Received: 17 October 2020 | Revised: 3 April 2021 | Accepted: 6 May 2021

IET Software

DOI: 10.1049/sfw2.12028

The Institution of Engineering and Technology WILEY

REVIEW

Systematic review of question answering over knowledge bases

Arnaldo Pereira[1] | Alina Trifan[1] | Rui Pedro Lopes[2] | José Luís Oliveira[1]

[1]Institute of Electronics and Informatics Engineering of Aveiro (IEETA), University of Aveiro, Aveiro, Portugal

[2]Research Centre in Digitalization and Intelligent Robotics (CeDRI), Polytechnic Institute of Bragança, Bragança, Portugal

Correspondence

Arnaldo Pereira, Institute of Electronics and Informatics Engineering of Aveiro (IEETA), University of Aveiro, 3810-193 Aveiro, Portugal.
Email: arnaldop@ua.pt

Funding information

Fundação para a Ciência e a Tecnologia, Grant/Award Number: UIDB/00127/2020; Fundação para a Ciência e a Tecnologia, Grant/Award Number: PD/BD/142877/2018

Abstract

Over the years, a growing number of semantic data repositories have been made available on the web. However, this has created new challenges in exploiting these resources efficiently. Querying services require knowledge beyond the typical user's expertise, which is a critical issue in adopting semantic information solutions. Several proposals to overcome this difficulty have suggested using question answering (QA) systems to provide user-friendly interfaces and allow natural language use. Because question answering over knowledge bases (KBQAs) is a very active research topic, a comprehensive view of the field is essential. The purpose of this study was to conduct a systematic review of methods and systems for KBQAs to identify their main advantages and limitations. The inclusion criteria rationale was English full-text articles published since 2015 on methods and systems for KBQAs. Sixty-six articles were reviewed to describe their underlying reference architectures.

1 | INTRODUCTION

Question answering (QA) refers to systems that allow users to use natural language (NL) interfaces to ask questions and receive concise answers. The first solution was created in the 1960s to answer questions asked in English about baseball games from information saved in a list-structured database [1]. Some years later, with the emergence of the relational data model, considerable effort was put into developing natural language interfaces for databases (NLIDB). However, just five years after the creation of the World Wide Web, Androutsopoulos et al. reported the lack of interest in investigating NLIDB [2]. In those days, the focus was on information retrieval techniques to create web search engines using the keyword-based search paradigm. Meanwhile, QA over text advanced [3], and the Semantic Web (SW) vision formulated in 2001 by Berners-Lee et al. [4] brought attention to semantic data.

Search engines have come to offer direct answers to some user questions in recent years [5]. Instead of just presenting a list of links to documents where the answer is likely to be found, the idea is to satisfy the need for information without further searching and navigation. Questions whose answer is an entity are the ideal candidate for this type of approach, and using large semantic databases that capture general knowledge has become of great value. In this context, triples extraction to answer questions is priceless and continues to motivate research work.

Since the formulation of SW principles and standards, semantic representations to capture knowledge in the life sciences have become common practice [6]. As a result, there was an explosion in the creation of ontologies and Resource Description Framework (RDF) databases made available online. The usual access to this data has been through visual navigation interfaces and commonly through SPARQL Protocol and RDF Query Language (SPARQL) endpoints, but these access methods have problems. The first approach is not rich enough to answer more complex questions, and the second is not suitable for users who have not mastered the use of formal querying languages. Therefore, this reality is pressing for new question answering over knowledge base (KBQA) solutions for biomedical data.

A couple of examples from the life sciences illustrate the use of these systems. Asiaee et al. applied a KBQA solution to parasite immunology [7], and Hamon et al. created a querying platform for linked biomedical data [8]. Other KBQA systems retrieve information from open knowledge databases, such as DBpedia or Wikidata, or use proprietary enterprise knowledge graphs, such as Google Knowledge Graph or Bing Satori [9].

We have identified the publication of surveys and overviews exclusively or partially about KBQA since 2015. Höffner et al. [10] surveyed 72 papers published until the middle of 2015 and mapped against a set of challenges. By the depth of its analysis and selection of articles, this study is a reference work to understand the contributions that had emerged until that moment. Other contributions, such as those of Mishra and Jain [11], Dimitrakis et al. [12], Affolter et al. [13], and Ojokoh and Adebisi [14], are narrative reviews, meaning that they do not use systematic search methods. Unlike the latter studies, our review follows a systematic search strategy that provides a greater guarantee that essential contributions have not gone unnoticed. We also focus on the presentation of KBQA reference architectures, which none of the previous works addresses. Our attention to architectural decomposition allows us to highlight state-of-the-art practices for particular subtasks.

The remainder of this paper comprises four sections. Section 2 overviews the current aspects of KBQA. In Section 3, we introduce the methods for retrieving and selecting the articles surveyed and present the quantitative results of that selection. In Section 4, we critically summarize our findings. We wind up the paper with the conclusions in Section 5.

2 | BACKGROUND

The purpose of QA systems is to allow users to create questions in NL without a formal query language. Considering the nature of the data sources, we can divide these solutions into three major groups: QA over unstructured data (e.g., text), QA over semi-structured data (e.g., tables), and QA over structured data (e.g. graph data sets). Hybrid systems operate in two or more kinds of data sources. Regarding the scope of data, on the one hand, we can consider domain-specific solutions when the data schema is narrowed down to a particular body of knowledge (e.g., biomedical data) that limits the question types that are accepted. On the other hand, open-domain systems consider data on generic subjects specified by general ontologies.

RDF is the data model for the SW, established in a suite of normative specifications by the World Wide Web Consortium [15]. An RDF triple (or statement) has three components: the subject, which is an Internationalized Resource Identifier (IRI) or a blank node (bnode); the predicate, which is an IRI; and the object, which is an IRI, a literal, or a bnode [16]. A set of RDF triples is an RDF graph, and an RDF data set is a collection of RDF graphs. Several RDF serialization formats are available, such as Turtle, TriG, and JSON-LD (JavaScript Object Notation for Linked Data).

We realize the real power of the triples when considering large data sets. An RDF store (or triplestore) is a proper database for the storage and retrieval of triples. For quick reference, we can list some well-known solutions: OpenLink Virtuoso (https://virtuoso.openlinksw.com/), Eclipse RDF4J (formerly Sesame) (http://rdf4j.org/), Apache Jena (https://jena.apache.org/), and GraphDB (http://graphdb.ontotext.com/). By default, we use SPARQL [17] for querying RDF graphs considering one of four query forms. The SELECT construction returns variables and their bindings directly, and the CONSTRUCT clause returns a single RDF graph. The ASK statement returns a boolean indicating whether a query pattern matches, and a DESCRIBE query returns an RDF graph describing the resources found [18].

Several benchmarks and evaluation campaigns have promoted the advancement of KBQA systems. The Question Answering on Linked Data (QALD) challenge launched in 2011 is the oldest running campaign, and its ninth edition provided a training data set with 408 questions in 11 different languages for the open-domain semantic QA over DBpedia task [19]. To shorten the size limitations of the QALD data set, the Large-Scale Complex Question Answering Dataset (LC-QuAD) provides 30,000 questions with corresponding SPARQL queries for DBpedia and Wikidata [20]. Free917 is another benchmark and consists of 917 utterances using Freebase with a meaning representation in a variant of lambda calculus [21]. To avoid using logical forms, Berant et al. [22] created the WebQuestions data set containing 5810 question-answer pairs, to which Yih et al. [23] added SPARQL queries to create WebQuestionsSP. Then, Bordes et al. [24] achieved, with SimpleQuestions, a significant scale-up of the numbers with 108,442 questions for possible rephrasing in the form (subject, relationship, ?). Finally, the BioASQ series of challenges has a task on domain-specific semantic QA on biomedical data to evaluate systems outputting relevant triples and text snippets [25].

To meet the challenges posed in implementing KBQA solutions, it is important to identify the most common architectures. From the analysis of the papers we selected for our work, we found that they are classified by four different architectures. Semantic parsing pipelines are solutions based on semantic parsing, which uses a pipe and filter style where data flows to generate a formal query from the original input in NL. It is the most straightforward architectural style of KBQA systems and relies on connecting components to form a pipeline, as shown in Figure 1.

The idea is to apply several data transformations from the question in NL to the logical form or formal query. To achieve that, we use natural language processing (NLP) techniques such as tokenization, named-entity recognition (NER), part-of-speech (POS) tagging, dependency parsing, and entity linking (EL). In addition, steps for query generation and answer generation are required.

An alternative way of using semantic parsing is based on the observation that executing a formal query is equivalent to finding a subgraph, as depicted in Figure 2.

Systems capable of answering complex questions (e.g., questions that cannot be reduced to a simple triple pattern) require a higher degree of sophistication than the systems presented so far. A template is a query skeleton with an arbitrary degree of complexity that fits the knowledge base (KB) to be questioned and has slots that must be filled with information from entities and relations. Naturally, the quality of the system depends on the effort put into creating the templates. In the early stages, the use of templates was mainly accomplished through manual annotations; in more recent times,

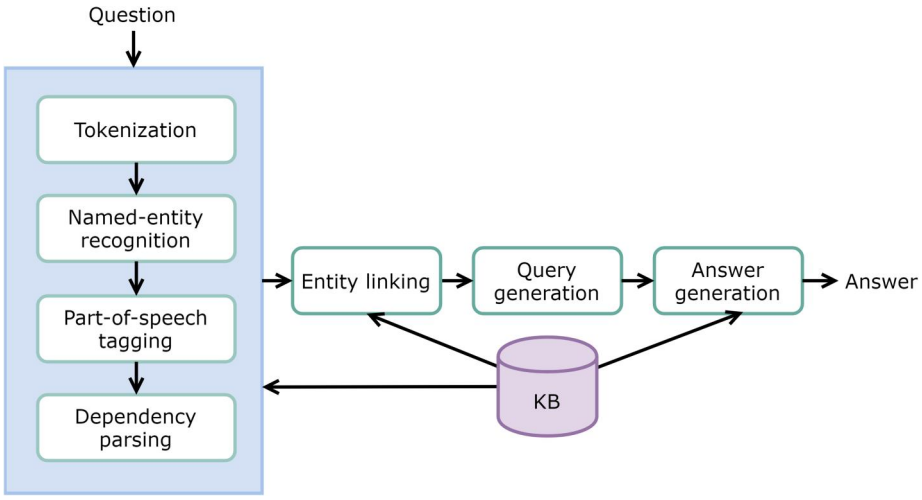

FIGURE 1 General architecture for semantic parsing pipelines. The direction of the arrows denotes the direction of the data flows

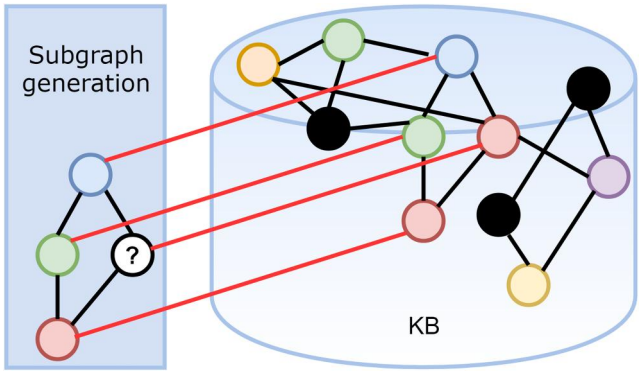

FIGURE 2 Subgraph matching approach

automatic approaches have emerged. These systems rely on the manual or automatic creation of a template database assuming an architectural configuration such as that shown in Figure 3.

In the offline phase, it is necessary to create templates. This involves considering pairs of questions and answers used to obtain successively more abstract representations that are used to generate pairs of question-query templates after alignment. The online phase is straightforward: a question is matched with a template to produce a query template, the slots are filled with entities and relations, and the answer is provided by issuing the query candidate.

End-to-end solutions perform sequence-to-sequence translation or apply methods to extract triples directly from the KB. The selection of the final answer is based on the representations of the questions in NL obtained by applying machine learning techniques, as can be seen in Figure 4. After extracting the candidate answers from the KB, they are evaluated against a predefined score using a specialized function.

Höffner et al. [10] highlighted significant challenges faced by semantic QA. The lexical gap occurs when the surface forms used in a question are different from those used in the KB. The ambiguity stemming from the fact that the same word can represent various entities is also problematic. Another significant problem is finding answers to questions manoeuvring several units combined in complex queries requiring ordered, aggregated, or filtered outputs. Equally challenging is multilingualism, which concerns two distinct realities that may or may not co-occur. The first involves the problem of using the same interface to ask questions in several NLs, and the second has to do with the possibility of the KB data being multilingual. In addition, systems relying on languages other than English end up receiving far less attention from the scientific community, limiting the number of available solutions. For instance, very few developers have participated in challenges like QALD with multilingual systems. Some systems try to prevent difficulties by using controlled natural languages (CNLs), which are constructions that restrict in some way the lexicon, syntax, or semantics of the NL from which they start. This review will not focus on multilingualism or the use of CNL interfaces.

3 | PAPER SELECTION

Computer science and software engineering can benefit from using systematic literature reviews to synthesize the best evidence about the state of the art, especially for mature topics for which a large number of studies may not be appropriately acknowledged [26]. In this work, we followed a strict methodology. To enable repeatability, we have created a replication package that is available at https://osf.io/hxyvw. We have used PRISMA (Preferred Reporting Items for Systematic Reviews and Meta-Analyses) guidelines [27] to report the protocol's execution and present the findings.

3.1 | Search methodology

The following research questions guided our study:

RQ1 Which methods of KBQA have been proposed?
RQ2 What are the solved and unsolved challenges?

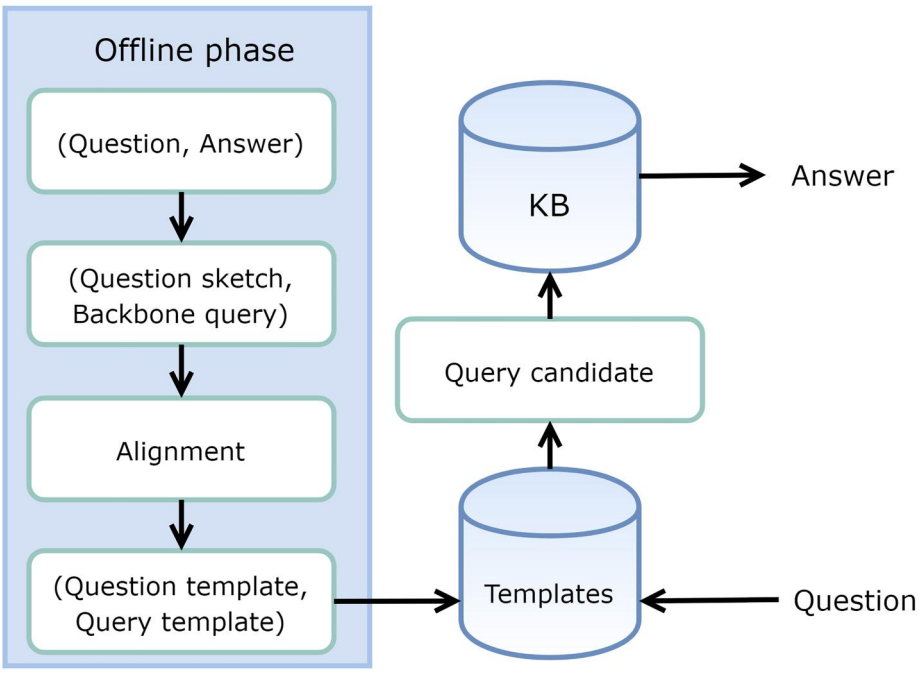

FIGURE 3 General architecture for template-based question answering over knowledge bases

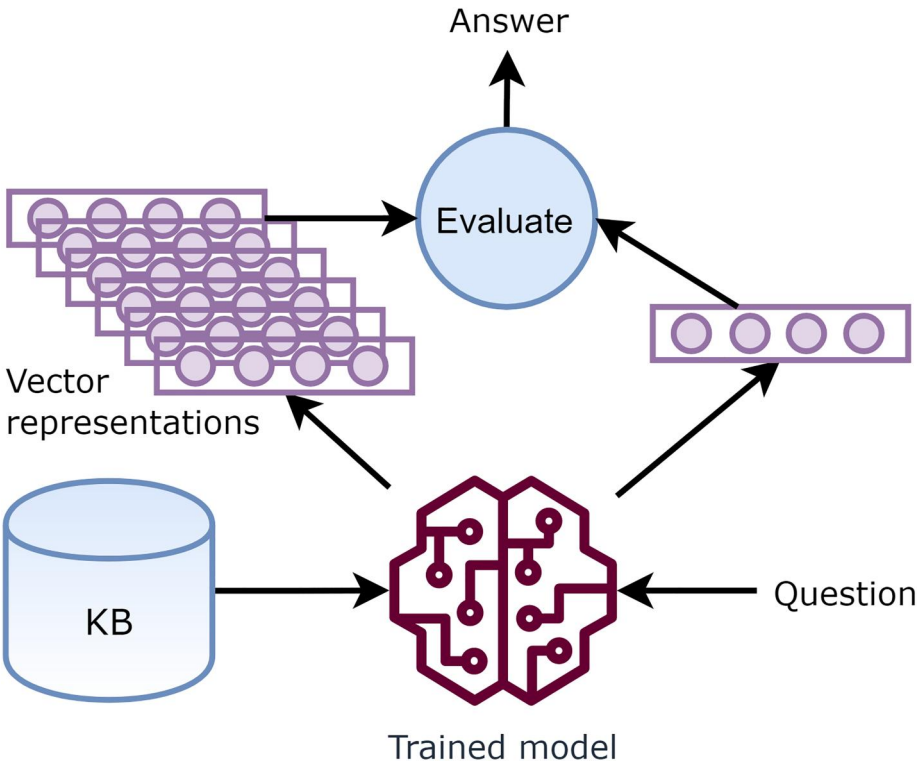

FIGURE 4 General architecture for question answering over knowledge bases based on information extraction

Going through past surveys and overviews [10–14], we collected the keywords shown in Table 1 mapped against the Population, Intervention, Comparison, Outcomes (PICO) structure [28]. As pointed out previously, Höffner et al. [10] presented a systematic study of KBQA systems until mid-2015. We have built upon that. Our search query followed the template

((Population OR Comparison)
AND
(Intervention OR Outcomes))

TABLE 1 Slot values for the population, intervention, comparison, and outomes template

PICO Slot	Values
Population	'Knowledge Base*' OR 'Knowledge Graph*' OR 'Semantic Web' OR 'Linked Data*' OR 'RDF Data*' OR 'data web'
Intervention	Question-Answer* OR 'natural language que*' OR 'Natural Language Interface'
Comparison	SPARQL OR 'Query Graph*'
Outcomes	QALD* OR SimpleQuestions OR WebQuestions OR WebQSP OR LC-QuAD

Abbreviations: SPARQL, SPARQL Protocol and RDF Query Language; QALD, Question Answering on Linked Data.

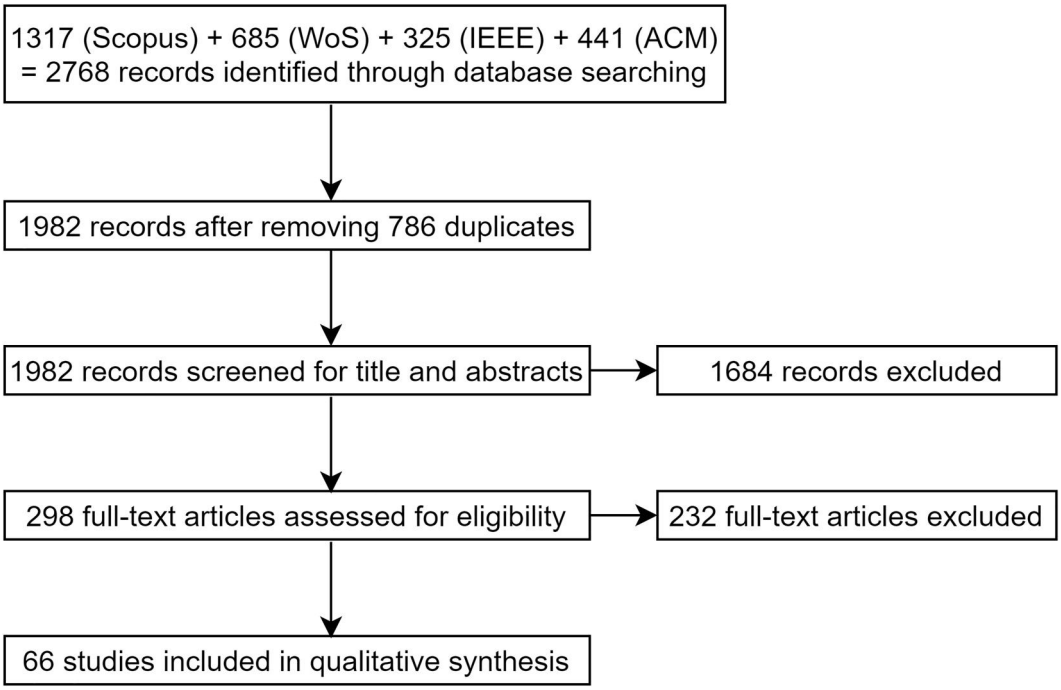

FIGURE 5 Study selection

Then we used Scopus, Web of Science, IEEE Xplore, and the ACM Digital Library to find KBQA papers. We emphasize that our research questions and search string allow us to retrieve all papers reviewed by Höffner et al. [10], in addition to the most recent ones. Next, we considered the following inclusion criteria:

I1 Studies on methods and systems for question answering over knowledge bases.
I2 Papers focussing on a specific challenge of KBQA solutions (answering complex questions, module reusability etc.).

For the exclusion criteria, we have

E1 Books, surveys and overviews, tutorials, talks, panel sessions, conference reviews, editorials, abstracts, a summary of a workshop or challenge, dissertations, or grey literature. Not available in English. Being unable to retrieve the full text.
E2 If faced with multiple papers by the same author about the same subject, we will keep only those needed to report the main contribution.

The main threats to validity are losing relevant studies in the search step and risking the rejection of relevant studies in the review phase. The PICO strategy minimizes the first problem, ensuring high recall. In addition, we have queried four bibliographic databases. To mitigate the second problem, we allocated two people (the first two authors) to select and review papers. We have performed content analysis to collect information about the challenges addressed, proposed solutions, future work, and architectural styles. The fourth author participated as a referee when disagreements arose.

3.2 | Selection results

Figure 5 shows the flow diagram of the paper selection procedure. After applying the inclusion and exclusion criteria, we selected the 66 papers listed in Table 2. Figure 6 presents the relations of the keywords of the papers selected for qualitative analysis. Naturally, the term 'question answering' appears in the most articles. It is interesting to note that the 'Semantic Web' entry is very prominent only

TABLE 2 Publications selected

ID	Paper name	Architectural style	Year
1	Answering questions with complex semantic constraints on open knowledge bases	Sem. parsing pipe.	2015
2	Applying semantic parsing to question answering over linked data: Addressing the lexical gap [29]	Sem. parsing pipe.	2015
3	HAWK—hybrid question answering using linked data	Sem. parsing pipe.	2015
4	How to build templates for RDF question/answering - An uncertain graph similarity join approach [30]	Template-based	2015
5	ISOFT at QALD-5: Hybrid question answering system over linked data and text data	Sem. parsing pipe.	2015
6	More accurate question answering on Freebase	Template-based	2015
7	QAnswer—Enhanced entity matching for question answering over linked data [31]	Sem. parsing pipe.	2015
8	Question answering over Freebase with multi-column convolutional neural networks [32]	Info. extraction	2015
9	Question answering via phrasal semantic parsing	Sem. parsing pipe.	2015
10	Semantic parsing via staged query graph generation: Question answering with knowledge base [33]	Subgraph matching	2015
11	SemGraphQA@QALD-5: LIMSI participation at QALD-5@CLEF	Subgraph matching	2015
12	SINA: Semantic interpretation of user queries for question answering on interlinked data	Sem. parsing pipe.	2015
13	TR Discover: A natural language interface for querying and analysing interlinked datasets [34]	Sem. parsing pipe.	2015
14	AskNow: A framework for natural language query formalization in SPARQL [35]	Sem. parsing pipe.	2016
15	CFO: Conditional focussed neural question answering with large-scale knowledge bases	Info. extraction	2016
16	Character-level question answering with attention	Info. extraction	2016
17	Constraint-based question answering with knowledge graph	Subgraph matching	2016
18	GRU-RNN based question answering over knowledge base	Info. extraction	2016
19	Hybrid question answering over knowledge base and free text	Sem. parsing pipe.	2016
20	Knowledge base question answering based on deep learning models	Info. extraction	2016
21	Neural generative question answering	Info. extraction	2016
22	Qanary—A methodology for vocabulary-driven open question answering systems [36]	Sem. parsing pipe.	2016
23	QuerioDALI: Question answering over dynamic and linked knowledge graphs [37]	Sem. parsing pipe.	2016
24	Question answering on Freebase via relation extraction and textual evidence [38]	Info. extraction	2016
25	The value of semantic parse labelling for knowledge base question answering [23]	Sem. parsing pipe.	2016
26	When a knowledge base is not enough: Question answering over knowledge bases with external text data [39]	Info. extraction	2016
27	An end-to-end model for question answering over knowledge base with cross-attention combining global knowledge [40]	Info. extraction	2017
28	Automated template generation for question answering over knowledge graphs [41]	Template-based	2017
29	End-to-end representation learning for question answering with weak supervision	Subgraph matching	2017
30	Improved neural relation detection for knowledge base question answering [42]	Info. extraction	2017
31	Introducing feedback in Qanary: How users can Interact with QA systems [43]	Sem. parsing pipe.	2017
32	KBQA: Learning question answering over QA corpora and knowledge bases	Template-based	2017
33	Matching natural language relations to knowledge graph properties for question answering	Sem. parsing pipe.	2017
34	Natural language supported relation matching for question answering with knowledge graphs	Info. extraction	2017
35	Neural network-based question answering over knowledge graphs on word and character levels [9]	Info. extraction	2017
36	QAESTRO—semantic-based composition of question answering pipelines [44]	Sem. parsing pipe.	2017
37	Querying biomedical linked data with natural language questions [8]	Sem. parsing pipe.	2017

TABLE 2 (Continued)

ID	Paper name	Architectural style	Year
38	Question answering on knowledge bases and text using universal schema and memory networks	Info. extraction	2017
39	Trill: A reusable front-end for QA systems [45]	Sem. parsing pipe.	2017
40	An attention-based word-level Interaction model for knowledge base relation detection	Info. extraction	2018
41	Answering natural language questions by subgraph matching over knowledge graphs [46]	Subgraph matching	2018
42	Formal query generation for question answering over knowledge bases [47]	Sem. parsing pipe.	2018
43	Frankenstein: A platform enabling reuse of question answering components [48]	Sem. parsing pipe.	2018
44	Never-ending learning for open-domain question answering over knowledge bases [49]	Template-based	2018
45	Novel knowledge-based system with relation detection and textual evidence for question answering research	Sem. parsing pipe.	2018
46	Question answering over knowledge graphs: Question understanding via template decomposition	Template-based	2018
47	Svega: Answering natural language questions over knowledge base with semantic matching	Subgraph matching	2018
48	Why reinvent the wheel: Let's build question answering systems together [50]	Sem. parsing pipe.	2018
49	Answer-enhanced path-aware relation detection over knowledge base	Info. extraction	2019
50	Complex query augmentation for question answering over knowledge graphs [51]	Sem. parsing pipe.	2019
51	ComQA: Question answering over knowledge base via semantic matching	Subgraph matching	2019
52	Deep query ranking for question answering over knowledge bases	Sem. Parsing pipe.	2019
53	Handling modifiers in question answering over knowledge graphs	Sem. parsing pipe.	2019
54	Knowledge base question answering with a matching-aggregation model and question-specific contextual relations	Info. extraction	2019
55	Knowledge base question answering with attentive pooling for question representation	Info. extraction	2019
56	Learning to answer complex questions over knowledge bases with query composition	Subgraph matching	2019
57	Learning to rank query graphs for complex question answering over knowledge graphs [52]	Subgraph matching	2019
58	Message passing for complex question answering over knowledge graphs	Subgraph matching	2019
59	Pretrained transformers for simple question answering over knowledge graphs [53]	Info. extraction	2019
60	A BERT-based approach with relation-aware attention for knowledge base question answering [54]	Info. extraction	2020
61	A state-transition framework to answer complex questions over knowledge base	Subgraph matching	2020
62	Data-driven construction of SPARQL queries by approximate question graph alignment in question answering over knowledge graphs	Subgraph matching	2020
63	Exploring sequence-to-sequence models for SPARQL pattern composition [55]	Template-based	2020
64	Formal query building with query structure prediction for complex question answering over knowledge base	Info. extraction	2020
65	Improving question answering over incomplete KBs with knowledge-aware reader	Info. extraction	2020
66	Knowledge base question answering via encoding of complex query graphs	Info. extraction	2020

until 2017. The most recent significant cluster is linked to the term 'knowledge graph', showing a greater interest in this type of structure, which is very suitable for applying deep learning techniques.

Figure 7 shows the distribution of the selected articles divided by types of architecture and distributed over years.

As we can see, there is a consistent decline in the use of pipeline-based approaches. On the other hand, after an increase in subgraph matching solutions, we saw a slight drop in 2020. After a boom in 2016, the proposals for information extraction fell to a plateau still higher than the other proposals. Finally, template-based systems fluctuated to an annual maximum of two proposals in 2017 and 2018.

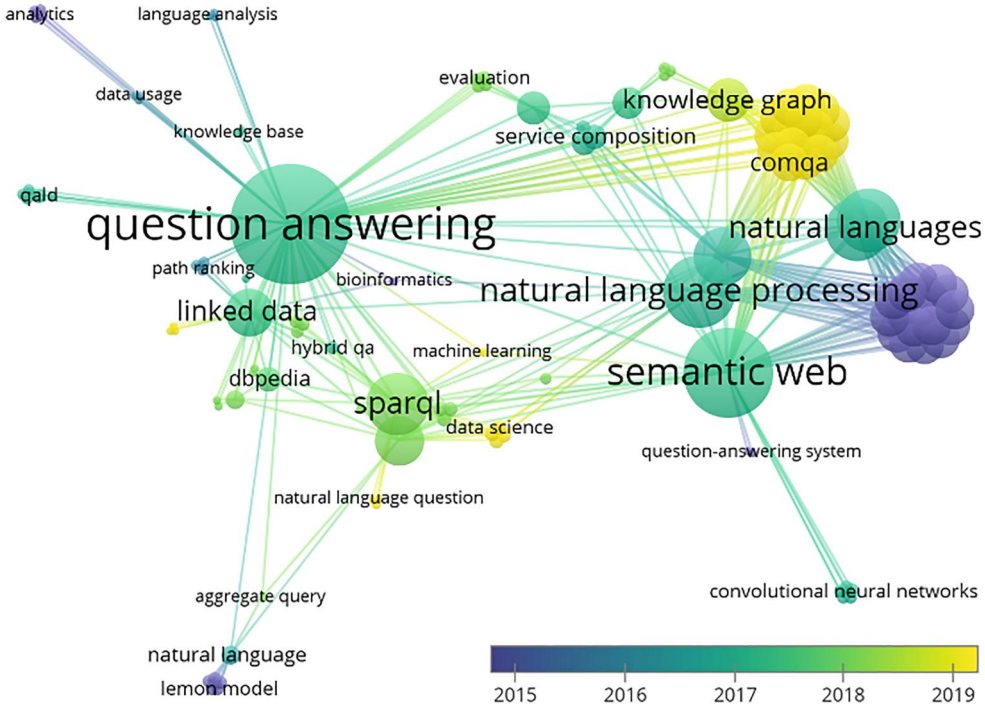

FIGURE 6 Relationships between keywords

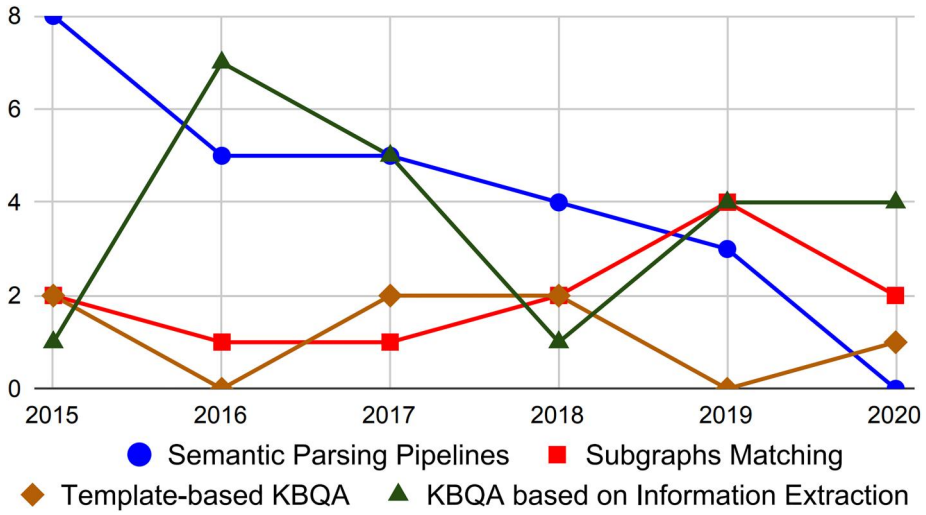

FIGURE 7 Distribution of papers by year and architecture

4 | QA OVER KNOWLEDGE BASES SOLUTIONS

4.1 | Methods (Research Question 1)

4.1.1 | Semantic parsing pipelines

Hamon et al. [8] obtain answers from linked biomedical data using a four-step method that begins with the linguistic and semantic annotation of the input question. Similarly, a simple pipeline is proposed by Lopez et al. [37] for the QuerioDALI solution. In the first step, the system performs an NER to classify named entities related to the KB domain. At the next level, an EL filter binds a unique identity to each of the entities identified in the previous step. Then, the system uses fusion and ranking of possible answers.

The lexical gap is a central issue in solving QA challenges. This refers to the mismatch between the vocabularies: on one hand, the user's vocabulary, and on the other hand, the vocabulary used in the data set of interest. The importance of the problem led to the emergence of several strategies.

Hakimov et al. [29] use a combinatorial categorical grammar with handcrafted lexical items and lambda-type calculus expressions to obtain semantic representations. In this

way, the input utterances must comply with the grammar. As is naturally emphasized by the authors, performance improves according to the lexicon size. The same conclusion is reached by Yih et al. [23], which shows that learning from labelled semantic parsers significantly improves overall performance. TR Discover, a solution by Song et al. [34], uses feature-based grammar. First, it parses an NL question to its first-order logic representation, which is, in turn, translated into SPARQL. Dubey et al. [35] also use a logical representation. Users can put queries in English to a target RDF KB. They are first normalized into an intermediary canonical syntactic form, called normalized query structure, and then translated into SPARQL queries.

The direct use of an ontology makes it possible to reduce ambiguity. Ruseti et al. [31] use the structured KB DBpedia and a Wikipedia-based approach to match phrases from the question to entities in the ontology. Different solutions for matching each type of entity were developed, and the most probable interpretation is converted in a SPARQL query. Missing properties or types can also be inferred if they were not matched in the previous step.

The query-generation (QG) process of a QA pipeline occurs after the entity and relation linking subtasks. Zafar et al. [47] start with the set of identified entities and relationships and generate walks on the KB by using the adjacent relations within one-hop distance. Valid walks are the ones containing all the starting entities. Finally, candidate walks are evaluated against the type of question, and a SPARQL query is created. To extend QG to ordinal and filter questions, Abdelkawi et al. [51] add extra constraints to the list of all possible answers.

Within the WDAqua Marie Skłodowska Curie ITN (https://github.com/WDAqua) effort to advance the field of QA, several contributions can be reported related to KBQA. Both et al. [36] start from the realization that QA systems are very complex and usually monolithic to present Qanary (https://github.com/WDAqua/Qanary), a vocabulary-driven methodology to allow decoupling of the different components and thus achieve reconfiguration and reuse. First, the Web Annotation Data Model (https://www.w3.org/TR/annotation-model/) is used to create a vocabulary covering the common abstractions related to the author's idea of a QA pipeline. In addition, the input and output of filters are described to achieve interoperability, forcing the components to have the same interface, like in a uniform pipe and filter architecture. Considering that no vocabulary can describe all existing components, the burden of creating a new description is naturally pushed to the creators of the components, which can compromise the adoption of this methodology. The problem of adapting the input and output of each module to comply with the shared vocabulary is also burdensome. Diefenbach et al. [45] present a reusable user interface to call the Qanary APIs. To allow user feedback, a way to change the descriptions of the module inputs at runtime was proposed that used timestamps to avoid conflict between Qanary annotations [43].

The idea of creating a generic (pipeline) architecture for QA on linked data to foster cooperation among developers is championed by QAestro (https://github.com/WDAqua/QAestro) [44], a proposal competing with Qanary that can be used to combine building blocks in tailored systems, allowing a semantic description of both QA components and requirements. Several important subtasks are covered, such as tokenization, POS tagging, NER, EL, dependency parsing, triple generation, data mapping, QG, and answer generation. Question type identification, answer type identification, query ranking, and syntactic parsing are also available.

Embracing the quest for component reuse, Frankenstein (https://github.com/WDAqua/Frankenstein) [48] is a platform that collects several core components to solve QA tasks and enable the creation of different QA pipelines, more precisely 380 at the time of the paper's publication. Highlighting the fact that modern QA systems rely on the flexible integration of many specialized filters, Singh et al. suggest that the construction of the pipeline could be considered an optimization problem [50], where each component could be selected from a set of options for NER and EL, relation extraction and query building. The prediction of the best-performing components facing a new NL question is tackled as a supervised learning problem in a training set of labelled questions.

The use of semantic pipelines for KBQA is the oldest and most documented approach in the literature and is preferred by authors who intend to integrate NL interfaces into their systems quickly. Reinforcing this statement is the existence of frameworks that allow decoupling of the different components used to filter the data, thus offering greater customization. It is also the easiest way for those who do not want to invest a great deal to develop more technically elaborate solutions, usually with better performance. We can investigate each filter independently because they are of interest in many other applications, not just in QA. For instance, Shen et al. [56] surveyed EL issues, techniques, and solutions. Nevertheless, this way of solving the problem seems to be reaching its maturity, and in our opinion, more important future developments will almost certainly come from other approaches.

4.1.2 | Subgraph matching

Some proposals depart very little from the classic pipeline, building the query subgraph using a semantic tree, whereas others move away sharply by constructing the subgraph step by step from a starting entity. Hu et al. [46] start by finding the semantic tree, and then after extracting the semantic relations, they build a semantic query graph. More elaborately, Yih et al. [33] propose staged query graph generation, a solution that formulates a query graph by solving a search problem. A general query subgraph is supported by the existing entities in the KB, an existential node not mappable to the KB, and a node for identifying possible aggregation functionality. The solution revolves around creating an inferential chain starting with a root entity node and using legitimate actions to grow a query graph. The first step is to find root candidates by using a lexicon to perform EL over the input query. The next step considers the lexicon again to extract the expected answer.

From relating the root entity and the kind of answer, it is possible to create a set of candidate subgraphs constrained by an aggregation function. Finally, a convolutional neural network is used to select the best candidate. For this last classification task, we can use the proposal by Gauray et al. [52], which considers a self-care mechanism that explores the intrinsic structure of subgraphs.

4.1.3 | Template-based KBQA

Zheng et al. [30] started from an initial set of NL questions and formal queries to propose a technique based on studying the similarity of graphs generated from the utterances and SPARQL queries to match the best candidate pairs to form a database with templates. Savenkov et al. [39] used external text data to explore the central topic of the question and select the best query candidates using a predefined collection of query templates. However, considering a set of manually adjusted templates is necessarily limiting, for instance, when new relations are added to the KB. To overcome this limitation, Abujabal et al. [41] proposed the QUINT system, which automatically learns role-aligned utterance-query templates from user questions paired with their answers. When QUINT answers a question, it visualizes the complete derivation sequence from the NL utterance to the final answer. The derivation explains how the syntactic structure of the question was used to derive the structure of a SPARQL query and how the phrases in the question were used to instantiate different parts of the query. When an answer seems unsatisfactory, the derivation provides valuable insights for reformulating the question. An evolution of QUINT is Never Ending QA (NEQA) [49]. NEQA automatically learns templates mapping syntactic structures to semantic ones from a small number of training question-answer pairs. Once deployed, continuous learning is triggered in cases where templates are insufficient. Using a semantic similarity function between questions and a judicious invocation of non-expert user feedback, NEQA learns new templates that capture previously-unseen syntactic structures, gradually extending its template repository.

We note that the literature offers few proposals for this type of system, which strikes us as quite strange because it allows answers to a wide range of questions. Investing in research to create wider lexicons to be used in the production of templates promises the creation of systems with even higher performance regarding complex questions. However, it seems that the research effort is shifting to end-to-end systems, which we will talk about in the next subsection.

4.1.4 | KBQA based on information extraction

Several KBQA solutions using some form of a deep neural network have been reported. Dong et al. [32] introduced a multicolumn convolutional neural network to understand questions from three different aspects, answer path, answer context, and answer type, and learn their distributed representations. Meanwhile, the system enables us to jointly learn low-dimensional embeddings of entities and relations in the KB. This approach can be expanded and enriched if we consider more dimensions to convert into vector representations. Xu et al. [38] present a neural network-based relation extractor to retrieve the candidate answers from Freebase and then infer from Wikipedia to validate these answers. More precisely, the process involves dividing the original question into subquestions by applying a set of syntactic patterns. Then, for each subquestion, EL and relation extraction is performed and refined by a joint inference model. After retrieving a set of candidate answers, the final solution is obtained by inference on Wikipedia, searching on the page of the topic entity for evidence about candidate answers.

The model proposed by Lukovnikov et al. [9] learns to rank subject–predicate pairs to enable the retrieval of relevant facts given a question. The network contains a nested word and character-level question encoder that allows the handling of new and rare words without compromising the exploitation of word-level semantics. This neural network approach generates a single process solution that avoids complex NLP pipeline constructions and error propagation, and it can be retrained or reused for different domains. In scenarios where training data is limited, overfitting compromises network performance. To tackle this problem, instead of using a bidirectional long short-term memory (LSTM) network to create the language representation model, Lukovnikov et al. [53], Luo et al. [54], and Panchbhai et al. [55] independently evaluated the use of Bidirectional Encoder Representations from Transformers (BERT) [57], the current most performant solution for NL understanding tasks.

Hao et al. [40] present a model to represent the questions and their corresponding scores dynamically according to the various candidate answer aspects via the cross-attention mechanism. In addition, they leverage the global knowledge inside the underlying KB, aiming to integrate this information into the representation of the answers. As a result, it could alleviate the out-of-vocabulary problem, which helps the cross-attention model to represent the question more precisely.

Relation detection is essential to extract candidate answer triples. Yu et al. [42] use deep residual bidirectional LSTM networks to compare questions and relation names considering different abstraction hierarchies. This relation detector integrates EL for mutual enhancement, similar to the joint inference feature of Xu et al. [38].

The creation of models to generate vector representations of features of interest from KB allows avoiding the use of semantic pipelines. As there are multiple architectures of deep neural networks and varied ways of digesting the information to be processed, the literature already reports several possibilities, and many more will appear shortly. LSTMs with attention have great room for further development. On the other hand, we noticed that transfer learning using pretrained models is still underrepresented in new system implementations. Finally, the arrival of new and better-performing models allows better results but at computational costs that are not always bearable.

TABLE 3 Question answering over knowledge bases challenges and solutions

Challenges	Solutions
Answering simple questions	BERT transformer (59). CNN (29). Formal logic (13, 14). Manual templates (6). Seq2Seq (15, 16, 21, 35). Simple pipeline (23, 37)
Answering complex questions	Hybrid system (3, 5). More SPARQL modifiers compliance (53). N-tuple assertions (1). Query ranking (57). Siamese CNNs (17). Simple queries composition (56). Subgraph matching (10, 51, 61). Templates (46, 63). Unsupervised message passing (58)
Entity linking	BERT transformer (60). Distant supervision (7). Joint entity and relation linking (41)
KB incompleteness	Hybrid system (19, 38, 65)
Modular design, module reusability	Integration framework (22, 36). Modules collection (43). Optimal module selection (48)
Relation linking	BERT transformer (60) Hierarchical RNN (30). Joint entity and RL (41). LSTM (45). Siamese LSTM (49, 55)
System tunings	User interaction (31). User-interface (39). Query builder module (42, 50)
Training data scarcity	Automatic labelling (25, 28). Distant supervision (24, 26). Multicolumn CNN (8)

Abbreviations: BERT, Bidirectional Encoder Representations from Transformers; CNN, convolutional neural network; KB, knowledge base; LSTM, long short-term memory; RL, relation linking; RNN, recurrent neural network; SPARQL, SPARQL Protocol and RDF Query Language.

TABLE 4 Remaining challenges, future work

Challenges. Future work	Research directions
Answering complex questions (8, 15, 10, 16, 19, 21, 25, 42, 45, 53, 54, 57, 59, 64)	Conversational agent (21). Data augmentation (57). More SPARQL Protocol and RDF Query Language modifiers compliance (53). More training data (8, 10). Reinforcement learning (64)
Knowledge base incompleteness (26, 35, 49, 55, 60)	Hybrid system (26). More external knowledge, more training data (35, 49, 55, 60)

4.2 | Challenges (Research Question 2)

Several obstacles have prevented the full adoption of KBQA systems. Table 3 presents a summary of the challenges KBQA has faced.

The preferred technique for solving simple questions is sequence-to-sequence translators using neural networks. An encoder converts the NL question to a vector representation, and then a decoder outputs a query in a formal language. It is also possible to extract features by processing convolutions. There is also a paper reporting using BERT linguistic model, but using a transformer is clearly unsuitable because of the high computational cost.

Research on this topic is marginal, and as such, we are led to believe that it has been successfully solved.

Research on complex questions is richer in the proposals, starting with systems that propose adding support to another set of SPARQL modifiers. More sophisticated techniques such as the generation of templates or the use of subgraphs are also on the agenda. The use of the information extraction approach using some neural model is common practice. We also find hybrid systems that use KB data and free text. This technique is also used to mitigate KB incompleteness. The renewed interest in both topics indicates that these challenges are not closed. Entity and relation linking are unsolved issues, although the joint entity and relation linking approach shows promise. Automatic labelling and distant supervision usually help in obtaining more training data.

In general, almost all papers promise to tune their proposals for better performance. However, two major problems remain open, as presented in Table 4.

Future work to tackle the answers to complex questions revolves around exploring solutions that allow real-time feedback to the system, such as implementing a conversational agent or shifting to reinforcement learning so that new knowledge adds can be continuous. On the other hand, KB incompleteness also limits these systems' usability. Hybrid systems that use free text to address this problem have been explored, but there is still a long way to go. We need more training data and more external knowledge.

5 | CONCLUSION

This systematic study collected information on methods and challenges of QA over KBs, a topic that has gained traction in the search engine industry. We analysed 66 documents to classify KBQA systems according to their architectural styles. We reported 25 semantic parsing pipeline systems, 12 using subgraph matching, 7 based on templates, and 22 performing information extraction. We look at the challenges ahead and identify some directions for future research. Two primary challenges remain that are particularly sensitive to the success of this technology. On the one hand, it is necessary to answer increasingly complex questions, and on the other hand, we need to deal with the natural incompleteness of KBs. Our

study concluded that hybrid systems and the adoption of recent advanced machine learning techniques promise significant advances in the field.

ACKNOWLEDGEMENTS

This publication was funded by the National Funds through the FCT, in the context of the project UIDB/00127/2020. FCT—Portuguese Foundation for Science and Technology supports Arnaldo Pereira (Ph.D. Grant PD/BD/142877/2018). Fundação para a Ciência e a Tecnologia. Grant Number: PD/BD/142877/2018.

CONFLICT OF INTEREST

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

ORCID

Arnaldo Pereira https://orcid.org/0000-0003-3361-6269
Alina Trifan https://orcid.org/0000-0001-7613-1435
Rui Pedro Lopes https://orcid.org/0000-0002-9170-5078
José Luís Oliveira https://orcid.org/0000-0002-6672-6176

REFERENCES

1. Green, B.F., et al.: Baseball: an automatic question-answerer. In: Western Joint IRE-AIEE-ACM Computer Conference, IRE-AIEE-ACM. vol. 19, pp. 219–224. (1961)
2. Androutsopoulos, I., Ritchie, G.D., Thanisch, P.: Natural language interfaces to databases - an introduction. Nat. Lang. Eng. 1(1), 29–81 (1995)
3. Hirschman, L., Gaizauskas, R.: Natural language question answering: the view from here. Nat. Lang. Eng. 7(04), 275–300 (2001)
4. Berners-Lee, T., Hendler, J., Lassila, O.: The semantic web. Sci. Am. 284(5), 34–43 (2001)
5. Guha, R., Rob M.C., Eric, M.: Semantic search. In: Proceedings of the 12th International Conference on World Wide Web, pp. 700–709. (2003)
6. Pereira, A., et al.: SCALEUS-FD: a FAIR data tool for biomedical applications. BioMed. Res. Int. 2020, 1–8 (2020). Article ID 3041498
7. Asiaee, A.H., et al.: A framework for ontology-based question answering with application to parasite immunology. J. Biomed. Semant. 6, 25 (2015)
8. Hamon, T., Grabar, N., Mougin, F.: Querying biomedical Linked data with natural language questions. Semant. Web. 8(4), 581–599 (2017)
9. Lukovnikov, D., et al.: Neural network-based question answering over knowledge graphs on word and character level. In: Proceedings of the 26th International Conference on World Wide Web, pp. 1211–1220. (2017)
10. Höffner, K., et al.: Survey on challenges of question answering in the semantic web. Semant. Web. 8(6), 895–920 (2017)
11. Mishra, A., Jain, S.K.: A survey on question answering systems with classification. J. King Saud Univ. Comp. & Info. Sci. 28(3), 345–361 (2016)
12. Dimitrakis, E., Sgontzos, K., Tzitzikas, Y.: A survey on question answering systems over linked data and documents. J. Intell. Inf. Syst. 55, 233–259 (2019)
13. Affolter, K., Stockinger, K., Bernstein, A.: A comparative survey of recent natural language interfaces for databases. VLDB J. 28(5), 793–819 (2019)
14. Ojokoh, B., Adebisi, E.: A review of question answering systems. J. Web. Eng. 17(8), 717–758 (2019)
15. Schreiber, G., Raimond, Y.: RDF 1.1 Primer. W3C working group note. W3C (Jun. 2014)
16. Cyganiak, R., Wood, D., Lanthaler, M.: RDF 1.1 Concepts and abstract syntax. W3C Recommendation. W3C (Feb. 2014)
17. W3C SPARQL Working Group: SPARQL 1.1 overview. W3C Recommendation (Mar. 2013)
18. Harris, S., Seaborne, A.: SPARQL 1.1 Query Language. W3C Recommendation. W3C (Mar. 2013)
19. Usbeck, R., et al.: 9th challenge on question answering over linked data (QALD-9). In: CEUR Workshop Proceedings. vol. 2241, pp. 58–64. (2018)
20. Dubey, M., et al.: LC-QuAD 2.0: a large dataset for complex question answering over Wikidata and DBpedia. In: 18th International Semantic Web Conference, ISWC 2019, pp. 69–78. (2019)
21. Cai, Q., Yates, A.: Large-scale semantic parsing via schema matching and lexicon extension. In: Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics. vo. 1, pp. 423–433. Long Papers (2013)
22. Berant, J., et al.: Semantic parsing on Freebase from question-answer pairs In: Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, pp. 1533–1544. (2013)
23. Yih, W.T., et al.: The value of semantic parse labelling for knowledge base question answering. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, pp. 201–206. (2016)
24. Bordes, A., et al.: Large-scale simple question answering with memory networks. CoRR, vol. abs/1506.02075 (2015)
25. Tsatsaronis, G., et al.: An overview of the BIOASQ large-scale biomedical semantic indexing and question answering competition. BMC Bioinf. 16, 138 (2015)
26. Kitchenham, B.A., Dybå, T., Jørgensen, M.: Evidence-based software engineering. In: Proceedings of the 26th International Conference on Software Engineering, 273–281 (2004)
27. Moher, D., et al.: Preferred reporting items for systematic reviews and meta-analyses: the PRISMA statement. BMJ. 339(7716), b2535 (2009)
28. Thabane, L., et al.: Posing the research question: not so simple. Can. J. Anesth. 56(1), 71–79 (2009)
29. Hakimov, S., et al.: Applying semantic parsing to question answering over linked data: Addressing the lexical gap. In: Natural Language Processing and Information Systems, Nldb, vol. 9103, Springer-Verlag Berlin, Berlin (2015)
30. Zheng, W.G., et al.: How to build templates for RDF question/answering-an uncertain graph similarity join approach, pp. 1809–1824. Assoc Computing Machinery, New York (2015)
31. Ruseti, S., et al.: QAnswer - Enhanced entity matching for question answering over linked data, vol. 1391. CEUR (2015)
32. Dong, L., et al.: Question answering over Freebase with multi-column convolutional neural networks. In: Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing. vol. 1, pp. 260–269. (2015)
33. Yih, W.T., et al.: Semantic parsing via staged query graph generation: question answering with knowledge base, pp. 1321–1331.Assoc Computational Linguistics-Acl, Stroudsburg (2015)
34. Song, D.Z., et al.: TR Discover: A natural language interface for querying and analysing Interlinked datasets. In: Semantic Web - Iswc 2015, Pt Ii, vol. 9367. Springer International Publishing Ag, Cham (2015)
35. Dubey, M., et al.: AskNow: a framework for natural language query formalization in SPARQL Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), vol. 9678, pp. 300–316. (2016)
36. Both, A., et al.: Qanary - a methodology for vocabulary-driven open question answering systems. In Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), vol. 9678, pp. 625–641. (2016)
37. Lopez, V., et al.: QuerioDALI: question answering over dynamic and linked knowledge graphs. In: Semantic Web - Iswc 2016, Pt Ii, vol. 9982. Springer Int Publishing Ag, Cham (2016)
38. Xu, K., et al.: Question answering on freebase via relation extraction and textual evidence, pp. 2326–2336. Assoc Computational Linguistics-Acl, Stroudsburg (2016)
39. Savenkov, D., Agichtein, E.: When a knowledge base is not enough: question answering over knowledge bases with external text data, pp. 235–244. Assoc Computing Machinery, New York (2016)

40. Hao, Y., et al.: An end-to-end model for question answering over knowledge base with cross-attention combining global knowledge, pp. 221–231. Assoc Computational Linguistics-Acl, Vancouver (2017)
41. Abujabal, A., et al.: Automated template generation for question answering over knowledge graphs, pp. 1191–1200. Assoc Computing Machinery, New York (2017)
42. Yu, M., et al.: Improved neural relation detection for knowledge base question answering, pp. 571–581. Assoc Computational Linguistics-Acl, Stroudsburg (2017)
43. Diefenbach, D., et al.: Introducing feedback in qanary: how users can interact with QA systems. In: Lecture Notes in Computer Science, vol. 10577 LNCS, pp. 81–86. Springer Verlag (2017)
44. Singh, K., et al.: Qaestro - semantic-based composition of question answering pipelines. In Database and Expert Systems Applications, Dexa 2017, Pt I, vol. 10438. Springer International Publishing Ag, Cham (2017)
45. Diefenbach, D., et al.: Trill: a reusable front-end for QA systems. In Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), vol. 10577 LNCS, pp. 48–53. (2017)
46. Hu, S., et al.: Answering natural language questions by subgraph matching over knowledge graphs. IEEE Trans. Knowl. Data Eng. 30(5), 824–837 (2018)
47. Zafar, H., Napolitano, G., Lehmann, J.: Formal query generation for question answering over knowledge bases. In: Lecture Notes in Computer Science, vol. 10843 LNCS, pp. 714–728. Springer Verlag (2018)
48. Singh, K., et al.: Frankenstein: a platform enabling reuse of question answering components. In: Lecture Notes in Computer Science, vol. 10843 LNCS, pp. 624–638. Springer Verlag (2018)
49. Abujabal, A., et al.: Never-ending learning for open-domain question answering over knowledge bases, pp. 1053–1062. Assoc Computing Machinery, New York (2018)
50. Singh, K., et al.: Why reinvent the wheel - let's build question answering systems together, pp. 1247–1256. Assoc Computing Machinery, New York (2018)
51. Abdelkawi, A., et al.: Complex query augmentation for question answering over knowledge graphs. In: Lecture Notes in Computer Science, vol. 11877 LNCS, pp. 571–587. Springer (2019)
52. Maheshwari, G., et al.: Learning to rank query graphs for complex question answering over knowledge graphs. In: Lecture Notes on Computer Science, vol. 11778 LNCS, pp. 487–504. Springer (2019)
53. Lukovnikov, D., Fischer, A., Lehmann, J.: Pretrained transformers for simple question answering over knowledge graphs. In: Lecture Notes on Computer Science. 11778 LNCS, pp. 470–486. Springer (2019)
54. Luo, D., Su, J., Yu, S.: A BERT-based approach with relation-aware attention for knowledge base question answering. In 2020 International Joint Conference on Neural Networks (IJCNN), pp. 1–8. Glasgow (2020)
55. Panchbhai, A., Soru, T., Marx, E.: Exploring sequence-to-sequence models for SPARQL pattern composition. In: Villazón-Terrazas, B. et al.: (eds.) Knowledge Graphs and Semantic Web. KGSWC 2020. Communications in Computer and Information Science, vol. 1232, pp. 158–165. Springer, Cham (2020)
56. Shen, W., Wang, J., Han, J.: Entity linking with a knowledge base: issues, techniques, and solutions. IEEE Trans. Knowl. Data Eng. 27(2), 443–460 (2015)
57. Devlin, J., et al.: Pre-training of deep bidirectional transformers for language understanding. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. (Long and Short Papers), vol. 1, pp. 4171–4186. (2019)

How to cite this article: Pereira A, Trifan A, Lopes RP, Oliveira JL. Systematic review of question answering over knowledge bases. *IET Soft*. 2021;1–13. https://doi.org/10.1049/sfw2.12028