Applied Intelligence (2023) 53:4124–4144
https://doi.org/10.1007/s10489-022-03732-9

The state of the art in open domain complex question answering: a survey

Romina Etezadi[1] · **Mehrnoush Shamsfard**[1]

Accepted: 6 May 2022 / Published online: 6 June 2022

Abstract
Research on question answering (QA) systems has a long tradition. QA systems, as widely used systems in various applications, seek to find the answers to the given questions through the available resources. These systems are expected to be capable of answering various types of questions, including simple questions whose answers can be found in a single passage or sentence and complex questions which need more complicated reasoning to find the answer or their answer should be found by traversing several relations. Nowadays, answering complex questions from texts or structured data is a challenge in QA systems. In this paper, we have a comparative study on QA approaches and systems for answering complex questions. For this purpose, firstly, this paper discusses what a complex question is and surveys different types of constraints that may appear in complex questions. Furthermore, it addresses the challenges of these types of questions, the methods proposed to deal with them, and benchmark datasets used to evaluate their strengths and weaknesses.

Keywords Question answering · Complex question · Text based question answering · Knowledge based question answering

1 Introduction

Answering a wide range of real-world questions asked in natural languages has been received substantial interest for several decades. Such a growing interest has led to the creation of question answering (QA) systems. Despite numerous studies conducted to find a solution for this task, there exist challenges that remain largely unsolved. QA systems can be divided into different categories according to different aspects and dimensions. Some of these aspects are listed below:

Question Domain: One of the important aspects of QA systems is the question domain, which divides the systems into two major categories: *open-domain* and *closed-domain*. A closed-domain system (CDQA) focuses on answering questions that are asked in limited and bounded domains like medical or sports [17, 64]. In comparison, an open-domain system (ODQA) aims to answer questions in different domains. Such a system should access world knowledge to be able to answer questions about anything [14, 59, 107, 120].

Question Type: Question type refers to the type of responses that the question expects [8], including factoids (who/whom/when/where), hypothetical, confirmation (yes/no), causal (how/why), etc. [20, 71].

Question Complexity: Based on the complexity level of the question and the efforts needed to find the answer, questions are divided into simple and complex categories. The answer to simple questions is usually found in a single passage or sentence while complex questions need more complicated reasoning to find the answer through multiple passages or sentences or their answer should be found by traversing several relations in a knowledge base. In other words, Complex questions require inferring several semantic components or applying complex methods of natural language understanding (NLU) to retrieve the answers.

Previous surveys mostly concentrate on the methods that are employed for answering simple questions [4, 83, 103]. On the other hand, some other studies briefly discuss complex questions as well [3, 43, 84]. Also, some studies consider the existence of complex questions as one of the challenges in QA systems [20, 71], although their definition

✉ Mehrnoush Shamsfard
m-shams@sbu.ac.ir

Romina Etezadi
ro.etezadi@mail.sbu.ac.ir

[1] Faculty of Computer Science and Engineering, Shahid Beheshti University, Tehran, Iran

of complex questions is different from ours. This issue will be discussed further in Section 2. In another research [81], the shortcomings of QA systems in answering complex questions and textual inference are discussed. In [19], a survey is proposed for analyzing the methods of QA systems over knowledge bases, as well as mentioning the existence of complex questions.

Challenges for answering complex questions using knowledge bases are discussed in [32, 58], Although to the best of our knowledge, no previous research has investigated the challenges of answering complex questions using different resources. This issue is elaborated on in Section 2.

This paper describes the approaches and their complications for answering such questions in open-domain QA systems. Closed-domain QA systems mostly, on the other hand, use the template- or rule-based approaches to answer complex questions due to having a restricted domain [36, 65]. For this reason, we only take open-domain QA systems into account in this work and classify their algorithms into three categories based on the type of resource structure they utilize to find the answers. The main reason for choosing the resource structure as a classification criterion is that the challenges for facing the constraints and semantic complexities to retrieve the correct answers are different. This will be discussed more in Section 4.

The focus of this paper is mostly on work that has been done in English. However, we include several studies on other languages as well. Papers included in this survey were published from 2008 to 2020. We cover different approaches in various aspects, such as the structure of the resources they use, the architecture of deep learning models, or their main algorithm's steps. The remainder of this paper is organized as follows: Section 2 describes the definition of complex questions. Section 3 summarizes the benchmark datasets used in the papers to train or test complex QA systems. Section 4 presents different classes of approaches introduced to answer complex questions. Section 5 presents the evaluation metrics. Section 6 discusses the results of the relevant works. Section 7 we the future studies are discussed. Section 8 concludes the paper.

2 Complex questions

Reviewing the literature reveals different points of view regarding what exactly a complex question is. Based on the various definitions in the literature, we classify the complexity in complex questions into two sub-classes: *answer-retrieval-complexity* and *question-understanding-complexity*.

Answer-retrieval-complexity: Some researchers [20, 71] take *why*, *how*, *hypothetical*, *confirmation (yes-no)*, and *opinion* questions as complex questions in which the complexity is mostly on the answer retrieval part. This part may need reasoning over multiple clues in different documents. This type of complexity is also known as long-form question-answering in which NLU techniques are used alongside information retrieval methods to find the answer. The examples of this type of complexities are shown in the 8th to 12th row in Table 1.

Question-understanding-complexity: Questions that have multiple facts and multiple constraints such as order, aggregation, etc., are also viewed as complex questions that require inferring semantic components existing within the question. For example, to answer the question *"Who is the second richest person born in the United States?"*, we should pay attention to several semantic components. For instance, it must be known whether it is a person, is born in the U.S., and his/her wealth is in the second rank compared to others. It is of note

Table 1 Complex question classes with example. Questions using *How* can be also procedural or even in some cases factoid such as *"How are Barack Obama and Michelle Obama related?"*

Complexity type	Example
1. Multi-entity	What movies have **Ben Affleck** and **Matt Damon** appeared in together?
2. Multi-relation	What was the **death cause** of the **writer of** the Cranberries?
3. Time-explicit	Who was the United States president in **1996**?
4. Time-implicit	How old was Tom Cruise **when Lewis Allen died**?
5. Operational-ordinal	What is **the second** tallest mountain in the world?
6. Operational-aggregation	**How many** movies did Alfred Hitchcock direct?
7. Type	Which **city** did Lewis Allen born?
8. Why	Why did the United States attack Iraq?
9. How	How does a heart attack happen?
10. Hypothetical	What would happen if all the ice in the Arctic melts?
11. Confirmation	Is Mount Everest located in China?
12. Opinion	What are the opinions of people in America about racism?

that this sub-class can also result in a complex answer retrieval, but this complexity is also highly dependent on understanding the question correctly. Questions in the question-understanding-complexity category have different definitions according to the type of resources that QA systems employ to extract the answers. Local documents, web pages (unstructured data), tables, linked-data and knowledge bases such as Freebase [11] and DBpedia [5] (structured data) are the main types of resources that are widely used to find the answers. The examples of this type of complexities are shown in the first to 7th row in Table 1.

Throughout this paper, the complexity of a question is considered based on the following requirements based on the resource structure: 1) Finding multiple facts, relations, or predicates in knowledge bases, known as a multi-hop question. 2) Deep understanding of the relationship between various information in documents, passages, or sentences. 3) Adding different operations or dealing with multiple constraints to restrict the answers. Overall, based on previous work and literature, we categorize the question understanding complexity into the following five classes [7, 39, 42, 67]:

1) *Multi-hop complexity*. In this class, a question contains multiple relations or facts. This type of question is also known as multi-hop questions as they have multiple relations opposite to one- or single-hop questions that only have one relation.
2) *Multi-entity constraint*. In this class, a question has multiple entities in it. In the example of the first row of Table 1, *Ben Affleck* and *Matt Damon* are two target entities.
3) *Time constraint*. The answers to the questions in this class are at a certain time or time intervals. There are two types in this class: *Explicit temporal* and *Implicit temporal*. In explicit temporal, the time constraint is expressed in an explicit form like *1996* in the example of row 3 in Table 1. In comparison, in implicit temporal, the time constraint is expressed in an implicit form and it needs to be converted to an explicit form such as *when Lewis Allen died* in the example of row 4 in Table 1.
4) *Operational constraint*. Questions in this class require applying mathematical operations on the data to find the answers. This class is divided into two types, namely: *ordinal* and *aggregation*. In ordinal constraints, it is necessary to find a ranked set of related elements such as superlative, comparative, maximum, minimum, etc. In aggregation constraints, we need to calculate the total number of a set like questions that contain "how many", "number of", etc. Examples of these two types can be found in rows 5 and 6 in Table 1. Moreover, some other operations like changing units belong to this class as well. For instance, in question *"How many feet tall is Jeff Bezos?"*, the person's height can be in other units like centimeter and meter in the resource.
5) *Type constraint*. Possibly, the answer or any information in the question has a type. Therefore, recognizing the wrong type can cause retrieving incorrect answers. For example, for the question in the 7th row of Table 1, the answer should be a "city" name, instead of other locations such as country or town.

Due to the existence of these constraints, one should face many challenges to find the proper answers:

- Detecting constraints correctly.
- Finding the right order if there are multiple constraints; for example, in question "What is the second highest mountain in Europe?", we can first get all the mountains with their corresponding heights and then sort them and find if it is located in Europe. Or, we first can get all the mountains and select the ones that are in Europe and then sort them in the correct order.
- Finding the right composition of constraints [45] when it comes to building logical forms.
- Inferring multiple semantic clues and finding multiple related documents, passages, and sentences that lead to the correct answers.

In this paper, we aim to study the recent methods that are introduced to deal with these challenges. However, before that, the developed datasets for this task are discussed.

3 Datasets

There are different benchmark datasets for examining the capability of QA systems to answer complex questions. Although these datasets are mostly in the English language, there are some datasets for other languages such as PeCoQ [27] for Persian. Also, there are some multi-lingual datasets which mostly include English as well [60]. Some of these datasets contain a combination of simple and complex questions, while others contain only complex questions.

- **Final Jeopardy!**[1]: Jeopardy! is an American television show in which players should answer questions from different topics that have multiple facts about the answer. Final Jeopardy! is the final level of the program, in which the dataset contains questions that are even harder for Jeopardy! players and is used for constructing QA systems such as IBM Watson as well.
- **WebQuestions** [10]: This dataset has both simple and complex questions. It is worth mentioning that 85%

[1] Registered trademark of Jeopardy Productions Inc.

of its questions are simple ones [111]. WebQuestions contains 3,778 training and 2,032 test examples.

- **ComplexQuestions***[2] [7]: This dataset contains 2100 complex questions (1300 on training and 800 on test set). It is the result of the combination of *complexQuestions* [115] which has 300 questions, and the complex questions in *WebQuestions*, as well as 878 manually created questions.
- **QALD**[3]: QALD is a series of QA campaigns for evaluating QA systems over linked-data for different languages. Benchmark datasets of QALD have different complexities. The most recent one is QALD-9.
- **ComplexQuestions** [1]: This dataset has 150 complex questions and answers but no training data. It is used as test data to evaluate the capability of their QA system, which automatically learns templates from question-answer pairs of simpler questions, in answering complex ones. Overall, this dataset is constructed from the collection of questions in WikiAnswers [29].
- **ComplexWebQuestions**[4] [93]: This dataset is built with the help of *WebQuestionsSP* [114] which contains questions with their corresponding SPARQL query. The builders of this dataset added constraints (such as superlatives, comparatives, conjunctions, and compositions) to the existing SPARQL query and create more complex questions. There are 34,689 instances in it (27,639 data on training, 3,519 on development, and 3,531 on test set).
- **MetaQA** [119][5]: It is a series of datasets that contain more than 400K questions (329,282 training, 39,138 development, and 39,093 test data) for both single- and multi-hop reasoning. About 28.5% of the whole dataset contains simple questions.
- **HotpotQA** [110][6]: This dataset contains 113k Wikipedia-based complex question-answer pairs for multi-hop reasoning. There are also about 18K single-hop questions in the training set. HotpotQA consists of three main sets: training set (training-easy, training-medium, training-hard), dev set, test set (test-distractor, test-fullwiki).
- **LC-QuAD 2.0**[7] [24]: This dataset, which was first created with, has 5000 questions with their SPARQL query based in DBpedia [95]. However, the latest version 2.0 contains over 30,000 complex questions (24,180 training and 6,046 test) with their corresponding paraphrasing and SPARQL queries for Wikidata [101] and DBpedia knowledge bases.
- **SQuAD** [78]: SQuAD consists of more than 100,000 question-answer pairs based on more than 500 Wikipedia articles. This dataset is mostly used for training and evaluating reading comprehension methods.
- **WikiSQL**[8][121]: This dataset includes 80,654 hand-annotated instances of natural language questions with their corresponding SQL queries. This dataset is mostly employed for systems that use tabular data. The builders of this dataset extracted tables used in WikiSQL from Wikipedia.
- **WikiTQ**[9][76]: WikiTableQuestions consist of 22,033 complex question-answer pairs and 2,108 tables from Wikipedia. This dataset is based on tabular data extracted from Wikipedia.
- **QA@CLEF**[10]: This dataset is built over multiple languages. It has different track editions from 2003 until 2014.
- **Mlqa** [60]: Mlqa consists of over 12,000 question-answer pairs in English. This dataset has seven languages: English, Arabic, German, Spanish, Hindi, Vietnamese and Simplified Chinese.
- **PeCoQ** [27]: This dataset has 10,000 Persian complex questions based on Farsbase [82] with their corresponding SPARQL query and two different paraphrasings.

As part of question answering tasks, there are other datasets that pose some challenges, such as WikiQA [109]. However, these datasets do not have complex questions. As a result, we do not discuss them. Complex question datasets and other datasets for reading comprehension such as SearchQA [25], QAngaroo [105], and MS MARCO [74] with their details are shown in Table 2.

In order to investigate CQA datasets further[11], we visualize the diversity of datasets according to the semantics of the questions. For this purpose, we first replace entities in the question with a dummy token $<E>$. After creating an embedding for each question using a sentence-transformer model [99], we calculate the cosine similarity between each pair of questions. The distribution of how similar the questions are in each dataset are shown is Fig. 1. Figure 1 shows that datasets with simple questions have more diversity (e.g. WebQuestions and WebQuestionsSP). Datasets that have multiple hops are less diverse and share semantics more (e.g. ComplexWebQuestions). Note that the

[2] https://github.com/JunweiBao/MulCQA/tree/ComplexQuestions

[3] http://qald.aksw.org

[4] https://www.tau-nlp.org/compwebq

[5] https://github.com/yuyuz/MetaQA

[6] https://hotpotqa.github.io/

[7] http://lc-quad.sda.tech/

[8] https://github.com/salesforce/WikiSQL

[9] https://ppasupat.github.io/WikiTableQuestions/

[10] http://nlp.uned.es/clef-qa/

[11] Visualizing datasets with fewer than 100K instances is only considered. For datasets that have more than 100K questions, the diagrams reach normal distribution.

Table 2 Summary of datasets. Columns "%CQ", "LF", and "ResForAns" stand for percentage of complex question in the dataset, logical form, resource for finding answers, respectively

Dataset	Total Size	%CQ	Source Of questions	ResForAns	SPARQL/LF
Jeopardy!	200K+	100%	Human writes	Wikipedia	✗
WebQuestions	5,810	15%	Search engine queries	Freebase	✗
WebQuestionsSP	5,810	15%	Adding SPARQL to WebQuestions	Freebase	✓
ComplexQuestions*	2,100	100%	Using previous complex questions and using query logs from search engine	Freebase	✗
QALD-6	450	100%	Humans annotating	DBpedia	✓
QALD-9	350+	100%	Manually and from previous challenges	DBpedia	✓
ComplexWebQuestions	34,689	100%	Adding constraints to WebQuestionsSP questions	Freebase/Web snippets	✓
MetaQA	400K	71.5%	Using templates on WikiMovies	MovieQA	✗
LC-QuAD 2.0	30,000	100%	Using templates over knowledge graph	DBpedia/Wikidata	✓
WikiSQL	80,654	100%	Creating by templates and then crowd-sourcing on AMT (hand-annotating)	Wikipedia Tables	✓
WikiTQ	22,033	100%	Randomly selecting Wikipedia tables and given to AMT workers to write questions about the tables	Wikipedia Tables	✓
HotpotQA	113K	84%	Crowdsourcing	Wikipedia Hyperlink Graph	✗
TriviaQA	650K	100%	14 trivia and quiz-league websites	Reading Comprehension[a]	✗
SQuAD2.0	150K	100%	Crowdsourcing	Reading Comprehensio	✗
SearchQA	140K	100%	Google search engine	Reading Comprehension	✗
QAngaroo	53K+	100%	WikiHop + MedHop	Reading Comprehension	✗
MS MARCO	1M+	100%	Bing's search logs	Reading Comprehension	✗

[a] In RC datasets the corresponding passage for extracting answers is given

distributions of a single dataset is the same whether it is a train, development, or test set. The semantic diversity primarily affects methods that attempt to improve semantic aspects rather than the syntactic features (complexity constraints) of answering complex questions. To evaluate these methods, datasets with more semantic diversity would be useful.

4 System categorization based on data resource

External resources have a significant contribution in constructing a system for answering users' questions. Many papers have proposed different techniques based on the structures of these resources. Methods using documents or web pages, unstructured resources, as the information resource [51, 93], face different challenges in finding the correct answers based on the constraints in a complex question than methods that employ structured resources [45, 55, 56] such as tables or knowledge graphs. Having a well-structured resource can facilitate finding the exact and correct answer, while they do not have all the information that one can find in the web pages or raw texts. Therefore, some methods have been presented that use both types of resources for finding the answers [91, 92, 106]. It is worth mentioning that there are methods that try to answer complex questions using all the resources they have including raw texts, tabels, and even images [94]. According to this explanation, we classify these approaches into the following three classes: 1) Methods using unstructured data, 2) methods using structured data, and 3) hybrid methods.

4.1 Answering questions from unstructured data

For answering natural language questions, the main resources are raw texts that can be found in documents (textual QA) and web pages (web-based QA) [73]. Many QA systems have been evaluated based on Text Retrieval Conference QA data (TREC-QA[12]) that their questions mostly contain a single fact [51]. Complex questions, on the other hand, need inference, and their answers are not in a single sentence, passage, or document. Due to this complication in answering such questions, various approaches have been introduced for finding related documents to find the relation of different aspects of

[12] https://trec.nist.gov/data/qamain.html

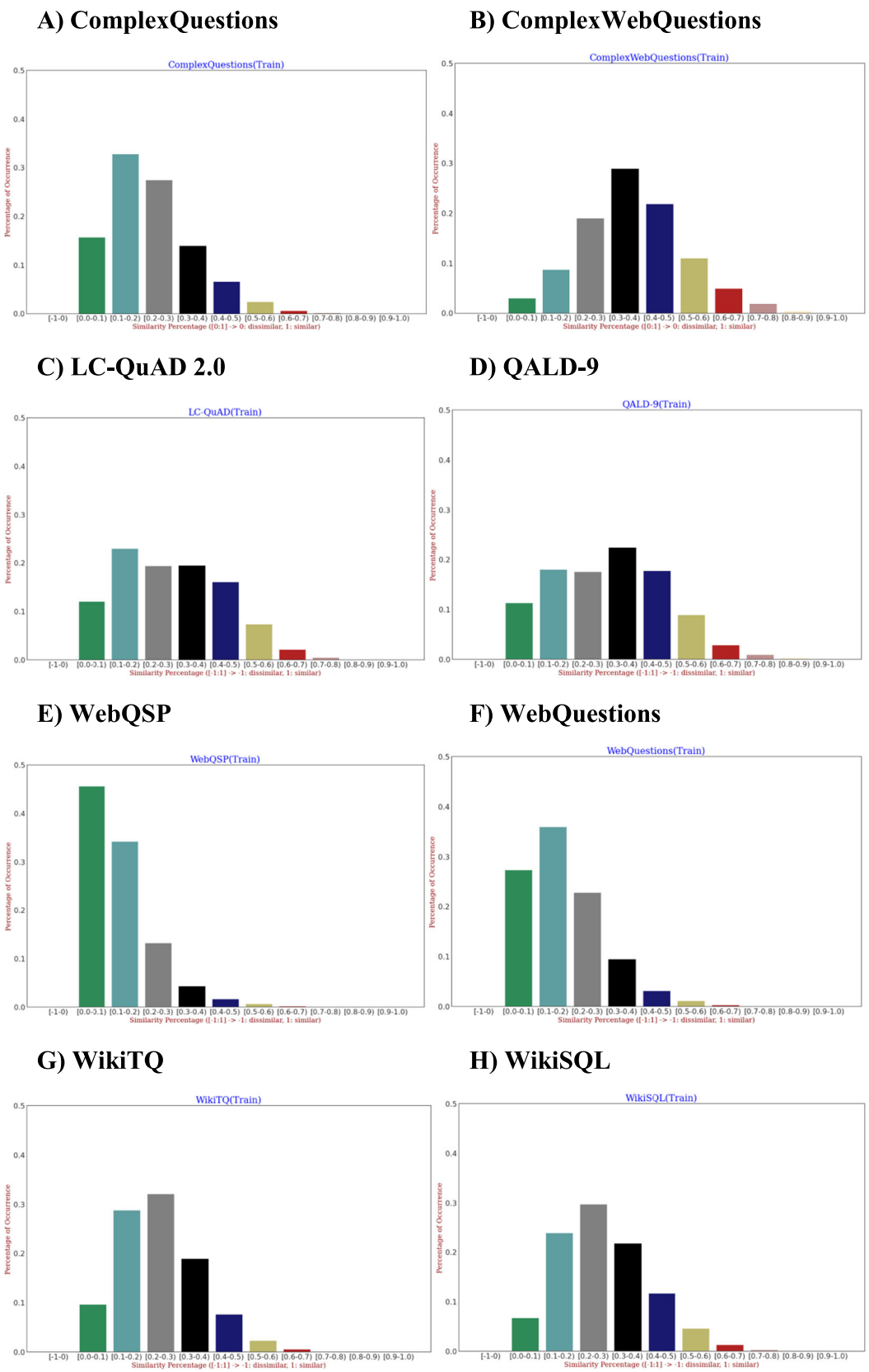


Fig. 1 Each bar shows the percentage of occurence of the similarity (*y* axes). The *x* axes present the similarity boundries (-1 for least similar and 1 for most similar): pink:[-1,0], green:[0.1,0.2), light blue:[0.2,0.3), gray:[0.3,0.4), black:[0.4,0.5), dark blue:[0.5,0.6), beige:[0.6,0.7), red:[0.7,0.8), brown:[0.8,0.9), yellow:[0.9, 1)

Springer

the question that lead to the answer. Different methods have been proposed to answer conplex questions using raw texts using decomposition or reading comprehension techniques. Furthermore, there are methods based on retrieval algorithms that use query expansion to answer complex questions [26].

Decomposition: One of the most widely used methods in answering complex questions via unstructured data is *Decomposition techniques*. These techniques break a complex question into multiple simple questions so that the answers for each of them are in a different single sentence, passage, or document. By using raw text as the resource for finding the answer, three main steps should be taken: 1) Decomposing the complex question into multiple simple questions, 2) retrieving related documents and passages to the simple questions and get the candidate answers, and 3) reasoning over the answers, combining the answers (set operations) or finding the best ones. A complex question can be decomposed through two main approaches, namely: *syntactic* and *semantic* decompositions.

Syntactic decomposition focuses on the syntactic clues of the question that can either result in a unique decomposition or generate different admissible decompositions, while some of them may have no answer [52]. Therefore, it is important to define some patterns and rules to find accurate decomposition.

On the other hand, semantic decomposition uses knowledge templates to find the sub-questions that are answerable by resources and documents. One major drawback of semantic decomposition is that it is restricted to the domain areas in which they have the knowledge templates. The first step in using semantic decomposition is to either have knowledge templates or create them from the raw texts.

In [37, 38] a German QA system is presented in which the documents are analyzed by a syntacto-semantic parser to extract the corresponding semantic network for each sentence. These researchers used German questions in QA@CLEF 2004 and 2005 datasets containing *wh-*, *count-*, and *definition* questions. The question is the first parsed with the same parser that was used for analyzing the documents. Next, the semantic representation of the question is extracted to find the answers by matching the questions' representation with the semantic network of the documents (semantic network matching). Afterward, by giving the semantic representations of the matched document, the question, and the question type, the system uses some rules to generate candidate answers. Finally, the best answer is selected by checking the frequency and the elaboration of the answers. Following the previous works, in [72] a system is introduced for answering *factoid*, *temporal*, and *definition* questions in QA@CLEF 2004 data. In this work, for answering definition questions, they used structural linguistic patterns such as appositions and abbreviation-extension patterns to find the definition parts in the text. A syntactic decomposition technique is utilized for implicit temporal questions to decompose the independent parts into simple sub-questions. Next, the candidate answers to the sub-questions are retrieved by the methods that handle factoid questions. The final answer is selected in an answer fusion step. They use the web (World Wide Web) to get the best answer that has a high total frequency count (TFC) for answer validation. In [39] a system was proposed that covers more types of complex questions in QA@CLEF 2004 till 2008 by defining various semantic decomposition classes based on the semantic network they have built based on the documents [40]. The whole steps of decomposition are based on the semantic networks, not the natural language. According to these researchers, these decomposition methods can be used in other languages as long as there is a parser that can create the same semantic network from raw texts as they did.

In [51], a multiple-fact QA system was proposed. They used lexico-syntactic features to identify different facts in the questions and decompose complex questions into multiple simple questions. Then, they employed IBM Watson to enhance answering the simple questions. In this work the decomposition is classified into two types: *parallel* and *nested*. In *parallel* decomposition, the sub-questions can be answered independently, while in *nested*, the sub-questions are processed in sequence to find the answer. Four key steps are proposed which have different algorithms based on being a parallel or nested question. These steps are as follows: 1) Identifying decomposition parts using syntactic clues, 2) adding information context to the sub-questions as some of the sub-questions may not carry the information needed to limit their answer, 3) retrieving a list of answers to the sub-questions with a confidence score from the QA system, and 4) using a candidate re-ranker that combines the answers based on the question being either parallel or nested. This recomposition strategy employs a heuristic approach for nested questions and machine learning techniques for parallel questions to get the final ranking by using candidate sub-answers' confident scores.

Today, deep learning and end-to-end systems have been the subject of intense research and various efforts have been made to build a system with this architecture. In [93], an end-to-end system with a sequence-to-sequence model was prepared. This system gets a question and produces its corresponding *computation tree* by semantic decomposition. The leaves of the computation tree are strings resembling the sub-questions and its nodes are functions for combining answers of its corresponding

children to reach the correct answer. For answering the sub-questions a reading comprehension technique is utilized to extract the answer from a list of web snippets retrieved from search engines. They also define some functions that combine the answers with proper logic to reach the correct final answer. [93] also address some challenges in answering questions with operational constraints and negation questions by raw texts involve some problems. For example, obtaining all entities with their corresponding values from the raw text for performing operational constraints on them can be a time-consuming task. Furthermore, to negate questions such as *What countries are not in the OECD?*, an open set of entities emerges that cannot be analyzed through set subtraction.

In [97], an effective and interpretable Select, Answer and Explain (SAE) system is proposed to solve the multi-document reading comprehension problem. The question and N documents are given. Based on the relevance score of the given documents, the BERT [18] model selects the top k ranked document. All selected documents are concatenated into one context input. Then, the question and the context are forwarded into another BERT+Multi-Perceptron model to find the start and end points of the answer span. Finally, to find the supporting sentence, a GNN model is used over sentence embeddings. Embeddings are the nodes of the graph, and the edges are constructed based on the named entities and noun phrases presented in the question and sentences. A multi-relational Graph Convolution Networks (GCN) based message passing strategy is employed to update the graph node features, and the final node features are input to an MLP to get the classification logit of each sentence. Supporting sentence is one that score 1 in the classification output.

As can be inferred, the major difference between the algorithms presented for answering complex questions from raw texts is the decomposition stage, which shows how they handle the constraints. After getting the simple sub-questions, existing QA systems, information retrieval techniques, or reading comprehension methods can be employed to obtain the answers. Eventually, the last step is recomposing the answers by defining various techniques to obtain the final correct answer.

Reading comprehension: Other methods are presented for reading comprehension tasks that try to answer complex questions without breaking them into simple ones and they use given raw text as the outer resource. They use deep models to infer and find the answer span of the given context.

In [86], a reading comprehension model, BIDAF, is proposed with the help of an attention mechanism. The question (query) and the corresponding paragraph (context) are passed through six layers. The first three layers compute features from the query and the context at different levels of granularity. In this process, the attention layer, the fourth layer, is responsible for linking and fusing information from the paragraph context and words which are in the question. The fifth layer captures the interaction among the context words conditioned on the query. In the final layer, the output layer finds a sub-phrase of the paragraph to answer the query. It is worth mentioning that the structure of BIDAF can be adapted to different tasks by modifying the last layer according to the task.

In [116], QANet is proposed, which has similarities with the BIDAF model. QANet uses convolutional neural network (CNN) and attention mechanisms. The use of CNN makes the process faster. Therefore, this allows the model to be trained with more data. The extra data is generated by back-translation from a neural machine translation model.

4.2 Answering questions from structured data

Some primary problems with the algorithms using unstructured data are that they can not necessarily find the exact answer, and they need additional efforts to extract the exact answers (reader) from the retrieved raw text (retriever). Besides, dealing with some operational constraints may need laborious works. To mitigate these weaknesses, several studies extract all information (that is in the raw text of documents and web pages) to create well-structured data or use existing knowledge bases. There are studies which have created large structured knowledge bases or knowledge graphs such as Freebase [11], DBpedia [5], YAGO [90], NELL [12], and Vault [23]. KB-QA refers to the studies on using knowledge bases to answer questions. Some other QA systems use *tables (tabular data)* as a semi-structured resource for answering questions.

There are two major types of KB-QA: *curated (or closed KB) systems* and *open KB systems*. The difference between these systems is in the type of knowledge bases they use. The first type of system utilizes curated KBs like Freebase, DBpedia, and YAGO in which information and knowledge are encoded as entities and relations with unique IDs called RDF[13] triples. They are also known as semantic webs or RDF knowledge bases. These knowledge bases have a predefined schema and are sometimes edited (not necessarily created) manually making them accurate and easy to use. However, there are some drawbacks when using these KBs such as [115]:

1) Query transformation (the question should be transformed into a structured query such as SPARQL)
2) Incompleteness

[13] Resource Description Framework

The second type of system employs open information extraction (Open IE) techniques to build semi-structured data that are stored in the form of n-tuples assertion (n $\geq$ 3). It is worth mentioning that these facts are noisy and the same entity or relation can be found with a different form of noun phrases in more than one RDF triple; e.g., *<Michael Jeffrey Jordan, place-of-birth, U.S.>* and *<Jordan, born-in, United States>*. Due to these faults, it is not sure whether we obtain all the facts about a specific entity by executing a query over the open KBs [33]. One advantage of open KBs is that if some pieces of knowledge do not exist in the KB they can be retrieved from the raw text in documents or web-pages on the fly. Different KBs are presented in Table 3. There are more details about the number of relations, entities, and RDF triples for curated knowledge bases in [31].

The third type of system use tables that are closed to QA systems using databases or KGs. The main characteristic of tabular data is that they are not normalized, which leads to schema mismatching error.

Regarding the presence of three different types of structured data, different approaches are introduced for finding answers to complex questions. These approaches are discussed in the following sections.

4.2.1 Open KB-QA Systems

There are QA systems that work with open KBs [29, 30] but they are not capable of answering questions with complex semantic components. Therefore, new approaches with open knowledge bases are introduced to handle complex semantic constraints that appear in the questions.

In [115], a new open knowledge base called *nOKB* was introduced by mentioning that complex questions have complex semantic contents and they may need n-tuple assertions of multiple arguments. This base has an n-tuple assertion structure. Their main algorithm of this system includes four steps: 1) Question paraphrasing: In this step, the informal question is converted to a formalized one. This formalization is for having a question with a similar syntactic and vocabulary structure with open KB tuples. The proposed new paraphrasing templates cover shortcomings of [30] for paraphrasing questions with complex semantic contents. 2) Question parsing: It parses the paraphrased questions and creates multiple tuple queries ($<sbj, rel, Args = arg_1, arg_2, ..., arg_{n-2}>$) that one of its fields, entities or relations in the tuple, is unidentified as it is the unknown answer of the question. 3) Querying over open KB: This step gets the tuple queries and executes them against the existing open KB and retrieves a set of candidate answers. 4) Answer Raking, which collects all the answers that were retrieved in the previous stage. Each answer has many features as it is derived along the paraphrasing-parsing-querying process. If an answer appears in more than one query, then the best features among them are selected. Finally, there is a linear model that gets these features along with the question to predict the best answer among them. The first stage of this algorithm is its most important part because poor paraphrasing results in getting wrong tuples in the second step. The workflow of their system (TAQA) is exemplified in Fig. 2.

Table 3 Different samples of knowledge bases

Knowledge base	Type
Freebase	Curated KB
DBpedia	Curated KB
YAGO	Curated KB
Vault	Curated KB
Wikidata	Curated KB
ProBase [104]	Curated KB
ReVerb [28]	Open KB
NELL	Open KB
OPIEC [35]	Open KB

In [53], an algorithm was proposed to answer multiple-choice complex questions of 4th- and 8th-grade science exam datasets. The algorithm has the following steps: 1) Finding tuples that are related to the complex question by *tf-idf* score; here, the question is assumed as a query and tuples as a document. Also, tuples that do not support any answer choices of the question in one of their fields are removed. In this stage, the top 50 tuples are selected. 2) Connecting these tuples to create graphs with nodes, as entities, and edges, as relations. Figure 3 presents a graph that connects the terms in the question with the answer choice. 3) Finding the best graph that leads to the correct answer. For this step, an objective function is used to score the best matches.

In [55], the researchers come up with an approach that uses the random walk technique over open KB to answer multi-choice complex questions in the 6th- to 9th-grade science exam. The random walk employed by these researchers starts with the existing nodes in question. Next, it is followed by walking through the relations and nodes to find the answer nodes while scoring them through the walk that it took to reach them. Node importance, edge probability, and teleportation probability are measured through supervised and unsupervised methods. In this way, it is possible to guide the random walk to the right answer. Figure 4 presents where the algorithm starts (S nodes) and how it reaches the answer choices (W is the wrong answer and R is the correct one) by the existing RDF triples.

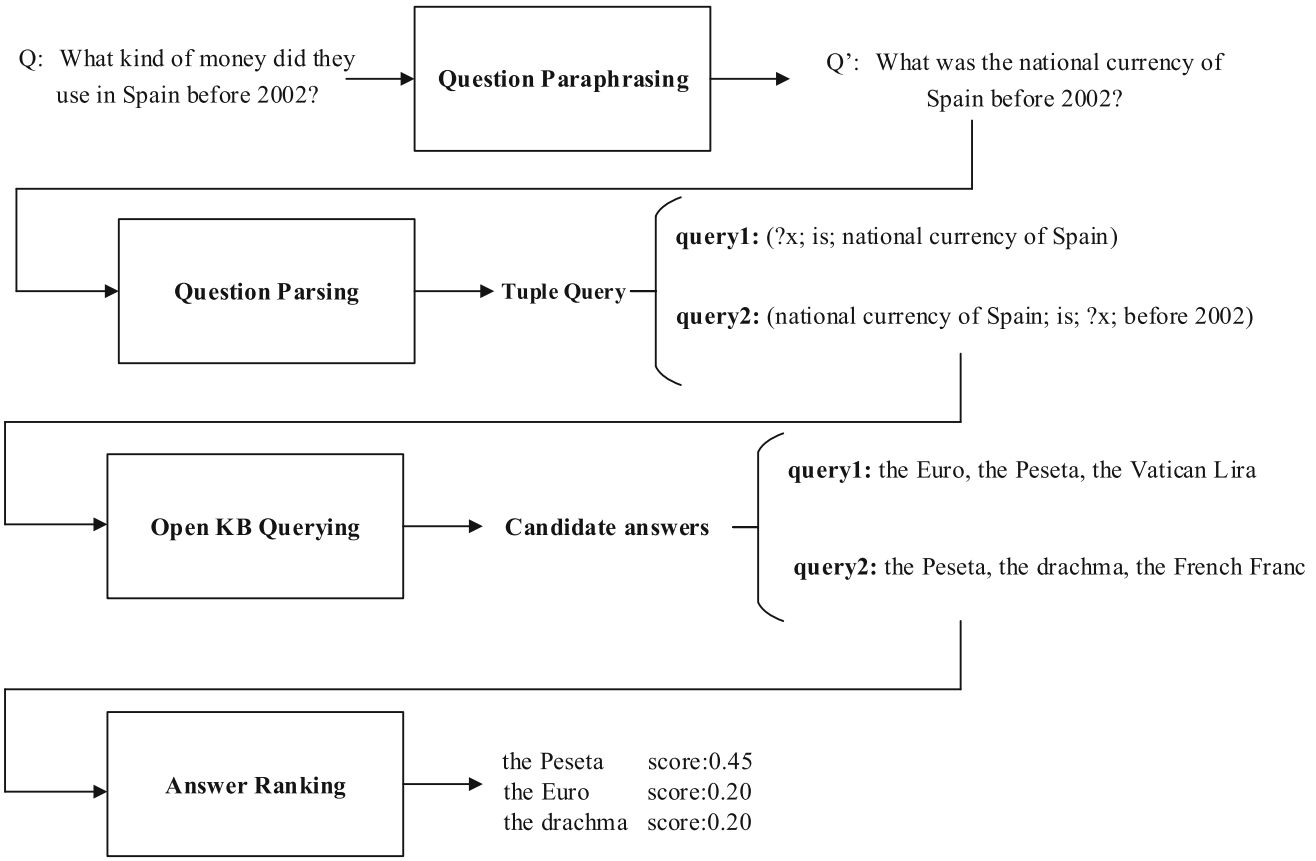


Fig. 2 TAQA's workflow for answering the question *What kind of money did they use in Spain before 2002?* [115]

4.2.2 Curated KB Systems

For decades, one of the most popular resources used in open-domain QA systems has been curated knowledge bases. Although many approaches have been proposed for answering simple questions based on curated KBs, answering complex questions is still far from a closed problem. A question that has multiple entities, relations, or complex semantics is considered complex. The methods introduced for this purpose can be categorized into three main classes: 1) Semantic parsing based, 2) Embedding based, and 3) Information retrieval-based. In this section, we discuss the algorithms of the first two classes that are proposed to handle complex questions. the third class, which is discussed in Subsection 4.3, contains hybrid methods that use both structured and unstructured resources.

Semantic parsing-based approaches: Semantic parsing approaches try to capture different semantic components and build the corresponding logical form of the question, like λ-DCS [61], query graph, or executable queries such as SPARQL. The main goal of these approaches is to reach a gold logical form that best describes all aspects of the question. Therefore, all semantic components and constraints should be in the created logical form. This feature causes retrieving exact and correct answers from the knowledge base by converting the question's logical form into executable queries like SPARQL. As described earlier, decomposition techniques involve more steps to get the final answer, such as retrieving the answer from text and recomposing sub-answers. Nevertheless, semantic parsing approaches combine all this information in a logical form, which allows it to be

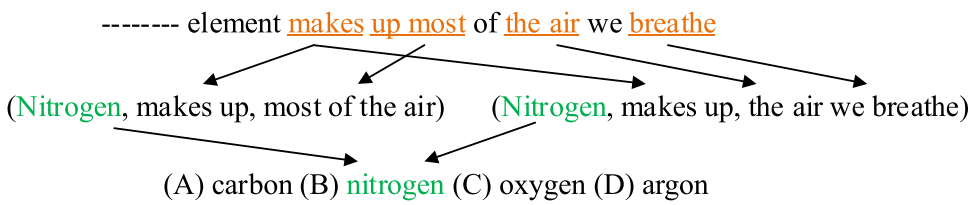


Fig. 3 An example that connects the right answer to the given question by the tuples of the using knowledge base [53]

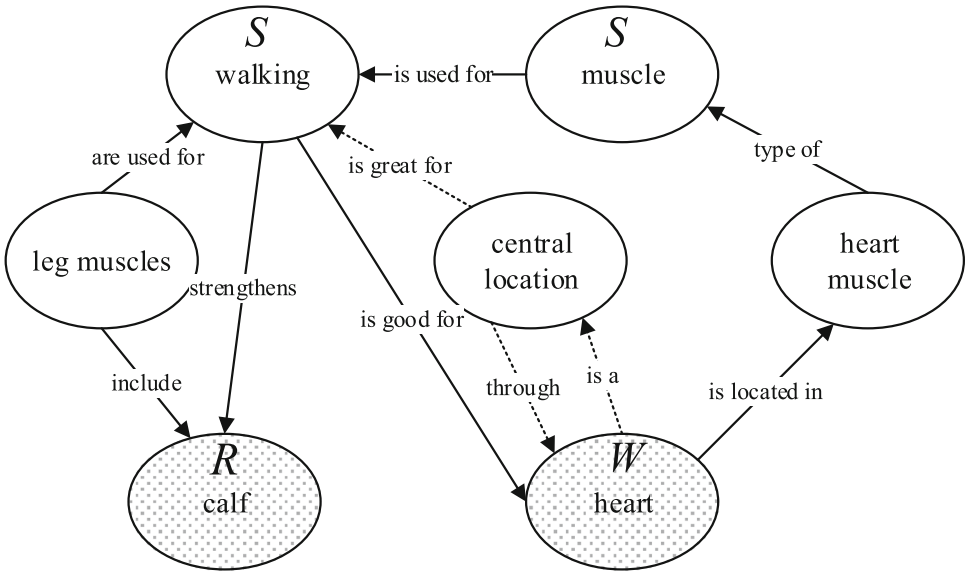


Fig. 4 A random walk for the question: *Which muscle is used for walking? a) heart b) calf* [55]

considered at the same time in the retrieval stage. In this regard, some approaches need extra work for matching the created semantic representation to corresponding entities and relations in the knowledge base that they are going to use [9, 54, 80]. On the other hand, some other approaches proposed a semantic parser that includes the knowledge base when building the parse for the question [6, 112] and does not need matching.

In [113], the researchers introduced a semantic parsing framework that creates questions' query graphs that can be mapped to logical forms in λ-calculus. The process of semantic parsing in this work is a query graph generation that can be formed as a search problem with stage states and actions. The state is a query graph and action is the way to expand the graph. The state diagram of this approach is presented in Fig. 5. It is of note that the nodes and edges of the graph are the exact entities and relations in the KB and there is no need for graph matching. This can be achieved by using a convolutional neural network (CNN) framework [34, 87, 88] for finding a core inferential chain.[14] For matching named entities in questions to entities in KGs entity linking tools [108] are used as well. Some actions have been defined to attach the constraints of a complex question to the query graph. Since staged query graph generation (STAGG), produces multiple candidate query graphs, there is a reward function that assigns a score to these candidates and then the best one is executed against the KB to retrieve the answer. In query graph generation approaches, constraints are injected within the query graph, especially for the operational constraint that can easily be decoded to logic forms such as SPARQL.

[14]The chain of relations that connects the topic entity, the root of the query graph, to the answer node

In [7], a semantic parsing system was offered that translates complex multi-constraint questions (*MulCQ*) to a multi-constraint query graph (*MulCG*) in 4 main steps: 1) Basic query graph generation: Each entity in the question is treated as a topic entity and its path to the answer is extracted by using a CNN model similar to the previous algorithm. 2) Constraint detection and binding: There is a rule-based constrain detection method for finding the constraints in a question and a rule-based constraint binding method for inserting the constraints into the basic query graph that is generated. 3) Search space generation: It provides all the premutation of the constraints that can be bound to different basic query graphs. 4) Features and rankings. Due to having multiple ways to assemble the query graph and find different basic query graph as a core chain, a linear scoring function is used to rank these candidate graphs. Eventually, the best query graph is executed against the KB.

In [67], a new semantic parsing system was introduced for generating query graphs. This system is more efficient and captures more types of constraints than the one proposed in [7] and provides a method that encodes both the complex query graph and the question into a vector representation. The cosine similarity between these two vectors is an extra strong feature for finding the best representative query graph among the generated ones.

In [45], a semantic parsing method was proposed based on state transition to translate the complex question to a semantic query graph. The first step of this method is node recognition that uses BLSTM-CRF model (for literal and variable nodes) [47] in addition to using the entity-linking tool S-MART (for entity and type nodes). There are four operations (i.e. connect, merge, expand, and fold) that construct the query graph by using a linear method

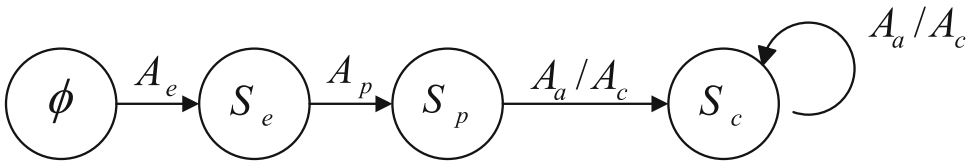


Fig. 5 ϕ, S_e, S_p, and S_c are the states that shows empty graph, graph with single node, graph with *core inferential chain*, and complex graph with additional constraints, respectivly. A_e, A_p, A_a, and A_c are the actions for picking an entity node, determine the core inferential chain, adding aggregation node, and adding constraints, respectivly [113]

to score the state while transiting to reach the best state. Then, the final state, with the highest score among other states, is applied to retrieve the answer by executing the KB. This approach can handle some challenges such as co-reference (with merge operation), nodes with hidden information (with expand operation), and deleting useless nodes (with fold operation). The semantic query generation for the question *"Which cosmonauts died in the same place they were born in?"* is illustrated in Fig. 6.

In [117], an end-to-end hierarchical semantic parsing was introduced based on a sequence-to-sequence model. This method decomposes the complex question into a sequence of sub-questions and processes them to produce the logical forms. This work focuses on creating semantic parsing by paying attention to question decomposition. These researchers compared their work with other papers that focus on creating the semantic parsing of utterances (not only questions) [21, 22].

In [79], a system was proposed that decomposes the complex question via a divide-and-conquer technique based on the entities in the question. This system works after obtaining the question representation with some learned templates from the knowledge base. Next, it extracts the answers to each sub-question from the knowledge base separately. For semantic matching patterns and finding the correlation from a sequence of answers, they introduced a deep learning network called Template representation based Convolutional Recurrent Neural Network (T-CRNN).

In [89], a method was introduced based on syntactic decomposition. It builds the sub-query graph of each sub-question and combines the sub-query graphs with different combinations to get the complete query graph candidates. For finding the correct answer, a ranking model is used to rank the best query graph.

In [56], motivated by the works [7, 67, 113], a modified staged query graph generation was prepared using reinforcement learning. In this work, a beam search was employed to generate the candidate query graphs iteratively. The researchers considered more than two relations in the core chain rather than considering only up to two hops in previous works. Having longer hops to consider means making the search space larger. To reduce the search space, they added the constraint iteratively and did not wait to obtain the core relation path first. Allowing the constraints to be considered before going further into the graph would narrow the search space as we need to search only the relations related to the constraints. There are three main actions to grow the query graph: *connect*, *extend*, and *aggregate*. An example of these actions is presented in Fig. 7.

In [46], a Meta-RL approach was proposed that converts a complex question into a sequence of actions. It has two main parts: a programmer and an interpreter. The programmer is a Seq2Seq model whose input and output are question sequence and action sequence, respectively. The interpreter executes the sequence of actions to produce answers.

In [77], an approach was proposed which produces a query graph. The entire knowledge graph is used in the paper as a starting point, and it is gradually shrunk to find the desired query graph. The shrinking has three stages: 1) Relation Subgraph Extractor that captures only those relations in a graph that are in the given question with a classifier. 2) Query Graph Generator that gets a set

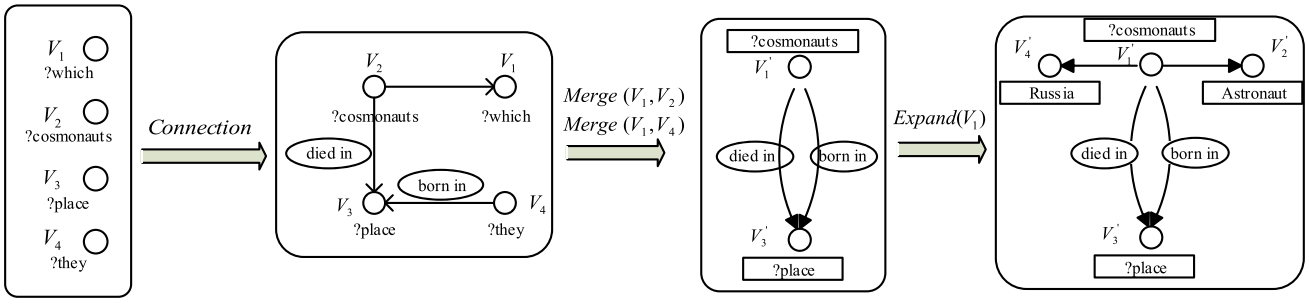


Fig. 6 The flow of the algorithm to find the semantic query graph [45]

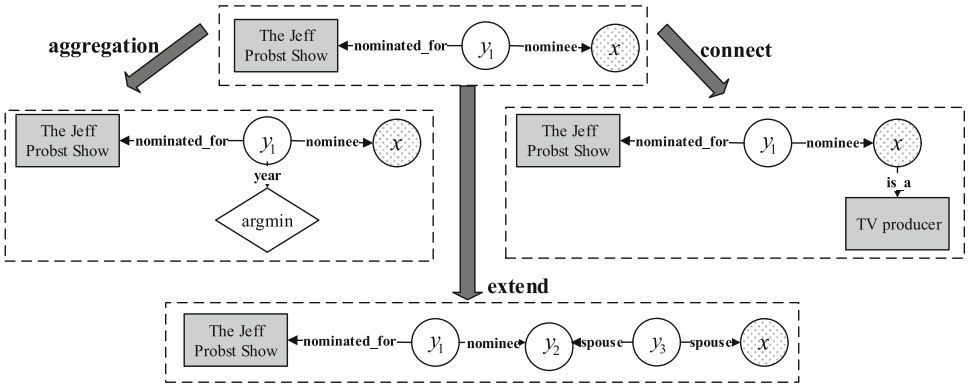


Fig. 7 The example of *connect*, *extend*, and *aggregate* for the question *"Who is the first wife of the TV producer that was nominated for the Jeff Probst Show?"* [56]

of candidate query graphs by narrowing down the search space. Such candidates should lead to answers with high F1 scores using a logistic regression model. 3) Query Graph Ranker that ranks the candidate query graphs using Albert to compute the matching score between the given candidate query graph and the question. In the end, the candidate with the highest score is mapped to the SPARQL query.

In [15], an approach was proposed to generate query graph with respect to the query structure. In contrast to the others, this model reduces the amount of noisy generated queries. There are two stages. In the first stage, an encoder-decoder is used to predict the argument of predetermined operation in each generative step. The second stage ranks the candidate query graphs.

In [50], an end-to-end method was proposed to answer temporal complex questions. This method has two stages. The first step computes question-relevant compact subgraphs within the KG, and judiciously enhances them with pertinent temporal facts, using Group Steiner Trees and fine-tuned BERT models. The second step constructs relational graph convolutional networks (R-GCNs) from the first step's output, and enhances the R-GCNs with time-aware entity embeddings and attention over temporal relations.

Embedding-based approaches: Embedding-based methods try to learn the continuous vector representations of the question and knowledge base triples that the inference is base on these embeddings. One of the primary advantages of embedding based approaches is working effectively toward complex reasoning. As mentioned in Section 4.2, knowledge bases suffer from incompleteness that some pieces of information or relations may not be in them. Therefore, the inference should be based on existing information. Some approaches that utilize knowledge graph embedding can be more efficient in handling this weakness and handling complex reasoning [85]. It is worth mentioning that in recent years some methods only focus on answering multi-hop questions (without any constraint) via Graph Neural Network (GNN) [44, 100], reinforcement learning [16, 63], and memory network [48, 57].

In [49], an embedding-based approach is introduced based on adopting a memory network. Question and its corresponding KB facts are represented as a vector using word embeddings that are fed into an L layer memory network as the inputs and the output of the network provides the score of related facts. These researchers used a set of rules for handling some constraints such as temporal and superlative qualifiers.

In [14], a bidirectional attention memory network (BAMnet) was proposed. This network has four modules: 1) Input module, which gets the question and encodes it with bidirectional LSTM network, 2) Memory module, which generates the candidate answers according to the topic entity in the question, 3) Reasoning module, which is two-layered bidirectional attention to captures the interaction between questions and KB. The idea of this network comes from similar methods used in reading comprehension; and 4) Answer module, which finds the best answer by computing the score between question and candidate answer representation.

In [85], the researcher proposed an EmbedKGQA system that employs knowledge graph embeddings for answering multi-hop questions. It consists of three main modules: 1) KG embedding module, which builds the embeddings of all the entities in the knowledge graph with the help of ComplEx embeddings [96], 2) Question embedding module, which finds the embedding of the given question using RoBERTa [66], and 3) Answer selection module, which lowers the size of the set of candidate answers and finds the final answer by learning a scoring function. The flow of this algorithm is in Fig. 8.

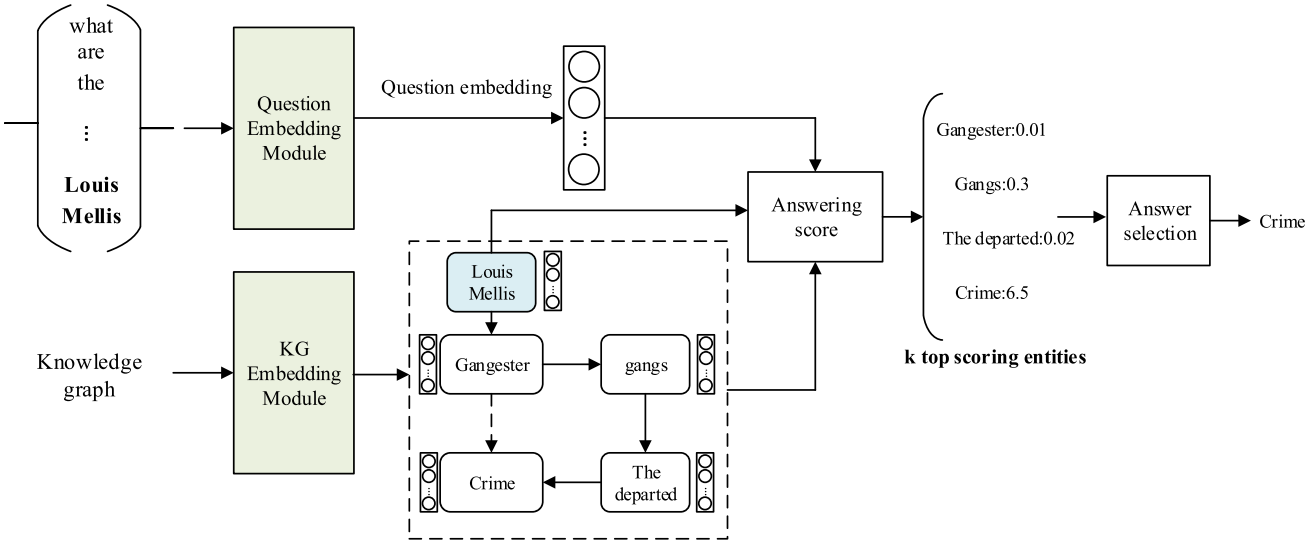


Fig. 8 Example of finding the answer for question *What are the genres of movies written by Louis Mellis?* [85]

4.2.3 QA Systems using tabular data

Tables are among the popular and simple structures for many purposes but not always efficient for saving data. Some papers have proposed a semantic parsing technique to build the logical forms of questions or executable queries [41, 68, 69]. Also, some other studies have introduced weakly supervised techniques to find the answers [2, 70, 102] or use end-to-end models [98].

In [41], a weakly supervised system was built based on BERT to predict the denotation by selecting table cells and apply aggregation optionally. There are two classification layers for table cells selection and aggregation operations on selected cells.

4.3 Hybrid methods

In this section, we discuss the approaches that use both KBs and documents to retrieve the answers and deal with constraints within complex questions. As described in the previous lines, getting the semantic representation of a sentence can be more feasible with structured data. However, building a structured knowledge base is expensive and it may not have all the information. More importantly, building the correct semantic representation of a complex question is a laborious task.

In [106], a method was presented that uses Freebase and Wikipedia to answer the questions. This method is performed in two steps: 1) Extracting candidate answers from the knowledge base using entity linking and relation extraction (using deep learning, MCCNN model) and 2) Further inference with Wikipedia to eliminate the wrong answers and find the correct ones. For constraint handling, this method employs a syntactic decomposition to create sub-questions for complex questions with multiple relations and entities. Furthermore, these researchers tried to find the answers for the operational constraints in the raw text instead of dealing with them mathematically.

In [92], the PullNet network was proposed to produce a *question subgraph* to answer a question. The subgraph is constructed iteratively using both corpus and knowledge base. A question's subgraph is a heterogeneous graph that contains a set of vertices and edges. The vertices are entity nodes, text nodes, and fact nodes. The text node is a single sentence that contains the mentioned entity. To build such subgraphs iteratively, they proposed two main operations: *pull* operations and *classify* operations. *Pull* operations either retrieve information from KB or corpus. *Classify* operations are applied to the nodes for finding the probability of nodes that need to be extended in the next iteration. After the t-th iteration of expansion, a classify operation is used to find the answer node. This network is built on GRAFT-Net [91].

5 Evaluation metrics

There are different metrics for evaluating a QA system. We explain 9 evaluation metrics. The formula is based on true positive (TP), true negative (TN), false positive (FP), and false negative (FN) examples. Every metric has its own strength and weaknesses [13]. We classify evaluation metrics into two categories: 1) Answer-based evaluation. 2) Generation-based evaluation.

5.1 Answer-based evaluation

Common metrics that are presented in this section mostly used when the output of the models are the answer entities.

Accuracy: This metric tells that how many questions are answered correctly (considering retrieving all the answers of a given question).

$$accu = \frac{Number\ of\ questions\ answered\ correctly}{Total\ number\ of\ questions} \tag{1}$$

Precision: Precision (also called positive predictive value) is the fraction of correct answers among the total retrieved answers.

$$precision = \frac{TP}{TP + FP} \tag{2}$$

Recall: Recall is the fraction of correct answers that were retrieved among the total actual answers.

$$recall = \frac{TP}{TP + FN} \tag{3}$$

F1: F1 is a function of precision and recall that includes both metrics effects.

$$f_\beta = (1 + \beta^2) \times \frac{recall \times precision}{(\beta^2 \times precision) + recall} \tag{4}$$

Hit@1: The percentage of questions that the model's first predicted answer is a correct answer (i.e., have a rank of 1).

MRR: MRR is the Mean Reciprocal Rank that takes the position of an answer (the "rank") into account. MRR is used mostly for document retrieval in QA.

$$f1 = \frac{1}{|Q|} \Sigma_{i=1}^{|Q|} \frac{1}{rank_i} \tag{5}$$

Where Q is the number of questions.

5.2 Generation-based evaluation

Common metrics that are presented in this section are mostly used when the output of the models are a generated text such as SPARQL query or answer text.

Exact Match (EM): The exact match ratio is a very strict measure of the model performance. It increases only when the model correctly identifies answers that are identical to the correct answers (this metric is called a Reader-Metric).

$$EM\ ratio = \frac{Number\ of\ qustions\ with\ exact\ match}{Total\ number\ of\ questions} \tag{6}$$

BLEU: This is a metric for evaluating the quality of the generated text based on n-gram matches which are position-independent [75]. In complex question asnwering, this metric is used to evaluate the generated SPARQL queries.

$$p_n = \frac{\sum_{C \in candidates} \sum_{n-gram \in C} Count_{clip}(n\text{–}gram)}{\sum_{C' \in candidates} \sum_{n-gram' \in C'} Count(n\text{–}gram')}$$
$$BP = \begin{cases} 1 & \text{if } c > r \\ e^{1-r/c} & \text{if } c \leq r \end{cases}$$
$$BLEU = BP \times \exp\left(\sum_{n=1}^{N} w_n \log p_n \right) \tag{7}$$

Where $candidates$ are the generated texts or SPARQL queries. The function $Count$ compute the number of n-grams in the candidate that matche the n-gram in the target or reference sentence(s). On the other hand, the function $Count_{clip}$ is *min(Count, Max_ref_Count)*, Max_ref_Count is the largest count observed in any single reference for that n-gram. c is the length of the candidate translation and r be the effective reference length.

ROUGE: It stands for Recall-Oriented Understudy for Gisting Evaluation [62]. ROUGE-N is an n-gram recall between a candidate text and a set of reference text. It is essentially a set of metrics for evaluating automatic summarization of texts as well as machine translations. In complex question answering, this metric is used to evaluate the generated SPARQL queries.

$$ROUGE - N = \frac{\sum_{S \in ReferenceText} \sum_{gram_n \in S} Count_{match}(gram_n)}{\sum_{S \in ReferenceText} \sum_{gram_n \in S} Count(gram_n)} \tag{8}$$

Where n stands for the length of the n-gram, $gram_n$, and $Count_{match}(gram_n)$ is the maximum number of n-grams co-occurring in a candidate text and a set of reference texts.

BERTScore: Given a reference sentence $x =< x_1, ..., x_k >$ and a candidate sentence $\hat{x} =< \hat{x_1}, ..., \hat{x_l} >$, contextual embeddings is used to represent the tokens, and compute matching using cosine similarity, optionally weighted with inverse document frequency scores [118].

6 Performance and results

In this section, we discuss the performance of some of the systems explained in Sections 4.1 and 4.2 on the popular benchmark datasets. In addition to Table 4 which shows the performance on Meta-QA, the performance of other systems and datasets are shown in Table 5. Results that are shown in Table 5 are mostly based on the F1 score. However, other metrics are presented to give the readers

Table 4 Results on MetaQA with hits@1 metric. The evaluation of the [92] was without using text data [85]

Paper	MetaQA KG-Full			MetaQA KG-50%		
	1-hop	2-hop	3-hop	1-hop	2-hop	3-hop
[85]	97.0	99.9	91.4	65.1	52.1	59.7
[92]	97.5	98.8	94.8	83.9	91.8	70.3

a sense of magnitude about the best results obtained by each metric. Among the datasets discussed in Section 3, only those that are used most commonly in CQA are illustrated to observe the rate of performance improvement by different methods. As can be seen from the obtained results, answering complex questions is a task far from a solved problem. Key findings extracted from reviewing the results in Table 5 are:

1) An important aspect in evaluating systems is the dataset that is used for both training and testing. As it was shown in Fig. 1, datasets have different diversity. As it is expected less diverse datasets can result in more growth in the value of evaluation metric once a new state-of-the-art algorithm is presented. In Table 5, the growth which is made by different models on commonly used datasets can be seen. Changes in ComplexQuestions is less than ComplexWebQuestions (based on Fig. 1 ComplexQuestions in more diverse that ComplexWebQuestions). Therefore, same system can change the evaluation metric larger for less diverse datasets with simple changes. Hence, it is suggested that for better evaluation different datasets should be used.
2) By comparing the results in [115] and [49], it is inferred that using curated KB has better performance in the field of answering complex questions. However, as discussed earlier, curated KBs suffer from incompleteness and most systems using these resources cannot answer questions that do not have their information in the corresponding KB. Nevertheless, some approaches like [85] can cover this shortcoming. The results on the

Table 5 Results of different approaches on the benchmark datasets. All these results have come from the exact result that was declared in the corresponding paper

Dataset	Paper	Resource type	Metric + Result
WebQuestions	[113]	curated KB	F1 : 52.5%
	[115]	open KB	accuracy : 45.8%
	[7]	curated KB	F1 : 54.36%
	[106]	KB + text	F1 : 53.3%
	[49]	curated KB	accuracy : 55.6%
	[45]	curated KB	F1 : 53.6%
	[67]	curated KB	F1 : 52.7%
	[14]	curated KB	F1 : 51.8%
	[89]	curated KB	F1 : 52.72%
	[15]	curated KB	F1 : 53.4%
ComplexQuestions*	[7]	curated KB	F1 : 42.33%
	[67]	curated KB	F1 : 42.84%
	[93]	text	F1 : 40.9%
	[89]	curated KB	F1 : 43.05%
	[56]	curated KB	F1 : 43.3%
	[15]	curated KB	F1 : 43.1%
ComplexWebQuestions	Human	–	percision@1 : 63.0%
	[93]	text	percision@1 : 27.5%
	[117]	–	BLEU : 66.18%
	[92]	KB / text	Hit@1 : 45.9% / 13.8%
	[56]	curated KB	percision@1 : 44.1% / F1 : 40.4%
	[77]	curated KB	F1 : 46.2%

MetaQA dataset by removing 50% of the KG are given in Table 4.

3) It is of note that a system can work well on answering complex questions while its performance on answering simple questions may not be as good as that of the complex ones. For example, the method proposed in [89] does not provide the highest F-measure on WebQuestions, which also contains simple questions.

7 Future direction

In light of recent studies and the challenges they encounter, we discuss several future research topics that can assist in improving complex question answering systems.

7.1 Datasets

There are many datasets for complex question answering. However, they have different features which make them reliant on the resource or algorithm being used. We discussed previously that different resources present different challenges. In spite of this, it is suggested that a baseline dataset should be developed that can be used for all existing methods. Thus, algorithms can be better analyzed in terms of reasoning and inference.

Many papers have found errors in datasets related to dataset faults such as incorrect paraphrasing (e.g. ComplexWebQuestions), no correct answers (e.g. ComplexWebQuestions and WikiQA), and ambiguous questions. It is suggensted to correct these faults.

7.2 Sub-Tasks improvement

Most methods presented have different steps to find the answer. Improvements can be made to each step independently, resulting in better overall performance. For instance, finding the answers using knowledge graphs can present challenges, such as graph matching, entity linking, and relation extraction. Recent methods eliminate graph matching by using knowledge graphs directly in their algorithms. Entity linking and relation extraction are two main tasks that are used in KBQA systems. Based on the error analysis presented in the papers, these two tasks cause the main errors. Therefore, having a robust EL and RE can improve the performance without changing the whole algorithm. In [14], it was shown that the performance of their system is improved from 51.8% to 55.7% only by assuming that the gold topic entity is known. There are also other examples such as improving question or contex representation, improving complexity type recognizer.

8 Conclusion

In this study, a survey is presented about complex open-domain question answering systems. We talk about what a complex question is and the challenges it required to be answered, followed by categorizing the constraints into 5 classes. Approaches utilized in these systems were discussed and categorized into three classes based on the type of resources they utilize to find the answer: 1) Methods using unstructured data, 2) Methods using structured data, and 3) hybrid methods. The results showed that algorithms have different ideas with respect to the resource type. Finally, the performance of some of the approaches tested on the benchmark datasets was discussed and the findings were presented. Challenges for handling complexities are still unsolved. Choosing the best query graph among candidate query graphs in KBQA systems is a part that still needs to be worked on. Creating large question-answer datasets with different complexities can affect deep learning approaches for both decomposing question into simple ones when using unstructured resources or creating logical forms when there is a structured resource.

References

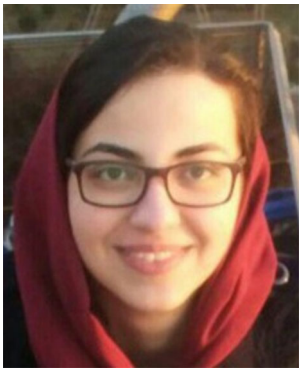
1. Abujabal A, Yahya M, Riedewald M, Weikum G (2017) Automated template generation for question answering over knowledge graphs. In: WWW '17 Proceedings of the 26th international conference on world wide web, pp 1191–1200
2. Agarwal R, Liang C, Schuurmans D, Norouzi M (2019) Learning to generalize from sparse and underspecified rewards. In: International conference on machine learning, pp 130–140
3. Alkholy EMN, Haggag MH, Aboutabl A (2018) Question answering systems: Analysis and survey. International Journal of Computer Science & Engineering Survey 9(6):1–13
4. Allam AMN, Haggag MH (2012) The question answering systems: A survey. International Journal of Research and Reviews in Information Sciences (IJRRIS) 2(3)
5. Auer S, Bizer C, Kobilarov G, Lehmann J, Cyganiak R, Ives Z (2007) Dbpedia: a nucleus for a web of open data. International Semantic Web Conference 4825:722–735
6. Bao J, Duan N, Zhou M, Zhao T (2014) Knowledge-based question answering as machine translation. In: Proceedings of the 52nd annual meeting of the association for computational linguistics (Volume 1: Long Papers), vol 1, pp 967–976
7. Bao JW, Duan N, Yan Z, Zhou M, Zhao T (2016) Constraint-based question answering with knowledge graph. In: Proceedings of COLING 2016, the 26th international conference on computational linguistics: technical papers, pp 2503–2514
8. Benamara F (2004) Cooperative question answering in restricted domains: the webcoop experiment. In: Proceedings of the conference on question answering in restricted domains
9. Berant J, Liang P (2014) Semantic parsing via paraphrasing. In: Proceedings of the 52nd annual meeting of the association for computational linguistics (Volume 1: Long Papers), vol 1, pp 1415–1425

10. Berant J, Chou A, Frostig R, Liang P (2013) Semantic parsing on freebase from question-answer pairs. In: Proceedings of the 2013 conference on empirical methods in natural language processing, pp 1533–1544
11. Bollacker K, Evans C, Paritosh P, Sturge T, Taylor J (2008) Freebase: a collaboratively created graph database for structuring human knowledge. In: Proceedings of the 2008 ACM SIGMOD international conference on Management of data, pp 1247–1250
12. Carlson A, Betteridge J, Kisiel B, Settles B, Hruschka ER, Mitchell TM (2010) Toward an architecture for never-ending language learning. In: AAAI'10 Proceedings of the Twenty-Fourth AAAI conference on artificial intelligence, pp 1306–1313
13. Chen A, Stanovsky G, Singh S, Gardner M (2019) Evaluating question answering evaluation. In: Proceedings of the 2nd workshop on machine reading for question answering, pp 119–124
14. Chen Y, Wu L, Zaki M (2019) Bidirectional attentive memory networks for question answering over knowledge bases. In: NAACL-HLT 2019: Annual Conference of the North american chapter of the association for computational linguistics, pp 2913–2923
15. Chen Y, Li H, Hua Y, Qi G (2021) Formal query building with query structure prediction for complex question answering over knowledge base. arXiv:210903614
16. Das R, Dhuliawala S, Zaheer M, Vilnis L, Durugkar I, Krishnamurthy A, Smola A, McCallum A (2018) Go for a walk and arrive at the answer: Reasoning over paths in knowledge bases using reinforcement learning. In: International conference on learning representations
17. Devi M, Dua M (2017) Adans: An agriculture domain question answering system using ontologies. In: 2017 international conference on computing, communication and automation (ICCCA), pp 122–127
18. Devlin J, Chang MW, Lee K, Toutanova KN (2018) Bert: Pre-training of deep bidirectional transformers for language understanding
19. Diefenbach D, López V, Singh KD, Maret P (2018) Core techniques of question answering systems over knowledge bases: a survey. Knowl Inf Syst 55(3):529–569
20. Dimitrakis E, Sgontzos K, Tzitzikas Y (2019) A survey on question answering systems over linked data and documents. Journal of Intelligent Information Systems, pp 1–27
21. Dong L, Lapata M (2016) Language to logical form with neural attention. In: Proceedings of the 54th annual meeting of the association for computational linguistics (Volume 1: Long Papers), vol 1, pp 33–43
22. Dong L, Lapata M (2018) Coarse-to-fine decoding for neural semantic parsing. In: ACL 2018: 56Th annual meeting of the association for computational linguistics, vol 1, pp 731–742
23. Dong X, Gabrilovich E, Heitz G, Horn W, Lao N, Murphy K, Strohmann T, Sun S, Zhang W (2014) Knowledge vault: a web-scale approach to probabilistic knowledge fusion. In: Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining, pp 601–610
24. Dubey M, Banerjee D, Abdelkawi A, Lehmann J (2019) Lc-quad 2.0 : A large dataset for complex question answering over wikidata and dbpedia. In: 18th international semantic web conference, pp 69–78
25. Dunn M, Sagun L, Higgins M, Güney VU, Cirik V, Cho K (2017) Searchqa: A new q&a dataset augmented with context from a search engine. ArXiv
26. Esposito M, Damiano E, Minutolo A, De Pietro G, Fujita H (2020) Hybrid query expansion using lexical resources and word embeddings for sentence retrieval in question answering. Inf Sci 514:88–105
27. Etezadi R, Shamsfard M (2020) pecoq: A dataset for persian complex question answering over knowledge graph. In: 2020 11Th international conference on information and knowledge technology (IKT), IEEE, pp 102–106
28. Fader A, Soderland S, Etzioni O (2011) Identifying relations for open information extraction. In: Proceedings of the 2011 conference on empirical methods in natural language processing, pp 1535–1545
29. Fader A, Zettlemoyer L, Etzioni O (2013) Paraphrase-driven learning for open question answering. In: Proceedings of the 51st annual meeting of the association for computational linguistics (Volume 1: Long Papers), vol 1, pp 1608–1618
30. Fader A, Zettlemoyer L, Etzioni O (2014) Open question answering over curated and extracted knowledge bases. In: Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining, pp 1156–1165
31. Färber M, Rettinger A (2018) Which knowledge graph is best for me. arXiv:180911099
32. Fu B, Qiu Y, Tang C, Li Y, Yu H, Sun J (2020) A survey on complex question answering over knowledge base: Recent advances and challenges. arXiv:200713069
33. Galárraga L, Heitz G, Murphy K, Suchanek FM (2014) Canonicalizing open knowledge bases. In: Proceedings of the 23rd ACM international conference on conference on information and knowledge management, pp 1679–1688
34. Gao J, Pantel P, Gamon M, He X, Deng L (2014) Modeling interestingness with deep neural networks
35. Gashteovski K, Wanner S, Hertling S, Broscheit S, Gemulla R (2019) Opiec: An open information extraction corpus. In: AKBC 2019 : 1st conference on automated knowledge base construction
36. Green Jr BF, Wolf AK, Chomsky C, Laughery K (1961) Baseball: an automatic question-answerer. In: Papers presented at the May 9-11, 1961, western joint IRE-AIEE-ACM computer conference, pp 219–224
37. Hartrumpf S (2004) Question answering using sentence parsing and semantic network matching. cross language evaluation forum pp 512–521
38. Hartrumpf S (2005) Extending knowledge and deepening linguistic processing for the question answering system insicht. cross language evaluation forum pp 361–369
39. Hartrumpf S (2008) Semantic decomposition for question answering. In: Proceedings of the 2008 conference on ECAI 2008: 18th European conference on artificial intelligence, pp 313–317
40. Helbig H (2005) Knowledge representation and the semantics of natural language
41. Herzig J, Nowak PK, Müller T, Piccinno F, Eisenschlos JM (2020) Tapas: Weakly supervised table parsing via pre-training. In: Proceedings of the 58th annual meeting of the association for computational linguistics, pp 4320–4333
42. Höffner K, Lehmann J, Usbeck R (2016) Cubeqa—question answering on rdf data cubes. In: International semantic web conference, vol 1, pp 325–340
43. Höffner K, Walter S, Marx E, Usbeck R, Lehmann J, Ngonga Ngomo AC (2017) Survey on challenges of Question Answering in the Semantic Web. Semantic Web Journal 8(6):895–920
44. Hu R, Rohrbach A, Darrell T, Saenko K (2019) Language-conditioned graph networks for relational reasoning. In: Proceedings of the IEEE/CVF international conference on computer vision, pp 10294–10303
45. Hu S, Zou L, Zhang X (2018) A state-transition framework to answer complex questions over knowledge base. In: EMNLP 2018: 2018 conference on empirical methods in natural language processing, pp 2098–2108

46. Hua Y, Li YF, Haffari G, Qi G, Wu T (2020) Few-shot complex knowledge base question answering via meta reinforcement learning. In: Proceedings of the 2020 conference on empirical methods in natural language processing (EMNLP), pp 5827–5837
47. Huang Z, Xu W, Yu K (2015) Bidirectional lstm-crf models for sequence tagging. arXiv:150801991
48. Hudson DA, Manning CD (2018) Compositional attention networks for machine reasoning. In: International conference on learning representations
49. Jain S (2016) Question answering over knowledge base using factual memory networks. In: Proceedings of the NAACL student research workshop, pp 109–115
50. Jia Z, Pramanik S, Saha Roy R, Weikum G (2021) Complex temporal question answering on knowledge graphs. In: Proceedings of the 30th ACM international conference on information & knowledge management, pp 792–802
51. Kalyanpur A, Patwardhan S, Boguraev B, Lally A, Chu-Carroll J (2012) Fact-based question decomposition in deepqa. IBM J Res Dev 56(3.4):13–1
52. Katz B, Borchardt G, Felshin S (2005) Syntactic and semantic decomposition strategies for question answering from multiple resources. In: Proceedings of the AAAI 2005 workshop on inference for textual question answering, AAAI Press Menlo Park, CA, pp 35–41
53. Khot T, Sabharwal A, Clark P (2017) Answering complex questions using open information extraction. In: Proceedings of the 55th annual meeting of the association for computational linguistics (Volume 2: Short Papers), vol 2, pp 311–316
54. Kwiatkowski T, Choi E, Artzi Y, Zettlemoyer L (2013) Scaling semantic parsers with on-the-fly ontology matching. In: Proceedings of the 2013 conference on empirical methods in natural language processing, pp 1545–1556
55. Kwon H, Trivedi H, Jansen P, Surdeanu M, Balasubramanian N (2018) Controlling information aggregation for complex question answering. In: 40Th european conference on information retrieval, ECIR, vol 2018, pp 750–757
56. Lan Y, Jiang J (2020) Query graph generation for answering multi-hop complex questions from knowledge bases. In: ACL 2020: 58th annual meeting of the association for computational linguistics, pp 969–974
57. Lan Y, Wang S, Jiang J (2019) Multi-hop knowledge base question answering with an iterative sequence matching model. In: 2019 IEEE international conference on data mining (ICDM), IEEE, pp 359–368
58. Lan Y, He G, Jiang J, Jiang J, Zhao WX, Wen JR (2021) Complex knowledge base question answering: A survey. arXiv:210806688
59. Lee J, Seo M, Hajishirzi H, Kang J (2020) Contextualized sparse representations for real-time open-domain question answering. In: ACL 2020: 58th annual meeting of the Association for Computational Linguistics
60. Lewis P, Oguz B, Rinott R, Riedel S, Schwenk H (2020) MLQA: Evaluating Cross-lingual extractive question answering In: Proceedings of the 58th annual meeting of the association for computational linguistics, association for computational linguistics, Online
61. Liang P (2013) Lambda dependency-based compositional semantics. arXiv:13094408
62. Lin CY (2004) Rouge: A package for automatic evaluation of summaries. In: Text summarization branches out, pp 74–81
63. Lin XV, Socher R, Xiong C (2018) Multi-hop knowledge graph reasoning with reward shaping. In: Proceedings of the 2018 conference on empirical methods in natural language processing, pp 3243–3253
64. Liu H, Hu Q, Zhang Y, Xing C, Sheng M (2017) A knowledge-based health question answering system. In: International conference on smart health, pp 286–291
65. Liu Y, Hao Y, Zhu X, Li J (2015) A question answering system built on domain knowledge base. In: International conference on web-age information management, pp 111–122
66. Liu Y, Ott M, Goyal N, Du J, Joshi M, Chen D, Levy O, Lewis M, Zettlemoyer L, Stoyanov V (2019) Roberta: A robustly optimized bert pretraining approach. arXiv:190711692
67. Luo K, Lin F, Luo X, Zhu K (2018) Knowledge base question answering via encoding of complex query graphs. In: EMNLP 2018: 2018 conference on empirical methods in natural language processing, pp 2185–2194
68. Lyu Q, Chakrabarti K, Hathi S, Kundu S, Zhang J, Chen Z (2020) Hybrid ranking network for text-to-sql. arXiv:200804759
69. Ma J, Yan Z, Pang S, Zhang Y, Shen J (2020) Mention extraction and linking for sql query generation. In: Proceedings of the 2020 conference on empirical methods in natural language processing (EMNLP), pp 6936–6942
70. Min S, Chen D, Hajishirzi H, Zettlemoyer L (2019) A discrete hard em approach for weakly supervised question answering. In: Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP), pp 2851–2864
71. Mishra A, Jain SK (2016) A survey on question answering systems with classification. Journal of King Saud University-Computer and Information Sciences 28(3):345–361
72. Neumann G, Sacaleanu B (2005) Experiments on cross–linguality and question–type driven strategy selection for open–domain qa. cross language evaluation forum pp 429–438
73. Neumann G, Xu F (2003) Mining answers in german web pages. In: Proceedings IEEE/WIC international conference on web intelligence (WI 2003), pp 125–131
74. Nguyen T, Rosenberg M, Song X, Gao J, Tiwary S, Majumder R, Deng L (2016) Ms marco: a human generated machine reading comprehension dataset. In: CoCo@ NIPS
75. Papineni K, Roukos S, Ward T, Zhu WJ (2002) Bleu: a method for automatic evaluation of machine translation. In: Proceedings of the 40th annual meeting of the Association for Computational Linguistics, pp 311–318
76. Pasupat P, Liang P (2015) Compositional semantic parsing on semi-structured tables. In: Proceedings of the 53rd annual meeting of the association for computational linguistics and the 7th international joint conference on natural language processing (Volume 1: Long Papers), pp 1470–1480
77. Qin K, Li C, Pavlu V, Aslam J (2021) Improving query graph generation for complex question answering over knowledge base
78. Rajpurkar P, Zhang J, Lopyrev K, Liang P (2016) Squad: 100,000+ questions for machine comprehension of text. In: Proceedings of the 2016 conference on empirical methods in natural language processing, pp 2383–2392
79. Reddy ACO, Madhavi K (2020) Convolutional recurrent neural network with template based representation for complex question answering. International Journal of Electrical and Computer Engineering 10(3):2710–2718
80. Reddy S, Lapata M, Steedman M (2014) Large-scale semantic parsing without question-answer pairs. Transactions of the Association for Computational Linguistics 2(1):377–392
81. Rodrigo A (2017) A study about the future evaluation of question-answering systems. Knowledge Based Systems 137:83–93
82. Sajadi MB, Minaei B, Hadian A (2018) Farsbase: A cross-domain farsi knowledge graph. SEMANTICS Posters&Demos

83. Salunkhe A (2020) Evolution of techniques for question answering over knowledge base: a survey. International Journal of Computer Applications 177(34):9–14
84. Sasikumar U (2014) A survey of natural language question answering system. International Journal of Computer Applications 108(15):42–46
85. Saxena A, Tripathi A, Talukdar P (2020) Improving multi-hop question answering over knowledge graphs using knowledge base embeddings. In: ACL 2020: 58th annual meeting of the association for computational linguistics, pp 4498–4507
86. Seo MJ, Kembhavi A, Farhadi A, Hajishirzi H (2017) Bidirectional attention flow for machine comprehension/ In: 5Th international conference on learning representations, ICLR 2017, toulon, france, april 24-26, 2017, Conference Track Proceedings, OpenReview.net
87. Shen Y, He X, Gao J, Deng L, Mesnil G (2014) A latent semantic model with convolutional-pooling structure for information retrieval. In: Proceedings of the 23rd ACM international conference on conference on information and knowledge management, pp 101–110
88. Shen Y, He X, Gao J, Deng L, Mesnil G (2014) Learning semantic representations using convolutional neural networks for web search. In: Proceedings of the 23rd international conference on world wide web, pp 373–374
89. Shin S, Lee KH (2020) Processing knowledge graph-based complex questions through question decomposition and recomposition. Inf Sci 523:234–244
90. Suchanek FM, Kasneci G, Weikum G (2007) Yago: a core of semantic knowledge. In: Proceedings of the 16th international conference on World Wide Web, pp 697–706
91. Sun H, Dhingra B, Zaheer M, Mazaitis K, Salakhutdinov R, Cohen W (2018) Open domain question answering using early fusion of knowledge bases and text. In: EMNLP 2018: 2018 conference on empirical methods in natural language processing, pp 4231–4242
92. Sun H, Bedrax-Weiss T, Cohen W (2019) Pullnet: Open domain question answering with iterative retrieval on knowledge bases and text. In: 2019 conference on empirical methods in natural language processing, pp 2380–2390
93. Talmor A, Berant J (2018) The web as a knowledge-base for answering complex questions. In: NAACL HLT 2018: 16Th annual conference of the north american chapter of the association for computational linguistics: human language technologies, vol 1, pp 641–651
94. Talmor A, Yoran O, Catav A, Lahav D, Wang Y, Asai A, Ilharco G, Hajishirzi H, Berant J (2021) Multimodal{qa}: complex question answering over text tables and images. In: International conference on learning representations
95. Trivedi P, Maheshwari G, Dubey M, Lehmann J (2017) Lc-quad: a corpus for complex question answering over knowledge graphs. In: International semantic web conference, vol 2, pp 210–218
96. Trouillon T, Welbl J, Riedel S, Gaussier É, Bouchard G (2016) Complex embeddings for simple link prediction
97. Tu M, Huang K, Wang G, Huang J, He X, Zhou B (2020) Select, answer and explain: Interpretable multi-hop reading comprehension over multiple documents. In: Proceedings of the AAAI conference on artificial intelligence, vol 34, pp 9073–9080
98. Vakulenko S, Savenkov V (2017) Tableqa: Question answering on tabular data. SEMANTICS Posters&Demos
99. Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser L, Polosukhin I (2017) Attention is all you need. In: NIPS, pp 6000–6010
100. Veličković P, Cucurull G, Casanova A, Romero A, Liò P, Bengio Y (2018) Graph attention networks. In: International conference on learning representations
101. Vrandečić D, Krötzsch M (2014) Wikidata: a free collaborative knowledgebase. Communications of The ACM 57(10):78–85
102. Wang B, Titov I, Lapata M (2019) Learning semantic parsers from denotations with latent structured alignments and abstract programs. In: Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP), pp 3765–3776
103. Wang M (2006) A survey of answer extraction techniques in factoid question answering. Computational Linguistics 1(1):1–14
104. Wang Z, Huang J, Li H, Liu B, Shao B, Wang H, Wang J, Wang Y, Wu W, Xiao J et al (2010) Probase: a universal knowledge base for semantic search. Microsoft Research Asia
105. Welbl J, Stenetorp P, Riedel S (2018) Constructing datasets for multi-hop reading comprehension across documents. Transactions of the Association for Computational Linguistics 6:287–302
106. Xu K, Reddy S, Feng Y, Huang S, Zhao D (2016) Question answering on freebase via relation extraction and textual evidence. In: Proceedings of the 54th annual meeting of the association for computational linguistics (Volume 1: Long Papers), vol 1, pp 2326–2336
107. Yang W, Xie Y, Lin A, Li X, Tan L, Xiong K, Li M, Lin J (2019) End-to-end open-domain question answering with bertserini. In: NAACL-HLT 2019: Annual conference of the north american chapter of the association for computational linguistics, pp 72–77
108. Yang Y, Chang MW (2015) S-mart: Novel tree-based structured learning algorithms applied to tweet entity linking. In: Proceedings of the 53rd annual meeting of the association for computational linguistics and the 7th international joint conference on natural language processing (Volume 1: Long Papers), vol 1, pp 504–513
109. Yang Y, tau Yih W, Meek C (2015) Wikiqa: A challenge dataset for open-domain question answering. In: Proceedings of the 2015 conference on empirical methods in natural language processing, pp 2013–2018
110. Yang Z, Qi P, Zhang S, Bengio Y, Cohen WW, Salakhutdinov R, Manning CD (2018) Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In: Proceedings of the 2018 conference on empirical methods in natural language processing, pp 2369–2380
111. Yao X (2015) Lean question answering over freebase from scratch. In: Proceedings of the 2015 conference of the North american chapter of the association for computational linguistics: demonstrations, pp 66–70
112. Yao X, Durme BV (2014) Information extraction over structured data: Question answering with freebase. In: Proceedings of the 52nd annual meeting of the association for computational linguistics (Volume 1: Long Papers), vol 1, pp 956–966
113. Yih W, Chang MW, He X, Gao J (2015) Semantic parsing via staged query graph generation: Question answering with knowledge base. In: Proceedings of the 53rd annual meeting of the association for computational linguistics and the 7th international joint conference on natural language processing (Volume 1: Long Papers), vol 1, pp 1321–1331
114. Yih W, Richardson M, Meek C, Chang MW, Suh J (2016) The value of semantic parse labeling for knowledge base question answering. In: Proceedings of the 54th annual meeting of

the association for computational linguistics (Volume 2: Short Papers), vol 2, pp 201–206
115. Yin P, Duan N, Kao B, Bao J, Zhou M (2015) Answering questions with complex semantic constraints on open knowledge bases. In: Proceedings of the 24th ACM international on conference on information and knowledge management, pp 1301–1310
116. Yu AW, Dohan D, Le Q, Luong T, Zhao R, Chen K (2018) Fast and accurate reading comprehension by combining self-attention and convolution. In: International conference on learning representations
117. Zhang H, Cai J, Xu J, Wang J (2019) Complex question decomposition for semantic parsing. In: ACL 2019 : The 57th annual meeting of the association for computational linguistics, pp 4477–4486
118. Zhang* T, Kishore* V, Wu* F, Weinberger KQ, Artzi Y (2020) Bertscore: Evaluating text generation with bert. In: International conference on learning representations
119. Zhang Y, Dai H, Kozareva Z, Smola A, Song L (2018) Variational reasoning for question answering with knowledge graph. In: AAAI-18 AAAI conference on artificial intelligence, pp 6069–6076
120. Zhao W, Chung T, Goyal A, Metallinou A (2019) Simple question answering with subgraph ranking and joint-scoring. In: NAACL-HLT 2019: Annual conference of the north american chapter of the association for computational linguistics, pp 324–334
121. Zhong V, Xiong C, Socher R (2017) Seq2sql: Generating structured queries from natural language using reinforcement learning. CoRR arXiv:abs/1709.00103



Romina Etezadi has graduated with a master's degree in Computer Science-Artificial Intelligence from Shahid Beheshti University, Tehran, Iran. She is currently an NLP researcher. She is particularly interested in natural language processing, NLP's applications, knowledge graphs, machine learning, and deep learning.

Mehrnoush Shamsfard has received her BS and MSc both on computer software engineering from Sharif University of Technology and her PhD in Computer Engineering- Artificial Intelligence from AmirKabir University of Technology, Tehran, Iran. Currently, she is an associate professor of Faculty of computer science and engineering in Shahid Beheshti University, and also the head of NLP research Laboratory of this faculty. Her main fields of interest are natural language processing, knowledge and ontology engineering, text mining and semantic and intelligent web.