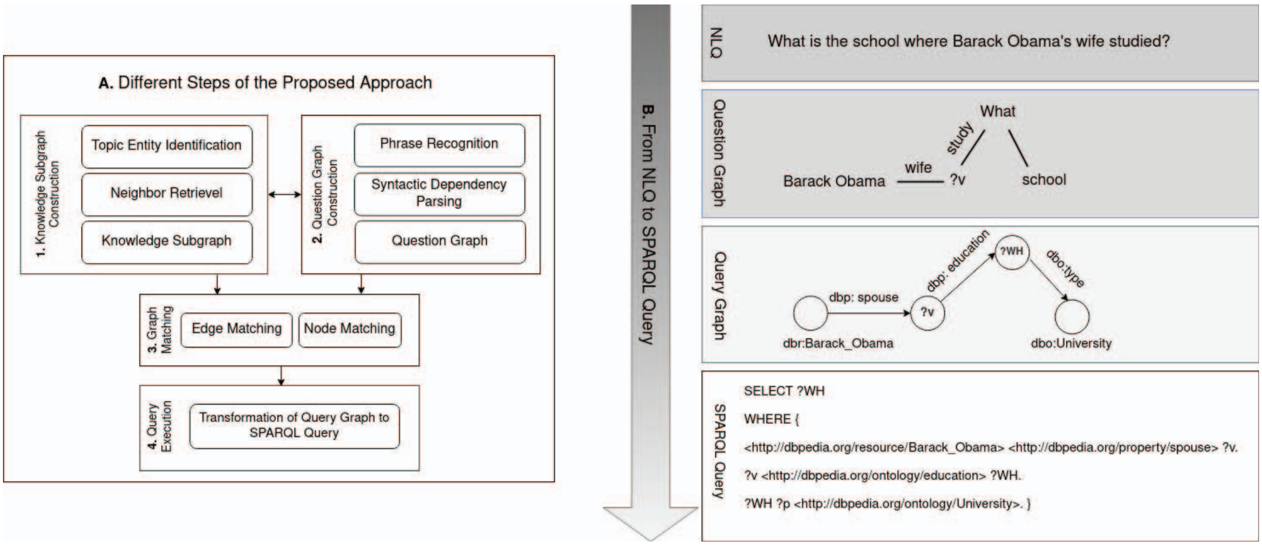

Fig. 1. Overview of the proposed approach

the NLQ. Also, the entity corresponding to the topic mention is called the topic entity. The topic mention and topic entity are denoted as T_m and T_e, respectively.

In this research study, DBpedia spotlight (DBS) [22] is adopted to recognize topic entities for questions.

Assumption 1: Given a NLQ, there is at least one topic mention T_m to reflect the main focus of the question.

Example 1: The entity mention and topic entity of the question “Where is Syngman Rhee buried?” are “Syngman Rhee” and <http://dbpedia.org/resource/Syngman_Rhee>, respectively.

2) Neighbourhood Retrieval: After identifying the topic entity of a NLQ, all the entities in the KG with the maximum distance m in KG, starting from the topic entity, are extracted (m equals the maximum path length in the question graph, which is discussed in Section III-B).

Definition 2: A neighbourhood graph is defined as $G_n = (N_N, E_N)$ where N_N is a set of entities around the topic entity T_e with distance $d <= m$ from T_e, E_N is a set of edges with distance $d < m$ from T_e, and m is the longest path in the question graph.

3) Knowledge Subgraph: The BiDPPR, a bi-directed propagation technique inspired by PPR [23] and the directed propagation method in [12], is summarized as following:

- A personalized vector which biases towards the topic entity is defined according to the neighbourhood graph G_n. The entries of the personalized vector are set to $\frac{1}{|N_N|}$ for all non-topic entities, and the corresponding entries of topic entities are set to $1+\frac{1}{|N_N|}$ where $|N_N|$ is the number of entities in G_n.

$$pr_v^{(0)} = \begin{cases} \frac{1}{|N_N|} + 1 & for\, topic\, entities \\ \frac{1}{|N_N|} & otherwise \end{cases} \qquad (1)$$

- The pre-trained word embeddings BERT [24] is applied to generate the embedding of the question and edges' surface-forms [5] and calculate the weights of the edges. The cosine similarity between the embeddings is considered as the edge's weight.
- The adjacency matrix of the neighbourhood graph G_N is generated according to the edge weights. If M denotes the adjacency matrix, the transpose of M is considered as M^T, including the inverse of relations between entities. The BiDPPR calculation is presented as:

$$pr^{(t)} = (1-\alpha)pr^{(t-1)} + \alpha\Big(\omega_1 \sum_r \sum_{<n,r,v>} w_r \cdot pr_n^{(t-1)} + \omega_2 \sum_r \sum_{<v,r,n>} w_r^{(t)} \cdot pr_n^{(t-1)}\Big) \qquad (2)$$

Where w_r and w_r^t denote the weights of the edge r in both directions based on the matrices M and M^T. Also, ω_1 and ω_2 are assumed as the coefficient ratio for the incoming edges and outgoing edges, respectively.

After T iterations, the top-k entities by BiDPPR score and any edges between them are selected as the relevant facts to shape the question knowledge subgraph. We give a formal definition of knowledge subgraph in Definition 3 as follows:

Definition 3: A knowledge subgraph is defined as $G_k = (N_K, E_K)$ where $N_K \subset N_N$ and $E_K \subset E_N$ and N_K includes the entities with the highest BiDPPR scores which are more likely to be answer entities.

B. Question Graph Construction

This section discusses how to generate question graphs as intermediate forms to bridge the gap between given question

Authorized licensed use limited to: Wageningen UR. Downloaded on December 07,2023 at 12:53:21 UTC from IEEE Xplore. Restrictions apply.

sentences and SPARQL queries.

Definition 4: Given a NLQ q, a question graph is defined as an undirected labeled graph $G_q = (N_q, E_q)$ to present the intention of q where N_q and E_q denote the nodes and edges.

1) Node Recognition: For each input question, we extract three kinds of question nodes, including (1) variable nodes, (2) constant nodes and (3) type nodes.

The constant nodes are basically the named entities mentioned in a question sentence. The variable nodes include the main variable node, which refers to the WH-word of the question, and zero or several middle-variable nodes inserted based on the transformation rules once the question graph is being constructed. To define the type nodes, the noun phrases of the question sentence are extracted. Then, we use a mention-class dictionary to map the noun phrases into the classes of the schema of the underlying KG. Each noun phrase which includes a corresponding class, is assumed as a type node. Those noun phrases not mapped to the classes are later used to shape relations between the question nodes.

A mention-class dictionary consists of all the classes used in the underlying KG along with their hypernyms and synonyms. We use NLTK WordNet[3] to find hypernyms and synonyms of each class.

Example 2: The question sentence "Which books by Kerouac were published by Viking Press?" includes "Kerouac" and "Viking Press" as the constant nodes, "Which" and "book" as the variable node and type node, respectively.

2) Syntactic Parsing: Syntactic parsing (or dependency parsing) is the task of assigning a syntactic structure to a sentence, and a dependency tree is the most widely used syntactic structure. We define the following transformation rules to prune and simplify the generated dependency tree:

- The unnecessary dependency relations (e.g., passive auxiliary, open clausal complement) are filtered to transform the dependence tree to a more simplified syntactic structure [25].
- If a sub-tree only includes the leaf proper-noun-phrase (NNP) nodes, the sub-tree is reduced to a merged node, including the combination of sub-tree's nodes.
  Example 3: Given the question "When did the Boston Tea Party happen?", the nodes "Boston", "Tea", and "Party" are merged into the "Boston Tea Party" node as shown in Figure 2.

3) Question Graph: After pruning the dependency tree, the question graph is created. The nodes of a question graph are constituted from question nodes, and edges are formed based on the transformation rules as follows.

- When a type node is connected to the main variable node in a dependency tree, all the nodes linked to the type node will be connected to the main variable node.
  Example 4: The type node "books" and main variable node "Which" are connected to each other in the dependency tree of the question *"Which books by Kerouac were published by Viking Press?"*, thus the nodes "Kerouac"

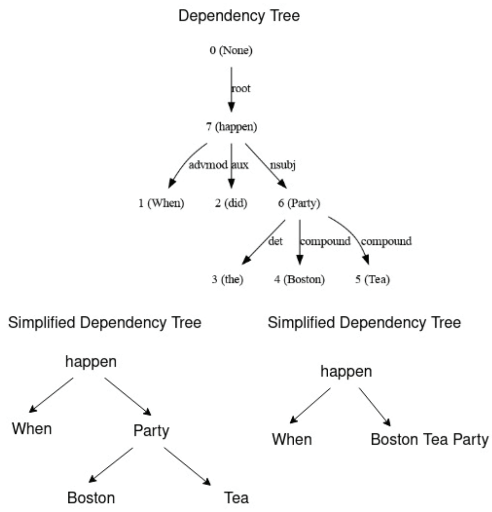

Fig. 2. Dependency tree and simplified dependency tree of the question "When did the Boston Tea Party happen?"

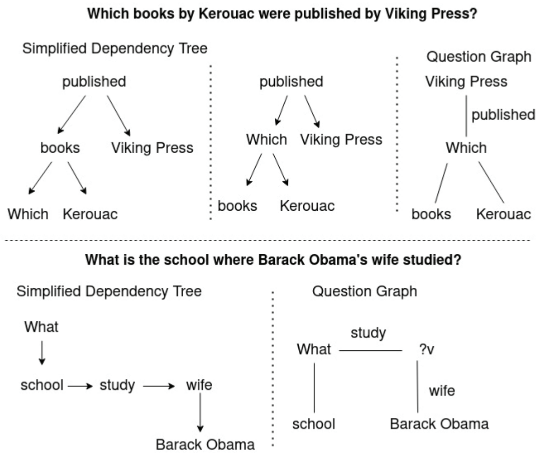

Fig. 3. Examples of applying the transformation rules

  and "published" will be linked to the node "Which" as shown in Figure 3.
- Two question nodes are linked through an edge in a question graph if the simple path between those two nodes in the dependency tree does not include any other question node. Additionally, the words along the simple path shape the relation [20].
  Example 5: Since there is no question node along the simple path between "Which" and "Viking Press", as shown in Figure 3, these two nodes are connected through an edge in the question graph, represented by the edge label "published".
- If a constant node is connected to a noun-phrase that is not of type node, then a middle-variable node is inserted. The middle-variable node is connected to the constant

[3]https://www.nltk.org/_modules/nltk/corpus/reader/wordnet

Authorized licensed use limited to: Wageningen UR. Downloaded on December 07,2023 at 12:53:21 UTC from IEEE Xplore. Restrictions apply.

node, and the noun-phrase node is transformed to the edge between two nodes.

Example 6: Given the question *"What is the school where Barack Obama's wife studied?"*, the constant node "Barack Obama" and the noun-phrase "wife" are transformed into the relation "wife" with the connected nodes "Barack Obama" and middle-variable node ?v as depicted in Figure 3.

In our work, we divide the edges' labels in question graphs into three groups, including (1) the edges with known labels, (2) the edges with unknown labels, and (3) the edges describing rdf:type.

Example 7: As shown in Figure 3, the edge between nodes "Which" and "Viking Press" includes the known label "published", and the edge between nodes "Which" and "Kerouac" has an unknown label. Additionally, the edge between nodes "What" and "school" shows the node "What" is a type (an instance) of node "School" (rdf:type).

Since a question graph has to be matched over a KG, we compute the maximum path length in the question graph and then consider it as the number of required hops m to retrieve the neighbourhood graph (more details in Section III-A2).

Assumption 2: The maximum path length in the G_q is considered as the number of required hops m to build the neighbourhood graph G_n.

C. Graph Matching

Graph matching is the task of finding a similarity between two graphs through exact or inexact graph matching. The exact graph matching is restricted to finding a one-to-one mapping between two graphs with the name number of nodes, while the inexact graph matching is applied to the cases where the number of nodes is different [26], [27]. Thus, the task of finding the similarity between the question graph and knowledge subgraph is an inexact graph matching problem.

In our work, the directed graph found by node matching and edge matching is named the query graph[4].

Definition 5: A query graph is defined as $G_{qry} = (N_{qry}, E_{qry})$ where is found based on inexact graph matching between the question graph $G_q = (N_q, E_q)$ and knowledge subgraph $G_k = (N_K, E_K)$.

Initially, a query graph includes a topic entity associated with the topic mention and then extends in I iterations. The query graph is expanded in each iteration by matching one more edge or node from the question graph with the knowledge subgraph. Thus, the first matching subgraph which is formed in the iteration process is considered as the query graph. The iterative process is terminated when edge matching and node matching are performed for all of the edges and nodes in the question graph. Since unknown edges and variable nodes can be associated with any nodes and edges in the knowledge subgraph, edge matching and node matching are not performed in these cases.

[4]The nodes and edges of a question graph are spans of the question while the nodes and edges of query graph are the enties or propoertes from

1) Edge Matching: For each edge in G_q, structural and semantic similarities are utilised to identify its associated edge in G_k [21]. We compute the structural similarity of an edge-pair, and if two edges are similar in the structure, the semantic similarity of the edges is computed.

Definition 6: An edge-pair is denoted as (r_q, r_k) consisting of two edges r_q and r_k where $r_q \in G_q$ and $r_k \in G_k$.

Structural similarity is defined according to the distance of edges from the topic mention T_m and topic entity T_e in G_k and G_q, respectively.

Definition 7: An edge r_q in G_q is structurally similar to an edge r_k in G_k, if the length of simple path between r_q and T_m is equal to the length of simple path between r_k and T_e.

Given an edge-pair (r_q, r_k), semantic similarity is computed if r_q and r_k are in the same distance. Semantic similarity is associated with the similarity between edges based on the likeness of their meanings. We use the edges' surface-forms in the knowledge subgraph and the edges' labels in the question graph to compute semantic similarities. Rather than applying one measure for semantic similarity, we explore the idea of combining existing measures into a semantic similarity ensemble. We take advantage of word embedding BERT, WordNet [28], and Levenstshtein distance [29]. Given an edge-pair, the embeddings of the surface-form and label are generated using the pre-trained word embedding BERT, and then their cosine similarity is calculated as a similarity score. WordNet is applied to generate two sets of synonyms according to the surface-form and label. Then, we compute the Levenstshtein distance of the elements among two sets, and normalize them as another score of similarity. Algorithm 1 presents the edge matching process.

2) Node Matching: Similar to the edge matching, we use the structural and semantic similarities to map each node in G_q to a corresponding node in G_k.

Thus, given each non-variable node in G_q, the nodes with a similar structure are found in G_k, and then the semantic similarity of those node-pairs is calculated to select the best match in G_q. Note that the applied algorithm for node matching is similar to the edge matching process in Algorithm 1.

D. Query Execution

Given a query graph, the SPARQL query is constructed through simple conversion rules and then executed over the KG. The primary conversion rules include (1) each edge with its nodes connected presents a triple in which the head node and tail node are the subject and object of the triple, (2) the nodes or edges with the URI resources or literal values are inserted into the triples with the URIs or literal values, (3) the variable nodes or unknown-labeled edges are inserted with the variable names, and (4) the main variable (WH-node) which is the answer of the factoid question, needs to be fetched in the SELECT part of the query.

We apply a set of simple rules to handle questions, including aggregation phrases. We assume there are only a few aggregation phrases (how many, how much, greater than, less than, and etc) and then define a set of simple rules to

Authorized licensed use limited to: Wageningen UR. Downloaded on December 07,2023 at 12:53:21 UTC from IEEE Xplore. Restrictions apply.

add constraints to the queries. For example, in the question "How many moons does Mars have?", the COUNT rule is used to count the main variable.

Algorithm 1: Edge Matching Process

Input: Question graph G_q, Knowledge subgraph G_k
Result: Vector o including the matched edges from G_k
for *each edge* e_k *in* G_k **do**
  Compute d_k as the length of simple path between e_k and topic entity T_e;
  Find S_{ek} as the set of synonyms for e_k;
end
for *each edge* e_q *in* G_q **do**
  if e_q *has a known label* **then**
    Find S_{eq} as the set of synonyms for e_q;
    for *each edge* e_k *in* G_k **do**
      if d_k *equals* d_q **then**
        Compute s_{cs} as the cosine similarity between the embeddings of e_q and e_k;
        Compute and normalize the Levenstshtein distance between elements of sets S_{ek} and S_{eq}, and select the maximum as s_{ld};
        Calculate S_{sim} as the liner combination of s_{ld} and s_{cs};
        Store S_{sim} as the similarity score of the edge-pair (e_q, e_k) in vector v
      end
    end
    Find (e_q, m_k) with the highest S_{sim} from v;
    Store m_k as the matched edge of e_q in vector o
  end
end
Return vector o

IV. Experimental Study

The proposed approach is evaluated on DBPedia [30] as the KG using the questions of QALD-6 as the QA dataset. The results are compared with all systems in the QALD-6 competition as well as GAQA and GAnswer.

A. Dataset

a) KG: In our experiment, DBPedia as an open-domain large-scaled KG is adopted. DBPedia is extracted from structured data in Wikipedia through a crowd sourcing community.

b) QA Benchmark: The Question Answering over Linked Data challenge (QALD) provides up-to-date benchmarks for assessing and comparing QASs over DBPedia. The QALD-6 which is the sixth installment of the QALD challenge, includes 350 training questions and 100 test questions.

B. Metrics

We adopt three evaluation metrics, including recall, precision and F1-score, formulated as the following:

$$\text{Recall} = \frac{\text{correctly retrieved answers}}{\text{gold standard answers}} \tag{3}$$

$$\text{Precision} = \frac{\text{correctly obtained answers}}{\text{returned answers}} \tag{4}$$

$$\text{F1} = 2 * \frac{\text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}} \tag{5}$$

C. Results and Discussion

Table I presents the experimental results, where 'Right' denotes the number of questions that are correctly answered. Our proposed approach, denoted GDQA (Graph Driven Question Answering), achieves 0.75%, 0.82%, and 0.78% for the recall, precision, and F1, respectively and answers 72 questions correctly, outperforming the participants in the QALD-6 competitions. Compared to GAnswer (NFF and RFF), QDQA improves the recall by 5% and correctly answers more questions by 4%.

Although the results of GAQA and CANaLI [31] show slightly better performance than our results, GAQA requires users to submit questions as question graphs (see Section II) and CANaLI uses a controlled natural language (CNL) to restrict the grammar used to pose input questions.

TABLE I
Main Results on QALD-6 Test Set

	Processed	Right	Recall	Precision	F1
GDQA	100	72	0.75	0.82	0.78
GAQA	100	74	0.83	0.87	0.84
NFF	100	68	0.70	0.89	0.78
RFF	100	40	0.43	0.77	0.55
CANaLI	100	83	0.89	0.89	0.89
UTQA	100	63	0.69	0.82	0.75
KWGAnswer	100	52	0.59	0.85	0.70
SemGraphQA	100	20	0.25	0.70	0.37
UIQA1	44	21	0.63	0.54	0.25
UIQA2	36	14	0.53	0.43	0.17

According to 28% questions that our proposed approach can not answer correctly, the failure analysis of the work is provided. We find that there are two main reasons for the failure to answer questions. The first failure reason is question graph construction. As an example, the question graph of the question "How many grand-children did Jacques Cousteau have?" includes "How many" and "Jacques Cousteau" as the variable node and constant node, respectively and "grand-children" as the relation while the answer entities are accessed through the question graph: {<How many, child, ?v>, <?v, child, Jacques Cousteau>}. The second reason is finding semantic similar nodes or edges in the graph matching step. For example, in the question "Where is Syngman Rhee buried?", the relation "burn" in the question graph has to match the relation "restingPlace" in the knowledge subgraph while the edge matching is failed to find the matching. Since the other reasons, such as knowledge subgraph construction, and complex aggregation, happen far much less than the first two reasons, their ratio is reported in total.

From this experiment, we find out that our approach achieves high performance in answering questions over KGs. While our approach is tested on an open-domain KG, it can be generalised across diverse KGs, which we regard as an advantage of the work. In addition, end-users are not restricted to posing their questions in a specific format (e.g., graphs or CNL).

Authorized licensed use limited to: Wageningen UR. Downloaded on December 07,2023 at 12:53:21 UTC from IEEE Xplore. Restrictions apply.

We also acknowledge the following limitations of the work. We assume that every question includes at least one topic mention, whereas there may be no topic mention in some cases (e.g., "Who died in the same place where they were born?"). Moreover, although our approach can answer some aggregation questions (e.g., count questions), it may not answer questions with more complex aggregation functions.

V. Conclusions

QA over KGs provides an easy and effective way for end-users to access KGs using NLQs rather than complicated structured query languages such as SPARQL. We propose a graph-driven approach to answer NLQs in four steps: (1) knowledge subgraph construction, (2) question graph construction, (3) matching graphs, and (4) query execution. Our experimental results show that the proposed approach achieves significantly higher recall, precision and F1-score in comparison to the QALD competition's results; however, there still exists much space for improvement of QA over KGs.

A possible direction for our future work is to utilise semantic parsing as well as syntactic parsing in the question graph generation step and address the limitations, which are highlighted in Section IV-C.

Acknowledgment

This work has been partially funded by the project WordLiftNG within the Eureka, Eurostars Programme (grant agreement number 877857 with the Austrian Research Promotion Agency (FFG)) and the project KI-Net within the Interreg Österreich-Bayern 2014-2020 programme (grant agreement number AB 292). Additionally, we would like to express our gratitude to Oleksandra Roche-Newton for providing feedback that aided in the manuscript's improvement.

References

[1] G. Klyne, "Resource description framework (rdf): Concepts and abstract syntax," *http://www. w3. org/TR/2004/REC-rdf-concepts-20040210/*, 2004.

[2] A. Ait-Mlouk and L. Jiang, "Kbot: a knowledge graph based chatbot for natural language understanding over linked data," *IEEE Access*, vol. 8, pp. 149 220–149 230, 2020.

[3] L. Shiqi, K. Stockinger, M. Anisimova, M. Gil *et al.*, "Querying knowledge graphs in natural language," *Journal of Big Data*, vol. 8, no. 1, 2021.

[4] K. Angele, M. Meitinger, M. Bußjäger, S. Föhl, and A. Fensel, "Graphsparql: a graphql interface for linked data," in *Proceedings of the 37th ACM/SIGAPP Symposium on Applied Computing*, 2022, pp. 778–785.

[5] S. Aghaei, K. Angele, and A. Fensel, "Building knowledge subgraphs in question answering over knowledge graphs," *in Proceedings of the 22nd International Conference on Web Engineering*, pp. 1–15, 2022.

[6] S. Aghaei, "Question answering over knowledge graphs," *Proceedings of the 15th International Rule Challenge, 7th Industry Track, and 5th Doctoral Consortium*, 2021.

[7] W. Zheng, L. Zou, X. Lian, J. X. Yu, S. Song, and D. Zhao, "How to build templates for rdf question/answering: An uncertain graph similarity join approach," in *Proceedings of the 2015 ACM SIGMOD international conference on management of data*, 2015, pp. 1809–1824.

[8] A. Abujabal, R. Saha Roy, M. Yahya, and G. Weikum, "Never-ending learning for open-domain question answering over knowledge bases," in *Proceedings of the 2018 World Wide Web Conference*, 2018, pp. 1053–1062.

[9] A. Abujabal, M. Yahya, M. Riedewald, and G. Weikum, "Automated template generation for question answering over knowledge graphs," in *Proceedings of the 26th international conference on world wide web*, 2017, pp. 1191–1200.

[10] W. Cui, Y. Xiao, H. Wang, Y. Song, S.-w. Hwang, and W. Wang, "Kbqa: learning question answering over qa corpora and knowledge bases," *arXiv preprint arXiv:1903.02419*, 2019.

[11] Y. Lan, G. He, J. Jiang, J. Jiang, W. X. Zhao, and J.-R. Wen, "A survey on complex knowledge base question answering: Methods, challenges and solutions," *arXiv preprint arXiv:2105.11644*, 2021.

[12] H. Sun, B. Dhingra, M. Zaheer, K. Mazaitis, R. Salakhutdinov, and W. W. Cohen, "Open domain question answering using early fusion of knowledge bases and text," *arXiv preprint arXiv:1809.00782*, 2018.

[13] H. Sun, T. Bedrax-Weiss, and W. W. Cohen, "Pullnet: Open domain question answering with iterative retrieval on knowledge bases and text," *arXiv preprint arXiv:1904.09537*, 2019.

[14] G. He, Y. Lan, J. Jiang, W. X. Zhao, and J.-R. Wen, "Improving multi-hop knowledge base question answering by learning intermediate supervision signals," in *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, 2021, pp. 553–561.

[15] B. Fu, Y. Qiu, C. Tang, Y. Li, H. Yu, and J. Sun, "A survey on complex question answering over knowledge base: Recent advances and challenges," *arXiv preprint arXiv:2007.13069*, 2020.

[16] L. Dong and M. Lapata, "Language to logical form with neural attention," *arXiv preprint arXiv:1601.01280*, 2016.

[17] K. Xu, L. Wu, Z. Wang, M. Yu, L. Chen, and V. Sheinin, "Exploiting rich syntactic information for semantic parsing with graph-to-sequence model," *arXiv preprint arXiv:1808.07624*, 2018.

[18] C. Liang, J. Berant, Q. Le, K. D. Forbus, and N. Lao, "Neural symbolic machines: Learning semantic parsers on freebase with weak supervision," *arXiv preprint arXiv:1611.00020*, 2016.

[19] W. Zheng, H. Cheng, J. X. Yu, L. Zou, and K. Zhao, "Interactive natural language question answering over knowledge graphs," *Information Sciences*, vol. 481, pp. 141–159, 2019.

[20] S. Hu, L. Zou, J. X. Yu, H. Wang, and D. Zhao, "Answering natural language questions by subgraph matching over knowledge graphs," *IEEE Transactions on Knowledge and Data Engineering*, vol. 30, no. 5, pp. 824–837, 2017.

[21] M. Bakhshi, M. Nematbakhsh, M. Mohsenzadeh, and A. M. Rahmani, "Data-driven construction of sparql queries by approximate question graph alignment in question answering over knowledge graphs," *Expert Systems with Applications*, vol. 146, p. 113205, 2020.

[22] P. N. Mendes, M. Jakob, A. García-Silva, and C. Bizer, "Dbpedia spotlight: shedding light on the web of documents," in *Proceedings of the 7th international conference on semantic systems*, 2011, pp. 1–8.

[23] L. Page, S. Brin, R. Motwani, and T. Winograd, "The pagerank citation ranking: Bringing order to the web." Stanford InfoLab, Tech. Rep., 1999.

[24] N. Reimers and I. Gurevych, "Sentence-bert: Sentence embeddings using siamese bert-networks," *arXiv preprint arXiv:1908.10084*, 2019.

[25] R. Beaumont, B. Grau, and A.-L. Ligozat, "Semgraphqa@ qald5: Limsi participation at qald5@ clef." in *Proceedings of the sixth conference and labs of the evaluation forum*, 2015.

[26] D. L. Sussman, Y. Park, C. E. Priebe, and V. Lyzinski, "Matched filters for noisy induced subgraph detection," *IEEE transactions on pattern analysis and machine intelligence*, vol. 42, no. 11, pp. 2887–2900, 2019.

[27] K. Riesen, X. Jiang, and H. Bunke, "Exact and inexact graph matching: Methodology and applications," *Managing and Mining Graph Data*, pp. 217–247, 2010.

[28] R. Poli, M. Healy, and A. Kameas, "Theory and applications of ontology: Computer applications." Springer, 2010, pp. 231–243.

[29] V. I. Levenshtein *et al.*, "Binary codes capable of correcting deletions, insertions, and reversals," *Soviet physics doklady*, vol. 10, no. 8, pp. 707–710, 1966.

[30] J. Lehmann, R. Isele, M. Jakob, A. Jentzsch, D. Kontokostas, P. N. Mendes, S. Hellmann, M. Morsey, P. Van Kleef, S. Auer *et al.*, "Dbpedia–a large-scale, multilingual knowledge base extracted from wikipedia," *Semantic web*, vol. 6, no. 2, pp. 167–195, 2015.

[31] G. M. Mazzeo and C. Zaniolo, "Answering controlled natural language questions on rdf knowledge bases." in *Proceedings of the International Conference on Extending Database Technology*, 2016, pp. 608–611.

Authorized licensed use limited to: Wageningen UR. Downloaded on December 07,2023 at 12:53:21 UTC from IEEE Xplore. Restrictions apply.