Explore What LLM Does Not Know in Complex Question Answering

Xin Lin[1,2], **Zhenya Huang**[1,2,3], **Zhiqiang Zhang**[5], **Jun Zhou**[4], **Enhong Chen**[1,2*]
[1]School of Computer Science and Technology, University of Science and Technology of China, Hefei, China
[2]State Key Laboratory of Cognitive Intelligence, Hefei, China
[3]Institute of Artificial Intelligence, Hefei Comprehensive National Science Center, Hefei, China
[4]Zhejiang University, Hangzhou, China
[5]Independent Researcher
linx@mail.ustc.edu.cn, {huangzhy, cheneh}@ustc.edu.cn, {zzqsmall, junzhougucas}@gmail.com

Abstract

Complex question answering (QA) is a challenging task in artificial intelligence research which requires reasoning based on related knowledge. The retrieval-augmented generation (RAG) based on large language models (LLMs) have become one promising solution in QA. To facilitate RAG more effectively, the LLM needs to precisely evaluate knowledge required in QA. That is, first, the LLM needs to examine its knowledge boundary (what the LLM does not know) to retrieve external knowledge as supplement. Second, the LLM needs to evaluate the utility of the retrieved knowledge (whether it helps in reasoning) for robust RAG. To this end, in this paper, we propose a novel Question Answering with Knowledge Evaluation (KEQA) framework to promote the effectiveness and efficiency of RAG in QA. First, inspired by quizzes in classroom, we propose a quiz-based method to precisely examine the knowledge state of the uninterpretable LLM for QA. We ask indicative quizzes on each required knowledge, and inspect whether the LLM can consistently answer the quiz to examine its knowledge boundary. Second, we retrieve the unknown knowledge from external source, and evaluate its utility to pick the helpful ones for reasoning. We design a reasoning-based metric to evaluate utility, and construct a demonstration set in training data for reference to guide knowledge picking in inference. We conduct extensive experiments on four widely-used QA datasets, and the results demonstrate the effectiveness of the proposed method.

1 Introduction

Complex question answering (QA) is a key task in artificial intelligence (AI) research (Lehnert 1978), which aims to answer questions based on related knowledge. Therefore, the QA systems are required to master multiple knowledge, and perform complex reasoning over these knowledge, making it a challenging task. As shown in Figure 1, to answer the question "what is the birth date of the person Richard Callaghan coached ...", the QA system should know "who did Richard Callaghan coached ..." ("Tara Lipinski"), and "what is the birth date of Tara Lipinski" ("June 10, 1982").

Recently, the large language models (LLMs) have become the most promising solution for QA due to numerous pre-trained knowledge stored in huge parameters and strong reasoning abilities (Achiam et al. 2023; Zhou et al. 2023; Zheng et al. 2024). The retrieval-augmented generation (RAG), which first retrieves knowledge and then performs generation, further supplements the LLMs with external knowledge for more accurate and reliable QA (Wang et al. 2023b; Jiang et al. 2023). To facilitate RAG more effectively, the LLM needs to accurately evaluate knowledge required in QA. First, the LLM needs to precisely examine its *knowledge boundary*, i.e., what the LLM has already known and what it does not know, and adopts different actions. For example, in Figure 1, the LLM has already known the first knowledge K_1 "who did Richard Callaghan coached ..." which could be directly used in QA, but has not mastered K_2 "what is the birth date of Tara Lipinski" which needs to be supplemented from external to answer the question. Second, the LLM needs to evaluate whether the knowledge could help to answer the question (i.e., the *utility*) especially for external knowledge. As shown in Figure 1, all three knowledge (P_1, P_2 and P_3) are relevant to the question ("Tara Lipinski"), but only P_1 helps in QA with the birth date and the rest are helpless and may even mislead reason-

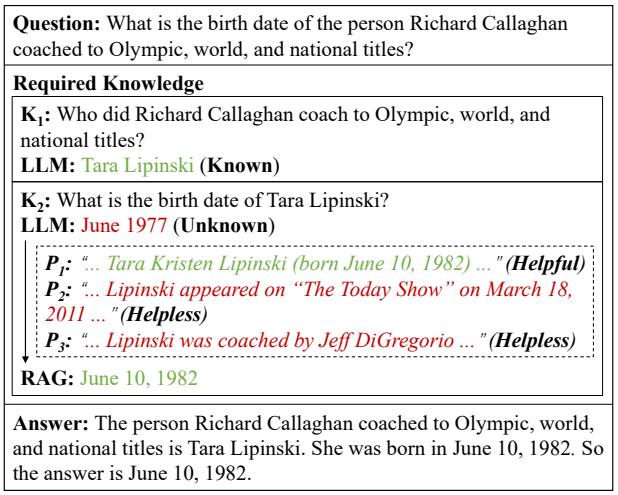


Figure 1: An example of one complex question requiring knowledge K_1 and K_2. The LLM has mastered K_1, while K_2 needs to be retrieved from external source.

*Corresponding Author.

ing. Therefore, we hope to design a framework for accurate knowledge evaluation to promote RAG in complex QA.

However, there are several technical challenges along this line. First, it is hard to precisely examine the knowledge boundary of the uninterpretable LLMs in complex reasoning. Existing methods mainly prompt the LLM to dynamically determine whether to retrieve external knowledge (Wang et al. 2023b; Jiang et al. 2023; Asai et al. 2024), which has been proven to be inaccurate, as LLMs tend to be over-confident and unknown knowledge may be mistakenly treated as known (Ren et al. 2023; Yin et al. 2023). Another more reasonable method is to conduct retrieval when the generation probability is low (Jiang et al. 2023). However, it focuses on the uncertain tokens, and is hard to precisely find what knowledge is missing. Moreover, it can not work when the probability is unavailable. How to precisely examine the knowledge state of the LLM is a challenging problem. Second, identifying the utility of knowledge in QA may be difficult for complex questions. Different from relevancy that can be directly observed from the semantics or literal contents of the knowledge and question (e.g., they both contain "Tara Lipinski" in Figure 1), the utility of knowledge in QA may be affected by reasoning logic (whether it is necessary in one reasoning step, such as P_2 and P_3 are not used in Figure 1) and the ability of LLM (whether the LLM masters the knowledge and whether the LLM may be affected by the knowledge). We can hardly analyze whether the knowledge is helpful unless we perform reasoning with the knowledge and observe the outputs. How to evaluate the utility of the knowledge in QA is another non-trivial technical challenge.

To this end, we explore the knowledge boundary of the LLM and the utility of knowledge in reasoning, and propose a novel *Q*uestion *A*nswering with *K*nowledge *E*valuation (KEQA) framework to promote RAG for complex QA. First, inspired by the quizzes in classroom, we propose a quiz-based method to precisely examine whether the LLM masters knowledge required in QA. We generate quizzes on each required knowledge to evaluate the LLM, and confirm the results by inspecting whether the LLM can consistently answer the quiz. We only retrieve the unknown knowledge that the LLM fails in the quiz from external source for efficiency. Second, to evaluate the utility of retrieved knowledge, we design a metric to compute the utility of the knowledge based on reasoning outputs. We compute the knowledge utility in the training data, and construct a demonstration set for reference. In inference, we use the reference set to guide the LLM to evaluate the utility of each retrieved knowledge and pick the helpful ones in reasoning for robust RAG. Finally, we conduct extensive experiments on four widely-used datasets in QA to evaluate the KEQA framework, and the results demonstrate that the proposed method can achieve higher performances and efficiency, and more robust RAG as well.

2 Related Work

Question Answering. Question answering is a key task in AI and natural language processing (NLP) research. Studies on QA have evolved from early rule-based methods (Hirschman et al. 1999), neural network methods (Cui et al. 2017; Lin et al. 2024a; Liu et al. 2023b), pre-trained methods (Bian et al. 2021; Lin et al. 2024b; Liu et al. 2023a) to recent LLM-based methods (Yang et al. 2022; Liu et al. 2024; Xue et al. 2024b). Recently, LLMs have shown strong reasoning abilities in various NLP tasks, and become the most promising solution for QA. The researchers have designed several methods to further evoke the pre-trained knowledge in LLMs, and improve reasoning. For example, Kojima et al. (Kojima et al. 2022) discovered that prompts like "Let's think step-by-step" could let the LLMs output the reasoning process (chain of thought, CoT) and increase accuracy, and Wei et al. (Wei et al. 2022) further used CoTs as demonstrations for stable improvements. Wang et al. (Wang et al. 2023a) proposed self-consistency by voting between multiple reasoning for robust results and higher performances. Researchers further improved CoT by explicitly decomposing the complex questions to plan the reasoning logic (Xue et al. 2024a). Zhou et al. (Zhou et al. 2023) and Dua et al. (Dua et al. 2022) proposed least-to-most prompting and successive prompting respectively by asking and answering sub-questions step-by-step to solve complex questions. Moreover, researchers also designed the retrieval-augmented generation and verification to avoid factual faults with external knowledge (Trivedi et al. 2023; Dhuliawala et al. 2024), and self-refine to refine the outputs and increase accuracy (Madaan et al. 2023).

Retrieval-Augmented Generation. Researchers find that LLMs often generate erroneous facts in the output, which is called hallucination. The retrieval-augmented generation (RAG) is the most widely-used method to address the problem. Vanilla RAG first retrieves external knowledge with the question, and then generates the output based on the knowledge (Lewis et al. 2020). In multi-step reasoning tasks where one may need to retrieve with the intermediate output, Trivedi et al. (Trivedi et al. 2023) and Feng et al. (Feng et al. 2024) extended the vanilla RAG by iteratively performing retrieval and generation on the generated and retrieved results. Jiang et al. (Jiang et al. 2023) further proposed the FLARE framework which dynamically determined whether retrieval was required based on the generation probability and performed retrieval only when necessary. In this way, it could achieve good tradeoff between retrieval costs and performances. In addition, researchers also improved RAG from other perspectives. For example, Ma et al. (Ma et al. 2023) and Wang, Yang, and Wei (Wang, Yang, and Wei 2023) improved retrieval by rewriting the query. Sun et al. (Sun et al. 2023) and Yu et al. (Yu et al. 2023) replaced external retrieval with passage generation from LLM, and Yoran et al. (Yoran et al. 2024) explored several methods to make RAG more robust to irrelevant retrieval results.

Our work differs from existing methods as follows. First, existing methods seldom precisely examine the knowledge state of LLMs, which may limit performances or efficiency of RAG, while we design a quiz-based knowledge evaluation for LLMs, which is more precise and could work on black-box LLMs without probability. Second, existing methods mainly improve the relevancy of the retrieved knowledge for robustness, while we further study their utility in reasoning to pick those more helpful ones to promote RAG.

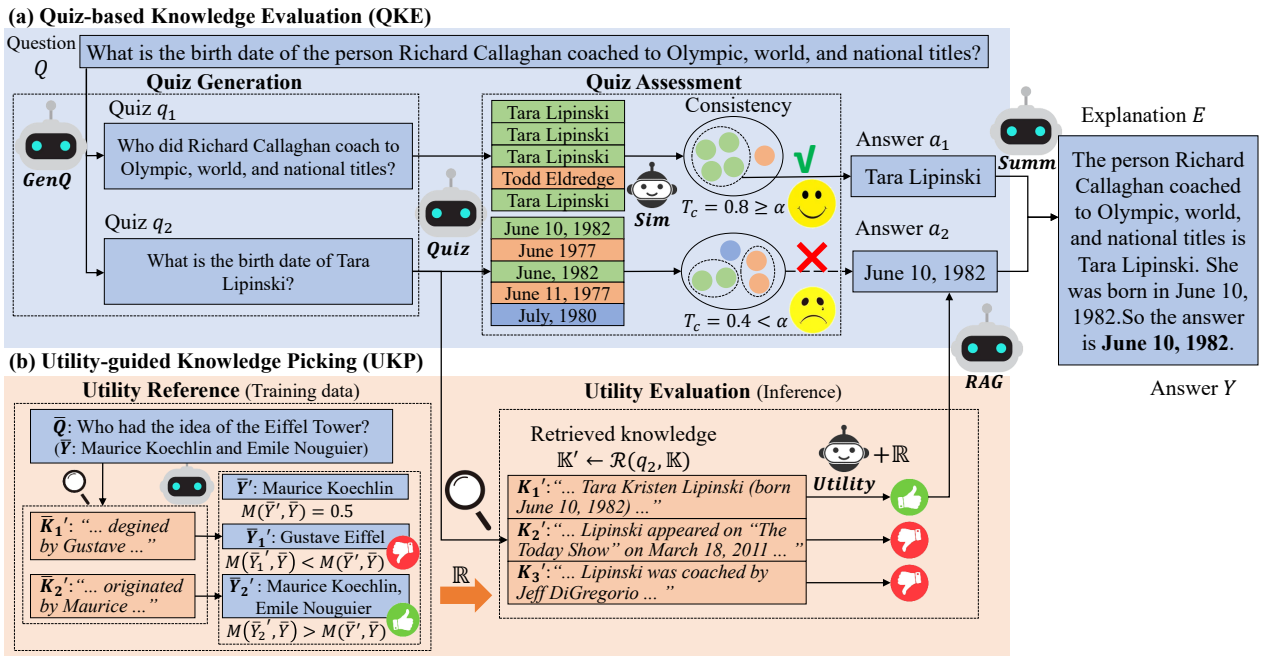


Figure 2: The KEQA framework, which is composed of (a) Quiz-based Knowledge Evaluation to evaluate knowledge state of the LLM and (b) Utility-guided Knowledge Picking to select the helpful retrieved knowledge based on the utility.

3 KEQA Framework

3.1 Problem Definition

Question answering consists of the question Q (e.g., "what is the birth date of ...?") and the answer Y ("June 10, 1982") to Q, which are both in natural language. An external knowledge source $\mathbb{K} = \{K_1, K_2, \cdots, K_n\}$ is given along with the QA data, where $K_i \in \mathbb{K}$ may be a passage in corpus, knowledge triple in knowledge graph or webpage online depending on $\mathbb{K}$. In this paper, we focus on the passages.

Given the knowledge source $\mathbb{K}$ and the question Q, our goal is to retrieve knowledge $\mathbb{K}^* = \{K_1^*, K_2^*, \cdots, K_m^*\}$ from $\mathbb{K}$ with a retriever $\mathcal{R}$ if necessary, and generate one explanation E with a LLM $\mathcal{L}$ to infer the answer Y to Q. In this paper, we hope to first evaluate whether $\mathcal{L}$ masters knowledge required to answer Q, and only retrieve the unknown ones from $\mathbb{K}$ to generate the answer Y.

3.2 Framework Overview

We propose a novel *Q*uestion *A*nswering with *K*nowledge *E*valuation (KEQA) framework to promote retrieval and reasoning in question answering. As shown in Figure 2, KEQA is composed of the Quiz-based Knowledge Evaluation (QKE) and the Utility-guided Knowledge Picking (UKP). QKE examines the knowledge boundary of the LLM $\mathcal{L}$ for the question Q, and UKP retrieves the unknown knowledge that are helpful in reasoning from $\mathbb{K}$ to augment answer generation. More specifically, first, QKE generates the quizzes on each knowledge required by Q to examine the LLM, and inspects whether the LLM $\mathcal{L}$ could answer these quizzes. Next, for quizzes that $\mathcal{L}$ fails, UKP retrieves related knowledge from knowledge source K with the retriever $\mathcal{R}$, evaluates the utility of each retrieved knowledge, and picks helpful ones to answer the failed quizzes. Finally, KEQA generates the explanation E and answer Y to Q with $\mathcal{L}$ based on the answers of the quizzes. In the following sections, we will introduce the QKE and UKP in detail.

3.3 Quiz-Based Knowledge Evaluation

Retrieval may be quite expensive in costs or time latency for RAG, so it is practical to examine the *knowledge boundary* of the LLM and only retrieve the unknown knowledge to reduce retrieval (Jiang et al. 2023). However, it is hard to achieve the goal on the huge uninterpretable parameters of the LLM. Therefore, inspired by the quizzes in classroom to evaluate students' knowledge proficiency, we design a quiz-based knowledge evaluation to examine the knowledge state of the LLM in a result-oriented manner for the target question Q. We ask the LLM knowledge-related questions, and inspect whether the LLM could answer these questions as shown in Figure 2(a). We call each question one *quiz*.

Quiz Generation. The complex question Q (e.g., "what is the birth ... titles" in Figure 2) is not a suitable quiz to evaluate the LLM. If the LLM fails in the question, it is hard to determine whether the LLM misses required knowledge and which knowledge it misses, as multiple reasons may lead to the failure such as incorrect reasoning logic, inconsistent generation, and absence of any required knowledge. Therefore, we need simpler indicative quizzes that is only related to one piece of required knowledge and does not need complex reasoning (e.g., "what is the birth date of Tara Lipinski"). Obviously it is reasonable to assume that the LLM

fails the quiz only due to absence of the related knowledge.

Given the complex question Q, we use the LLM $\mathcal{L}$ to generate the quizzes by decomposing Q. We carefully craft demonstrations to prompt $\mathcal{L}$ to decompose Q into simple sub-questions $q_1, q_2, \cdots, q_s$. Each sub-question q_i is expected to be directly answered with related knowledge, and thus used as one quiz. Following (Radhakrishnan et al. 2023; Zhou et al. 2023), we adopt the fixed demonstrations, where we select the questions considering their answer and reasoning types for diversity, and manually write the quizzes for quality. As the follow-up quizzes may depend on results of previous ones, we use the number of the quiz to represent its answer in question decomposition, and recover to the answer in reasoning. For example, the question Q in Figure 2 is decomposed into two quizzes: q_1 "who did Richard Callaghan ..." and q_2 "what is the birth date of #1". The process can be formally represented as

$$\{q_1, q_2, \cdots, q_s\} \leftarrow GenQ(Q, \mathcal{L}). \tag{1}$$

Note that typical question decomposition methods (Zhou et al. 2023) mainly focus on planning the reasoning logic, while we aim to generate simple related quizzes to examine knowledge state with decomposition as a suitable technique.

Quiz Assessment. Given the quizzes, we can examine the LLM by investigating whether the LLM could answer the quizzes. However, we have no references such as the ground truth, external knowledge, or generation probability (which may be unavailable in black-box LLM) to assess the predicted answer. Therefore, inspired by self-consistency (Wang et al. 2023a), we assume that the LLM could output consistent answers with the same meanings in multiple tries if it masters related knowledge; otherwise, the LLM would randomly guess and output different answers. Along this line, another challenge is to judge whether two answers are consistent (i.e., have the same meaning), as the same answer may have different literal contents. For example, in Figure 2, "June 10, 1982" and "June, 1982" are consistent answers to q_2. We should compare the answers based on semantics rather than literal contents.

Formally, given the quizzes for Q, we prompt the LLM $\mathcal{L}$ to answer each q_i without external knowledge:

$$a_i \leftarrow Quiz(q_i, \mathcal{L}). \tag{2}$$

To assess the answer a_i, we inspect whether the LLM could reach a consistency in N_c tries, i.e., the proportion T_c of the consistent answers reaches the threshold α. To achieve the goal, we first judge whether two answers a_i and a_j are consistent to the quiz q with a semantic discriminator $\mathcal{D}_s$ such as LLaMA (Touvron et al. 2023) for computation efficiency:

$$same \leftarrow Sim(q, a_i, a_j, \mathcal{D}_s). \tag{3}$$

For example, "June 10, 1982" and "June, 1982" are consistent to q_2, but "June, 1982" and "June 1977" are not. After that, we design an efficient algorithm to assess the consistency score T_c of N_c answers in Algorithm 1. Given the consistent answer A_c and consistency score T_c from the algorithm, if $T_c \geq alpha$, $\mathcal{L}$ is considered to master knowledge related to q; otherwise, $\mathcal{L}$ has not mastered related knowledge, and needs to retrieved from external source $\mathbb{K}$. For example, in Figure 2, $\mathcal{L}$ generates 5 answers for q_1, and 4 of them are "Tara Lipinski" which account for $T_c = 0.8 \geq \alpha$ (Assume $\alpha = 0.8$), so $\mathcal{L}$ masters knowledge related to q_1; while in the 5 answers for q_2, only 2 of them are consistent ("June 1997" or "June, 1982") accounting for $T_c = 0.4 < \alpha$, so $\mathcal{L}$ has not mastered related knowledge to q_2.

Compared with existing self-consistency methods (Wang et al. 2023a) that mainly concern the final answer, KEQA uses it as a metric for intermediate knowledge evaluation and further considers the open answers.

Algorithm 1: Consistency-based assessment

Input: Quiz q, LLM $\mathcal{L}$, semantic discriminator $\mathcal{D}_s$
Parameter: Number of tries N_c
Output: Answer A_c, consistency score T_c

$CS \leftarrow \emptyset$
for $i = 1 \rightarrow N_c$ **do**
  $a_i \leftarrow Quiz(q, \mathcal{L})$
  for $AS \in CS$ **do**
    $same \leftarrow Sim(q, a_i, AS[0], \mathcal{D}_s)$
    if $same == True$ **then**
      $AS \leftarrow AS \cup \{a_i\}$
      break
    end if
  end for
  if a_i does not match any $AS \in CS$ **then**
    $CS \leftarrow CS \cup \{\{a_i\}\}$
  end if
end for
$AS_c \leftarrow \max_{AS \in CS} |AS|$
$A_c \leftarrow AS_c[0], T_c \leftarrow |AS_c|/N_c$
return A_c, T_c

3.4 Utility-Guided Knowledge Picking

To reduce retrieval and promote efficiency, we only retrieve the unknown knowledge that $\mathcal{L}$ fails from external source $\mathbb{K}$. Most retrievers maximize the relevancy of the retrieved knowledge, but can not ensure they help in QA (as shown in Figure 1), so we need to further evaluate the *utility* of each knowledge and pick the helpful ones. As it may be difficult to directly tell the utility of the knowledge in answering the question, we design a metric to evaluate the utility based on the reasoning outputs. We construct a demonstration set in the training data for reference, and use it to guide the utility evaluation in inference as shown in Figure 2(b).

Utility Reference. Helpful knowledge can increase the correctness of the output, so we define the utility of knowledge K' to question Q as whether K' increases the correctness of the predicted answer Y' for Q. We can compute the utility given the true answer in the training data. For QA pair $(\overline{Q}, \overline{Y}) \in \mathbb{D}_{train}$, we first retrieve candidate knowledge for $\overline{Q}$ as $\overline{\mathbb{K}}' \leftarrow \mathcal{R}(\overline{Q}, \mathbb{K})$, and let $\mathcal{L}$ predict the answer $\overline{Y}'$ to $\overline{Q}$ with each $\overline{K}' \in \overline{\mathbb{K}}'$. We measure the correctness of $\overline{Y}'$ with

Algorithm 2: KEQA inference

Input: Question Q, LLM $\mathcal{L}$, knowledge source $\mathbb{K}$, utility reference $\mathbb{R}$, knowledge retriever $\mathcal{R}$, reference retriever $\mathcal{R}_u$, semantic discriminator $\mathcal{D}_s$, utility discriminator $\mathcal{D}_u$
Parameter: Number of tries N_c, consistency threshold α
Output: Explanation E, answer Y

$QS \leftarrow GenQ(Q, \mathcal{L})$
$QAS \leftarrow \emptyset$
for $q \in QS$ **do**
  $A_c, T_c \leftarrow Consist\_assess(q, \mathcal{L}, \mathcal{D}_s, N_c)$
  if $T_c \geq \alpha$ **then**
    $QAS \leftarrow QAS \cup \{(q, A_c)\}$
  else
    $\mathbb{K}' \leftarrow \mathcal{R}(q, \mathbb{K})$
    $\mathbb{K}^* \leftarrow \emptyset$
    for $K' \in \mathbb{K}'$ **do**
      $\mathbb{R}' \leftarrow \mathcal{R}_u(q, K', \mathbb{R})$
      if $Util(q, K') == 1$ **then**
        $\mathbb{K}^* \leftarrow \mathbb{K}^* \cup \{K'\}$
      end if
    end for
    $a \leftarrow RAG(q, \mathbb{K}^*, \mathcal{L})$
    $QAS \leftarrow QAS \cup \{(q, a)\}$
  end if
end for
$E, Y \leftarrow Summ(Q, QAS, \mathcal{L})$
return E, Y

a metric $\mathcal{M}$ based on the true answer $\overline{Y}$. The utility label of $\overline{K}'$ to question $\overline{Q}$ can be formulated as:

$$U(\overline{Q},\overline{K}') \leftarrow \begin{cases} 1 & Cor(\overline{Q}, \overline{\mathbb{K}}^* \cup \{\overline{K}'\}) > Cor(\overline{Q}, \overline{\mathbb{K}}^*) \\ 0 & Cor(\overline{Q}, \overline{\mathbb{K}}^* \cup \{\overline{K}'\}) = Cor(\overline{Q}, \overline{\mathbb{K}}^*) \\ -1 & Cor(\overline{Q}, \overline{\mathbb{K}}^* \cup \{\overline{K}'\}) < Cor(\overline{Q}, \overline{\mathbb{K}}^*) \end{cases},$$
$$Cor(\overline{Q}, \overline{\mathbb{K}}^*) = \mathcal{M}(\overline{Y}', \overline{Y}) = \mathcal{M}(Ans(\overline{Q}, \overline{\mathbb{K}}^*, \mathcal{L}), \overline{Y}), \tag{4}$$

where $\overline{\mathbb{K}}^*$ is knowledge set used to predict $\overline{Y}'$. We examine each $\overline{K}' \in \overline{\mathbb{K}}'$ in descending relevancy order, and include previously confirmed helpful knowledge in $\overline{\mathbb{K}}^*$. $\mathcal{M}$ can be any metrics, and in this paper we adopt the widely-used F1 score. For example, in Figure 2, $\overline{K}'_1$ decreases the correctness of $\overline{Y}'_1$ compared with $\overline{Y}'$ (without $\overline{K}'_1$), so it is helpless; while $\overline{K}'_2$ increases the correctness and is thus helpful.

As samples with $U(\overline{Q}, \overline{K}') = 0$ contain less information ($\overline{K}'$ is helpless, or $\overline{\mathbb{K}}^*$ has contained sufficient knowledge), we only reserve samples with $U(\overline{Q}, \overline{K}') \in \{1, -1\}$ as reference $\mathbb{R}$ to guide utility evaluation in inference:

$$\mathbb{R} \leftarrow \{(\overline{Q}, \overline{K}', U(\overline{Q}, \overline{K}')) | U(\overline{Q}, \overline{K}') \in \{1, -1\}\}. \tag{5}$$

Utility Evaluation. In inference, for each quiz q that $\mathcal{L}$ fails in QKE, we first retrieve candidate knowledge set $\mathbb{K}'$ from external source $\mathbb{K}$ with the retriever $\mathcal{R}$ (e.g., BM25) and query q as $\mathbb{K}' \leftarrow \mathcal{R}(q, \mathbb{K}) \subset \mathbb{K}$. After that, we evaluate the utility of each $K' \in \mathbb{K}'$ by using a utility discriminator $\mathcal{D}_u$ (e.g., LLaMA) with demonstrations $\mathbb{R}'$ from $\mathbb{R}$. We search $\mathbb{R}'$ by maximizing the BERT-based semantic similarity to (q, K') as $\mathcal{R}_u$, and adopt FAISS (Douze et al. 2024) to accelerate computing. The process can be formulated as:

$$\begin{aligned} Util(q, K') &\leftarrow Utility(q, K', \mathbb{R}', \mathcal{D}_u), \\ \mathbb{R}' &\leftarrow \mathcal{R}_u(q, K', \mathbb{R}) \subset \mathbb{R}. \end{aligned} \tag{6}$$

We only pick the retrieved knowledge $K' \in \mathbb{K}'$ that is helpful with $Util(q, K') = 1$ to answer the quiz q with $\mathcal{L}$:

$$a \leftarrow RAG(q, \mathbb{K}^*, \mathcal{L}), \mathbb{K}^* = \{K' \in \mathbb{K}' | Util(q, K') = 1\}. \tag{7}$$

After answering all quizzes $q_i \in GenQ(Q, \mathcal{L})$, we summarize all answers a_i to quizzes q_i, and conclude one overall explanation E and final answer Y to the question Q with $\mathcal{L}$:

$$E, Y \leftarrow Summ(Q, \{(q_i, a_i)\}, \mathcal{L}). \tag{8}$$

The whole inference process of the proposed KEQA framework is demonstrated in Algorithm 2.

4 Experiments

In this section, we conduct extensive experiments on four QA benchmarks to evaluate KEQA framework. [1]

4.1 Experimental Setup

We use four benchmarks for QA including both one-hop and multi-hop QA tasks. We use the NaturalQuestions (NQ) (Kwiatkowski et al. 2019) for one-hop QA, and StrategyQA (Geva et al. 2021), HotpotQA (Yang et al. 2018) and 2WikiMultihopQA (2WMQA) (Ho et al. 2020)for multi-hop QA. As the test data in these datasets do not contain ground truth annotations, we use the train data of StrategyQA and dev data of other datasets, and sample 500 instances for each dataset to reduce the costs of running experiments following previous work (Trivedi et al. 2023; Jiang et al. 2023).

We use accuracy (ACC) to evaluate the performances on StrategyQA with only yes-or-no questions, and F1 score and Exact Match (EM) on other datasets with open answers following (Jiang et al. 2023; Trivedi et al. 2023). Note that answers of these datasets are quite short, so similarity metrics for long answers like BLEU are not quite appropriate, and the word-level F1 score actually performs a similar 1-gram evaluation for the short answers.

To implement the KEQA framework, we use gpt-3.5-turbo[2] as the LLM $\mathcal{L}$, and BM25 algorithm implemented in Elasticsearch[3] as the retriever $\mathcal{R}$ following (Jiang et al. 2023; Trivedi et al. 2023). We use Wikipedia dump from Dec 20, 2018 in (Karpukhin et al. 2020) as the knowledge source $\mathbb{K}$ following (Jiang et al. 2023; Asai et al. 2024). For the semantic and utility discriminator $\mathcal{D}_s$ and $\mathcal{D}_u$, we both adopt Llama-2-7b-chat-hf[4]. Reference retriever $\mathcal{R}_u$ is implemented with Bert and FAISS. In QKE, we set N_c and

[1] Our codes are available at https://github.com/l-xin/KEQA.
[2] https://openai.com/api/
[3] https://www.elastic.co/elasticsearch
[4] https://huggingface.co/meta-llama/Llama-2-7b-chat-hf

Dataset	NQ		StrategyQA	HotpotQA		2WMQA	
Metric	F1	EM	ACC	F1	EM	F1	EM
Vanilla GPT-3.5	0.427	0.294	0.468	0.380	0.264	0.313	0.224
Zero-shot CoT	0.454	0.296	0.510	0.353	0.260	0.320	0.218
Few-shot CoT	0.445	0.292	0.620	0.373	0.254	0.360	0.224
Vanilla RAG	0.385	0.258	0.516	0.387	0.254	0.314	0.244
ReAct	0.335	0.212	0.554	0.390	0.270	0.305	0.204
IRCoT	0.344	0.216	0.622	0.361	0.232	0.318	0.202
FLARE	0.455	0.318	0.662	0.391	0.268	0.364	0.246
Self-Rag	0.387	0.270	0.632	0.357	0.220	0.311	0.210
SearChain	0.337	0.214	0.616	0.349	0.216	0.313	0.222
Rowen	0.452	0.286	0.666	0.382	0.240	0.307	0.212
SlimPLM	0.442	0.280	0.566	0.393	0.266	0.368	0.242
KEQA	**0.483***	**0.352***	**0.680***	**0.400***	**0.278***	**0.405***	**0.326***
KEQA *w/o* QKE	0.409	0.284	0.644	0.352	0.232	0.396	0.258
KEQA *w/o* UKP	0.453	0.302	0.678	0.350	0.250	0.398	0.314
KEQA *w/o* $\mathbb{R}$	0.456	0.316	0.676	0.356	0.252	0.398	0.302
KEQA *w* random $\mathbb{R}'$	0.474	0.324	0.666	0.375	0.262	0.385	0.288
KEQA *w* SE	0.475	0.342	0.678	0.388	0.272	0.397	0.316

Table 1: Overall results on four datasets. Existing state-of-the-art results are underlined, and the best results are **bold**. * indicates a p-value < 0.05 in the paired t-test with the strong baseline.

α for consistency to 5 and 0.8. In UKP, we retrieve top-10 candidate knowledge from $\mathbb{K}$ before knowledge picking, and top-8 demonstrations as $\mathbb{R}'$ from $\mathbb{R}$. We run all experiments on a Linux server with two 2.20 GHz Intel Xeon E5-2650 CPUs and an NVIDIA A100 GPU.

We compare KEQA with GPT-3.5 on direct prompt, zero-shot CoT (Kojima et al. 2022) and few-shot CoT (Wei et al. 2022). For RAG baselines, we adopt the vanilla RAG and seven advanced RAG methods, i.e., ReAct (Yao et al. 2023), IRCoT (Trivedi et al. 2023), FLARE (Jiang et al. 2023), Self-Rag (Asai et al. 2024), SearChain (Xu et al. 2024), Rowen (Ding et al. 2024), and SlimPLM (Tan et al. 2024). We rerun all baseline methods under the same settings (e.g., LLM and retriever) of KEQA for fair comparison.

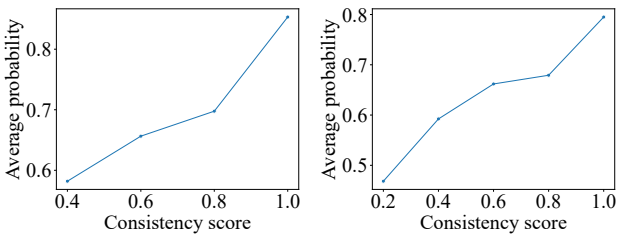
4.2 Experimental Results

Overall Results. We compare KEQA with all baselines, and report the results in Table 1. From the results, there are several observations. First, KEQA outperforms all baselines, which demonstrates the effectiveness of the proposed method. The proposed method can precisely examine the knowledge state of the LLM and augment LLM with helpful knowledge to promote the performances. We statistically test the improvements over strong baselines with paired t-test, and find the improvements to be significant with $p < 0.05$. Second, generally there is an obvious gap between the F1 and EM scores due to the open answers, where similar answers are treated totally different on EM. Third, we find that RAG methods do not always outperform the non-RAG baselines especially on simple tasks, which may be due to the noises in retrieval results. In most cases, RAG performs better due to more information. Last, generally adaptive RAG methods which dynamically determine whether to retrieve (e.g., FLARE) perform better than those always conduct retrieval (e.g., IRCoT), which proves the effectiveness of retrieval-on-demand to avoid noises.

Dataset	NQ	StrategyQA
LLaMA $\mathcal{D}_s$	0.611	0.774
BERTScore	0.559	0.671

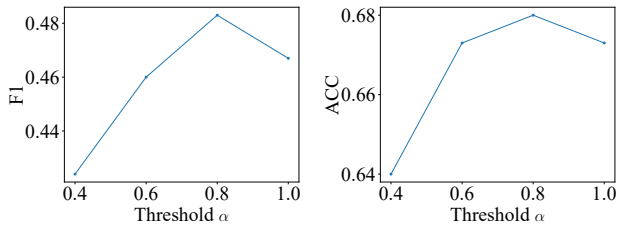
Table 2: Performances of LLaMA $\mathcal{D}_s$ and BERTScore.

Ablation Study. We introduce five variants of KEQA to investigate the effectiveness of each module. KEQA *w/o* QKE treats all knowledge to be unknown and performs UKP on all quizzes. KEQA *w/o* UKP uses all retrieved results for failed quizzes. KEQA *w/o* $\mathbb{R}$ absolutely uses $\mathcal{D}_u$ for utility evaluation without $\mathbb{R}$. KEQA *w* random $\mathbb{R}'$ randomly samples demonstrations $\mathbb{R}'$ from $\mathbb{R}$. KEQA *w* SE further extracts key sentences from passages in addition to KEQA. We also report the performances of the variants in Table 1. We can get the following conclusions. First, the performances decrease when each module is missing, proving their effectiveness. Second, the performances decrease the most when QKE is missing, which proves that precise knowledge evaluation and retrieve-on-demand is a key in RAG to avoid noises. Next, KEQA *w/o* $\mathbb{R}$ performs better than KEQA *w/o* UKP, which shows that $\mathcal{D}_u$ can also filter out some noises, and the similar demonstrations from $\mathbb{R}$ further enhances its ability. However, the randomly sampled demonstrations may degrade the performances, and even mislead $\mathcal{D}_u$. Last, KEQA *w* SE performs slightly worse than KEQA, which may be due to key sentences not extracted from the passages, proving that the passage is a suitable knowledge



(a) NQ (b) StrategyQA

Figure 3: Average probabilities over consistency scores T_c.



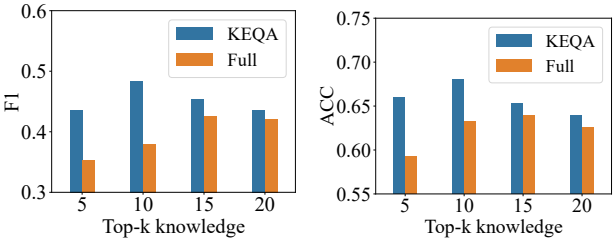
(a) NQ (b) StrategyQA

Figure 4: Performances over consistency score threshold α.

granularity for RAG.

Quiz Analysis. In this section, we investigate the rationality and effectiveness of the quiz-based knowledge evaluation. Due to space limitation, we only report results on NQ and StrategyQA here. We first compute the average probabilities of generating the answers to quizzes over different consistency scores T_c, and report the results in Figure 3. From the results, we can find that with increasing consistency score, the generation probability keeps increasing. It means that generally a higher consistency score implicates a higher generation probability of the answer to the quiz, i.e., the LLM has mastered related knowledge and is more confident on the outputs. Therefore, the consistency score is a reasonable metric to evaluate the knowledge state of the LLM and determine whether to retrieve.

We also investigate what consistency score can implicate that the LLM has mastered related knowledge, i.e., the effects of threshold α on the performances. We change the threshold α from 0.4 to 1.0 and observe the performances of KEQA in Figure 4. From the results, we can find the following observations. First, when the threshold is lower (from 0.4 to 0.8), the performances increase with growing threshold. It is reasonable as the low consistency score means that the LLM may not master related knowledge, which may be missed if not retrieved from external source. Next, when the threshold is high enough, the performances stop increasing and begin to decrease. The reason may be that some accidental factors cause the LLM to give unexpected outputs on confident quizzes and thus not reaches the threshold, and the unnecessary retrieval introduces some noises to the reasoning. It is important to keep a balance between sufficient



(a) NQ (b) StrategyQA

Figure 5: Performances over top-k knowledge.

Dataset	NQ	StrategyQA
KEQA *w* DPR	0.491	0.690
RAG *w* DPR	0.486	0.662

Table 3: Performances of KEQA and RAG with DPR.

confidence and acceptable errors in knowledge evaluation.

We further study the effectiveness of the semantic discriminator $\mathcal{D}_s$ for quiz assessment. We use the GPT-3.5 as an ideal discriminator to annotate the consistency label between answers as ground truth, and evaluate the LLaMA-based $\mathcal{D}_s$ compared with BERTScore as shown in Table 2. From the results, we can see that the LLaMA $\mathcal{D}_s$ could achieve an acceptable performance on discriminating the consistency compared with BERTScore. More complex techniques, such as larger base model, fine-tuned method and in-context demonstrations, can be used for further improvement.

Retrieval Analysis. In this section, we study th effect of the retrieval on the performances of KEQA. We first investigate how the retrieved knowledge could affect the performances. We change the top-k knowledge retrieved from $\mathbb{K}$ before knowledge picking from 5 to 20, and report the results of KEQA (i.e. KEQA) and a variant of KEQA that retrieves for all quizzes and uses all retrieval results (i.e., Full) in Figure 5. We can draw the following findings from the results. First, the top-k knowledge does affect the results of KEQA, but the influences may be limited. KEQA could pick a minor helpful knowledge from the results and discard the rest, so the number of helpless knowledge have little impact on KEQA. Second, KEQA stably outperforms the variant, which proves the effectiveness of the knowledge evaluation and knowledge picking in reducing the misleading noises. Last, more knowledge does not mean to increase the performances even on the variant, as the retrieved knowledge tends to be helpless and even irrelevant with growing retrieval counts, which may mislead the reasoning rather than augmenting the generation.

We also study whether KEQA could work with more powerful retrievers. We implement $\mathcal{R}$ with the widely-used dense retriever DPR (Karpukhin et al. 2020), and compare the performances of KEQA and vanilla RAG in Table 3. We can get the following observations. First, comparing the performances of KEQA and RAG on DPR and BM25, we can

Dataset	NQ	StrategyQA
KEQA *w* GPT-4	0.515	0.760
RAG *w* GPT-4	0.496	0.733
KEQA *w* LLaMA	0.315	0.486
RAG *w* LLaMA	0.295	0.446

Table 4: Performances of KEQA with GPT-4 and LLaMA.

Dataset	NQ				StrategyQA			
Cost	$\mathcal{R}$	$\mathcal{L}$	token	time	$\mathcal{R}$	$\mathcal{L}$	token	time
KEQA	0.138	5.08	239	6.28	0.212	16.44	1016	17.21
IRCoT	1.46	1.46	2024	2.67	1.88	1.88	2374	3.15
FLARE	0.81	1.81	1367	2.45	1.22	2.22	1742	3.45

Table 5: Average retrieval and LLM costs for each question.

find that DPR performs better than BM25 with both KEQA and DPR, which shows the importance of retrieval quality in RAG. Next, KEQA outperforms the vanilla RAG with DPR and BM25, proving that KEQA could also work on more powerful retrievers to promote RAG.

Generalizability Analysis. We hope to study whether the KEQA could work on LLMs with different abilities. We adopt the more powerful GPT-4 and less powerful LLaMA 7B, and compare the performances of KEQA and vanilla RAG in Table 4. The observations are as follows. First, GPT-4 performs much better than GPT-3.5 on both KEQA and vanilla RAG, and GPT-3.5 outperforms LLaMA, which proves that the ability of the LLM is the key to the performances of RAG. Next, KEQA outperforms vanilla RAG on all three LLM backbones, which proves the generalizability of KEQA to work with LLMs with different abilities.

Efficiency Analysis. Finally, we investigate the costs of retrieval and LLM in KEQA, and compare with two advanced RAG, IRCoT and FLARE in Table 5. We represent the retrieval cost as the number of retrieval calls ($\mathcal{R}$), and LLM cost as the number of LLM calls ($\mathcal{L}$) and tokens (token). We also include the latency of the whole reasoning process (time). From the results, we can get the following observations. First, we can obviously find that KEQA requests much less retrieval, which can not only reduce costs of retrieval, but also avoid noises in the retrieval results. Next, although the LLM calls of KEQA are larger, these LLM calls are majorly simple knowledge quizzes without complex reasoning, so the inputs and outputs can be quite short without heavy demonstration payload and long generation. Therefore, the overall tokens are much less. Last, KEQA seems to cost much more time compared with IRCoT and FLARE. We study the time spent on each step in detail, and find that quiz answering and assessment take up most time for 11.75s and 2.46s, while all other steps take only 2.93s. Note that the most time-consuming quiz answering and assessment can be fully parallelized for up to 5 times speedup. Therefore, although KEQA has much higher latency as shown in the table, it could achieve no much higher latency if optimized in parallel. In summary, our method can achieve high efficiency on both retrieval and LLM.

Question	Which film has the director who was born earlier, The Assassination Of Trotsky or My Life Is Hell?
KEQA	The director of The Assassination Of Trotsky, Joseph Losey, was born on January 14, 1909. The director of My Life Is Hell, *Josiane Balasko*, was born on April 15, 1950. Joseph Losey was born earlier than Josiane Balasko. So the answer is The Assassination Of Trotsky.
q_1: Who is the director of The Assassination Of Trotsky?	
a_1: Joseph Losey	
q_2: When was Joseph Losey born?	
a_2: January 14, 1909	
q_3: Who is the director of My Life Is Hell?	
a_3: *Josiane Balasko*	
Retrieval: My Life Is Hell is a 1991 French comedy film directed by Josiane Balasko ...	
q_4: When was the director of Josiane Balasko born?	
a_4: April 15, 1950	
q_5: Is January 14, 1909 earlier than April 15, 1950?	
a_5: Yes	

Table 6: One case generated by KEQA.

Case Study. We also provide a case in Table 6 to demonstrate the reasoning steps of KEQA, and mark the information from retrieved knowledge in italic. KEQA could detect that the LLM does not know the director of My Life Is Hell and refer to external knowledge for the answer "Josiane Balasko". For other knowledge that the LLM has already known, KEQA does not perform retrieval to reduce costs and avoid noises. Therefore, KEQA could achieve effective and efficient RAG for complex QA.

5 Conclusion

In this paper, we proposed a novel Question Answering with Knowledge Evaluation (KEQA) framework to promote knowledge retrieval and reasoning in question answering. We designed the quiz-based knowledge evaluation to generate knowledge-related quizzes and evaluate whether the LLM mastered knowledge required in reasoning. After that, we retrieved the unknown knowledge from external source, and evaluated their utility in reasoning to pick helpful ones to answer the question. Experimental experiments on four QA datasets demonstrated that the proposed KEQA could reach higher performances, and achieve high efficiency on both retrieval and LLM reasoning.

Acknowledgments

This research was partially supported by grants from the National Natural Science Foundation of China (No.62477044, U20A20229), and the Key Technologies R&D Program of Anhui Province (No.202423k09020039), and the Fundamental Research Funds for the Central Universities.

References

Achiam, J.; Adler, S.; Agarwal, S.; Ahmad, L.; Akkaya, I.; Aleman, F. L.; Almeida, D.; Altenschmidt, J.; Altman, S.; Anadkat, S.; et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Asai, A.; Wu, Z.; Wang, Y.; Sil, A.; and Hajishirzi, H. 2024. Self-RAG: Learning to Retrieve, Generate, and Critique through Self-Reflection. In *The Twelfth International Conference on Learning Representations*.

Bian, N.; Han, X.; Chen, B.; and Sun, L. 2021. Benchmarking knowledge-enhanced commonsense question answering via knowledge-to-text transformation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, 12574–12582.

Cui, Y.; Chen, Z.; Wei, S.; Wang, S.; Liu, T.; and Hu, G. 2017. Attention-over-Attention Neural Networks for Reading Comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 593–602.

Dhuliawala, S.; Komeili, M.; Xu, J.; Raileanu, R.; Li, X.; Celikyilmaz, A.; and Weston, J. E. 2024. Chain-of-Verification Reduces Hallucination in Large Language Models. In *ICLR 2024 Workshop on Reliable and Responsible Foundation Models*.

Ding, H.; Pang, L.; Wei, Z.; Shen, H.; and Cheng, X. 2024. Retrieve only when it needs: Adaptive retrieval augmentation for hallucination mitigation in large language models. *arXiv preprint arXiv:2402.10612*.

Douze, M.; Guzhva, A.; Deng, C.; Johnson, J.; Szilvasy, G.; Mazaré, P.-E.; Lomeli, M.; Hosseini, L.; and Jégou, H. 2024. The faiss library. *arXiv preprint arXiv:2401.08281*.

Dua, D.; Gupta, S.; Singh, S.; and Gardner, M. 2022. Successive Prompting for Decomposing Complex Questions. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, 1251–1265.

Feng, Z.; Feng, X.; Zhao, D.; Yang, M.; and Qin, B. 2024. Retrieval-generation synergy augmented large language models. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 11661–11665. IEEE.

Geva, M.; Khashabi, D.; Segal, E.; Khot, T.; Roth, D.; and Berant, J. 2021. Did aristotle use a laptop? a question answering benchmark with implicit reasoning strategies. *Transactions of the Association for Computational Linguistics*, 9: 346–361.

Hirschman, L.; Light, M.; Breck, E.; and Burger, J. D. 1999. Deep read: A reading comprehension system. In *Proceedings of the 37th annual meeting of the Association for Computational Linguistics*, 325–332.

Ho, X.; Nguyen, A.-K. D.; Sugawara, S.; and Aizawa, A. 2020. Constructing A Multi-hop QA Dataset for Comprehensive Evaluation of Reasoning Steps. In *Proceedings of the 28th International Conference on Computational Linguistics*, 6609–6625.

Jiang, Z.; Xu, F. F.; Gao, L.; Sun, Z.; Liu, Q.; Dwivedi-Yu, J.; Yang, Y.; Callan, J.; and Neubig, G. 2023. Active Retrieval Augmented Generation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, 7969–7992.

Karpukhin, V.; Oguz, B.; Min, S.; Lewis, P.; Wu, L.; Edunov, S.; Chen, D.; and Yih, W.-t. 2020. Dense Passage Retrieval for Open-Domain Question Answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 6769–6781.

Kojima, T.; Gu, S. S.; Reid, M.; Matsuo, Y.; and Iwasawa, Y. 2022. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35: 22199–22213.

Kwiatkowski, T.; Palomaki, J.; Redfield, O.; Collins, M.; Parikh, A.; Alberti, C.; Epstein, D.; Polosukhin, I.; Devlin, J.; Lee, K.; et al. 2019. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7: 453–466.

Lehnert, W. G. 1978. *The Process of Question Answering: A Computer Simulation of Cognition*. Lawrence Erlbaum Associates.

Lewis, P.; Perez, E.; Piktus, A.; Petroni, F.; Karpukhin, V.; Goyal, N.; Küttler, H.; Lewis, M.; Yih, W.-t.; Rocktäschel, T.; et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33: 9459–9474.

Lin, X.; Huang, Z.; Zhao, H.; Chen, E.; Liu, Q.; Lian, D.; Li, X.; and Wang, H. 2024a. Learning Relation-Enhanced Hierarchical Solver for Math Word Problems. *IEEE Transactions on Neural Networks and Learning Systems*, 35(10): 13830–13844.

Lin, X.; Su, T.; Huang, Z.; Xue, S.; Liu, H.; and Chen, E. 2024b. A Knowledge-Injected Curriculum Pretraining Framework for Question Answering. In *Proceedings of the ACM on Web Conference 2024*, 1986–1997.

Liu, J.; Huang, Z.; Ma, Z.; Liu, Q.; Chen, E.; Su, T.; and Liu, H. 2023a. Guiding Mathematical Reasoning via Mastering Commonsense Formula Knowledge. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 1477–1488.

Liu, J.; Huang, Z.; Xiao, T.; Sha, J.; Wu, J.; Liu, Q.; Wang, S.; and Chen, E. 2024. SocraticLM: Exploring Socratic Personalized Teaching with Large Language Models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Liu, J.; Huang, Z.; Zhai, C.; and Liu, Q. 2023b. Learning by applying: A general framework for mathematical reasoning via enhancing explicit knowledge learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, 4497–4506.

Ma, X.; Gong, Y.; He, P.; Zhao, H.; and Duan, N. 2023. Query Rewriting in Retrieval-Augmented Large Language Models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, 5303–5315.

Madaan, A.; Tandon, N.; Gupta, P.; Hallinan, S.; Gao, L.; Wiegreffe, S.; Alon, U.; Dziri, N.; Prabhumoye, S.; Yang,

Y.; et al. 2023. Self-refine: Iterative refinement with self-feedback. *Advances in Neural Information Processing Systems*, 36.

Radhakrishnan, A.; Nguyen, K.; Chen, A.; Chen, C.; Denison, C.; Hernandez, D.; Durmus, E.; Hubinger, E.; Kernion, J.; Lukošiūtė, K.; et al. 2023. Question decomposition improves the faithfulness of model-generated reasoning. *arXiv preprint arXiv:2307.11768*.

Ren, R.; Wang, Y.; Qu, Y.; Zhao, W. X.; Liu, J.; Tian, H.; Wu, H.; Wen, J.-R.; and Wang, H. 2023. Investigating the factual knowledge boundary of large language models with retrieval augmentation. *arXiv preprint arXiv:2307.11019*.

Sun, Z.; Wang, X.; Tay, Y.; Yang, Y.; and Zhou, D. 2023. Recitation-Augmented Language Models. In *The Eleventh International Conference on Learning Representations*.

Tan, J.; Dou, Z.; Zhu, Y.; Guo, P.; Fang, K.; and Wen, J.-R. 2024. Small Models, Big Insights: Leveraging Slim Proxy Models To Decide When and What to Retrieve for LLMs. *arXiv preprint arXiv:2402.12052*.

Touvron, H.; Martin, L.; Stone, K.; Albert, P.; Almahairi, A.; Babaei, Y.; Bashlykov, N.; Batra, S.; Bhargava, P.; Bhosale, S.; et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Trivedi, H.; Balasubramanian, N.; Khot, T.; and Sabharwal, A. 2023. Interleaving Retrieval with Chain-of-Thought Reasoning for Knowledge-Intensive Multi-Step Questions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 10014–10037.

Wang, L.; Yang, N.; and Wei, F. 2023. Query2doc: Query Expansion with Large Language Models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, 9414–9423.

Wang, X.; Wei, J.; Schuurmans, D.; Le, Q. V.; Chi, E. H.; Narang, S.; Chowdhery, A.; and Zhou, D. 2023a. Self-Consistency Improves Chain of Thought Reasoning in Language Models. In *The Eleventh International Conference on Learning Representations*.

Wang, Y.; Li, P.; Sun, M.; and Liu, Y. 2023b. Self-Knowledge Guided Retrieval Augmentation for Large Language Models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, 10303–10315.

Wei, J.; Wang, X.; Schuurmans, D.; Bosma, M.; Xia, F.; Chi, E.; Le, Q. V.; Zhou, D.; et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35: 24824–24837.

Xu, S.; Pang, L.; Shen, H.; Cheng, X.; and Chua, T.-S. 2024. Search-in-the-Chain: Interactively Enhancing Large Language Models with Search for Knowledge-intensive Tasks. In *Proceedings of the ACM on Web Conference 2024*, 1362–1373.

Xue, S.; Huang, Z.; Lin, X.; Liu, J.; Qin, L.; Su, T.; Liu, H.; and Liu, Q. 2024a. Enhancing the Completeness of Rationales for Multi-Step Question Answering. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, 2753–2763.

Xue, S.; Huang, Z.; Liu, J.; Lin, X.; Ning, Y.; Jin, B.; Li, X.; and Liu, Q. 2024b. Decompose, Analyze and Rethink: Solving Intricate Problems with Human-like Reasoning Cycle. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Yang, Z.; Gan, Z.; Wang, J.; Hu, X.; Lu, Y.; Liu, Z.; and Wang, L. 2022. An empirical study of gpt-3 for few-shot knowledge-based vqa. In *Proceedings of the AAAI conference on artificial intelligence*, volume 36, 3081–3089.

Yang, Z.; Qi, P.; Zhang, S.; Bengio, Y.; Cohen, W.; Salakhutdinov, R.; and Manning, C. D. 2018. HotpotQA: A Dataset for Diverse, Explainable Multi-hop Question Answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 2369–2380.

Yao, S.; Zhao, J.; Yu, D.; Du, N.; Shafran, I.; Narasimhan, K. R.; and Cao, Y. 2023. ReAct: Synergizing Reasoning and Acting in Language Models. In *The Eleventh International Conference on Learning Representations*.

Yin, Z.; Sun, Q.; Guo, Q.; Wu, J.; Qiu, X.; and Huang, X.-J. 2023. Do Large Language Models Know What They Don't Know? In *Findings of the Association for Computational Linguistics: ACL 2023*, 8653–8665.

Yoran, O.; Wolfson, T.; Ram, O.; and Berant, J. 2024. Making Retrieval-Augmented Language Models Robust to Irrelevant Context. In *The Twelfth International Conference on Learning Representations*.

Yu, W.; Iter, D.; Wang, S.; Xu, Y.; Ju, M.; Sanyal, S.; Zhu, C.; Zeng, M.; and Jiang, M. 2023. Generate rather than Retrieve: Large Language Models are Strong Context Generators. In *The Eleventh International Conference on Learning Representations*.

Zheng, L.; Fei, H.; Li, F.; Li, B.; Liao, L.; Ji, D.; and Teng, C. 2024. Reverse multi-choice dialogue commonsense inference with graph-of-thought. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, 19688–19696.

Zhou, D.; Schärli, N.; Hou, L.; Wei, J.; Scales, N.; Wang, X.; Schuurmans, D.; Cui, C.; Bousquet, O.; Le, Q. V.; et al. 2023. Least-to-Most Prompting Enables Complex Reasoning in Large Language Models. In *The Eleventh International Conference on Learning Representations*.