Complex Knowledge Base Question Answering: A Survey

Yunshi Lan*, Gaole He*, Jinhao Jiang, Jing Jiang, Wayne Xin Zhao *Member, IEEE*, and Ji-Rong Wen *Senior Member, IEEE*,

Abstract— Knowledge base question answering (KBQA) aims to answer a question over a knowledge base (KB). Early studies mainly focused on answering simple questions over KBs and achieved great success. However, their performance on complex questions are still far from satisfaction. Therefore, in recent years, researchers propose a large number of novel methods, which looked into the challenges of answering complex questions. In this survey, we review recent advances on KBQA with the focus on solving complex questions, which usually contain multiple subjects, express compound relations, or involve numerical operations. In detail, we begin with introducing the complex KBQA task and relevant background. Then, we present two mainstream categories of methods for complex KBQA, namely semantic parsing-based (SP-based) methods and information retrieval-based (IR-based) methods. Specifically, we illustrate their procedures with flow designs and discuss their difference and similarity. Next, we summarize the challenges that these two categories of methods encounter when answering complex questions, and explicate advanced solutions as well as techniques used in existing work. After that, we discuss the potential impact of pre-trained language models (PLMs) on complex KBQA. To help readers catch up with SOTA methods, we also provide a comprehensive evaluation and resource about complex KBQA task. Finally, we conclude and discuss several promising directions related to complex KBQA for future research.

Index Terms—Knowledge base question answering, knowledge base, question answering, natural language processing, survey.

1 Introduction

Knowledge base (KB) is a structured database that contains a collection of facts (alias triples) in the form (*subject*, *relation*, *object*). Large-scale KBs, such as Freebase [1], DBPedia [2], Wikidata [3], and YAGO [4], have been constructed to serve many downstream tasks. Among them, knowledge base question answering (KBQA) is a task that aims to answer natural language questions with KBs acting as its knowledge source. Nowadays, KBQA has attracted intensive attention from researchers as it plays an important role in many intelligent applications. For example, Amazon Alexa, Apple Siri, and Microsoft Cortana are integrated with the function of answering factoid questions from users. Chatbots such as Microsoft Xiaoice and Zo have also demonstrated a high degree of conversational capability, where factoid question answering frequently occurs [5], [6].

Early work on KBQA focused on answering a simple question, where only a single fact is involved. For example, "*Who was the nominee of The Jeff Probst Show?*" is a simple question which includes the subject "*The Jeff Probst Show*", the relation "*nominee*" and queries about the object entity "*Jeff Probst*" of fact "*(The Jeff Probst Show, nominee, Jeff Probst)*" in KBs. It is not trivial to retrieve the correct entity from the large-scale KBs, which consists of millions or even billions of facts. Therefore, researchers have spent much effort in proposing different models to answer simple questions over KBs [7], [8], [9], [10], [11].

Recently, researchers started paying more attention to answering ***complex questions*** over KBs, *i.e.*, the complex KBQA task [12], [13]. Complex questions usually contain

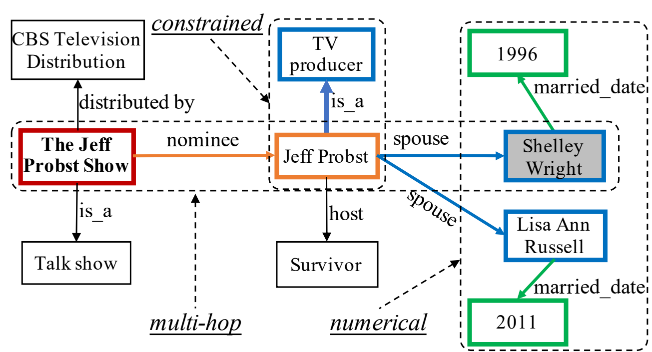

FIGURE 1: An example of complex KBQA for the question "*Who is the first wife of TV producer that was nominated for The Jeff Probst Show?*". We present the related KB subgraph for this question. The ground truth path heading to the answer is annotated with colored borders. The topic entity and the answer entity are shown in the bold font and shaded box respectively. "multi-hop" reasoning, "constrained" relations, and "numerical" operation are highlighted in black dotted box. We use different colors to indicate different reasoning hops to reach each entity from the topic entity.

- * Y. Lan and G. He contribute equally to this work.
- Yunshi Lan is with School of Data Science and Engineering, East China Normal University. E-mail: yslan@dase.ecnu.edu.cn
- Jing Jiang is with School of Computing and Information System, Singapore Management University. E-mail: jingjiang@smu.edu.sg.
- G.He is with the School of Information, Renmin University of China, and Beijing Key Laboratory of Big Data Management and Analysis Methods. E-mail: hegaole@ruc.edu.cn
- W.X. Zhao (corresponding author) , Jinhao Jiang and J. Wen are with Gaoling School of Artificial Intelligence, Renmin University of China, and Beijing Key Laboratory of Big Data Management and Analysis Methods. E-mail: batmanfly@gmail.com.

Manuscript revised xxx.

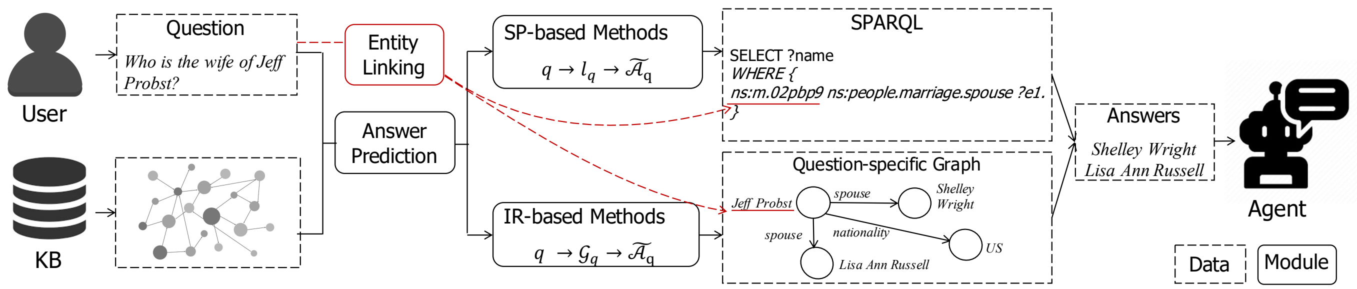

FIGURE 2: Architecture of KBQA systems. The entity linking procedure is shown in red color.

multiple subjects, express compound relations, or include numerical operations. Take the question in Figure 1 as an example. This example question starts with the subject "*The Jeff Probst Show*". Instead of querying a single fact, the question requires the composition of two relations, namely, "*nominee*" and "*spouse*". This query is also associated with an entity type constraint "*(Jeff Probst, is a, TV producer)*". The final answer should be further aggregated by selecting the possible candidates with the earliest marriage date. Generally, complex questions are questions involving ***multi-hop reasoning***, ***constrained relations*** or ***numerical operations***.

Tracing back to the solutions for simple KBQA task, a number of studies from two mainstream approaches have been proposed. We show the overall architecture of simple KBQA systems in Figure 2. These two approaches first recognize the subject in a question and link it to an entity in the KB (referred to as the ***topic entity***). Then they derive the answers within the neighborhood of the topic entity by either executing a parsed logic form or reasoning in a question-specific graph extracted from the KB. The two categories of methods are commonly known as ***semantic parsing-based*** **(SP-based)** methods and ***information retrieval-based*** **(IR-based)** methods in prior work [7], [8], [11], [14]. They design different working mechanisms to solve the KBQA task. The former approach represents a question by a symbolic logic form, and then executes it against the KB to obtain the final answers. The latter approach constructs a question-specific graph delivering the comprehensive information related to the question, and generates the final answers with the help of the extracted graph.

However, when applying the two mainstream approaches to the complex KBQA task, complex questions bring in challenges on different parts of the approaches. We identify the main challenges as follows:

- Parsers used in existing SP-based methods are difficult to cover diverse complex queries (*e.g.*, multi-hop reasoning, constrained relations, and numerical operations). Similarly, previous IR-based methods may fail to answer a complex query, as their reasoning is performed over small-scale subgragh without traceable reasoning To cover as many complex questions as possible, the development of an expressive logic form for SP-based methods and strong reasoning ability for IR-based methods is highly demanded.
- More relations and subjects in complex questions indicate a larger search space of potential logic forms for parsing, which will dramatically increase the computational cost. Meanwhile, more relations and subjects prevent IR-based methods from retrieving all relevant facts for reasoning, which makes the common incomplete KB issue become severer.
- When questions become complicated from both semantic and syntactic aspects, models are required to have strong capabilities of natural language understanding and generalization. Comparing the question "*Who is the first wife of TV producer that was nominated for The Jeff Probst Show?*" with another question "*Who is the wife of the first TV producer that was nominated for The Jeff Probst Show?*", the models are supposed to understand that the ordinal number "*first*" is used to constrain "*wife*" or the phrase "*TV producer*".
- It is expensive to label the ground truth paths heading to the answers (see the annotated ground truth path in Figure 1) for KBQA task. Generally, only question-answer pairs are provided. This indicates SP-based methods and IR-based methods have to be trained without the annotation of correct logic forms and reasoning paths. Such weak supervision signals bring difficulties to both approaches due to the lack of necessary guidance during the intermediate reasoning process.

Regarding the related surveys, we observe Wu *et al.* [15] and Chakraborty *et al.* [16] reviewed the existing work on simple KBQA. Furthermore, Fu *et al.* [17] investigated the current advances on complex KBQA. They provided a general view of advanced methods only from the perspective of techniques and more focused on application scenarios in e-commerce domain. Different from these surveys, our work tries to identify the challenges encountered in previous studies, and extensively discuss existing solutions in a comprehensive and well-organized manner. Specifically, we categorize the methods for complex KBQA into two mainstream approaches based on their working mechanisms. We decompose the overall procedure of the two approaches into a series of functional modules, and analyze the challenges in each module. Such a way is particularly helpful for readers to understand the potential challenges and solutions for complex KBQA. It is worth noting that this survey is an extended version of the short survey [18]. As a comparison, this survey has several main differences: (1) We add more recent-published papers and refine the description of challenges as well as solutions with a fine-grained taxonomy. (2) We provide deep discussions of the two mainstream categories including a comprehensive comparison of their core modules and a unified point view of symbolic reasoning. (3) We add a new section to discuss the role of cutting-edge pre-trained language techniques for complex KBQA and give a more thorough outlook on several promising research directions. (4) We attach a companion page with all methods, chronological information, open resources and reported results to help readers quickly catch up with the development of this task. (5) We introduce traditional approaches, preliminary knowledge, and evaluation protocol of KBQA in multiple aspects and a more concrete way, which goes far beyond the scope of the short survey.

The remainder of this survey is organized as follows. We first introduce the task formulation and preliminary knowledge about the task in Section 2. Next, we introduce the two mainstream categories of methods for complex KBQA followed by a systematic comparison and a unified view of them in Section 3. Following the categorization, we figure out typical challenges and solutions for SP-based and IR-based methods in Section 4 and Section 5, respectively. In Section 6, we highlight the impact of pre-trianed language models on complex KBQA. After that, we summarize evaluation protocol, leaderboard and relevant resource of complex KBQA methods in Section 7. Then, we discuss several recent trends in Section 8. Finally, we conclude and summarize the contribution of this survey in Section 9.

2 PRELIMINARY

In this section, we first briefly introduce KBs and task formulation of KBQA, then we talk about the traditional approaches for KBQA systems.

2.1 Knowledge Base

As mentioned earlier, KB is usually in the format of triples. They are designed to support modeling relationships between entities. Take Freebase [19] as an example for KB. Each entity in Freebase has a unique ID (refereed to as ***mid***), one or more types, and uses properties from these types in order to provide facts [3]. For example, the Freebase entity for person Jeff Probst has the mid "$m.02pbp9$"[1] and the type "$people.marriage$" that allows the entity to have a fact with "$people.marriage.spouse$" as the property and "$m.0j6d0bg$" (psychotherapist Shelley Wright) as the value. Freebase incorporates compound value types (CVTs) to represent n-ary ($n > 2$) relational facts [1] like "*Jeff Probst was married to Shelley Wright in 1996*", where three entities, namely "*Jeff Probst*", "*Shelley Wright*", and "*1996*", are involved in a single statement. Different from entities which can be aligned with real world objects or concepts, CVTs are artificially created for such n-ary facts.

In practice, large-scale open KBs (*e.g.,* Freebase and DBPedia) are published under Resource Description Framework (RDF) to support structured query language [3], [20]. To facilitate access to large-scale KBs, the query language SPARQL is frequently used to retrieve and manipulate data stored in KBs [3]. In Figure 2, we have shown an executable SPARQL to obtain the spouses of entity "*Jeff Probst*".

Different KBs are designed with different purposes, and have varying properties under different schema design. For example, Freebase is created mainly by community members and harvested from many resources including Wikipedia. YAGO [21] takes Wikipedia and WordNet [22] as the knowledge resources and covers taxonomy of more general concepts. WikiData [3] is a multilingual KB which integrates multiple resources of KBs with high coverage and quality. A more comprehensive comparison between open KBs is available at [23].

1. PREFIX: $< http://rdf.freebase.com/ns/ >$

2.2 Task Formulation

Formally, we denote a KB as $\mathcal{G} = \{\langle e, r, e' \rangle | e, e' \in \mathcal{E}, r \in \mathcal{R}\}$, where $\langle e, r, e' \rangle$ denotes that relation r exists between subject e and object e', $\mathcal{E}$ and $\mathcal{R}$ denote the entity set and relation set, respectively.

Given the available KB $\mathcal{G}$, this task aims to answer natural language questions $q = \{w_1, w_2, ..., w_m\}$ in the format of a sequence of tokens (typically organized with a unique vocabulary $\mathcal{V}$) and we denote the predicted answers as $\tilde{\mathcal{A}}_q$. Specially, existing studies assume the correct answers $\mathcal{A}_q$ can be derived from the entity set $\mathcal{E}$ of the KB or a natural language sequence (*i.e.,* the surface name of entities). Unlike answers of simple KBQA task, which are entities directly connected to the topic entity, the answers of the complex KBQA task are entities multiple hops away from the topic entities or even the aggregation of them. Generally, a KBQA system is trained using a dataset $\mathcal{D} = \{(q, \mathcal{A}_q)\}$.

2.3 Traditional Approaches

General KBQA systems for simple questions have a pipeline framework as displayed in Figure 2. The preliminary step is to identify topic entity e_q of the question q, which aims at linking a question to its related entities in the KBs. In this step, named entity recognition, disambiguation and linking are performed. It is usually done using some off-the-shelf entity linking tools, such as S-MART [24], DBpedia Spotlight [25], and AIDA [26]. Subsequently, an answer prediction module is leveraged to predict the answer $\tilde{\mathcal{A}}_q$ taking q as the input.

For simple KBQA task, the predicted answers are usually located within the neighborhood of the topic entities. Different features, as well as methods, are proposed to rank these candidate entities. Early attempts on solving simple KBQA task employed existing semantic parsing tools to parse a simple natural language question into an uninstantiated logic form, and then adapted it to KB schema by aligning the lexicons. This step results in an executable logic form l_q for q. In detail, the existing semantic parsing tools usually follow Combinatory Categorical Grammars (CCGs) [27], [28], [29] to build domain-independent logic forms. Then different methods [27], [28], [29], [30], [31] are proposed to perform schema matching and lexicon extension, which results in logic forms grounded with KB schema. For simple KBQA task, this logic form is usually a single triple starting from the topic entity and connecting to the answer entities. As early methods heavily rely on rule-based mapping, which is hard to be generalized to large-scale datasets [32], [33], [34]. Follow-up work proposed some scoring functions to automatically learn the lexicon coverage between the logic forms and the questions [35], [36]. With the development of deep learning, several advanced neural networks such as Convolutional Neural Network [37], Hierarchical Residual BiLSTM [9], Match-Aggregation Module [38], and Neural

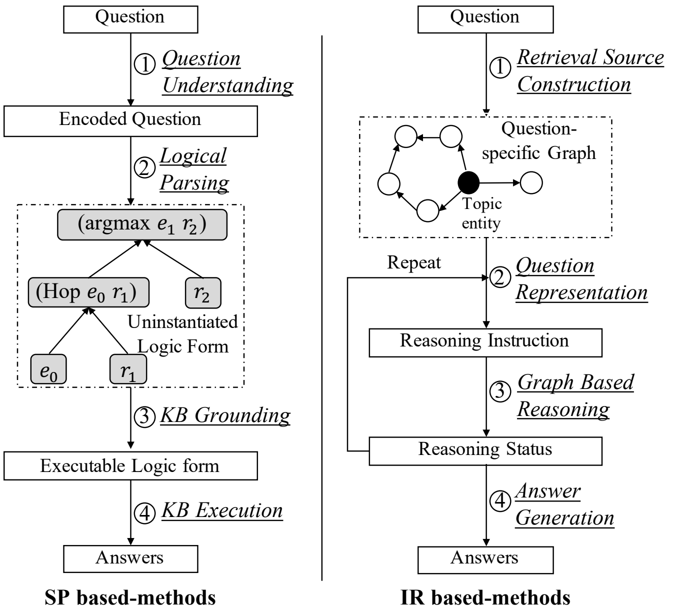

FIGURE 3: Illustration of two mainstream approaches for complex KBQA.

Module Network [39] are utilized to measure the semantic similarities. This line of work is known as semantic parsing-based methods.

Information retrieval-based methods were also developed over the decades. They retrieve a question-specific graph $\mathcal{G}_q$ from the entire KB. Generally, entities one hop away from the topic entity and their connected relations form the subgraph for solving a simple question. The question and candidate answers in the subgraph can be represented as low-dimensional dense vectors. Different ranking functions are proposed to rank these candidate answers and top-ranked entities are considered as the predicted answers [7], [10], [40], [41]. Afterwards, Memory Network [42] is employed to generate the final answer entities [43], [44]. More recent work [8], [45], [46] employs attention mechanism or multi-column modules to this framework to boost the ranking accuracy. In Figure 2, we have displayed different pipelines and intermediate outputs of the two methods.

3 Two Mainstream Approaches

Complex KBQA systems follow the same overall architecture as shown in Figure 2, where the entity linking is first performed. Subsequently, as introduced in Section 1, SP-based and IR-based methods are two mainstream approaches to answering complex questions. SP-based methods parse a question into a logic form and execute it against KBs for finding the answers. IR-based methods retrieve a question-specific graph and apply some ranking algorithms to select entities from top positions or directly generate answers with text decoder. To summarize, the two approaches follow either a *parse-then-execute* paradigm or a *retrieve-and-generate* paradigm. To show the difference between the two paradigms, we illustrate their question answering procedures with detailed modules in Figure 3.

3.1 Semantic Parsing-based Methods

As shown in Figure 3, we summarize the procedure of SP-based methods into the following four modules:

(1) They understand a question via a *question understanding* module, which is to conduct the semantic and syntactic analysis and obtain an encoded question for the subsequent parsing step. We denote this module as follows:

$$\tilde{q} = Question\_Understanding(q),$$

where $\tilde{q}$ is the encoded question that captures semantic and syntactic information of the natural language question. It can be distributed representation, structural representation or their combination. Intuitively, neural networks (*e.g.*, LSTM [47], GRU [48], and PLMs) are employed to act as the question understanding module and obtain hidden states to represent the question. Meanwhile, some syntactic parsing is performed to extract structural properties of the question.

(2) A *logical parsing* module is utilized to transfer the encoded question into an uninstantiated logic form:

$$\bar{l}_q = Logical\_Parsing(\tilde{q}),$$

where $\bar{l}_q$ is the uninstantiated logic form without the detailed entities and relations filled in. The grammar and constituents of logic forms can be different with specific designs of a system. Here, $\bar{l}_q$ can be obtained by either generating a sequence of tokens or ranking a set of candidates. In practice, Seq2seq models or featured based ranking models are employed to generate $\bar{l}_q$ based on the encoded question.

(3) To execute against KBs, the logic form is further instantiated and validated by conducting some semantic alignments to structured KBs via *KB grounding*. Note that, in some work [35], [49], the logical parsing and KB grounding are simultaneously performed, where logic forms are validated in KBs while partially parsed:

$$l_q = KB\_Grounding(\bar{l}_q, \mathcal{G}).$$

After this step, $\bar{l}_q$ is instantiated with the entities and relations in $\mathcal{G}$ so that we obtain an executable logic form l_q. It is worth noting that l_q always contains e_q, which are detected via entity linking module. Its format is not restricted to the SPARQL query but always transferable to SPARQL.

(4) Eventually, the parsed logic form is executed against KBs to generate predicted answers via a *KB execution* module:

$$\tilde{\mathcal{A}}_q = KB\_Execution(l_q),$$

where $\tilde{\mathcal{A}}_q$ is the predicted answers for the given question q. This module is usually implemented via an existing executor.

During training, the logic form l_q is treated as the intermediate output. The methods are trained using the KBQA datasets in the format of $\mathcal{D} = \{(q, \mathcal{A}_q)\}$, where the objective is set to generate a logic form matching the semantics of the question and resulting in correct answers.

3.2 Information Retrieval-based Methods

Similarly, we summarize the procedure of IR-based methods into four modules as illustrated in Figure 3:

(1) Starting from the topic entity e_q, the system first extracts a question-specific graph from KBs. Ideally, this graph includes all question-related entities and relations as nodes and edges respectively. Without explicitly generating an executable logic form, IR-based methods perform reasoning over the graph. We represent a retrieval source construction module taking as input of both the question and KB as:

$$\mathcal{G}_q = \textit{Retrieval\_Source\_Construction}(q, \mathcal{G}),$$

where $\mathcal{G}_q$ is the question-specific graph extracted from $\mathcal{G}$. As the size of subgraph grow exponentially with the distance to topic entities, some filtering tricks (*e.g.*, personalized Pagerank) are adopted to keep the graph size in a computation-affordable scale [50], [51].

(2) Next, the system encodes input questions via a question representation module. This module analyzes the semantics of the question and outputs reasoning instructions, which are usually represented as vectors. Typically, question q are encoded into hidden vectors $\boldsymbol{q}$ with neural networks (*e.g.,* LSTM, GRU, and PLMs) and then combined with other methods (*e.g.,* attention mechanism) to generate a vector as instruction:

$$\boldsymbol{i}^{(k)} = \textit{Question\_Representation}(\boldsymbol{i}^{(k-1)}, q, \mathcal{G}_q)$$

Here, $\{\boldsymbol{i}^{(k)}, k = 1, ..., n\}$ is the instruction vector of k-th reasoning that encodes the semantic and syntactic information of the natural language question. Both multi-step reasoning and one-step matching are applicable, which results in varying reasoning steps n.

(3) A graph based reasoning module conducts semantic matching via vector based computation to propagate and aggregate the information along the neighboring entities within the graph. The reasoning status $\{s^{(k)}, k = 1, ..., n\}$, which has diverse definitions in different methods (*e.g.*, distributions of predicted entities and representations of relations), is updated based on the reasoning instruction:

$$s^{(k)} = \textit{Graph\_Based\_Reasoning}(s^{(k-1)}, \boldsymbol{i}^{(k)}, \mathcal{G}_q),$$

where $s^{(k)}$ is the reasoning status which is considered as the status of k-th reasoning step on graph. Recently, several studies [44], [46] repeat Step (2) and (3) for multiple times to perform the reasoning.

(4) An answer generation module is utilized to generate answers according to the reasoning status at the end of reasoning. There are mainly two types of such generators: (1) entity ranking generator which ranks the entities to obtain top-ranked entities as predicted answers, (2) text generator which generates free text answers with vocabulary $\mathcal{V}$. This module can be formalized as:

$$\tilde{\mathcal{A}}_q = \textit{Answer\_Generation}(s^{(n)}, \mathcal{G}_q, \mathcal{V}),$$

where $s^{(n)}$ denotes the reasoning status at the last step. In the entity ranking paradigm, the entities contained in $\mathcal{G}_q$ are candidates for answer prediction $\tilde{\mathcal{A}}_q$. In many cases, $\tilde{\mathcal{A}}_q$ is obtained through selecting the entities with a score larger than the pre-defined threshold, where the score is derived from $s^{(n)}$. While in text generation paradigm, the answers are generated from vocabulary $\mathcal{V}$ as a sequence of tokens.

During training, the objective of entity ranking generator is usually to rank the correct entities higher than others in $\mathcal{G}_q$. In comparison, the text generator is usually trained to generate gold answers (name of correct entities).

3.3 A Comparison of Core Modules

Comparing the procedures of SP-based and IR-based methods, we note that these two methods have different designs of core modules and working mechanisms, but they also share similarities from multiple aspects.

Difference: SP-based methods rely heavily on the logical parsing module, which produces an expressive logic form for each question. In practice, many commercial KBQA systems developed upon SP-based methods require expertise to provide feedback to the generated logic forms so that the system can be further improved [52]. However, considering the expensive cost and expertise for obtaining annotated logic form, SP-based methods are usually trained under a weak supervision setting in research. Compared with IR-based methods, SP-based methods have the advantage of showing interpretability with explicit evidence about reasoning and defending perturbation of the question. However, the logical parsing module is bound by the design of the logic form and the capability of the parsing techniques, which is the key of performance improvement.

As a comparison, IR-based methods first employ a retrieval module to obtain the question-specific graph, and then conduct complex reasoning on the graph structure with graph based reasoning module. The answers are eventually predicted via an answer generation module. The performance of the IR-based methods partially depends on the recall of the retrieval module as the subsequent reasoning takes the retrieved graph as input. Meanwhile, the the graph based reasoning and answer generation module play key roles in making accurate prediction. Instead of generating intermediate logic forms, IR-based methods directly generate entities or free text as prediction. So they naturally fit into the end-to-end training paradigm and could be optimized easier compared with SP-based methods. Nevertheless, the blackbox style of the reasoning module makes the reasoning process less interpretable, which decreases the robustness and hinders users from interacting with the system.

Similarity: SP-based and IR-based methods both contain parameter-free modules, which are KB grounding, KB execution modules, and retrieval source construction module, respectively. While they are generally not learned from KBQA datasets, their performance has a great impact on the final KBQA performance. Both categories of methods make use of detected topic entities. SP-based methods leverage them to instantiate the logic form in the KB grounding module, while IR-based methods utilize them to narrow down the reasoning scale in the retrieval source module. Besides, both SP-based and IR-based methods emphasize the importance of natural language understanding with a question understanding (representation) module. The output of such modules substantially influences the subsequent parsing or reasoning process.

3.4 A Unified View - Neural Symbolic Reasoning

Above, we categorize the two mainstream approaches based on their different working mechanisms. In this part, we try to analyze them with a unified point view of neural symbolic reasoning. Such a discussion may deepen the understanding of complex KBQA methods.

Neural networks, which model the implicit correlations of data in continuous vector space [28], [53], have been proved highly effective in recent decades. In comparison, symbolic reasoning attempts to mine the discrete logic from data with explainability [54], [55]. Neural symbolic reasoning manages to combine the effectiveness of neural networks with the explainability of symbolic reasoning. Specifically, it proposes to conduct reasoning with neural networks, and provides explicit inference evidence to explain the results as well as the reasoning process. In both SP-based and IR-based categories, some methods fit into the neural-symbolic reasoning paradigm.

For SP-based methods, the reasoning is performed on a schema graph of parsed targets, where the nodes are components defined in the logic forms, and the edges are combining rules of grammar. We follow the parsing grammar in a representative SP-based method [49] and draw an example of question reasoning, which is displayed as the output of logical parsing module in Figure 3. As we can see, the schema graph consists of nodes, which are functions (*e.g.*, "Hop") and variables (*e.g.*, "e" and "r") defined by grammar. The rule — if the nodes are variables "e_0" and "r_1" then their parent node would be a function "Hop", denotes the implicit edge between the functions and entity variables. This generates a partial logic form "(Hop e_0 r_1)" resulting in a set of intermediate entities. Similarly, the rule "(Hop e_0 r_1)" could combine with another variable "r_2" denotes the implicit edge between the functions and partial logic forms. Above procedure infers the logic form by modeling the combining rule of grammar. The message is delivered along the implicit edges between the variables and functions.

The reasoning graph of IR-based methods is the knowledge graph, where the nodes are certain entities and edges are their relations. As the question specific graph shown in Figure 3, when it comes to the inference procedure, the entity receives message from its neighbors. In most cases, the message does not evenly propagate from its neighbors. The relation labels are treated as the important features to decide how much information should be delivered based on the semantic similarities between the relations and questions.

The unified point view of these two mainstream approaches indicates that the two mainstream approaches can be unified as reasoning over a graph, which differs in the definition of the graph. This combines discrete symbolic representations with neural networks to support inference in the continuous vector space [56].

4 Semantic Parsing-based Methods

In this part, we discuss the challenges and solutions for semantic parsing-based methods. The taxonomy of challenges and solutions can be visualized with Figure 5.

4.1 Overview

As introduced in Section 3, SP-based methods follow a parse-then-execute procedure via a series of modules, namely question understanding, logical parsing, KB grounding, and KB execution. These modules will encounter different challenges for complex KBQA. Firstly, question understanding becomes more difficult when the questions are complicated in both semantic and syntactic aspects. Secondly, logical parsing has to cover diverse query types of complex questions. Moreover, a complex question involving more relations and subjects will dramatically increase the possible search space for parsing, which makes the parsing less effective. Thirdly, the manual annotation of logic forms are both expensive and labor-intensive, and it is challenging to train the SP-based methods with weak supervision signals (*i.e.*, question-answer pairs).

In the following parts, we will introduce how prior studies deal with these challenges and summarize advanced techniques proposed by them.

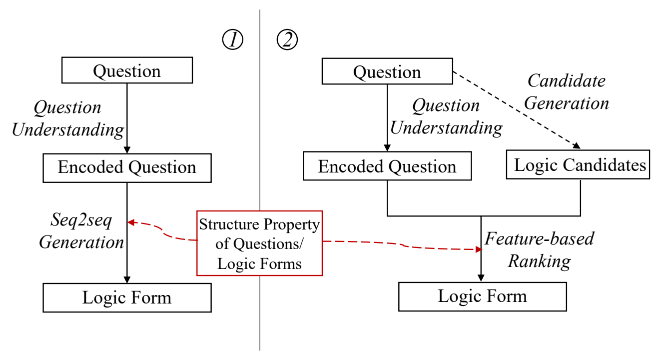

FIGURE 4: Illustration of two lines of research which leverage structure properties for better understanding of complex question.

4.2 Understanding Complex Semantics and Syntax

As the first step of SP-based methods, question understanding module converts unstructured text into encoded question, which benefits the downstream parsing. Compared with simple questions, complex questions are featured with compositional semantics and more complex query types, which increase the difficulty in linguistic analysis.

4.2.1 *Understanding complex semantics of questions*

The complex semantics of questions indicates a complex dependency pattern of sentences, which expresses the relation between constituents. Knowing the core part of the sentence structure could be beneficial for question understanding. Incorporating structure property of questions is an intuitive strategy to achieve this goal.

Incorporating structure property of questions to seq2seq generation. Many existing methods rely on syntactic parsing, such as dependencies [13], [57], [58] and Abstract Meaning Representation (AMR) [59], to provide better alignment between question constituents and logic form elements (*e.g.*,

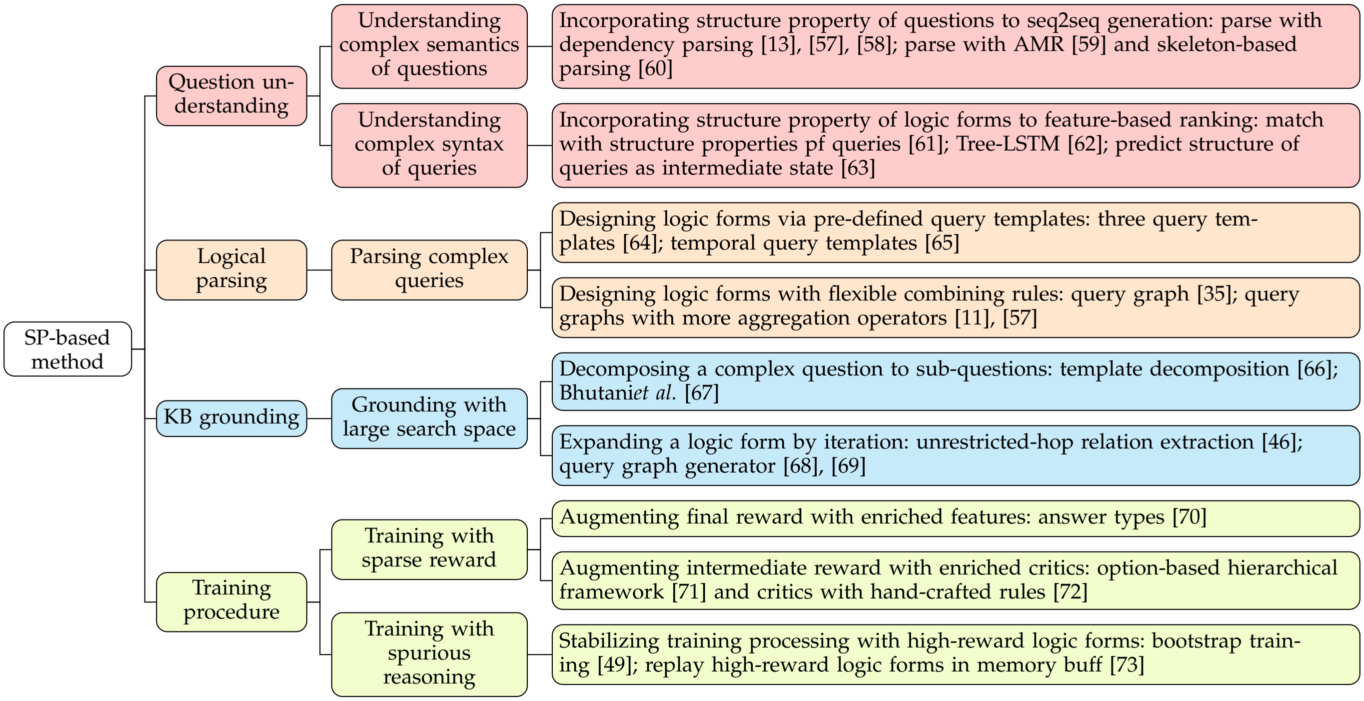

FIGURE 5: The main content of SP-based methods. The hierarchical structure is arranged with: SP-based method → module → challenge → solution.

entity, relation, entity types, and attributes). This line of research is illustrated at the left side of Figure 4. In order to represent long-range dependencies between the answer and the topic entity in question, Luo *et al.* [13] extracted the dependency path between them. By encoding the directional dependency path, they concatenated both syntactic features and local semantic features together to form global question representation. Similarly, Abujabal *et al.* [57] leveraged dependency parse to cope with compositional utterances and only focused on important tokens contained by parsed dependency path when creating query templates. Instead of directly creating logic forms upon the dependency paths, Abujabal *et al.* [58] leveraged dependency parse to analyse the composition of the utterances and aligned it with the logic form. Kapanipathi *et al.* [59] introduced AMR to help understand questions, the benefits are two-fold: (1) AMR is effective in disambiguating the natural language utterances. (2) AMR parsing module is highly abstract and helps to understand the questions in a KG-independent way. However, the accuracy of producing syntactic parsing is still not satisfying on complex questions, especially for those with long-distance dependency.

In order to alleviate the inaccurate syntactic parsing of complex questions, Sun *et al.* [60] leveraged the skeleton-based parsing to obtain the trunk of a complex question, which is a simple question with several branches (*i.e.*, head word of original text-spans) to be expanded. For example, the trunk for question "*What movie that Miley Cyrus acted in had a director named Tom Vaughan ?*" is "*What movie had a director ?*", and attributive clauses in question will be regarded as the branches of the trunk. Under such a skeleton structure, only simple questions are to be parsed further, which is more likely to obtain accurate parsing results.

4.2.2 *Understanding complex syntax of queries*

It is important to understand questions by analysing their complex semantics. It is also crucial to analyse the syntax of queries and ensure that the generated logic forms could meet the complex syntax of queries. While above methods generate logic forms with Seq2seq framework, another line of work (shown at the right side of Figure 4) focuses on leveraging structural properties (*e.g.*, tree structure or graph structure of logic forms) for ranking candidate parsing.

Incorporating structure property of logic forms to feature-based ranking. Maheshwari *et al.* [74] proposed a novel ranking model which exploits the structure of query graphs and uses attention weights to explicitly compare predicates with natural language questions. Specifically, they proposed a fine-grained slot matching mechanism to conduct hopwise semantic matching between the question and each predicate in the core reasoning chain. Instead of capturing semantic correlations between a question and a simple relation chain, Zhu *et al.* [61] focused on structure properties of query and conducted KBQA with query-question matching. They employed a structure-aware encoder to model entity or relation context in a query, promoting the matching between queries and questions. Similarly, Zafar *et al.* [62] incorporated two Tree-LSTMs [75] to model dependency parse trees of questions and tree structure of candidate queries respectively, and leveraged structural similarity between them for comprehensive ranking.

Traditional methods adopted state-transition strategy to generate candidate query graphs. As this strategy ignores the structure of queries, a considerable number of invalid queries will be generated as candidates. To filter these queries out, Chen *et al.* [63] proposed to predict the query structure of the question and leverage the structure to re-

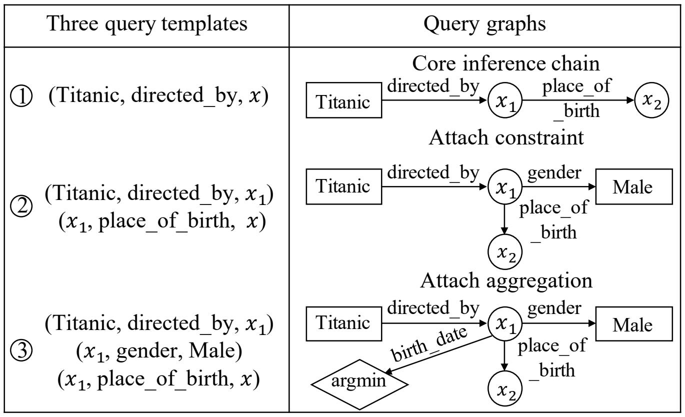

FIGURE 6: The illustration of possible parsing targets of for a complex question "*What's the birth place of the youngest male director of Titanic?*". x denotes the satisfied entity to be queried and x_1 denotes the intermediate entity included in the query.

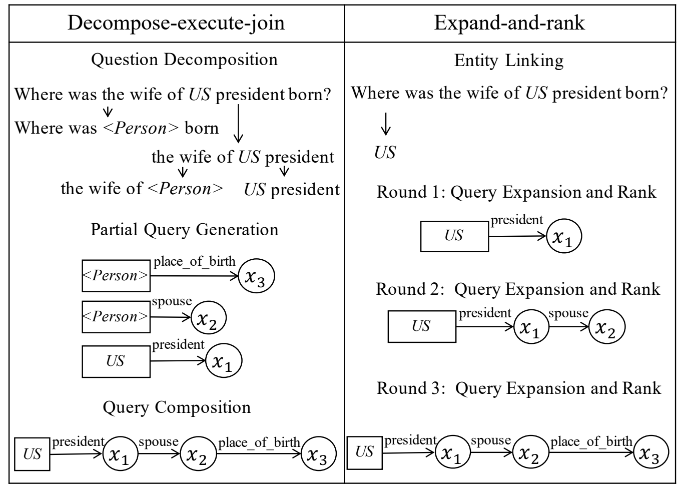

FIGURE 7: Illustration of two grounding strategies for a complex question "*Where was the wife of US president born?*".

strict the generation of the candidate queries. Specifically, they designed a series of operations to generate placeholders for types, numerical operators, predicates, and entities. After that, they can ground such uninstantiated logic forms with KBs and generate executable logic forms. With such a two-stage generation strategy, invalid logic forms with unmatched structure can be efficiently filtered out.

4.3 Parsing Complex Queries

To generate an executable logic form, traditional methods first utilized the existing parsers to convert a question into CCG derivation which is then mapped to a SPARQL via aligning predicates and arguments to relations and entities in the KBs. For example, the question "*Who directed Titanic ?*" can be parsed into "$TARGET(x) \wedge directed.arg1(e,x) \wedge directed.arg2(e, Titanic)$" via CCG parser. After that, predicate "$directed$" is aligned to relation "*directed_by*" and argument "$Titanic$" is aligned to entity "*Titanic*" in the KBs, respectively. Such a CCG derivation can be transferred to an executable SPARQL query. Traditional methods are sub-optimal for complex questions due to the ontology mismatching problem. Thus it is necessary to leverage the structure of KBs for accurate parsing, where parsing is performed along with the grounding of the KB.

Designing logic forms via pre-defined query templates. To satisfy the compositionality of the complex questions, researchers have developed diverse expressive logic forms as parsing targets. Recall the topic entities are recognized in the preliminary step, Bast *et al.* [64] started from the topic entities and designed three query templates as the parsing targets. We list these three query templates in Figure 6. The first two templates return entities which are 1-hop and 2-hop away from the topic entities "*Titanic*". The third template returns entities that are two hops away from the topic entities and constrained by another entity. A follow-up study concentrated on designing templates to answer temporal questions [65]. Although this work can successfully parse several types of complex questions, it suffers from the limited coverage issue.

Designing expressive logic forms with flexible combining rules. To design more expressive logic forms, Yih *et al.* [35] proposed query graph as the expressive parsing target. A query graph is a logic form in graph structure which closely matches with the KB schemas and is an alternative to an executable SPARQL. It consists of entities, variables, and functions, which correspond to grounded entities mentioned in questions, variables to query and aggregation operations, respectively. As illustrated in Figure 6, a set of core inference chains [35] starting from the topic entity are first identified. Constraint entities and aggregation operators are further attached to the path chains to make them adapt to more complex questions. Unlike the pre-defined templates, query graphs are not limited to the hop and constraint numbers. They have shown strong capabilities to express complex questions while they are still incapable to deal with long-tail complex question types. Based on more observations towards the long-tail data samples, follow-up work tried to improve the formulation of query graphs by involving syntactic annotation to enhance the structural complexity of the query graph [57], applying more aggregation operators such as merging, coreference resolution [12] to fit complex questions. Compared with query templates, logic forms with flexible combining rules could fit into a large variety of complex queries. A more expressive logic form indicates a more robust KBQA system which can handle questions with greater diversity.

4.4 Grounding with Large Search Space

To obtain executable logic forms, KB grounding module instantiates possible logic forms with a KB. As one entity in the KB can be linked to hundreds or even thousands of relations, it is unaffordable to explore and ground all the possible logic forms for a complex question considering both computational resource and time complexity.

Decomposing a complex question to sub-questions. Instead of enumerating logic forms with a single pass, researchers try to propose methods to generate the complex queries with multiple steps. Zheng *et al.* [66] proposed to first decompose a complex question into multiple simple questions, where each simple question was parsed into a simple logic form. The final answers are obtained with either the conjunction or composition of the partial logic forms. This ***decompose-execute-join*** strategy can effectively narrow down the search space. A similar approach was studied by Bhutani *et al.* [67]. As decomposing questions costs manual efforts, they reduced human annotation and identify the composition plan through an augmented pointer network [76]. The final answers are obtained via conjunction or composition of the answers of decomposed questions.

Expanding a logic form by iteration. Unlike decomposing a complex question to sub-questions, a number of studies adopted the ***expand-and-rank*** strategies to reduce the search space by expanding the logic forms in an iterative manner. Specifically, they collected all the query graphs that are 1-hop neighborhood of the topic entities as the candidate logic forms at the first iteration. These candidates are ranked based on their semantic similarities with the question. Top-ranked candidates are kept to do further expansion while low-ranked candidates are filtered out. At the following iterations, each top-ranked query graph in the beam is extended, which results in a new set of candidate query graphs that are more complicated. This procedure will repeat until the best query graph is obtained. Chen *et al.* [46] first utilized the hopwise greedy search to expand the most-likely query graphs. Lan *et al.* [68] proposed an incremental sequence matching module to iteratively parse the questions without revisiting the generated query graphs at each searching step. Above expansion is conducted in a linear manner, which is only effective in generating multi-hop relations. Lan *et al.* [69] defined three expansion actions for each iteration, which are extending, connecting, and aggregating to correspond to multi-hop reasoning, constrained relations, and numerical operations, respectively. Examples in Figure 7 show the different principles of these two strategies.

4.5 Training under Weak Supervision Signals

To cope with the issue of unlabeled reasoning paths, reinforcement learning (RL) based optimization has been used to maximize the expected reward [49], [71]. However, the insufficient training data makes it a challenge to train under weak supervision.

4.5.1 Training with sparse reward

Training via RL indicates that SP-based methods can only receive the feedback after the execution of the complete parsed logic form. This leads to a long exploration stage with severe sparse positive reward. To tackle this issue, methods are proposed to augment the final reward or intermediate reward.

Augmenting final reward with enriched features. Some research work adopted reward shaping strategy for parsing evaluation. Specifically, researchers augment reward of a logic form by involving more information of answers as the enriched features of the final prediction. Saha *et al.* [70] rewarded the model by the additional feedback when the predicted answers have the same type as ground truth. In this way, even the predicted answers are not exactly the ground truth, they could also encourage the model to search for the right answer type. This helps to avoid the sparse positive rewards during the exploration stage.

Augmenting intermediate reward with enriched critics. Besides rewards derived from the final prediction, intermediate rewards during the semantic parsing process may also help address this challenge. Recently, Qiu *et al.* [71] formulated query graph generation as a hierarchical decision problem, and proposed an option-based hierarchical framework to provide intermediate rewards for low-level agents. Through options over the decision process, the high-level agent sets goal for the low-level agent at intermediate steps. To evaluate whether intermediate states of the low-level agent meet the goal of the high-level agent, they measured the semantic similarity between the given question and the generated triple. To provide the policy with effective intermediate feedback, Qiu *et al.* [71] augmented the critic of query graphs with hand-crafted rules.

4.5.2 Dealing with spurious reasoning

At the early stage of training, it is difficult to find a logic form with the positive reward. Moreover, random exploration at the early stage easily lead to spurious reasoning, where logic forms result in correct answers but are semantically incorrect. Therefore, the early supervision of high-quality logic forms could be conducted to speed up the training and prevent models from the misguiding of spurious reasoning.

Stabilizing training processing with high-reward logic forms. To accelerate and stabilize the training process, Liang *et al.* [49] proposed to maintain pseudo-gold programs found by an iterative maximum-likelihood training process to bootstrap training. The training process contains two steps: (1) leveraging beam search mechanism to find pseudo-gold programs, and (2) optimizing the model under the supervision of the best program found in history. Hua *et al.* [73] followed a similar idea to evaluate the generated logic form by comparing it with the high-reward logic forms stored in the memory buffer. To make a trade-off between exploitation and exploration, they proposed the proximity reward and the novelty reward to encourage remembering the past high-reward logic forms and generating new logic forms to alleviate spurious reasoning respectively. Combining such bonus with terminal reward, models can obtain dense feedback along the learning phrase.

5 Information Retrieval-based Methods

In this section, we summarize the main challenges brought by complex questions for different modules of IR-based methods. The taxonomy of challenges and solutions can be visualized with Figure 8.

5.1 Overview

The overall procedure typically consists of the modules of retrieval source construction, question representation, graph based reasoning, and answer generation. These modules will encounter different challenges for complex KBQA. Firstly, the retrieval source module extracts a question-specific graph from KBs, which includes both relevant facts

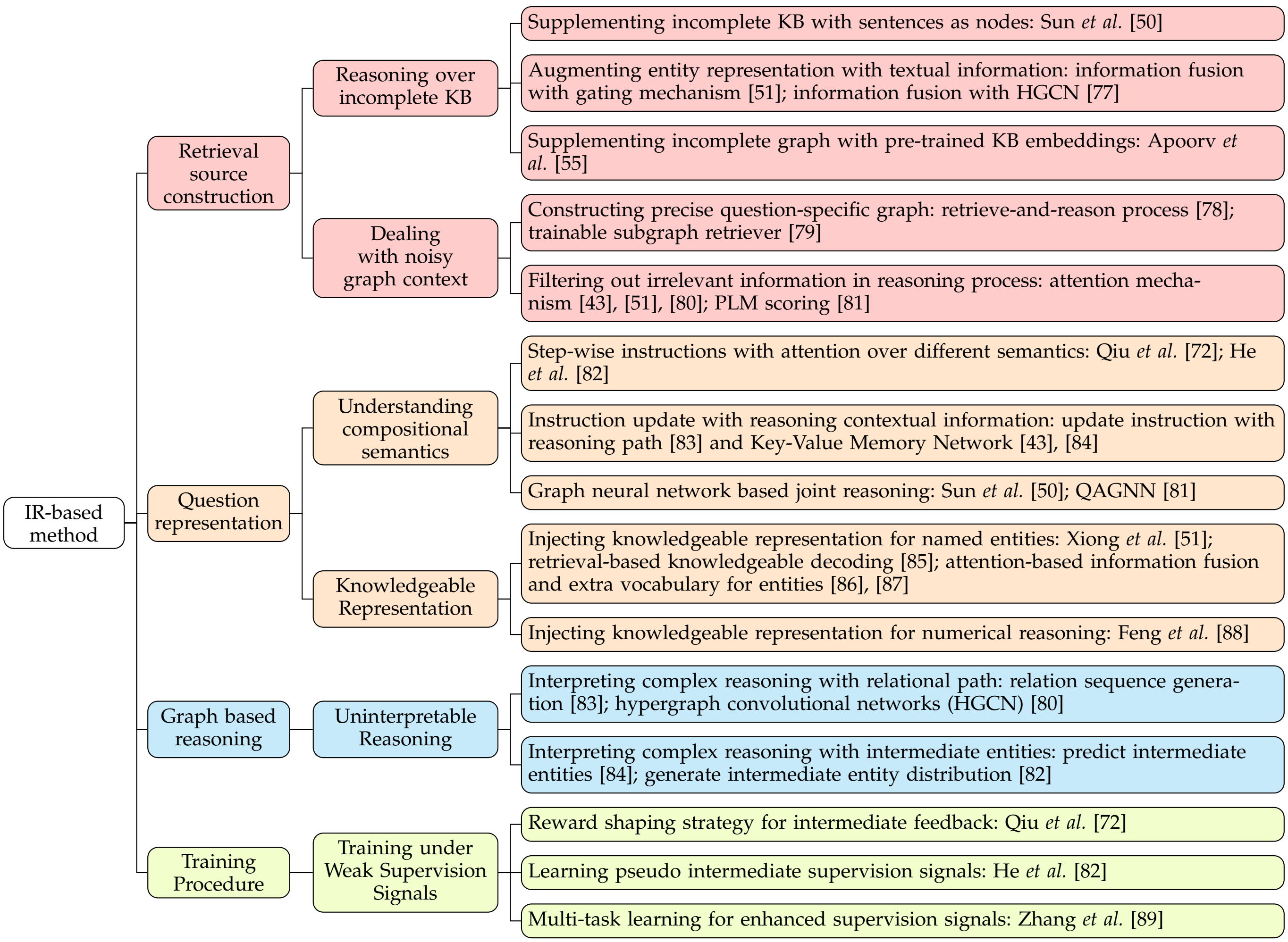

FIGURE 8: The main content of IR-based methods. The hierarchical structure is arranged with: IR-based method → module → challenge → solution.

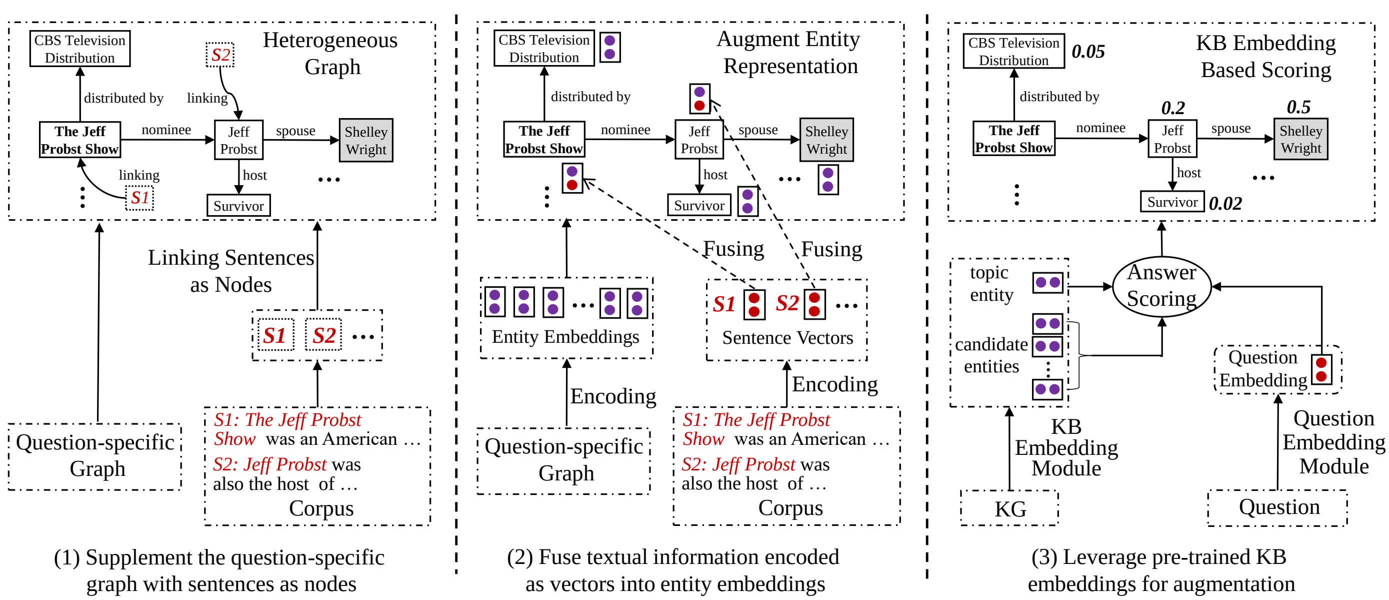

FIGURE 9: Illustration of three categories of methods to supplement the incomplete KB. All subfigures are drawn in a bottom-up style, where the input is placed at the bottom and supplemented graph is placed on the top. The topic entity and the answer entity are shown in the bold font and shaded box respectively.

and a wide range of noisy facts. Due to unneglectable incompleteness of source KBs [90], the correct reasoning paths may be absent from the extracted graph. The two issues are more likely to occur in the case of complex questions. Secondly, question representation module understands the question and generates instructions to guide the reasoning process. This step becomes challenging when the question is complicated. After that, reasoning on graph is conducted through semantic matching. When dealing with complex questions, such methods rank answers through semantic similarity without traceable reasoning in the graph, which hinders reasoning analysis and failure diagnosis.

The following parts illustrate how prior work deals with these challenges and the utilized advanced techniques.

5.2 Reasoning under Imperfect KB

In general, IR-based methods find answers by conducting reasoning on a graph structure. This graph structure is a question-specific graph extracted from a KB in most cases. However, such question-specific graphs are never perfect, due to incompleteness of KBs and the noisy graph context brought by heuristic graph generation strategy.

5.2.1 Reasoning over incomplete KB

It is vital for the question-specific graph to obtain high recall of correct reasoning paths. Since simple questions only require 1-hop reasoning on the neighborhood of the topic entity in the KB, IR-based methods are less likely to suffer from the inherent incompleteness of KBs [90] when solving the simple questions. By contrast, the correct reasoning paths for complex questions are of high probability to be absent from the question-specific graph and it turns to be a severe issue. To tackle with this challenge, researchers utilize auxiliary information to supplement the knowledge source. We divide the different supplementary methods into three categories and show the core differences in Figure 9.

Supplementing incomplete KB with sentences as nodes. Intuitively, a large amount of question-relevant text corpus retrieved from Wikipedia can provide a wide range of unstructured knowledge as supplementary evidence. Based on this observation, Sun *et al.* [50] proposed to complement the graph with extra question-relevant sentences as nodes and reason on the augmented heterogeneous graph (*i.e.*, the left side of Figure 9). According to the entities mentioned in sentences, they linked them to corresponding entities on the graph and viewed them as nodes.

Augmenting entity representation with textual information. Instead of directly complementing sentences to the question-specific graph as nodes, Xiong *et al.* [51] and Han *et al.* [77] proposed to fuse extra textual information into the entity representation as the second way (shown in the middle of Figure 9). Xiong *et al.* [51] proposed an end-to-end model, which consists of a subgraph reader to accumulate the knowledge of the question-specific graph and a text reader to obtain knowledge-aware information of sentences with the text-linked entity representations through a novel conditional gating mechanism. Such knowledge-aware information of sentences is further aggregated to enhance the entity representations to complement incomplete KB. Similarly, Han *et al.* [77] fused textual information of sentences into entity representations. In their settings, every sentence is regarded as a hyperedge connecting all of its involved entities and document can be viewed as a hypergraph. Based on hypergraph convolutional networks (HGCN) [91], they encoded the sentences in document and fused sentence representations into sentence-linked entity representations.

Supplementing incomplete graph with pre-trained KB embeddings. In knowledge base completion (KBC) task, knowledge base embeddings have been adopted to alleviate the sparsity of KB by performing missing link prediction. Inspired by that, Apoorv *et al.* [55] utilized pre-trained knowledge base embeddings to address the incomplete KB issue as shown at the right side of Figure 9. Specifically, they pre-trained KB embeddings (*i.e.*, entity and relation embeddings) with ComplEX [92] approach and predicted the answer via a triple scoring function taking the triples in the format of (topic entity, question, answer entity) as inputs. To make questions fit into original ComplEX scoring function, they map Roberta [93] embeddings of question into the complex space of same dimension. By leveraging the pre-trained knowledge from global KBs, they implicitly complemented the incomplete question-specific graph.

5.2.2 Dealing with noisy graph context

Since question-specific graphs are constructed in a heuristic way, it may introduce redundant and even question-irrelevant noisy graph context (both entities and sentence nodes). Compared with simple questions which require only 1-hop reasoning around topic entities, the question-specific graphs constructed for complex questions are more likely to involve noisy graph context. Reasoning over such noisy graphs poses a great challenge for complex questions, meanwhile it also reduces the efficiency of model training.

Constructing precise question-specific graph. An intuitive idea is to construct a relatively small and precise graph for downstream reasoning. To achieve this goal, Sun *et al.* [78] proposed to build the heterogeneous graph with an iterative retrieve-and-reason process under the supervision of shortest paths between the topic entities and answer entities. In a recent work, Zhang *et al.* [79] proposed a trainable subgraph retriever (SR) which retrieves relevant relational paths for subsequent reasoning. And their experimental results proved such precise graphs can bring significant performance gains for IR-based methods. While heuristic-based subgraph construction [50] was widely adopted in IR-based methods, the trainable subgraph retriever [79] is demonstrated to be the SOTA subgraph retriever toolkit in several benchmarks.

Filtering out irrelevant information in reasoning process. Besides constructing small and precise graphs for subsequent reasoning, some research work proposed to filter irrelevant information out along the reasoning process. Attention mechanisms, which are effective in eliminating irrelevant features, have been adopted by existing IR-based methods [43], [51], [80] to reserve relevant information during the reasoning process. Similarly, Yasunaga *et al.* [81] adopted pre-trained language model scoring of each node conditioned on question answering context as relevance scores to guide subsequent reasoning process.

5.3 Understanding Complex Semantics

Understanding the complex questions is the prerequisite for subsequent reasoning. However, complex questions contain compositional semantics and require specific knowledge (*e.g.*, named entities, ordinal reasoning) to answer. Due to such intrinsic properties of complex questions, methods designed for simple question understanding may be not fit for complex questions.

5.3.1 Understanding compositional semantics

IR-based methods usually generate initial question representation $\boldsymbol{q}$ by directly encoding questions as low-dimensional vectors through neural networks (*e.g.*, LSTM and GRU). Static reasoning instruction (*e.g.*, final hidden states of $\boldsymbol{q}$) obtained through above approach can not effectively represent the compositional semantics of complex questions, which poses challenges to guide the reasoning over the question-specific graph. In order to comprehensively understand questions, some studies dynamically update the reasoning instruction during the reasoning process.

Step-wise instructions with attention over different semantics. To make the reasoning models aware of the reasoning step, Qiu *et al.* [72] proposed to learn a step-aware representation through transforming initial question representation $\boldsymbol{q}$ with a single-layer perceptron. After obtaining step-aware question representation, attention mechanism is further incorporated to select useful information to generate instruction vectors. Similarly, He *et al.* [82] proposed to focus on different parts of the question with dynamic attention mechanism. Based on both step-aware question representation and previous reasoning instruction $\boldsymbol{i}^{(k-1)}$, they generated attention distribution over tokens of the question and updated the instruction vector.

Instruction update with reasoning contextual information. Besides explicitly recording the analyzed part of the question via attention, some other work proposed to update the reasoning instruction with information retrieved along the reasoning process. A typical example is generating explicit reasoning paths and updating instruction with generated paths. Zhou *et al.* [83] designed a model that takes the previous step reasoning instruction $\boldsymbol{i}^{(k)}$ as the input, and then predicts the intermediate relation $r^{(k)}$ from all relations in KB. After obtaining the predicted relation, the model updated the instruction vector as: $\boldsymbol{i}^{(k+1)} = \boldsymbol{i}^{(k)} - \boldsymbol{r}^{(k)}$, where the subtraction is meant to omit the analyzed information from the question. Thus, the updated reasoning instruction can hold unanalyzed parts of the question in the subsequent reasoning process.

Instead of generating explicit reasoning paths, Xu *et al.* [84] and Miller *et al.* [43] employed key-value memory network to achieve similar dynamic instruction update. Specifically, they first included all KB facts that contains with one of topic entities as subject into the memory. Then, they indexed the keys and values in the key-value memory, where keys are (subject, relation) pairs and values are corresponding object entities. A key addressing process is conducted to find the most suitable key and corresponding value for the instruction. With the addressed key and value, they concatenated their representations with the previous step reasoning instruction $\boldsymbol{i}^{(k)}$ and performed a linear transformation to obtain the updated reasoning instruction $\boldsymbol{i}^{(k+1)}$ to guide the next hop reasoning. In this way, the reasoning instruction will be updated over the memory.

Graph neural network based joint reasoning. Besides instruction update, another line of research addresses such compositional semantics with graph neural network (GNN) based reasoning. Sun *et al.* [50] proposed a GNN-based model GraftNet to reason complex questions over heterogeneous information sources. Through iterative GNN reasoning steps, the entity representations and reasoning instruction get updated in turn. The reasoning instruction conveys the knowledge of the topic entity which is dynamically updated over the reasoning process. Despite iterative update of reasoning instruction and graph neural network, Yasunaga *et al.* proposed [81] QAGNN model which reasoned complex questions with single graph neural network based joint reasoning. They constructed the question-specific graph with an extra question-answering context node which connects with all other nodes in the graph. All nodes are uniformly encoded with pre-trained language models (PLMs) as initial representation, and get updated along with graph neural network reasoning.

5.3.2 Knowledgeable Representation

Apart from compositional semantics, complex questions may also contain knowledge-intensive tokens or phrases (*e.g.*, named entities, ordinal constraint), which hinders natural language understanding for text-based semantic understanding. Besides question text, external knowledge is taken as the input to help understand these complex questions.

Injecting knowledgeable representation for named entities. In the natural language questions, the topic entities are always named entities which are not informative enough for understanding. To cope with such named entities, some existing work proposed to inject more informative representations obtained from knowledge bases. As a typical example, Xiong *et al.* [51] proposed to reformulate query representation in latent space with knowledge representation learned from the graph context of topic entities. Through ablation study, they verified the effectiveness of injecting such knowledgeable representation into question representation. Similar ideas were also adopted in knowledge-enhanced language model pre-training [94], [95].

While natural answers can be generated from popular seq2seq text generation framework, it is still hard to directly generate the named entities from token vocabulary. To address this gap, He *et al.* [85] first proposed a copying and retrieving mechanism to generate the natural answers from extra vocabulary for question tokens and entities in the question-specific graph. Similarly, Yin *et al.* [86] and Fu *et al.* [87] fed relational facts into structured memory slots, which served as extra vocabulary to generate named entities, and generate knowledgeable representation with attention-based information fusion.

Injecting knowledgeable representation for numerical reasoning. While multiple solutions are proposed to conduct multi-hop reasoning, little attention is paid to solving complex questions with numerical operations. To empower IR-based methods with numerical reasoning capability, Feng *et* *al.* [88] proposed to encode numerical properties (*i.e.*, the magnitude and ordinal properties of numbers) into entity representations. First, they manually defined a list of ordinal determiners (*e.g.*, first, largest) to detect ordinal constrained questions. For these detected questions, they enrich their question-specific graphs with extra numerical attribute triplets. Encoding these numerical attribute triplets with pre-trained number encoding modules, extra number embeddings can be used as model-agnostic plugins to conduct numerical reasoning for IR-based methods.

5.4 Uninterpretable Reasoning

Since the complex questions usually query multiple facts in sequence, the system is supposed to accurately predict answers over the graph based on a traceable reasoning process. While neural networks are powerful, blackbox style of reasoning module makes the reasoning process less interpretable and hard to incorporate user interaction for further improvement. To derive a more interpretable reasoning process, the reasoning is performed with a multi-step intermediate prediction. Along the reasoning process, the KBQA model generates a series of reasoning status $\{s^{(k)}, k = 1, ..., n\}$. While the final status is leveraged to generate the answer prediction, the intermediate status may help generate intermediate predictions (*i.e.*, matched relations or entities) for better interpretability. More importantly, intermediate predictions make it easier to detect spurious reasoning or error reasoning with user interaction.

Interpreting complex reasoning with relational path. Existing studies adopted different designs of reasoning status and reasoning modules to interpret the reasoning process. Specifically, Zhou *et al.* [83] formulated the multi-hop reasoning process as relation sequence generation and represented reasoning status using a vector. For each step, instruction vector and status vector are matched with relation candidates to generate probability distribution over all relations in KB. And weighted relation representation is then leveraged to update the status. By repeating this process, the model can achieve an interpretable reasoning process. Inspired by above work, Han *et al.* [80] proposed an interpretable model based on hypergraph convolutional networks (HGCN) to predict relation paths for explanation. They constructed a dense hypergraph by pinpointing a group of entities connected via same relation, which simulated human's hopwise relational reasoning. To train these two models, gold relation paths are leveraged. However, gold relation path annotations are unavailable in most cases, which makes their methods inapplicable to general datasets.

Interpreting complex reasoning with intermediate entities. Apart from relation paths, some research work predicted question-relevant entities at intermediate steps to explain multi-hop reasoning process. Xu *et al.* [84] elaborately adopted key-value memory network to achieve a traceable reasoning process. In their work, status $s^{(k)}$ is defined as the weighed sum of value representation, the weight of which is derived from key-instruction matching. To predict intermediate entities, their model followed traditional IR-based methods to score candidates given query $s^{(k)} + s^{(k-1)}$. As spurious long paths may connect topic entities with answer entities in KB, during training, they proposed to supervise intermediate entity prediction with the final answers. Such objective encourages the model to generate shortest reasoning path. Besides explicitly generating intermediate entities, He *et al.* [82] proposed to generate intermediate entity distribution to indicate the reasoning process. Their experimental results also showed that such intermediate supervision signals can effectively improve model accuracy.

5.5 Training under Weak Supervision Signals

Similar to the SP-based methods, it is difficult for IR-based methods to reason the correct answers without any annotations at intermediate steps, since the model cannot receive any feedback until the end of reasoning. It is found that this case may lead to spurious reasoning [82]. Due to the lack of intermediate state supervision signals, the reward obtained from spurious reasoning may mislead the model.

Reward shaping strategy for intermediate feedback. To train model under weak supervision signals, Qiu *et al.* [72] formulated multi-hop reasoning process over KBs as a process of expanding the reasoning path on graph. Based on the encoded decision history, the policy network leveraged attention mechanism to focus on the unique impact of different parts of a given question over triple selection. Similar as the strategies for SP-based methods, to alleviate the delayed and sparse reward problem caused by weak supervision signals, they adopted reward shaping strategy to evaluate reasoning paths and provide intermediate rewards. Specifically, they utilized semantic similarity between the question and the relation path to evaluate reasoning status at intermediate steps.

Learning pseudo intermediate supervision signals. Besides evaluating the reasoning status at intermediate steps, a more intuitive idea is to infer pseudo intermediate status and augment model training with such inferred signals. Inspired by bidirectional search algorithm on graph, He *et al.* [82] proposed to learn and augment intermediate supervision signals with bidirectional reasoning process. Taking entity distribution as suitable supervision signals at intermediate steps, they proposed to learn and leverage such signals under teacher-student framework. In the teacher network, the intermediate reasoning entity distributions are inferred and augmented with synchronized bidirectional reasoning process. With the learned intermediate entity distributions, the student network can receive feedback along the reasoning process and achieve better performance.

Multi-task learning for enhanced supervision signals. While most of existing work focused on enhancing the supervision signals at intermediate steps, few work paid attentions to the entity linking step. Most of existing work utilized off-the-shelf tools to locate the topic entity in question, causing error propagation. In order to accurately locate the topic entity without annotations, Zhang *et al.* [89] proposed to train entity linking module through a variational learning algorithm which jointly models topic entity recognition and subsequent reasoning over KBs. They also applied the REINFORCE algorithm with variance reduction technique to make the system end-to-end trainable.

6 PLM Applications on Complex KBQA

Unsupervised pre-training language models on large text corpora then fine-tuning pre-trained language models (PLMs) on downstream tasks has become a popular paradigm for natural language processing [96]. In complex KBQA task, both SP-based methods and IR-based methods have widely incorporated this technique. Furthermore, due to the powerful performance obtained from broad data at scale and capability to serve a wide range of downstream tasks, PLMs are recognized as "foundation models" [97] which may empower KBQA research to show a broader impact on the AI community.

Among all complex KBQA methods, part of methods directly generate answers without explicit reasoning. These methods, which are typically neural encoder-decoder models, can be well formalized under *knowledge-enhanced text generation* [98] framework. Both executable programs (*e.g.,* SPARQL and query graph, for SP-based methods) and free text (*i.e.,* surface name of answers, for IR-based methods) can be generated. With recent advances in PLMs, researchers have put more attention into utilizing them in answering complex questions.

PLM for precise and unified reasoning. Attracted by the powerful pre-trained language models, some researchers made adaptions to complex reasoning over graph structure for further involvement of PLM. While traditional reasoning over KB relies on the embeddings learned for entities and relations, such embeddings may fail to identify relevant parts of question answering context. To filter out noisy graph context in the retrieved subgraph, Yasunaga*et al.* [81] adopted PLM similarity scores to identify relevant knowledge given the question. For further joint reasoning of question answering context (*i.e.,* question-answer sequence) and knowledge graph, the node representations in the retrieved subgraph were initialized with PLM encoding of the concatenated sequence of question, answer, and node surface name.

PLM as knowledge source. While structured knowledge bases are always far from complete, PLMs may have the potential to predict the missing parts. PLMs have shown their capabilities to answer "fill-in-the-blank" cloze statements [99], which indicates that PLMs may learn relational knowledge from unsupervised pre-training. Petroni *et al.* [99] first analyzed the relational knowledge presented in a wide range of pre-trained language models and several pieces of follow-up work [100], [101] further demonstrated its effectiveness. These findings indicate that PLMs have great potential in serving as knowledge sources for question answering, which may play a complementary role for incomplete structural KB.

PLM for low-resource training. The robust and transferable natural language understanding capability obtained from unsupervised pre-training empowers PLMs to conquer the unaffordable need for training data in low-resource scenarios. In a recent study, Shi *et al.* [102] fine-tuned pre-trained sequence-to-sequence model on KQA Pro dataset [102] to generate SPARQLs and programs. The results show that PLM-based method (*i.e.,* BART-based generator [103]) achieved much superior performance (overall accuracy $> 87\%$) than attention-enhanced GRU encoder-decoder framework trained from scratch (overall accuracy $< 44\%$). Even reducing the training data to one tenth of original dataset, the PLM-based method still achieves much higher performance (overall accuracy $> 75\%$). While no external knowledge was incorporated to enhance the generation, the BART-based generator reached near-human performance and showed robustness to sparse training data. This observation implies that pre-trained language models have great potential in overcoming the sparsity of training data and incompleteness of knowledge bases.

PLM for cross-task generalization. Similar to complex KBQA, there is a series of tasks (*e.g.,* open-domain question answering, commonsense reasoning, tabular question answering) can be formed as leveraging structured knowledge to complete user requests. Due to the heterogeneous knowledge source, these tasks were studied by different communities. Recently, Xie *et al.* [104] proposed structured knowledge grounding (SKG) to unify these tasks and achieved (near) state-of-the-art performance with pre-trained language model T5 [105] on 21 benchmarks. With this PLM-based general-purpose approach, the challenges caused by the lack of training data in complex KBQA task can be solved by knowledge sharing and cross-task generalization.

7 Evaluation and Resource

In this section, we first introduce the evaluation protocol of KBQA systems. And then, we summarize some popular benchmarks for KBQA. At last, for tracking the research progress conveniently, we make a leaderboard for these benchmark datasets, which contains the evaluation results and resource links of the corresponding publications. We also attach a companion page [2] for comprehensive collection of the relevant publications, open-source codes, resources, and tools for KBQA.

7.1 Evaluation Protocol

In order to comprehensively evaluate KBQA systems, effective measurements from multiple aspects should be taken into consideration. Considering the goals to achieve, we categorize the measurement into three aspects: reliability, robustness, and system-user interaction [52].

Reliability: For each question, there is an answer set (one or multiple elements) as the ground truth. The KBQA system usually predicts entities with the top confidence score to form the answer set. If an answer predicted by the KBQA system exists in the answer set, it is a correct prediction. In previous studies [35], [49], [57], there are some classical evaluation metrics such as Precision, Recall, F_1, and Hits@1. For a question q, its Precision indicates the ratio of the correct predictions over all the predicted answers. It is formally defined as:

$$\text{Precision} = \frac{|\mathcal{A}_q \cap \tilde{\mathcal{A}}_q|}{|\tilde{\mathcal{A}}_q|},$$

2. https://github.com/RUCAIBox/Awesome-KBQA

where $\tilde{\mathcal{A}}_q$ is the predicted answers, and $\mathcal{A}_q$ is the ground truth. Recall is the ratio of the correct predictions over all the ground truth. It is computed as:

$$\text{Recall} = \frac{|\mathcal{A}_q \cap \tilde{\mathcal{A}}_q|}{|\mathcal{A}_q|}.$$

Ideally, we expect that the KBQA system has a higher Precision and Recall simultaneously. Thus F_1 score is most commonly used to give a comprehensive evaluation:

$$\text{F}_1 = \frac{2 * \text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}}.$$

Some other methods [43], [50], [51], [82] use Hits@1 to assess the fraction that the correct prediction rank higher than other entities. It is computed as:

$$\text{Hits@1} = \mathbb{I}(\tilde{a}_q \in \mathcal{A}_q),$$

where $\tilde{a}_q$ is the top 1 prediction in $\tilde{\mathcal{A}}_q$.

Robustness: Practical KBQA models are supposed to be built with strong generalizability to out-of-distribution questions at test time [14]. However, current KBQA datasets are mostly generated based on templates and lack of diversity [52]. And, the scale of training datasets is limited by the expensive labeling cost. Furthermore, the training data for KBQA system may hardly cover all possible user queries due to broad coverage and combinatorial explosion of queries. To promote the robustness of KBQA models, Gu *et al.* [14] proposed three levels of generalization (*i.e., i.i.d.*, *compositional*, and *zero-shot*) and released a large-scale KBQA dataset GrailQA to support further research. At a basic level, KBQA models are assumed to be trained and tested with questions drawn from the same distribution, which is what most existing studies focus on. In addition to that, robust KBQA models can generalize to novel compositions of seen schema items (*e.g.,* relations and entity types). To achieve better generalization and serve users, robust KBQA models are supposed to handle questions whose schema items or domains are not covered in the training stage.

System-user Interaction: While most of the current studies pay much attention to offline evaluation, the interaction between users and KBQA systems is neglected. On one hand, in the search scenarios, a user-friendly interface and acceptable response time should be taken into consideration. To evaluate this, the feedback of users should be collected and the efficiency of the system should be judged. On the other hand, users' search intents may be easily misunderstood by systems if only a single round service is provided. Therefore, it is important to evaluate the interaction capability of a KBQA system. For example, to check whether they could ask clarification questions to disambiguate users' queries and whether they could respond to the error reported from the users [58], [106]. So far, there is a lack of quantitative measurement of system-user interaction capability of the system, but human evaluation can be regarded as an efficient and comprehensive way.

7.2 Datasets And Leaderboard

Datasets. Over the decades, much effort has been devoted to constructing datasets for complex KBQA. We list the representative complex KBQA datasets for multiple popular KBs (*e.g.,* Freebase, DBpedia, Wikidata, and WikiMovies) in Table 1. In order to serve realistic applications, these datasets typically contain questions which require multiple KB facts to reason. Moreover, they might include numerical operations (*e.g.,* counting and ranking operations for comparative and superlative questions, respectively) and constraints (*e.g.,* entity and temporal keywords), which further increase the difficulty in reasoning the answers from KBs.

Overall, these datasets are constructed with the following steps: Given a topic entity in a KB as a question subject, simple questions are first created with diverse templates. Based on simple questions and neighborhood of topic entities in a KB, complex questions are further generated with predefined composition templates, and another work [102] also generates executable logic forms with templates. Meanwhile, answers are extracted with corresponding rules. In some cases, crowd workers are hired to paraphrase the canonical questions, and refine the generated logic forms, making the question expressions more diverse and fluent.

Leaderboard. In order to show the latest research progress in these KBQA benchmark datasets, we offer a leaderboard including the top-3 KBQA systems with respect to IR-based methods and SP-based methods. To give a fair comparison, the results are selected following two principles: 1) If one dataset has an official leaderboard, we only refer to the public results listed on the leaderboard. 2) otherwise, we select the top-3 results from the published papers accepted formally by conferences or journals before March 2022. Exceptionally, we do not report the results on the QALD series for easy display because they have multiple different versions, and we only report the 3-hop split of MetaQA Vanilla because it is more challenging than 1-hop and 2-hop splits. For LC-QuAD 2.0, we select the results reported in Zou *et al.* [124], where they produced these two results using the official public codes of respective methods. We leave the slots blank if there is no sufficient result according to the above principles. For more comprehensive evaluation of KBQA methods on all benchmarks, please refer to our companion page.

Analysis and discussions. Based on Table 1, we have following observations: (1) Both SP-based and IR-based methods are developed to handle complex KBQA challenges and there is no absolute agreement on which category is better. (2) While SP-based methods cover most benchmarks, IR-based methods focus on benchmarks which are mainly composed of multi-hop questions. The reason why SP-based methods are more commonly used in answering complex questions may be that SP-based methods generate flexible and expressive logic forms which are capable of covering all types of questions (*e.g.,* boolean, comparative). (3) We also observe for each category, the methods achieving outstanding performance are usually equipped with advanced techniques. The SP-based methods on the leaderboard leverage powerful question encoders (*e.g.,* PLMs) to help understand the questions and expressive logic forms to help parse complex queries. For IR-based methods, most SOTA methods adopt the step-wise dynamic instruction in question representation module and conduct multi-step reasoning with relational path modeling or GNN-based reasoning.

TABLE 1: Several KBQA benchmark datasets involving complex questions. "**LF**" denotes whether the dataset provides **L**ogic **F**orms like SPARQL, "**CO**" denotes whether the dataset contains questions with **CO**nstraints, "**NL**" represents whether the dataset incorporates crowd workers to rewrite questions in **N**atural **L**anguage and "**NU**" denotes whether the dataset contains the questions which require **NU**merical operations. Typically, SP-based methods adopt F_1 score as evaluation metric, while IR-based methods adopt $Hits@1$ (accuracy) as evaluation metric. The symbol of $\triangle$ and $\heartsuit$ indicates evaluation metric of $Hits@1$ (accuracy) and F_1 score respectively.

Datasets	KB	Size	LF	CO	NL	NU	SP-based			IR-based		
							Top-1	Top-2	Top-3	Top-1	Top-2	Top-3
WebQuestions [31]	Freebase	5,810	No	Yes	No	Yes	$62.9^{\heartsuit}$ [66]	$54.8^{\heartsuit}$ [71]	$54.6^{\heartsuit}$ [84]	$48.6^{\triangle}$ [84]	-	-
ComplexQuestions [107]	Freebase	2,100	No	Yes	No	Yes	$71.0^{\heartsuit}$ [66]	$54.3^{\heartsuit}$ [12]	$45.0^{\heartsuit}$ [71]	-	-	-
WebQuestionsSP [34]	Freebase	4,737	Yes	Yes	Yes	Yes	$82.9^{\heartsuit}$ [70]	$75.0^{\heartsuit}$ [108]	$74.6^{\heartsuit}$ [109]	$83.2^{\triangle}$ [79]	$74.3^{\triangle}$ [82]	$68.6^{\triangle}$ [88]
ComplexWebQuestions [76]	Freebase	34,689	Yes	Yes	Yes	Yes	$70.4^{\triangle}$ [110]	$44.1^{\triangle}$ [111]	$39.4^{\triangle}$ [112]	$53.9^{\triangle}$ [82]	$45.9^{\triangle}$ [78]	-
QALD series [113]	DBpedia	-	Yes	Yes	Yes	Yes	-	-	-	-	-	-
LC-QuAD [114]	DBpedia	5,000	Yes	Yes	Yes	Yes	$75.0^{\heartsuit}$ [62]	$74.8^{\heartsuit}$ [63]	$71.8^{\heartsuit}$ [115]	$33.0^{\heartsuit}$ [116]	-	-
LC-QuAD 2.0 [117]	DBpedia & Wikidata	30,000	Yes	Yes	Yes	Yes	$44.9^{\heartsuit}$ [63]	$52.6^{\heartsuit}$ [69]	-	-	-	-
MetaQA Vanilla [89]	WikiMovies	400k	No	No	No	No	$99.6^{\triangle}$ [68]	-	-	$100.0^{\triangle}$ [118]	$99.3^{\triangle}$ [119]	$98.9^{\triangle}$ [82]
CFQ [120]	Freebase	239,357	Yes	Yes	No	No	$67.3^{\triangle}$ [121]	$18.9^{\triangle}$ [120]	-	-	-	-
GrailQA [14]	Freebase	64,331	Yes	Yes	Yes	Yes	$74.4^{\heartsuit}$ [122]	$65.3^{\heartsuit}$ [123]	$58.0^{\heartsuit}$ [14]	-	-	-
KQA Pro [102]	Wikidata	117,970	Yes	Yes	Yes	Yes	$89.7^{\heartsuit}$ [102]	-	-	-	-	-

8 Recent Trends

In this section, we discuss several promising future directions for complex KBQA task:

进化KBQA **Evolutionary KBQA systems.** Existing KBQA systems are usually trained offline with specific datasets and then deployed online to handle user queries. However, most of existing KBQA systems neglect to learn from failure cases or unseen question templates received after deployment. At the same time, most of existing KBQA systems fail to catch up with the rapid growth of world knowledge and answer new questions. Therefore, a practical KBQA system is imperative to get performance improvement over time after online deployment. Online user interaction may provide deployed KBQA systems an opportunity to get further improvement. Based on this motivation, some work leverages user interaction to rectify answers generated by the KBQA system and further improve itself. With user feedback, Abujabal *et al.* [58] presented a continuous learning framework to learn new templates that capture previously unseen syntactic structures. Besides increasing the model's template bank, user feedback can also be leveraged to clarify ambiguous questions (*e.g.*, ambiguous phrases or ambiguous entities) [125]. Above methods provide an initial exploration to construct evolutionary KBQA systems with user feedback. Such approaches are effective and practical (*i.e.*, acceptable user cognitive burden and running cost), which may serve the industrial needs. Due to the wide applications of KBQA systems, more work and designs of user interaction with KBQA systems are in urgent need.

Robust KBQA systems. Existing studies on KBQA have conducted with the ideal hypothesis, where training data is sufficient and its distribution is identical with test set. However, this may not be desirable in practice due to data insufficiency and potential data distributional biases. To train robust KBQA systems in low-resource scenarios, meta-learning techniques [126] and knowledge transfer from high-resource scenarios [127] have been explored. We also highlighted the potential impact of PLMs in low-resource training and cross-task generalization (see Sec 6). As the manual annotations for KBQA systems are expensive and labor-intensive, there is a need for more studies about training robust KBQA systems in low-resource scenarios. Meanwhile, although existing methods usually hold the i.i.d. assumption, they may easily fail to deal with out-of-distribution (OOD) issue [128], [129], [130] on KBQA. With a systematic evaluation of GrailQA [14] dataset, Gu *et al.* [14] pointed out that existing baseline methods are vulnerable to compositional challenges. To promote higher level of robustness, researchers may gain more insights with addressing the three levels of generalization (*i.e.*, *i.i.d.*, *compositional*, and *zero-shot*) proposed by Gu *et al.* [14]. There is few work investigating robustness on complex KBQA task. It is still an open question of building robust KBQA systems with stronger generalization capability.

Conversational KBQA systems. Recent decades have seen the rapid development of AI-driven applications (*e.g.*, search engines and personal assistants) which are supposed to answer factoid questions. As users typically ask follow-up questions to explore a topic, deployed models are supposed to handle KBQA task in a conversational manner. In initial explorations of conversational KBQA, several pieces of work [6], [131], [132], [133] focused on ambiguity and difficulties brought by coreference and ellipsis phenomena. To track the focus of conversational KBQA, Lan *et al.* [111] proposed to model the flow of the focus via an entity transition graph. For a comprehensive understanding of conversation context, Plepi *et al.* [132], [134] leveraged Transformer [135] architecture taking as input of the previous turn of conversation history. While these studies addressed some challenges for conversational KBQA, it is still far from achieving human-level performance. More critical challenges should be identified and solved in the following research. Compared with traditional complex KBQA task, there could be more challenges in conversational KBQA task due to the conversation formulation and user involvement along the process. To our knowledge, there is no comprehensive taxonomy to cover the challenges in conversational KBQA task. While it is quite a new and challenging task, it may play an important role in future search engines and intelligent personal assistants.

More general knowledge bases. Due to KB incompleteness, researchers incorporated extra information (such as text [136], images [137], and human interactions [138]) to complement the knowledge bases, which would further address the information need of complex KBQA task. As text corpus is rich in semantics and easy to collect, researchers are fascinated by the idea of extracting knowledge from text corpus and answering questions with extracted knowledge. Researchers have explored various forms of knowledge obtained from text corpus, such as traditional relational triplet [139], virtual knowledge base (VKB) [140] which is stored as key-value memory, and PLMs as implicit knowledge base [99]. With these elaborate designs, more flexible and complementary knowledge can be obtained to solve complex KBQA tasks. Recently, a neglectable trend is to unify similar tasks with general architecture and achieve cross-task knolwedge transfer [104]. In the future, question answering related tasks may be explored with more general definition of KBs, such as synthetic, multilingual, and multi-modal KBs.

9 Conclusion

This survey attempted to provide an overview of typical challenges and corresponding solutions on complex KBQA. Particularly, task-related preliminary knowledge and traditional methods were first introduced. Then, we summarized the widely employed semantic parsing-based methods and information retrieval-based methods. We specified the challenges for these two categories of methods based on their working mechanism, and explicated the proposed solutions. Along with the taxonomy, we provide technical summaries to shed light on the applied advanced techniques for these two categories. Most of existing complex KBQA methods are generally summarized into these two categories. Please be aware that there are some other methods like [76], which focus on question decomposition instead of KB based reasoning or logic form generation. In the last section, we investigated several research trends related to complex KBQA task and emphasized many challenges are still open and under-explored. We believe that complex KBQA will continue to be an active and promising research area with wide applications, such as natural language understanding, compositional generalization, multi-hop reasoning. We hope this survey will give a comprehensive picture of cutting-edge methods for complex KBQA and encourage further contributions in this field.

References

[1] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor, "Freebase: A collaboratively created graph database for structuring human knowledge," in *SIGMOD*, 2008, pp. 1247–1250.
[2] J. Lehmann, R. Isele, M. Jakob, A. Jentzsch, D. Kontokostas, P. N. Mendes, S. Hellmann, M. Morsey, P. Van Kleef, S. Auer *et al.*, "Dbpedia–a large-scale, multilingual knowledge base extracted from wikipedia," *Semantic Web*, pp. 167–195, 2015.
[3] T. P. Tanon, D. Vrandecic, S. Schaffert, T. Steiner, and L. Pintscher, "From freebase to wikidata: The great migration," in *WWW*, 2016, pp. 1419–1428.
[4] F. M. Suchanek, G. Kasneci, and G. Weikum, "Yago: A core of semantic knowledge," in *WWW*, 2007, p. 697–706.
[5] H. Zhou, T. Young, M. Huang, H. Zhao, J. Xu, and X. Zhu, "Commonsense knowledge aware conversation generation with graph attention," in *IJCAI*, J. Lang, Ed., 2018, pp. 4623–4629.
[6] P. Christmann, R. S. Roy, A. Abujabal, J. Singh, and G. Weikum, "Look before you hop: Conversational question answering over knowledge graphs using judicious context expansion," in *CIKM*, 2019, pp. 729–738.
[7] A. Bordes, N. Usunier, S. Chopra, and J. Weston, "Large-scale simple question answering with memory networks," *CoRR*, 2015.
[8] L. Dong, F. Wei, M. Zhou, and K. Xu, "Question answering over freebase with multi-column convolutional neural networks," in *ACL*, 2015, pp. 260–269.
[9] M. Yu, W. Yin, K. S. Hasan, C. N. dos Santos, B. Xiang, and B. Zhou, "Improved neural relation detection for knowledge base question answering," in *ACL*, 2017, pp. 571–581.
[10] M. Petrochuk and L. Zettlemoyer, "Simplequestions nearly solved: A new upperbound and baseline approach," in *EMNLP*, 2018, pp. 554–558.
[11] S. Hu, L. Zou, J. X. Yu, H. Wang, and D. Zhao, "Answering natural language questions by subgraph matching over knowledge graphs," *TKDE*, vol. 30, pp. 824–837, 2018.
[12] S. Hu, L. Zou, and X. Zhang, "A state-transition framework to answer complex questions over knowledge base," in *EMNLP*, 2018, pp. 2098–2108.
[13] K. Luo, F. Lin, X. Luo, and K. Q. Zhu, "Knowledge base question answering via encoding of complex query graphs," in *EMNLP*, 2018, pp. 2185–2194.
[14] Y. Gu, S. Kase, M. Vanni, B. M. Sadler, P. Liang, X. Yan, and Y. Su, "Beyond I.I.D.: three levels of generalization for question answering on knowledge bases," in *WWW*, 2021, pp. 3477–3488.
[15] P. Wu, X. Zhang, and Z. Feng, "A survey of question answering over knowledge base," in *CCKS*, 2019, pp. 86–97.
[16] N. Chakraborty, D. Lukovnikov, G. Maheshwari, P. Trivedi, J. Lehmann, and A. Fischer, "Introduction to neural network based approaches for question answering over knowledge graphs," *CoRR*, 2019.
[17] B. Fu, Y. Qiu, C. Tang, Y. Li, H. Yu, and J. Sun, "A survey on complex question answering over knowledge base: Recent advances and challenges," *CoRR*, 2020.
[18] Y. Lan, G. He, J. Jiang, J. Jiang, W. X. Zhao, and J.-R. Wen, "A survey on complex knowledge base question answering: Methods, challenges and solutions," in *IJCAI*, 2021.
[19] Google, "Freebase data dumps," https://developers.google.com/freebase/data, 2016.
[20] R. Cyganiak, D. Wood, and M. Lanthaler, "Rdf 1.1 concepts and abstract syntax," in *WWW*, 2014.
[21] T. Rebele, F. M. Suchanek, J. Hoffart, J. Biega, E. Kuzey, and G. Weikum, "YAGO: A multilingual knowledge base from wikipedia, wordnet, and geonames," in *ISWC*, 2016, pp. 177–185.
[22] G. A. Miller, "Wordnet: A lexical database for english," *Commun. ACM*, vol. 38, no. 11, pp. 39–41, 1995.
[23] M. Färber, B. Ell, C. Menne, and A. Rettinger, "A comparative survey of dbpedia, freebase, opencyc, wikidata, and yago," *Semantic Web*, pp. 1–5, 2015.
[24] Y. Yang and M. Chang, "S-MART: Novel tree-based structured learning algorithms applied to tweet entity linking," in *ACL*, 2015, pp. 504–513.
[25] J. Daiber, M. Jakob, C. Hokamp, and P. N. Mendes, "Improving efficiency and accuracy in multilingual entity extraction," in *I-SEMANTICS*, 2013, p. 121–124.
[26] M. A. Yosef, J. Hoffart, I. Bordino, M. Spaniol, and G. Weikum, "AIDA: an online tool for accurate disambiguation of named entities in text and tables," *VLDB*, pp. 1450–1453, 2011.
[27] Q. Cai and A. Yates, "Large-scale semantic parsing via schema matching and lexicon extension," in *ACL*, 2013, pp. 423–433.
[28] T. Kwiatkowski, E. Choi, Y. Artzi, and L. Zettlemoyer, "Scaling semantic parsers with on-the-fly ontology matching," in *EMNLP*, 2013, pp. 1545–1556.
[29] S. Reddy, M. Lapata, and M. Steedman, "Large-scale semantic parsing without question-answer pairs," *TACL*, pp. 377–392, 2014.
[30] C. Unger, L. Bühmann, J. Lehmann, A. N. Ngomo, D. Gerber, and P. Cimiano, "Template-based question answering over RDF data," in *WWW*, 2012, pp. 639–648.
[31] J. Berant, A. Chou, R. Frostig, and P. Liang, "Semantic parsing on freebase from question-answer pairs," in *EMNLP*, 2013, pp. 1533–1544.
[32] W. Yih, X. He, and C. Meek, "Semantic parsing for single-relation question answering," in *ACL*, 2014, pp. 643–648.

[33] X. Yao and B. V. Durme, "Information extraction over structured data: Question answering with Freebase," in *ACL*, 2014, pp. 956–966.

[34] W. Yih, M. Richardson, C. Meek, M. Chang, and J. Suh, "The value of semantic parse labeling for knowledge base question answering," in *ACL*, 2016, pp. 201–206.

[35] W. Yih, M. Chang, X. He, and J. Gao, "Semantic parsing via staged query graph generation: Question answering with knowledge base," in *ACL*, 2015, pp. 1321–1331.

[36] S. Reddy, O. Täckström, M. Collins, T. Kwiatkowski, D. Das, M. Steedman, and M. Lapata, "Transforming dependency structures to logical forms for semantic parsing," *TACL*, pp. 127–140, 2016.

[37] W. Yin, M. Yu, B. Xiang, B. Zhou, and H. Schütze, "Simple question answering by attentive convolutional neural network," in *COLING*, 2016, pp. 1746–1756.

[38] Y. Lan, S. Wang, and J. Jiang, "Knowledge base question answering with a matching-aggregation model and question-specific contextual relations," *TASLP*, pp. 1629–1638, 2019.

[39] J. Andreas, M. Rohrbach, T. Darrell, and D. Klein, "Learning to compose neural networks for question answering," in *NAACL-HLT*, 2016, pp. 1545–1554.

[40] A. Bordes, J. Weston, and N. Usunier, "Open question answering with weakly supervised embedding models," in *Machine Learning and Knowledge Discovery in Databases*, 2014, pp. 165–180.

[41] A. Bordes, S. Chopra, and J. Weston, "Question answering with subgraph embeddings," in *EMNLP*, 2014, pp. 615–620.

[42] S. Sukhbaatar, A. Szlam, J. Weston, and R. Fergus, "End-to-end memory networks," in *NeurIPS*, 2015, pp. 2440–2448.

[43] A. H. Miller, A. Fisch, J. Dodge, A. Karimi, A. Bordes, and J. Weston, "Key-value memory networks for directly reading documents," in *EMNLP*, 2016, pp. 1400–1409.

[44] S. Jain, "Question answering over knowledge base using factual memory networks," in *NAACL-HLT*.

[45] Y. Hao, Y. Zhang, K. Liu, S. He, Z. Liu, H. Wu, and J. Zhao, "An end-to-end model for question answering over knowledge base with cross-attention combining global knowledge," in *ACL*, 2017, pp. 221–231.

[46] Z.-Y. Chen, C.-H. Chang, Y.-P. Chen, J. Nayak, and L.-W. Ku, "UHop: An unrestricted-hop relation extraction framework for knowledge-based question answering," in *NAACL-HLT*, 2019, pp. 345–356.

[47] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, p. 1735–1780, 1997.

[48] K. Cho, B. van Merriënboer, D. Bahdanau, and Y. Bengio, "On the properties of neural machine translation: Encoder–decoder approaches," in *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, 2014, pp. 103–111.

[49] C. Liang, J. Berant, Q. Le, K. D. Forbus, and N. Lao, "Neural symbolic machines: Learning semantic parsers on Freebase with weak supervision," in *ACL*, 2017, pp. 23–33.

[50] H. Sun, B. Dhingra, M. Zaheer, K. Mazaitis, R. Salakhutdinov, and W. W. Cohen, "Open domain question answering using early fusion of knowledge bases and text," in *EMNLP*, 2018, pp. 4231–4242.

[51] W. Xiong, M. Yu, S. Chang, X. Guo, and W. Y. Wang, "Improving question answering over incomplete kbs with knowledge-aware reader," in *ACL*, 2019, pp. 4258–4264.

[52] D. Diefenbach, V. López, K. D. Singh, and P. Maret, "Core techniques of question answering systems over knowledge bases: a survey," *Knowl. Inf. Syst.*, pp. 529–569, 2018.

[53] J. Berant and P. Liang, "Semantic parsing via paraphrasing," in *ACL*, 2014, pp. 1415–1425.

[54] D. Lukovnikov, A. Fischer, J. Lehmann, and S. Auer, "Neural network-based question answering over knowledge graphs on word and character level," in *Proceedings of the 26th International Conference on World Wide Web*, 2017, p. 1211–1220.

[55] A. Saxena, A. Tripathi, and P. P. Talukdar, "Improving multi-hop question answering over knowledge graphs using knowledge base embeddings," in *ACL*, 2020, pp. 4498–4507.

[56] J. Zhang, B. Chen, L. Zhang, X. Ke, and H. Ding, "Neural, symbolic and neural-symbolic reasoning on knowledge graphs," *CoRR*, 2021.

[57] A. Abujabal, M. Yahya, M. Riedewald, and G. Weikum, "Automated template generation for question answering over knowledge graphs," in *WWW*, 2017, pp. 1191–1200.

[58] A. Abujabal, R. S. Roy, M. Yahya, and G. Weikum, "Never-ending learning for open-domain question answering over knowledge bases," in *WWW*, 2018, pp. 1053–1062.

[59] P. Kapanipathi, I. Abdelaziz, S. Ravishankar, S. Roukos, A. G. Gray, R. F. Astudillo, M. Chang, C. Cornelio, S. Dana, A. Fokoue, D. Garg, A. Gliozzo, S. Gurajada, H. Karanam, N. Khan, D. Khandelwal, Y. Lee, Y. Li, F. P. S. Luus, N. Makondo, N. Mihindukulasooriya, T. Naseem, S. Neelam, L. Popa, R. G. Reddy, R. Riegel, G. Rossiello, U. Sharma, G. P. S. Bhargav, and M. Yu, "Leveraging abstract meaning representation for knowledge base question answering," in *Findings of ACL*, 2021.

[60] Y. Sun, L. Zhang, G. Cheng, and Y. Qu, "SPARQA: skeleton-based semantic parsing for complex questions over knowledge bases," in *AAAI*, 2020, pp. 8952–8959.

[61] S. Zhu, X. Cheng, and S. Su, "Knowledge-based question answering by tree-to-sequence learning," *Neurocomputing*, pp. 64–72, 2020.

[62] H. Zafar, G. Napolitano, and J. Lehmann, "Formal query generation for question answering over knowledge bases," in *ESWC*, 2018, pp. 714–728.

[63] Y. Chen, H. Li, Y. Hua, and G. Qi, "Formal query building with query structure prediction for complex question answering over knowledge base," in *IJCAI*, 2020, pp. 3751–3758.

[64] H. Bast and E. Haussmann, "More accurate question answering on freebase," in *CIKM*, 2015, p. 1431–1440.

[65] Z. Jia, A. Abujabal, R. Saha Roy, J. Strötgen, and G. Weikum, "Tequila: Temporal question answering over knowledge bases," in *CIKM*, 2018, p. 1807–1810.

[66] W. Zheng, J. X. Yu, L. Zou, and H. Cheng, "Question answering over knowledge graphs: Question understanding via template decomposition," 2018, pp. 1373–1386.

[67] N. Bhutani, X. Zheng, and H. V. Jagadish, "Learning to answer complex questions over knowledge bases with query composition," in *CIKM*, 2019, pp. 739–748.

[68] Y. Lan, S. Wang, and J. Jiang, "Multi-hop knowledge base question answering with an iterative sequence matching model," in *ICDM*, 2019, pp. 359–368.

[69] Y. Lan and J. Jiang, "Query graph generation for answering multi-hop complex questions from knowledge bases," in *ACL*, 2020, pp. 969–974.

[70] A. Saha, G. A. Ansari, A. Laddha, K. Sankaranarayanan, and S. Chakrabarti, "Complex program induction for querying knowledge bases in the absence of gold programs," *TACL*, pp. 185–200, 2019.

[71] Y. Qiu, K. Zhang, Y. Wang, X. Jin, L. Bai, S. Guan, and X. Cheng, "Hierarchical query graph generation for complex question answering over knowledge graph," in *CIKM*, 2020, pp. 1285–1294.

[72] Y. Qiu, Y. Wang, X. Jin, and K. Zhang, "Stepwise reasoning for multi-relation question answering over knowledge graph with weak supervision," in *WSDM*, 2020, pp. 474–482.

[73] Y. Hua, Y. Li, G. Qi, W. Wu, J. Zhang, and D. Qi, "Less is more: Data-efficient complex question answering over knowledge bases," *J. Web Semant.*, p. 100612, 2020.

[74] G. Maheshwari, P. Trivedi, D. Lukovnikov, N. Chakraborty, A. Fischer, and J. Lehmann, "Learning to rank query graphs for complex question answering over knowledge graphs," in *ISWC*, 2019, pp. 487–504.

[75] K. S. Tai, R. Socher, and C. D. Manning, "Improved semantic representations from tree-structured long short-term memory networks," in *ACL*, 2015, pp. 1556–1566.

[76] A. Talmor and J. Berant, "The web as a knowledge-base for answering complex questions," in *NAACL-HLT*, 2018, pp. 641–651.

[77] J. Han, B. Cheng, and X. Wang, "Open domain question answering based on text enhanced knowledge graph with hyperedge infusion," in *Findings of EMNLP*, 2020, pp. 1475–1481.

[78] H. Sun, T. Bedrax-Weiss, and W. W. Cohen, "Pullnet: Open domain question answering with iterative retrieval on knowledge bases and text," in *EMNLP*, 2019, pp. 2380–2390.

[79] J. Zhang, X. Zhang, J. Yu, J. Tang, J. Tang, C. Li, and H. Chen, "Subgraph retrieval enhanced model for multi-hop knowledge base question answering," *CoRR*, vol. abs/2202.13296, 2022.

[80] J. Han, B. Cheng, and X. Wang, "Two-phase hypergraph based reasoning with dynamic relations for multi-hop kbqa," in *IJCAI*, 2020, pp. 3615–3621.

[81] M. Yasunaga, H. Ren, A. Bosselut, P. Liang, and J. Leskovec, "QA-GNN: reasoning with language models and knowledge graphs

for question answering," in *NAACL-HLT 2021*, K. Toutanova, A. Rumshisky, L. Zettlemoyer, D. Hakkani-Tür, I. Beltagy, S. Bethard, R. Cotterell, T. Chakraborty, and Y. Zhou, Eds. Association for Computational Linguistics, 2021, pp. 535–546.

[82] G. He, Y. Lan, J. Jiang, W. X. Zhao, and J. Wen, "Improving multi-hop knowledge base question answering by learning intermediate supervision signals," in *WSDM*, 2021, pp. 553–561.

[83] M. Zhou, M. Huang, and X. Zhu, "An interpretable reasoning network for multi-relation question answering," in *COLING*, 2018, pp. 2010–2022.

[84] K. Xu, Y. Lai, Y. Feng, and Z. Wang, "Enhancing key-value memory neural networks for knowledge based question answering," in *NAACL-HLT*, 2019, pp. 2937–2947.

[85] S. He, C. Liu, K. Liu, and J. Zhao, "Generating natural answers by incorporating copying and retrieving mechanisms in sequence-to-sequence learning," in *ACL 2017*, R. Barzilay and M. Kan, Eds. Association for Computational Linguistics, 2017, pp. 199–208.

[86] J. Yin, X. Jiang, Z. Lu, L. Shang, H. Li, and X. Li, "Neural generative question answering," in *IJCAI 2016*, S. Kambhampati, Ed., 2016, pp. 2972–2978.

[87] Y. Fu and Y. Feng, "Natural answer generation with heterogeneous memory," in *NAACL-HLT 2018*, M. A. Walker, H. Ji, and A. Stent, Eds. Association for Computational Linguistics, 2018, pp. 185–195.

[88] Y. Feng, J. Zhang, G. He, W. X. Zhao, L. Liu, Q. Liu, C. Li, and H. Chen, "A pretraining numerical reasoning model for ordinal constrained question answering on knowledge base," in *Findings of the Association for Computational Linguistics: EMNLP 2021*, M. Moens, X. Huang, L. Specia, and S. W. Yih, Eds. Association for Computational Linguistics, 2021, pp. 1852–1861.

[89] Y. Zhang, H. Dai, Z. Kozareva, A. J. Smola, and L. Song, "Variational reasoning for question answering with knowledge graph," in *AAAI*, 2018, pp. 6069–6076.

[90] B. Min, R. Grishman, L. Wan, C. Wang, and D. Gondek, "Distant supervision for relation extraction with an incomplete knowledge base," in *NAACL-HLT*, 2013, pp. 777–782.

[91] Y. Feng, H. You, Z. Zhang, R. Ji, and Y. Gao, "Hypergraph neural networks," in *AAAI*, 2019, pp. 3558–3565.

[92] T. Trouillon, J. Welbl, S. Riedel, É. Gaussier, and G. Bouchard, "Complex embeddings for simple link prediction," in *ICML*, 2016, pp. 2071–2080.

[93] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "Roberta: A robustly optimized BERT pretraining approach," *CoRR*, 2019.

[94] Z. Zhang, X. Han, Z. Liu, X. Jiang, M. Sun, and Q. Liu, "ERNIE: enhanced language representation with informative entities," in *ACL 2019*, A. Korhonen, D. R. Traum, and L. Màrquez, Eds. Association for Computational Linguistics, 2019, pp. 1441–1451.

[95] M. E. Peters, M. Neumann, R. L. L. IV, R. Schwartz, V. Joshi, S. Singh, and N. A. Smith, "Knowledge enhanced contextual word representations," in *EMNLP-IJCNLP 2019*, K. Inui, J. Jiang, V. Ng, and X. Wan, Eds. Association for Computational Linguistics, 2019, pp. 43–54.

[96] R. Bommasani, D. A. Hudson, E. Adeli, R. Altman, S. Arora, S. von Arx, M. S. Bernstein, J. Bohg, A. Bosselut, E. Brunskill, E. Brynjolfsson, S. Buch, D. Card, R. Castellon, N. S. Chatterji, A. S. Chen, K. Creel, J. Q. Davis, D. Demszky, C. Donahue, M. Doumbouya, E. Durmus, S. Ermon, J. Etchemendy, K. Ethayarajh, L. Fei-Fei, C. Finn, T. Gale, L. Gillespie, K. Goel, N. D. Goodman, S. Grossman, N. Guha, T. Hashimoto, P. Henderson, J. Hewitt, D. E. Ho, J. Hong, K. Hsu, J. Huang, T. Icard, S. Jain, D. Jurafsky, P. Kalluri, S. Karamcheti, G. Keeling, F. Khani, O. Khattab, P. W. Koh, M. S. Krass, R. Krishna, R. Kuditipudi, and et al., "On the opportunities and risks of foundation models," *CoRR*, vol. abs/2108.07258, 2021.

[97] X. Han, Z. Zhang, N. Ding, Y. Gu, X. Liu, Y. Huo, J. Qiu, L. Zhang, W. Han, M. Huang, Q. Jin, Y. Lan, Y. Liu, Z. Liu, Z. Lu, X. Qiu, R. Song, J. Tang, J. Wen, J. Yuan, W. X. Zhao, and J. Zhu, "Pre-trained models: Past, present and future," *CoRR*, vol. abs/2106.07139, 2021.

[98] W. Yu, C. Zhu, Z. Li, Z. Hu, Q. Wang, H. Ji, and M. Jiang, "A survey of knowledge-enhanced text generation," *CoRR*, vol. abs/2010.04389, 2020.

[99] F. Petroni, T. Rocktäschel, S. Riedel, P. S. H. Lewis, A. Bakhtin, Y. Wu, and A. H. Miller, "Language models as knowledge bases?" in *EMNLP*, 2019, pp. 2463–2473.

[100] Z. Bouraoui, J. Camacho-Collados, and S. Schockaert, "Inducing relational knowledge from BERT," in *AAAI*, 2020, pp. 7456–7463.

[101] Z. Jiang, F. F. Xu, J. Araki, and G. Neubig, "How can we know what language models know," *TACL*, pp. 423–438, 2020.

[102] J. Shi, S. Cao, L. Pan, Y. Xiang, L. Hou, J. Li, H. Zhang, and B. He, "Kqa pro: A large diagnostic dataset for complex question answering over knowledge base," *CoRR*, 2020.

[103] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer, "BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension," in *ACL 2020*, D. Jurafsky, J. Chai, N. Schluter, and J. R. Tetreault, Eds. Association for Computational Linguistics, 2020, pp. 7871–7880.

[104] T. Xie, C. H. Wu, P. Shi, R. Zhong, T. Scholak, M. Yasunaga, C. Wu, M. Zhong, P. Yin, S. I. Wang, V. Zhong, B. Wang, C. Li, C. Boyle, A. Ni, Z. Yao, D. R. Radev, C. Xiong, L. Kong, R. Zhang, N. A. Smith, L. Zettlemoyer, and T. Yu, "Unifiedskg: Unifying and multi-tasking structured knowledge grounding with text-to-text language models," *CoRR*, vol. abs/2201.05966, 2022.

[105] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, "Exploring the limits of transfer learning with a unified text-to-text transformer," *J. Mach. Learn. Res.*, vol. 21, pp. 140:1–140:67, 2020.

[106] Z. Yao, Y. Tang, W. Yih, H. Sun, and Y. Su, "An imitation game for learning semantic parsers from user interaction," in *EMNLP*, 2020, pp. 6883–6902.

[107] J. Bao, N. Duan, Z. Yan, M. Zhou, and T. Zhao, "Constraint-based question answering with knowledge graph," in *COLING*, 2016, pp. 2503–2514.

[108] J. Zhang, L. Zhang, B. Hui, and L. Tian, "Improving complex knowledge base question answering via structural information learning," *Knowledge-Based Systems*, 2022.

[109] S. Cao, J. Shi, Z. Yao, L. Hou, J. Li, and J. Xiao, "Program transfer and ontology awareness for semantic parsing in KBQA," *CoRR*, 2021.

[110] R. Das, M. Zaheer, D. Thai, A. Godbole, E. Perez, J. Y. Lee, L. Tan, L. Polymenakos, and A. McCallum, "Case-based reasoning for natural language queries over knowledge bases," in *EMNLP*, M. Moens, X. Huang, L. Specia, and S. W. Yih, Eds., 2021.

[111] Y. Lan and J. Jiang, "Modeling transitions of focal entities for conversational knowledge base question answering," in *ACL*, 2021.

[112] Y. Lan, S. Wang, and J. Jiang, "Knowledge base question answering with topic units," in *IJCAI*, 2019, pp. 5046–5052.

[113] Orange, "Qald data challenge," http://qald.aksw.org/, 2011.

[114] P. Trivedi, G. Maheshwari, M. Dubey, and J. Lehmann, "Lc-quad: A corpus for complex question answering over knowledge graphs," in *ISWC*, 2017, pp. 210–218.

[115] W. Zheng and M. Zhang, "Automated query graph generation for querying knowledge graphs," in *CIKM*, G. Demartini, G. Zuccon, J. S. Culpepper, Z. Huang, and H. Tong, Eds., 2021.

[116] S. Vakulenko, J. D. F. Garcia, A. Polleres, M. de Rijke, and M. Cochez, "Message passing for complex question answering over knowledge graphs," in *CIKM*, 2019, pp. 1431–1440.

[117] M. Dubey, D. Banerjee, A. Abdelkawi, and J. Lehmann, "Lc-quad 2.0: A large dataset for complex question answering over wikidata and dbpedia," in *ISWC*, 2019, pp. 69–78.

[118] J. Shi, S. Cao, L. Hou, J. Li, and H. Zhang, "Transfernet: An effective and transparent framework for multi-hop question answering over relation graph," in *EMNLP*, M. Moens, X. Huang, L. Specia, and S. W. Yih, Eds., 2021.

[119] J. Cai, Z. Zhang, F. Wu, and J. Wang, "Deep cognitive reasoning network for multi-hop question answering over knowledge graphs," in *ACL/IJCNLP*, C. Zong, F. Xia, W. Li, and R. Navigli, Eds., 2021.

[120] D. Keysers, N. Schärli, N. Scales, H. Buisman, D. Furrer, S. Kashubin, N. Momchev, D. Sinopalnikov, L. Stafiniak, T. Tihon, D. Tsarkov, X. Wang, M. van Zee, and O. Bousquet, "Measuring compositional generalization: A comprehensive method on realistic data," in *ICLR*, 2020.

[121] Y. Guo, Z. Lin, J. Lou, and D. Zhang, "Hierarchical poset decoding for compositional generalization in language," in *NeurIPS*, 2020.

[122] X. Ye, S. Yavuz, K. Hashimoto, Y. Zhou, and C. Xiong, "Rng-kbqa: Generation augmented iterative ranking for knowledge base question answering," *CoRR*, vol. abs/2109.08678, 2021.

[123] S. Chen, Q. Liu, Z. Yu, C.-Y. Lin, J.-G. Lou, and F. Jiang, “ReTraCk: A flexible and efficient framework for knowledge base question answering,” in *ACL*, 2021.

[124] J. Zou, M. Yang, L. Zhang, Y. Xu, Q. Pan, F. Jiang, R. Qin, S. Wang, Y. He, S. Huang, and Z. Zhao, “A chinese multi-type complex questions answering dataset over wikidata,” *CoRR*, vol. abs/2111.06086, 2021.

[125] W. Zheng, H. Cheng, J. X. Yu, L. Zou, and K. Zhao, “Interactive natural language question answering over knowledge graphs,” *Inf. Sci.*, pp. 141–159, 2019.

[126] Y. Hua, Y.-F. Li, G. Haffari, G. Qi, and T. Wu, “Few-shot complex knowledge base question answering via meta reinforcement learning,” in *EMNLP*, 2020, pp. 5827–5837.

[127] Y. Zhou, X. Geng, T. Shen, W. Zhang, and D. Jiang, “Improving zero-shot cross-lingual transfer for multilingual question answering over knowledge graph,” in *NAACL-HLT*, 2021, pp. 5822–5834.

[128] M. Gardner, Y. Artzi, V. Basmova, J. Berant, B. Bogin, S. Chen, P. Dasigi, D. Dua, Y. Elazar, A. Gottumukkala, N. Gupta, H. Hajishirzi, G. Ilharco, D. Khashabi, K. Lin, J. Liu, N. F. Liu, P. Mulcaire, Q. Ning, S. Singh, N. A. Smith, S. Subramanian, R. Tsarfaty, E. Wallace, A. Zhang, and B. Zhou, “Evaluating nlp models via contrast sets,” *ArXiv*, 2020.

[129] D. Kaushik, E. Hovy, and Z. Lipton, “Learning the difference that makes a difference with counterfactually-augmented data,” in *ICLR*, 2020.

[130] I. Oren, J. Herzig, N. Gupta, M. Gardner, and J. Berant, “Improving compositional generalization in semantic parsing,” in *Findings of EMNLP*, 2020, pp. 2482–2495.

[131] D. Guo, D. Tang, N. Duan, M. Zhou, and J. Yin, “Dialog-to-action: Conversational question answering over a large-scale knowledge base,” in *NeurIPS*, 2018, pp. 2946–2955.

[132] J. Plepi, E. Kacupaj, K. Singh, H. Thakkar, and J. Lehmann, “Context transformer with stacked pointer networks for conversational question answering over knowledge graphs,” in *ESWC*, 2021, pp. 356–371.

[133] M. Kaiser, R. S. Roy, and G. Weikum, “Reinforcement learning from reformulations in conversational question answering over knowledge graphs,” 2021.

[134] E. Kacupaj, J. Plepi, K. Singh, H. Thakkar, J. Lehmann, and M. Maleshkova, “Conversational question answering over knowledge graphs with transformer and graph attention networks,” in *EACL*, 2021, pp. 850–862.

[135] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, “Attention is all you need,” in *NuerIPS*, 2017, pp. 5998–6008.

[136] R. Xie, Z. Liu, J. Jia, H. Luan, and M. Sun, “Representation learning of knowledge graphs with entity descriptions,” in *AAAI*, 2016, pp. 2659–2665.

[137] R. Xie, Z. Liu, H. Luan, and M. Sun, “Image-embodied knowledge representation learning,” in *IJCAI*, 2017, pp. 3140–3146.

[138] G. He, J. Li, W. X. Zhao, P. Liu, and J. Wen, “Mining implicit entity preference from user-item interaction data for knowledge graph completion via adversarial learning,” in *WWW*, 2020, pp. 740–751.

[139] X. Lu, S. Pramanik, R. S. Roy, A. Abujabal, Y. Wang, and G. Weikum, “Answering complex questions by joining multi-document evidence with quasi knowledge graphs,” in *SIGIR*, 2019, pp. 105–114.

[140] B. Dhingra, M. Zaheer, V. Balachandran, G. Neubig, R. Salakhutdinov, and W. W. Cohen, “Differentiable reasoning over a virtual knowledge base,” in *ICLR*, 2020.