Knowing You Don't Know: Learning When to Continue Search in Multi-round RAG through Self-Practicing

Diji Yang*
dyang39@ucsc.edu
University of California Santa Cruz
Santa Cruz, USA

Linda Zeng*
lindazeng979@gmail.com
The Harker School
San Jose, USA

Jinmeng Rao†
jinmengrao@google.com
Mineral.ai
Mountain View, USA

Yi Zhang
yiz@ucsc.edu
University of California Santa Cruz
Santa Cruz, USA

Abstract

Retrieval Augmented Generation (RAG) has shown strong capability in enhancing language models' knowledge and reducing AI generative hallucinations, driving its widespread use. However, complex tasks requiring multi-round retrieval remain challenging, and early attempts tend to be overly optimistic without a good sense of self-skepticism. Current multi-round RAG systems may continue searching even when enough information has already been retrieved, or they may provide incorrect answers without having sufficient information or knowledge. Existing solutions either require large amounts of expensive human-labeled process supervision data or lead to subpar performance.

This paper aims to address these limitations by introducing a new framework, **SIM-RAG**, to explicitly enhance RAG systems' self-awareness and multi-round retrieval capabilities. To train SIM-RAG, we first let a RAG system self-practice multi-round retrieval, augmenting existing question-answer pairs with intermediate inner monologue reasoning steps to generate synthetic training data. For each pair, the system may explore multiple retrieval paths, which are labeled as successful if they reach the correct answer and unsuccessful otherwise. Using this data, we train a lightweight information sufficiency Critic. At inference time, the Critic evaluates whether the RAG system has retrieved sufficient information at each round, guiding retrieval decisions and improving system-level self-awareness through in-context reinforcement learning.

Experiments across multiple prominent RAG benchmarks show that SIM-RAG is an effective multi-round RAG solution. Furthermore, this framework is system-efficient, adding a lightweight component to RAG without requiring modifications to existing LLMs or search engines, and data-efficient, eliminating the need for costly human-annotated mid-step retrieval process supervision data. [1]

CCS Concepts

• **Information systems → Question answering**; **Language models**.

Keywords

retrieval augmented generation, inner monologue, large language models, question answering, multi-round retrieval

ACM Reference Format:
Diji Yang, Linda Zeng, Jinmeng Rao, and Yi Zhang. 2025. Knowing You Don't Know: Learning When to Continue Search in Multi-round RAG through Self-Practicing. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '25), July 13–18, 2025, Padua, Italy.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3726302.3730018

1 Introduction

Large language models (LLMs) have shown decent results in multi-step reasoning benchmarks, such as mathematical competition [25], yet Retrieval Augmented Generation (RAG) systems still lag behind human performance in complex tasks that involve multi-round retrieval [36]. One of the major challenges in RAG is the need for strong self-awareness of its knowledge boundaries. In a closed-book reasoning setting, all knowledge is embedded inside the LLM and remains inherently static, regardless of how the problem is decomposed or how a Chain-of-Thought (CoT) is structured. In contrast, RAG involves external augmented information accessed through retrieval, potentially shifting the system's internal knowledge boundary. Furthermore, retrieval adds additional complexity and uncertainty, which may accumulate over extended reasoning sequences in multi-round RAG systems.

Human intelligence addresses this issue through meta-cognition (i.e., knowing when you don't know) [4, 21]. Humans can continuously assess their knowledge boundaries and adapt their search behaviors as needed in dynamic information environments, such as when using a search engine. After reviewing retrieved results at each time point, humans assess whether sufficient information has been gathered, decide whether further search is necessary, and issue new queries to better address the current information needs.

Meta-cognition is challenging for LLMs due to their noise sensitivity and limited self-awareness of knowledge boundaries [10, 21]. As illustrated in Figure 1, systems that rely on an LLM to decide

[*] Equal contribution.
[†] Now at Google DeepMind.
[1] All code and data are available at https://github.com/ucscirkm/SIM-RAG.

SIGIR '25, Padua, Italy.
© 2025 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-1592-1/25/07
https://doi.org/10.1145/3726302.3730018

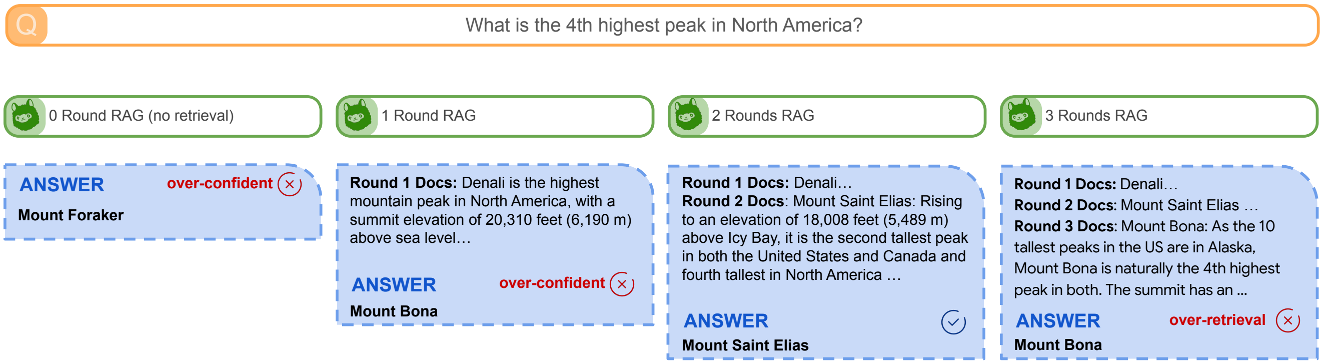
Q
What is the 4th highest peak in North America?
0 Round RAG (no retrieval)
1 Round RAG
2 Rounds RAG
3 Rounds RAG
ANSWER
over-confident
Mount Foraker
Round 1 Docs: Denali is the highest mountain peak in North America, with a summit elevation of 20,310 feet (6,190 m) above sea level…
ANSWER
over-confident
Mount Bona
Round 1 Docs: Denali…
Round 2 Docs: Mount Saint Elias: Rising to an elevation of 18,008 feet (5,489 m) above Icy Bay, it is the second tallest peak in both the United States and Canada and fourth tallest in North America …
ANSWER
Mount Saint Elias
Round 1 Docs: Denali…
Round 2 Docs: Mount Saint Elias …
Round 3 Docs: Mount Bona: As the 10 tallest peaks in the US are in Alaska, Mount Bona is naturally the 4th highest peak in both. The summit has an …
ANSWER
over-retrieval
Mount Bona

Figure 1: A key challenge in multi-round RAG systems [2] is determining the optimal stopping point for retrieval and then generating the answer. This figure illustrates two typical patterns that hurt performance: Over-Confidence (e.g., stopping too early, as seen in 0 and 1 Round RAG, where the system provides an incorrect answer based on limited context) and Over-Retrieval (e.g., retrieving unnecessary information in 3 Rounds RAG which introduces a long and overly complex context that confuses the LLM and leads to incorrect answers.).

the number of retrieval rounds make two types of errors: Over-Confidence, resulting in incorrect answers from insufficient information, and Over-Retrieval, where excessive and distracting information confuses the LLM. Thus, a core problem for multi-round RAG is *knowing you don't know* so that a system can either continue searching only if it is necessary or refrain from answering when the available information is insufficient to support a credible response. As an underexplored problem, recent research either requires a large amount of costly human-labeled supervision data [45] or produces suboptimal performance [1].

The optimization of a RAG system often employs outcome supervision to directly align the initial input with the final output [7, 18]. While outcome supervision with simple question-answer (QA) pairs has proven effective for single-step RAG [2, 11], when an LLM can rapidly learn to map a question to its direct answer or query, it appears inadequate for learning the optimal reasoning path in a multi-round RAG setting, when the answer or next-round query is context-dependent over rounds.

Machine Learning researchers have recently found that process supervision is a promising alternative to outcome supervision for enhancing self-awareness in complex reasoning tasks at the inference time thinking stage [17, 20, 25]. With a well-trained reward model from human-labeled CoT data, optimization can be performed by explicitly supervising intermediate reasoning steps via parameter tuning or training-free verbal reinforcement learning (RL) [23, 31, 44]. An early attempt from the IR community, IM-RAG [45], has explored process supervision for multi-round RAG by simulating the human Inner Monologue reasoning process (i.e., multi-round self-talk within one's mind). It optimizes each mid-step query or answer via actor-critic RL [16]. Despite its strong performance, the training relies on expensive human-annotated supporting documents to generate multi-round reasoning and retrieval training data (i.e., information-seeking chains with labels).

The lack of labeled training data is the main challenge in widely applying process supervision to RAG. Unlike other LLM tasks, such as coding or mathematical reasoning, annotating gold reasoning chains in RAG tasks is difficult because different LLMs may have varied internal knowledge, leading to distinct information needs even in the same context. Thus, human-annotated, LLM-independent information-seeking chains may not align with an LLM's behavior and knowledge, making high-quality multi-round RAG training data costly to label.

This work addresses the labeled data shortage problem when adapting process supervision to multi-round RAG systems. We propose **SIM-RAG** (**S**elf-practicing for **I**nner **M**onologue-based **R**etrieval **A**ugmented **G**eneration), a practical multi-round framework that can be learned through two stages. First, in the *Self-Practicing* stage, we generate synthetic process supervision data by distilling the system's inner monologues and corresponding process labels. This inner monologue captures the system's internal complex reasoning trajectory across its components and can also be interpreted as a form of dynamic reasoning chain. Unlike synthetic data generation using the strongest models, which focuses on producing near-human quality data [37], Self-Practicing generates data reflecting the given AI system's capability. Then, during the *Critic Training* stage, we use the generated data to train a Critic, which is system-specific and context-aware. When a SIM-RAG system is used at inference time, the Critic repeatedly checks the knowledge boundary based on available information and provides process supervision to optimize LLM's behavior via in-context RL [31], alleviating Over-Confidence and Over-Retrieval problems. In summary, our contributions are as follows:

- To simulate human-like meta-cognition in complex reasoning, we propose the **SIM-RAG** framework, which continuously assesses the system's knowledge boundaries and adapts its search behaviors accordingly. By using information sufficiency as a guiding principle for process supervision, SIM-RAG enhances the inference-time thinking capabilities

[2] In this figure, the multi-round RAG refers to a system that relies on LLM to determine how many rounds of retrieval are needed, including 0, 1, or multiple rounds.

of LLMs, enabling dynamic, multi-round retrieval and reasoning for complex tasks.

- To address the data challenges for RAG system training, we introduce the Self-Practicing Algorithm (Algorithm 1). This algorithm generates synthetic training data, providing a lower-cost alternative to human-annotated supporting documents or labeled information-seeking chains, and producing training data that more accurately reflects the given AI system's current capabilities.
- Our experiments on three standard benchmarks show that SIM-RAG is a lightweight and effective solution, capable of robust performance on complex reasoning tasks across diverse question-answering scenarios.

2 Related Work

2.1 Retrieval Augmented Generation

Retrieval Augmented Generation (RAG) enhances large language models (LLMs) by retrieving external knowledge during inference, addressing limitations such as reliance on fixed pre-trained knowledge bases and susceptibility to hallucinations [5]. Pre-trained LLMs often lack up-to-date or domain-specific information, while hallucinations arise when the model generates plausible but incorrect content. By incorporating external retrieval, RAG enables more accurate and grounded responses. In standard RAG systems, also known as single-round RAG, the retrieval process involves using the user's question or an LLM-generated query to search a knowledge base [2, 11, 28]. These systems are effective for tasks with straightforward information needs, where the required information is fully available in a single retrieval step and does not depend on iterative reasoning or multiple rounds of interaction.

However, many real-world tasks involve dynamic and complex reasoning, where the required information cannot be retrieved in a single step. For instance, answering a question may require synthesizing information from multiple documents, clarifying ambiguities, or filling gaps in the initial retrieval. In such cases, single-round RAG systems fall short, as they lack mechanisms to iteratively refine their retrieval and reasoning strategies [38]. This has motivated the development of multi-round retrieval and reasoning systems.

2.2 Multi-Round RAG

Multi-round RAG has shown great potential for tackling tasks involving dynamic and complex reasoning, where iterative interactions with external knowledge sources are necessary to refine responses. However, a core challenge in multi-round RAG is determining information sufficiency—deciding when the retrieved information is adequate to answer the query or if further retrieval steps are required [35, 45]. Existing works on multi-round RAG have explored various techniques to address this challenge.

Training-free system. Training-free methods have gained popularity due to their flexibility and ease of deployment, as they do not require task-specific optimization and can be seamlessly integrated into existing pipelines. These systems rely on the inherent capabilities of LLMs to determine when retrieval should stop. One approach involves reflection-based self-critique [31, 38], where the model evaluates its own knowledge through carefully designed prompts or in-context learning [39, 41]. This technique leverages the strong pre-trained knowledge of LLMs and allows the use of any LLM as a reasoning backbone. However, it is inherently constrained by the limitations of the underlying model, including a tendency to generate hallucinations or over-confident but incorrect responses [33, 46]. Some methods take advantage of the internal states of LLMs, such as token-level attention weights [35] or confidence scores [48], to decide retrieval sufficiency. These signals can offer insights into the model's reasoning process, but often require access to model weights and, consequently, cannot be used with closed-source LLMs. Additionally, the lack of interpretability in latent representations undermines trustworthiness, making these signals less suitable for applications [42] such as healthcare, where trustworthiness plays a critical role.

Recent studies have explored the use of well-trained models, such as GPT-4, as reward models [24]. However, this approach remains constrained by the inherent pre-training biases of the reward model due to the absence of task-specific optimization [34]. More broadly, the lack of training presents a double-edged sword: while it simplifies deployment and enhances ease of use, it simultaneously restricts the potential for further performance improvement

Learnable framework. Several learning frameworks for multi-round RAG have been proposed in the past two years. A recent work trains a classifier to categorize the difficulty of user queries into three classes [12], and applies different retrieval strategies to each class: no retrieval for simple queries, single-step retrieval for medium-difficult queries, and IR-CoT [38] for complex queries. Since this approach primarily focuses on choosing different retrieval strategies, it is not directly comparable to our work. Instead, we used IR-CoT as one of our baselines. Self-RAG [1] is inspired by the concept of Reinforcement Learning with Human Feedback (RLHF) [26]. It first trains a separate critic model using high-quality data to evaluate information sufficiency and then fine-tunes the LLM model in full size, equipping it with self-critique capabilities. Moreover, instead of standard online RLHF, Self-RAG uses outcome supervision [20] to generate sequences of text that include special tokens to trigger retrieval operations. Although this method shows potential, its substantial data and computational costs limit its practicality for deployment and domain-specific adaptation [43].

Recently, the Machine Learning community has shown that process supervision [20], which supervises intermediate reasoning steps, can significantly improve multi-step reasoning tasks compared to outcome supervision [17]. Motivated by these findings, IR researchers proposed IM-RAG [45], a framework that employs reinforcement learning via Proximal Policy Optimization (PPO) [29] to jointly optimize the reasoning chain and retrieval queries (i.e., learning the model's inner monologue) in multi-round RAG. IM-RAG has shown notable performance improvements. However, the method faces challenges in broader applicability due to its reliance on costly annotated support documents to define dataset-specific rule-based reward functions, and it lacks a principled mechanism for determining when to terminate the retrieval process.

2.3 LLM Self-Training for Complex Reasoning

With the increasing need for fresh data, recent advancements in post-training have shifted the focus toward using model-generated

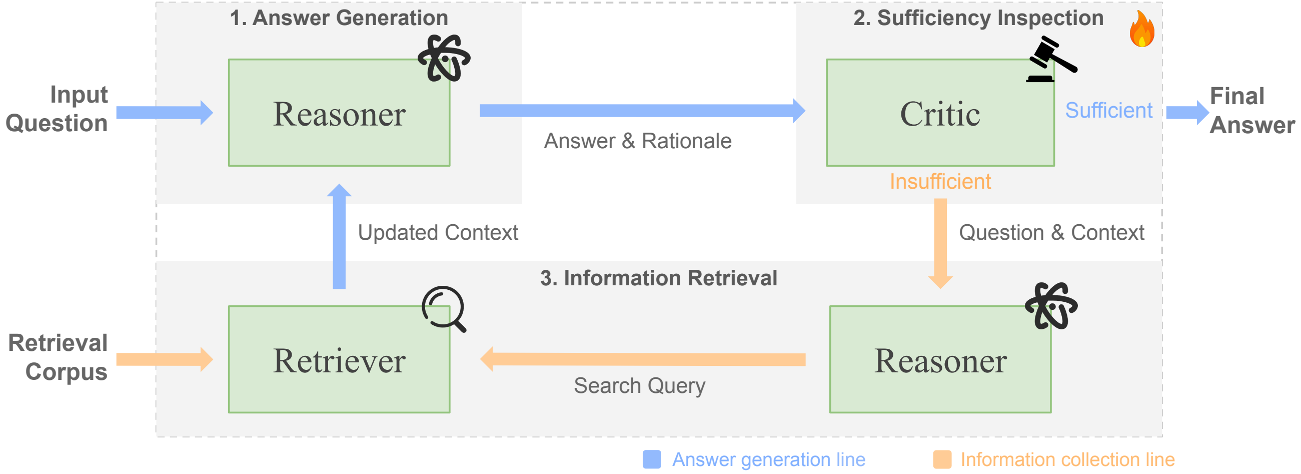

Figure 2: SIM-RAG overview at inference time, consisting of three main components: (1) Answer Generation, where the Reasoner generates an answer and rationale; (2) Sufficiency Inspection, where the Critic decides whether the generated answer to accept the answer or trigger refinement; and (3) Information Retrieval, where, if deemed insufficient, a query is generated to retrieve additional documents from the corpus to update the context. This iterative process continues until the Critic deems the answer sufficient or the maximum number of retrieval rounds is reached. The orange line represents the information collection path, while the blue line represents the answer generation path, visualizing the information flow between components.

data to improve reasoning (i.e., self-training) [9, 32, 50]. In this paradigm, LLMs generate multiple outputs for a given input and use a reward signal to identify and train on high-quality samples [52]. This iterative process enables models to improve their reasoning capability without relying exclusively on human-labeled data.

Adapting self-training to RAG has some new complications compared with pure language tasks (e.g., commonsense reasoning), which rely on static knowledge embedded within an LLM. In a multi-round RAG setting, newly retrieved information changes the knowledge boundary at each turn, the additional information can either support or hinder subsequent reasoning, and errors in early retrieval can propagate through later stages due to the sensitivity of LLMs to noisy contexts [27]. Optimizing self-awareness in RAG systems requires determining whether the current knowledge is adequate (when to retrieve) and issuing effective queries to acquire additional information that meets the current needs (what to retrieve). Because the knowledge boundary changes over turns and noise is added, multi-round RAG requires a higher level of self-awareness in the cognitive hierarchy [21] compared to multi-step reasoning tasks with static knowledge. This work tries to solve these unique challenges RAG systems face during self-training.

Practitioners have used LLM for self-critique when self-training, internalizing the critique capability. However, the research community is questioning whether LLM truly has enough self-awareness to satisfy self-critique [10, 33, 34]. Motivated by these findings, we choose to use an external Critic rather than self-critique. SIM-RAG employs a lightweight Critic that remains unattached from the LLM and is trained on a single task: checking the information sufficiency.

3 Methodology

This section introduces the SIM-RAG framework, outlining its design in Section 3.1. We then provide an in-depth explanation of two core stages for framework training: Self-Practicing for inner monologue distillation and labeling (Section 3.2) and Critic training (Section 3.3). Finally, we elaborate on the overarching rationale for the framework's design and its inference methodology (Section 3.4).

3.1 SIM-RAG

The overview of the SIM-RAG framework is illustrated in Figure 2, following the information flow during the inference-time thinking process. The system comprises three main components: (1) a Reasoner (an LLM) that generates queries or provides answers based on the context; (2) a Retriever (e.g., a search engine) that retrieves documents given the Reasoner's query; and (3) a learnable Critic (a lightweight discriminative model) that identifies when the Reasoner's current knowledge and reasoning are insufficient. As a fully modularized system, SIM-RAG organizes these components into three functional modules, described as follows in the order they are used during inference.

Answer Generation. In this stage, the Reasoner (LLM) receives the initial question or task Q from the user and any available context C from previous retrieval steps (an empty string at turn 0). The Reasoner produces an answer A' and a corresponding rationale r. Although the context is initially empty, the Reasoner benefits in subsequent turns from an accumulated context containing prior search queries and retrieved documents. The Reasoner's objective at this stage is to produce its best-guess answer based on all information currently available. Thus, the model chosen for the Reasoner can be any language model capable of answering questions, provided it can generate an answer and a rationale in natural language that can later be evaluated by the Critic.

Sufficiency Inspection. In complex reasoning problems, humans can continuously assess whether they have sufficient information and a correct answer as they progress through a long reasoning chain. This ability, known as meta-cognition, enables individuals to monitor and support their reasoning throughout the thinking process.

SIM-RAG employs a Critic to simulate a similar meta-cognitive function. After receiving the Reasoner's proposed answer-rationale pair (A', r), the Critic examines them alongside the initial question Q and the retrieved documents contained in the current context C. If the Critic determines that the answer A' is adequately supported by the evidence from (Q, C, r), the system returns A' as the final answer to the user. If the Critic judges A' to be insufficient, due to lack of information, inadequate support from retrieved data, inconsistencies with known facts, or similar issues, the system discards the current attempt and proceeds to the Information Retrieval module. This design helps prevent the propagation of flawed reasoning paths by ensuring that only well-supported answers are returned to the user.

Information Retrieval. After the Critic determines that the Reasoner is unable to answer the question based on all available information, the system triggers the Information Retrieval module. The Reasoner generates a search query q based on the user's question and the current context. This query is then passed to the Retriever, which returns the most relevant external knowledge. Both the search query and the returned documents are appended to the C, which will be fed into the next round of Answer Generation. By integrating newly retrieved information, the Reasoner may be better equipped to converge on a well-supported answer in subsequent iterations. Notably, the Reasoner in this stage is flexible and could be the same or a different LLM used in the Answer Generation block. For simplicity, and without loss of generality, we used the same LLM to generate both queries and answers in our experiments. However, in practice, LLMs optimized for each functionality may result in better performance, as developing a good answer may require different abilities from issuing a good query [45].

Iterative Framework. After updating the context C, the system loops back to the Answer Generation stage, with the newly retrieved information helping to expand the Reasoner's knowledge boundary. This iterative process, consisting of three steps per iteration, simulates a human-like search and reasoning loop that continually reevaluates the adequacy of its current explanation and dynamically seeks additional information as needed. The Answer Generation, Sufficiency Inspection, and Information Retrieval steps repeat until the Critic determines the answer is sufficiently grounded or the maximum number of iterations is reached to prevent an infinite loop. This cyclical, meta-cognitive design aims to maximize the correctness and completeness of the final response.

3.2 Self-Practicing

Our training pipeline starts with collecting training data for the supervised learning of the Critic. Given the Critic's task, the training data should consist of information-seeking chains with accurate labels and include sufficient quantity and diversity, enabling the Critic to learn how to assess mid-step information sufficiency and evaluate the correctness of the current answer in real-world scenarios. However, as discussed in Section 1, information sufficiency is subject to the LLM's knowledge and other available information. Thus, human-labeled, model-agnostic information-seeking chains may not match the real information-seeking behavior of an LLM whose internal knowledge scope does not align with the annotator's. To address this, we propose an approach to generate model-specific and context-aware synthetic data instead of real human-annotated data. Specifically, we let the RAG system self-practice a multi-round retrieval process to find the correct answer for a given question where the target answer is known. During this practice, the real interaction between the Reasoner and the Retriever enables us to collect and label inner monologue data.

Algorithm 1 Self-Practicing Algorithm. **Description.** This algorithm generates labeled inner monologue data by enabling the RAG system to perform self-practice. It automatically searches, attempts to generate answers, and checks whether the generated answer for each sequence of actions is correct.

```
1:  Notation:
2:  $\mathcal{IR}$: Information Retriever
3:  $\mathcal{R}$: Reasoner LLM
4:  $Q$: Question
5:  $A$: Ground-truth Answer
6:  $C$: Empty list to track the context
7:  $q_t$: Search query from LLM at the $t$-th turn
8:  $d_t$: Retrieved document at the $t$-th turn
9:  $r$: Rationale
10: $A'$: LLM generated answer
11: $T$: Maximum number of turns allowed
12: for each data point do
13:     $C \leftarrow \emptyset, i \leftarrow 0$          // initialize context and turn counter
14:     while t < T do
15:         $(A', r) \leftarrow \mathcal{R}(Q, C)$        // generate answer and rationale
16:         if $A' = A$ then
17:             Record $\{Q, C, A', r\}$ with critique label Accept
18:         else
19:             Record $\{Q, C, A', r\}$ with critique label Reject
20:         end if
21:         $t \leftarrow t + 1$
22:         $q_t \leftarrow \mathcal{R}(Q, C)$                 // generate retrieval query
23:         $d_t \leftarrow \mathcal{IR}(q_t)$                    // retrieve document
24:         $C \leftarrow C \cup \{q_t, d_t\}$                    // update context
25:     end while
26: end for
```

As shown in Algorithm 1, the Self-Practicing Algorithm requires only a source dataset containing the initial question Q and the final answer A, which are readily available in most existing QA datasets. The augmented information is tracked by the context list C, initialized as an empty list (line 13). For each data point, an iterative process simulates inner monologue between the Reasoner $\mathcal{R}$ and search engine $\mathcal{IR}$. In the t-th round, given all currently available information (i.e., embedded knowledge of LLM and the augmented retrieved information C), the Reasoner generates an answer A' and rationale r for the question Q (line 15). A correctness check between A and A' determines whether the critique label y for the data tuple $x = \{Q, C, A', r\}$ should be Accept or Reject (line 17 and 19). Regardless of the label, one data point is generated, and the process continues. For each raw (Q, A) pair, the algorithm generates T training data points of varying chain lengths and labels, where T is a predefined maximum number of turns (i.e., rounds of answering). If the exit condition is unmet, the Reasoner is re-prompted with Q and C to generate a search query q_t (line 22). The search engine $\mathcal{IR}$ is then called to retrieve passages/documents d_t given the query q_i (line 23). Next, the context is updated by concatenating the new query and document, $C = C \cup \{q_t, d_t\}$, before the next iteration (line 24).

As a result, given a source dataset with N question-answer pairs, the Self-Practicing Algorithm generates an augmented training set $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{M}$, where $M = N \times T$ is the size of $\mathcal{D}$. The generated data is annotated with current answer correctness (`Accept` or `Reject`) and entails information sufficiency for a given LLM under a multi-round RAG setting. Note that each generated data point simulates a sequence of a RAG system's behavior, with either positive or negative critique labels. This differs from standard LLM knowledge distillation methods, which treat all distilled data points as positive training examples. Moreover, the same raw (Q, A) pair can generate different sequences of RAG behaviors and corresponding labels across iterations when the available information changes. This increases both the quantity and diversity of the synthetic data.

3.3 Reasoning-Enhanced Critic Learning

The Critic's task is formulated as a straightforward binary classification problem for a given input x. While any classification algorithms could be used, we use pre-trained language models in this work due to their superior semantic understanding capabilities. Thus, the Critic training is to tune a generative language model by casting the classification task as a conditional generation problem.

Given the training set $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{M}$ from Section 3.2, we follow the conception of instruction fine-tuning and first update the input x_i to be the concatenation of general language instruction and task-specific input. During the training, the Critic is expected to predict the next token y_i given input x_i; thus, the training objective is to maximize the conditional likelihood of the binary labels (i.e. targeted tokens), as shown in Equation 1, where θ is the language model parameters.

$$\theta^* = \arg\max_{\theta} \log \mathcal{L}(\theta) = \arg\max_{\theta} \sum_{(x_i, y_i) \sim \mathcal{D}} [\log P(y_i | x_i; \theta)] \quad (1)$$

Through task-specific fine-tuning, the Critic quickly learns to make predictions from a large amount of labeled IM-distilled data.

3.4 Reasoning-Enhanced Inference

The goal of this section is to justify and elaborate on how the mechanism introduced in Sections 3.2 and 3.3 improves RAG's output by leveraging the Critic's meta-cognitive feedback. The integration of the trained Critic with the Reasoner and Retriever transforms SIM-RAG into an iterative reasoning framework, enabling it to dynamically refine responses based on feedback. The supervision of the LLM-based Reasoner over multiple rounds is verbal Reinforcement Learning (RL), where the Critic provides supervision in natural language (`Accept` or `Reject`) rather than numerical rewards or gradient updates. Analogous to traditional policy-based RL setups, this verbal reinforcement defines a policy as a combination of the agent's memory encoding and the selected LLM parameters [31]. Within our framework, this approach leverages the strengths of in-context learning [3], as the Reasoner can interpret and incorporate the Critic's feedback directly into its reasoning process by appending it to the input of the Reasoner. This in-context RL mechanism enables the Reasoner to dynamically adapt its behavior and decision-making process based on Q and C, including the supervision provided by the Critic, without requiring explicit parameter updates. This preserves the system's modularity and keeps training lightweight. By grounding iterative refinement in textual feedback, the framework encourages targeted improvements in reasoning paths and retrieval strategies.

From the system design perspective, SIM-RAG separates the Critic from the Reasoner to avoid self-critique bias. During inference time, the iterative cycle of feedback and context updates mirrors human reasoning under uncertainty, where gaps in knowledge are identified and addressed step-by-step.

4 Experiments

4.1 Task and Datasets

To comprehensively evaluate SIM-RAG across tasks of varying reasoning complexity, we conduct experiments on three highly distinct datasets covering single-hop and multi-hop QA tasks. For single-hop QA, we use TriviaQA [15], a widely used benchmark that focuses on factoid questions requiring reasoning over a single piece of evidence from Wikipedia. For multi-hop QA, we use HotpotQA [47] and 2WikiMultiHopQA [8]. HotpotQA requires synthesizing information from multiple documents to answer complex questions, while 2WikiMultiHopQA focuses on distinguishing closely related entities and incorporating fine-grained evidence. These datasets challenge SIM-RAG's capabilities in multi-round retrieval and reasoning over insufficient information. Following the standard evaluation methods, we report Exact Match (EM) and F1 scores for all datasets, using the Wikimedia dumps provided by each dataset as the retrieval corpus.

4.2 Implementation Details

We evaluate two versions of SIM-RAG using Llama3-8B and GPT-4 as the Reasoner. To fine-tune the Critic, we use two Flan-T5 models of different sizes, corresponding to two versions of SIM-RAG: Flan-T5-2.85B for the full version (SIM-RAG$_{full}$) and Flan-T5-783M for the lightweight version (SIM-RAG$_{lite}$). To ensure consistency with other well-known RAG frameworks, we use BM25 with Elasticsearch as the retriever across all experiments. Our pipeline can be replicated with two NVIDIA 3090 GPUs or equivalent hardware. We consider prompts, in-context examples, and the number of documents retrieved as hyperparameters and report them in our open-sourced code base to facilitate reproducibility.

4.3 Baselines

We compared SIM-RAG with eight baseline methods (Table 1). Naive Generation [14] and Standard RAG [14] are two basic methods. Naive Generation fully relies on the internal knowledge of the LLM without using any retrieval, whereas Standard RAG uses the initial question as a query to retrieve documents, which are then used to augment the LLM's response through prompting. We report the

			TriviaQA		HotPotQA		2Wiki	
	Method	LLM/Retriever	EM	F1	EM	F1	EM	F1
			Prompting					
(a)	NaiveGen [14]	Llama3	55.7	63.1	20.0	28.4	26.4	33.9
	StandardRAG [14]	Llama3/E5	58.9	68.3	25.1	35.3	8.6	21.0
	SEAKR [48]	Llama2/BM25	42.6	48.6	27.9	39.7	30.2	36.0
	DRAGIN [35]	Llama2/BM25	54.4	62.3	23.7	34.2	22.4	30.0
			Learned					
	Self-RAG [1]*	Llama3/E5	38.2	53.4	17.1	29.6	12.1	25.1
	Auto-RAG [49]*	Llama3/E5	56.7	60.7	31.9	44.3	38.8	46.9
			Prompting					
(b)	NaiveGen [14]	GPT-4	59.6	70.0	27.5	37.9	23.7	31.7
	StandardRAG [14]	GPT-4/BM25	60.3	65.9	35.7	45.5	28.7	33.8
	FLARE [13]*	Llama3/E5	55.8	63.2	19.7	28.0	25.8	33.9
	IR-CoT [38]*	Llama3/BM25	56.4	68.9	28.6	39.1	23.5	31.8
			Learned					
	SIM-RAG$_{lite}$	Llama3/BM25	69.2	74.5	27.8	37.9	32.1	38.0
	SIM-RAG$_{lite}$	GPT-4/BM25	77.3	82.8	37.0	49.7	44.5	53.3
	SIM-RAG$_{full}$	Llama-3/BM25	70.7	75.6	32.7	43.3	34.1	40.2
	SIM-RAG$_{full}$	GPT-4/BM25	77.5	82.7	39.8	52.2	46.1	54.6

Table 1: Comparison of RAG-based methods on three standard RAG datasets. Group (a): methods that require direct access to model weights; Group (b): methods that can be used with API-based LLMs[3]. Asterisk (*) indicates the results obtained from recent reproduction using newer models.

Llama3 and GPT-4 version baselines for both methods as a reference point for performance comparison. SEAKR [48], DRAGIN [35], Self-RAG [1], Auto-RAG [49], FLARE [13], IR-COT [38] are more advanced multi-round RAG methods. Some baselines are prompt-based, and others are learned.

The baselines include two groups: (a) methods requiring access to an LLM's internal weights and open-source models, and (b) methods that can use API-based closed-source models. In Group (a), SKEKR [48] and DRAGIN [35] do not involve fine-tuning; however, they rely on model internals, such as hidden-layer Gram matrices (SKEKR) or token-level attention weights (DRAGIN), for retrieval. Self-RAG [1] and Auto-RAG [49] fine-tune LLMs to support multi-round retrieval. While Self-RAG originally used Llama2, we report results based on its recent replication [14] using Llama3 for a fair comparison. Compared to SKEKR, DRAGIN, Self-RAG, and Auto-RAG methods, which require access to model weights, SIM-RAG offers greater flexibility without requiring open-source models.

Group (b) focuses on methods applicable to API-based closed-source LLMs, primarily relying on prompting. FLARE [13] utilizes probabilities or confidence scores of the next-generated-tokens to guide retrieval, while IR-CoT [38] interleaves retrieval with intermediate CoT reasoning steps, enabling effective multi-step question answering. For better comparison, we include replication results from [14] using Llama3. Baselines with E5-base-v2 [40] as the Retriever potentially have an advantage due to higher-quality retrieval engines; however, they also require more GPU resources.

4.4 Results

Table 1 summarizes the performance of various methods on three widely used RAG datasets. SIM-RAG$_{full}$ with GPT-4 consistently performs the best in all three datasets, outperforming all baseline methods by a large margin, including Self-RAG and Auto-RAG, which require extensive full model fine-tuning for LLMs. A group-wise comparison further highlights the strengths of SIM-RAG.

For multi-hop QA datasets, when using closed-source GPT-4 models, SIM-RAG$_{full}$ delivers the highest performance for both HotPotQA and 2Wiki. When using open-source Llama models, Auto-RAG performs the highest on 2Wiki, while SIM-RAG$_{full}$ is the highest on HotPotQA. Auto-RAG fine-tunes Llama3-8b for retrieval decisions and uses a learned retrieval E5, whereas our approach only fine-tunes a smaller Critic and uses a simpler BM25 retriever. If reducing computational costs or using closed-source LLM models is a priority, SIM-RAG would be the optimal choice.

In contrast, for single-hop QA, a key observation is that all advanced baselines tailored for multi-round RAG perform poorly on the straightforward TriviaQA dataset. None of them match the performance of Standard RAG, and only Auto-RAG and FLARE outperform Naive Generation. This exposes a critical limitation for these approaches: optimizing for complex multi-round retrieval tasks appears to undermine LLM's capability on simpler tasks. This may be due to the inherent biases of LLMs, particularly their difficulty with effective self-critique, as discussed in Section 2.2. These challenges lead to Over-Confidence or Over-Retrieval, making LLMs less competitive on simpler tasks, where even a standard RAG approach with a fixed number of retrieval steps performs better. In contrast, SIM-RAG uses an external model that specializes in "when to stop the iterative system." This distinction allows SIM-RAG$_{full}$ with Llama3 to outperform Standard RAG with Llama3 by a significant margin (70.7% vs. 58.9%) in the EM metric. Similarly, SIM-RAG$_{full}$ with GPT-4 achieves a 16.0% absolute improvement over the Standard RAG baseline, representing a relative improvement of 26.1%.

The study on Critic size further reveals that SIM-RAG is an effective framework even with a lightweight Critic (783M). For example, on HotPotQA, SIM-RAG$_{lite}$ significantly outperforms Self-RAG (27.8% vs. 17.1%) while using only one-tenth of the training parameters (783M vs. 7B).

These findings indicate that even a lightweight Critic can improve system performance, though SIM-RAG may benefit from a larger and more powerful Critic, particularly for complex multi-hop reasoning tasks.

4.5 Analysis on Critic Predictions

To further evaluate the Critic's prediction accuracy, we report SIM-RAG with GPT-4's confusion matrices of its binary classification performance on TriviaQA, HotPotQA, and 2Wiki datasets (Figure 3). The clear diagonal lines (True Positive and True Negative) highlight the Critic's ability to correctly predict whether to accept or reject a Reasoner's output, based on the ground-truth answer. From the results, the Critic demonstrates strong classification performance across all datasets, with notably high accuracy in rejecting incorrect answers, particularly for HotPotQA (63.9%) and 2Wiki (65.0%).

However, the True Positive and True Negative rates differ significantly between single-hop and multi-hop QA tasks. On TriviaQA,

[3] While Naive Generation and Standard RAG would technically fit in Group (b) (because they can be used via API), we list their Llama-based variants in Group (a) for easier comparison with other Llama-based methods.

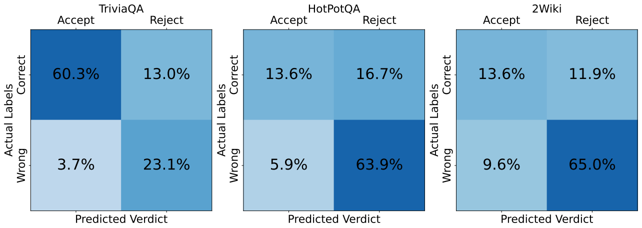

Figure 3: Confusion matrices of Critic predictions in SIM-RAG with GPT-4.

the Critic achieves 60.3% accuracy in correctly accepting answers, while on HotPotQA and 2Wiki, the accuracy drops to 13.6%. These discrepancies are expected and reflect the differing characteristics of the datasets. SIM-RAG demonstrates an ability to adapt across datasets, likely due to the distribution of the synthetically generated training data, which contains a higher proportion of positive examples in TriviaQA compared to the multi-hop datasets.

4.6 Ablation Study and Analysis

Impact of Critic Model Choice. We conducted an ablation study to explore whether a more powerful, general-purpose model with strong QA capabilities could serve as a better Critic. To this end, we replaced Flan-T5-783M with GPT-4 as the Critic in our SIM-RAG$_{lite}$ system, while using GPT-4 as the Reasoner. In this configuration, GPT-4 functions as both the Reasoner and Critic using different prompt settings, an approach commonly referred to as self-critique in literature [6]. The comparison is visualized in Figure 4. Notably, unlike other baselines discussed in Table 1, GPT-4-as-the-Critic achieves strong results on TriviaQA. This suggests that for simple tasks, SIM-RAG may use a more general, powerful model like GPT-4 to achieve comparable outcomes. However, the gap becomes significant for more complex multi-hop tasks. As shown in the bar chart, GPT-4-as-the-Critic underperforms Flan-T5 across both EM and F1 metrics by a substantial margin. This finding aligns with observations in the mathematical reasoning domain [10], where LLMs often struggle to provide reliable self-critiques on tasks involving complex reasoning. We suspect that a general-purpose LLM may be over-confident and generate too many false positives as the Critic. This is a more serious problem for multi-round QA tasks, where the percentage of actual "correct" predictions from the Reasoner is low (as suggested by the confusion matrices in Figure 3).

Analysis of Inner Monologue Turns. As described in Section 3.1, the SIM-RAG Self-Practicing Algorithm (Algorithm 1) defines an arbitrary maximum number of retrieval turns T during self-practice and inference. In practice, however, the number of turns is typically kept relatively low to strike a balance between the cost of inference (in terms of time and computational resources) and performance, while also taking into account the inherent limitations of LLM capabilities (more details in Section 5). To better understand the impact of the hyperparameter T, we conducted an ablation study varying T from 0 to 6. To save computational time, this experiment was performed on a subset of the HotPotQA development set, comprising 500 questions from the previously established RAG evaluation benchmark [53].

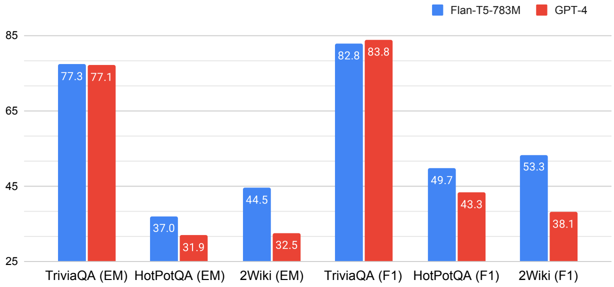

Figure 4: Ablation study on different Critic model choices. Both follow the SIM-RAG setting with GPT-4 as the Reasoner, but the blue bar uses learned Flan-T5-783M as a Critic, while the red bar uses GPT-4 as the Critic (a.k.a., self-critique).

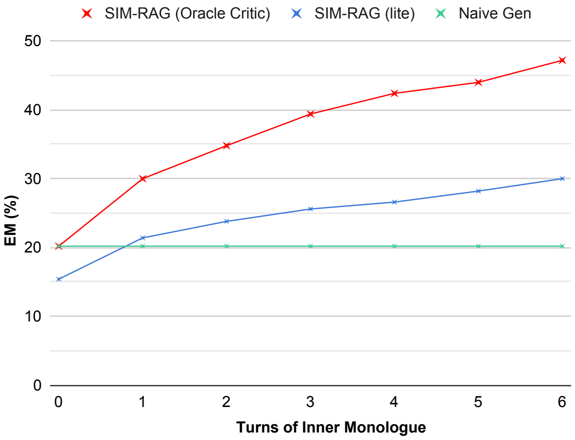

Figure 5: Ablation study on different Inner Monologue turns on HotPotQA. The green line represents the Naive Generation baseline, while the blue line shows the performance of SIM-RAG$_{lite}$. The red line indicates the EM score using the ground-truth label as an Oracle Critic.

Figure 5 shows that larger T leads to better performance, and there is much room to improve the performance of SIM-RAG by optimizing this hyperparameter. The green line represents the Naive Generation baseline, providing a reference point on LLM's zero-shot performance. The blue line shows SIM-RAG$_{lite}$'s actual performance across different turn settings, reflecting the system's ability to refine answers iteratively. Finally, the red line indicates the performance ceiling achievable using an Oracle Critic, which has access to the ground-truth label to determine whether to accept the current answer as the system's output.

By the final round, SIM-RAG achieves a 10.1% improvement over Naive Generation, corresponding to a 50.5% relative gain. Notably, using the Oracle Critic achieves 47.2%, representing the theoretical upper limit of SIM-RAG without modifying the Reasoner. This highlights the potential for further improvements in SIM-RAG through a better or larger Critic, as discussed in Section 4.4.

When $T = 0$, SIM-RAG yields fewer exact-match answers than Naive Generation, primarily because the Critic occasionally rejects correct answers. This outcome is expected, as SIM-RAG is designed to abstain from providing answers when evidence is insufficient, thereby reducing hallucinations (i.e., false positives) at the expense of occasionally suppressing correct responses (i.e., true positives or exact-match answers). We limit T to 6 in our experiments due to the context length constraint of the Critic. However, with ongoing advances in extending language model context windows, this limitation is expected to become less restrictive in the future.

Analysis on Self-Practicing Data. Figure 1 visualizes the inner monologue traces generated during self-practicing. These traces contribute to a rich and diverse training set, which naturally covers both insufficient and excessive retrieval cases. In particular, the self-practicing data captures how retrieval decisions influence reasoning outcomes. As illustrated in the four examples, the 0-Round and 1-Round traces show incorrect predictions due to missing evidence, while the 2-Round trace demonstrates a correct prediction when sufficient information is retrieved. Moreover, the 3-Round trace shows how over-retrieval introduces irrelevant information, distracting the model from correct prediction. Unlike humans, who can selectively attend to relevant information, LLMs are highly sensitive to noisy [30], misleading [51], or even long [22] inputs. Consequently, retrieving more information is not always beneficial for LLMs. This characteristic highlights the importance of information sufficiency checking and explains why self-practicing data is valuable in SIM-RAG.

5 Discussion

5.1 Strengths and Weaknesses

Task. As shown in Section 4.4, SIM-RAG excels in handling RAG tasks of varying complexity, from simple single-round retrieval to multi-round reasoning tasks. Its flexibility enables consistent outperformance of traditional baselines.

Domain adaptation. Like most training-free domain adaptation approaches for LLMs, SIM-RAG performs well on tasks that align with the Reasoner's pretraining corpus but may face challenges with domain-specific jargon or highly specialized terminology. The Critic, on the other hand, is system-specific. The behavior-driven training (discussed in Section 5.2) ensures it is well aligned with the system used to generate the synthetic training data (in Algorithm 1). However, if any major component of a RAG system, such as the Reasoner or Retriever. is replaced or significantly updated, the Critic may require retraining to maintain optimal performance.

Computational cost. The computational cost of SIM-RAG comprises two main components: training and inference. For training, the primary computational expense comes from data generation and Critic learning. The data generation phase requires $(T \times N)$ large model inferences, where T represents the predefined maximum number of retrieval rounds, and N denotes the size of the source dataset. The Critic training phase follows the standard resource demands of supervised learning. During inference, efficiency depends on the Reasoner's capabilities. If the question aligns with the LLM's pretrained knowledge, SIM-RAG is efficient. However, for unfamiliar domains, SIM-RAG may require more turns, highlighting the importance of domain adaptation to optimize performance and reduce inference time.

Failure Cases. Figure 1 provides an example of how our system can both benefit from and be hindered by the Critic rejecting responses, as false rejections may lead to mistakes due to Over-Retrieval in the 3-Round example. Beyond the Critic's behavior, as shown in Figure 5, even an Oracle Critic yields low accuracy, highlighting that failures also stem from broader limitations. These include the dataset's inherent difficulty, which challenges both the Reasoner's knowledge and its ability to formulate effective search queries, as well as the quality of the Retriever.

5.2 Critic Learning

From our empirical study, we found that introducing the Critic is a simpler problem than solving the task. Prior studies show that Flan-T5-783M (the Critic in SIM-RAG$_{lite}$) struggles to handle complex tasks through direct fine-tuning on training datasets [19]. Flan-T5-783M achieves only 14.7% EM score on the HotPotQA dataset via fine-tuning, compared to 20.1% achieved by zero-shot Llama3. However, Flan-T5-783M can be trained in SIM-RAG to do a decent job of serving as a Critic for more powerful LLMs (Llama3 and GPT-4). One possible reason is that the Critic only needs to model the relationships between the question, query, retrieved documents, the LLM's predicted answer and rationale, and the correctness of that prediction. That is, the Critic does not necessarily "know" the correct answer or how to generate the correct answer; instead, it has the easier task of learning to evaluate the coherence and validity of the LLM's output.

6 Conclusion and Future Work

In this paper, we propose the SIM-RAG framework, a lightweight yet effective approach to optimizing multi-round RAG by adding a critic module. This framework can work with both close-source and open-source RAG systems, offering great flexibility in real-world applications. Since SIM-RAG does not modify the parameters of the LLM, it functions as an inference-time enhancement to the RAG system, similar in goal to OpenAI-o1, but realized through a fundamentally different mechanism. In particular, this work introduces a novel Self-Practicing Algorithm to generate synthetic data to address the shortage of labeled, system-specific multi-round reasoning and retrieval training data. Experiments on three standard benchmarks validate SIM-RAG's effectiveness.

This work opens up additional opportunities for future enhancement of generative AI. Although SIM-RAG is a lightweight approach without requiring access to an LLM's weights, we recognize the potential for the trained Critic to be used as a reward model in future work to optimize other components of RAG (Retriever, Reasoner, etc.) through policy-based actor-critic reinforcement learning (e.g., RLHF). Moreover, the capability of an AI system to recognize its limitations—*Knowing you don't know*—is crucial for reducing hallucination and enhancing trustworthiness and reliability. While this paper focuses on multi-round RAG, we expect the novel Self-Practicing and Critic training techniques to be widely adopted to other AI problems.

References

[1] Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. 2023. Self-RAG: Learning to Retrieve, Generate, and Critique through Self-Reflection. In *The Twelfth International Conference on Learning Representations*.

[2] Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George Bm Van Den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, et al. 2022. Improving language models by retrieving from trillions of tokens. In *International conference on machine learning*. PMLR, 2206–2240.

[3] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems* 33 (2020), 1877–1901.

[4] John H Flavell. 1979. Metacognition and cognitive monitoring: A new area of cognitive–developmental inquiry. *American psychologist* 34, 10 (1979), 906.

[5] Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, and Haofen Wang. 2023. Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997* (2023).

[6] Zhibin Gou, Zhihong Shao, Yeyun Gong, Yelong Shen, Yujiu Yang, Nan Duan, and Weizhu Chen. 2023. Critic: Large language models can self-correct with tool-interactive critiquing. *arXiv preprint arXiv:2305.11738* (2023).

[7] Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Mingwei Chang. 2020. Retrieval augmented language model pre-training. In *International conference on machine learning*. PMLR, 3929–3938.

[8] Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. 2020. Constructing a multi-hop QA dataset for comprehensive evaluation of reasoning steps. *arXiv preprint arXiv:2011.01060* (2020).

[9] Arian Hosseini, Xingdi Yuan, Nikolay Malkin, Aaron Courville, Alessandro Sordoni, and Rishabh Agarwal. 2024. V-star: Training verifiers for self-taught reasoners. *arXiv preprint arXiv:2402.06457* (2024).

[10] Jie Huang, Xinyun Chen, Swaroop Mishra, Huaixiu Steven Zheng, Adams Wei Yu, Xinying Song, and Denny Zhou. 2024. Large Language Models Cannot Self-Correct Reasoning Yet. In *The Twelfth International Conference on Learning Representations*.

[11] Gautier Izacard, Patrick Lewis, Maria Lomeli, Lucas Hosseini, Fabio Petroni, Timo Schick, Jane Dwivedi-Yu, Armand Joulin, Sebastian Riedel, and Edouard Grave. 2023. Atlas: Few-shot learning with retrieval augmented language models. *Journal of Machine Learning Research* 24, 251 (2023), 1–43.

[12] Soyeong Jeong, Jinheon Baek, Sukmin Cho, Sung Ju Hwang, and Jong C Park. 2024. Adaptive-RAG: Learning to Adapt Retrieval-Augmented Large Language Models through Question Complexity. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*. 7029–7043.

[13] Zhengbao Jiang, Frank F Xu, Luyu Gao, Zhiqing Sun, Qian Liu, Jane Dwivedi-Yu, Yiming Yang, Jamie Callan, and Graham Neubig. 2023. Active retrieval augmented generation. *arXiv preprint arXiv:2305.06983* (2023).

[14] Jiajie Jin, Yutao Zhu, Xinyu Yang, Chenghao Zhang, and Zhicheng Dou. 2024. FlashRAG: A Modular Toolkit for Efficient Retrieval-Augmented Generation Research. *arXiv preprint arXiv:2405.13576* (2024).

[15] Mandar Joshi, Eunsol Choi, Daniel S Weld, and Luke Zettlemoyer. 2017. TriviaQA: A Large Scale Distantly Supervised Challenge Dataset for Reading Comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 1601–1611.

[16] Vijay Konda and John Tsitsiklis. 1999. Actor-Critic Algorithms. In *Advances in Neural Information Processing Systems*, S. Solla, T. Leen, and K. Müller (Eds.), Vol. 12. MIT Press. https://proceedings.neurips.cc/paper_files/paper/1999/file/6449f44a102fde848669bdd9eb6b76fa-Paper.pdf

[17] Aviral Kumar, Vincent Zhuang, Rishabh Agarwal, Yi Su, John D Co-Reyes, Avi Singh, Kate Baumli, Shariq Iqbal, Colton Bishop, Rebecca Roelofs, et al. 2024. Training Language Models to Self-Correct via Reinforcement Learning. *arXiv preprint arXiv:2409.12917* (2024).

[18] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems* 33 (2020), 9459–9474.

[19] Xiang Li, Shizhu He, Fangyu Lei, JunYang JunYang, Tianhuang Su, Kang Liu, and Jun Zhao. 2024. Teaching Small Language Models to Reason for Knowledge-Intensive Multi-Hop Question Answering. In *Findings of the Association for Computational Linguistics: ACL 2024*. Association for Computational Linguistics, 7804–7816. doi:10.18653/v1/2024.findings-acl.464

[20] Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2023. Let's Verify Step by Step. In *The Twelfth International Conference on Learning Representations*.

[21] Li Liu, Diji Yang, Sijia Zhong, Kalyana Suma Sree Tholeti, Lei Ding, Yi Zhang, and Leilani Gilpin. 2024. Right this way: Can VLMs Guide Us to See More to Answer Questions? *Advances in Neural Information Processing Systems* 37 (2024), 132946–132976.

[22] Nelson F Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2024. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics* 12 (2024), 157–173.

[23] Giovanni Monea, Antoine Bosselut, Kianté Brantley, and Yoav Artzi. 2024. LLMs Are In-Context Reinforcement Learners. *arXiv preprint arXiv:2410.05362* (2024).

[24] Thang Nguyen, Peter Chin, and Yu-Wing Tai. 2024. Reward-RAG: Enhancing RAG with Reward Driven Supervision. *arXiv preprint arXiv:2410.03780* (2024).

[25] OpenAI. 2024. OpenAI o1. https://openai.com/index/learning-to-reason-with-llms/

[26] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems* 35 (2022), 27730–27744.

[27] Fabio Petroni, Patrick Lewis, Aleksandra Piktus, Tim Rocktäschel, Yuxiang Wu, Alexander H Miller, and Sebastian Riedel. 2020. How context affects language models' factual predictions. *arXiv preprint arXiv:2005.04611* (2020).

[28] Ori Ram, Yoav Levine, Itay Dalmedigos, Dor Muhlgay, Amnon Shashua, Kevin Leyton-Brown, and Yoav Shoham. 2023. In-context retrieval-augmented language models. *Transactions of the Association for Computational Linguistics* 11 (2023), 1316–1331.

[29] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347* (2017).

[30] Freda Shi, Xinyun Chen, Kanishka Misra, Nathan Scales, David Dohan, Ed H Chi, Nathanael Schärli, and Denny Zhou. 2023. Large language models can be easily distracted by irrelevant context. In *International Conference on Machine Learning*. PMLR, 31210–31227.

[31] Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. 2024. Reflexion: Language agents with verbal reinforcement learning. *Advances in Neural Information Processing Systems* 36 (2024).

[32] Avi Singh, John D Co-Reyes, Rishabh Agarwal, Ankesh Anand, Piyush Patil, Xavier Garcia, Peter J Liu, James Harrison, Jaehoon Lee, Kelvin Xu, et al. 2024. Beyond Human Data: Scaling Self-Training for Problem-Solving with Language Models. *Transactions on Machine Learning Research* (2024).

[33] Kaya Stechly, Matthew Marquez, and Subbarao Kambhampati. 2023. Gpt-4 doesn't know it's wrong: An analysis of iterative prompting for reasoning problems. *arXiv preprint arXiv:2310.12397* (2023).

[34] Kaya Stechly, Karthik Valmeekam, and Subbarao Kambhampati. 2024. On the self-verification limitations of large language models on reasoning and planning tasks. *arXiv preprint arXiv:2402.08115* (2024).

[35] Weihang Su, Yichen Tang, Qingyao Ai, Zhijing Wu, and Yiqun Liu. 2024. Dragin: Dynamic retrieval augmented generation based on the real-time information needs of large language models. *arXiv preprint arXiv:2403.10081* (2024).

[36] Yixuan Tang and Yi Yang. 2024. Multihop-rag: Benchmarking retrieval-augmented generation for multi-hop queries. *arXiv preprint arXiv:2401.15391* (2024).

[37] Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. Stanford Alpaca: An Instruction-following LLaMA model. https://github.com/tatsu-lab/stanford_alpaca.

[38] Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 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.

[39] Keheng Wang, Feiyu Duan, Peiguang Li, Sirui Wang, and Xunliang Cai. 2024. LLMs Know What They Need: Leveraging a Missing Information Guided Framework to Empower Retrieval-Augmented Generation. *arXiv preprint arXiv:2404.14043* (2024).

[40] Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. 2022. Text embeddings by weakly-supervised contrastive pre-training. *arXiv preprint arXiv:2212.03533* (2022).

[41] Yile Wang, Peng Li, Maosong Sun, and Yang Liu. 2023. Self-Knowledge Guided Retrieval Augmentation for Large Language Models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*. 10303–10315.

[42] Miao Xiong, Zhiyuan Hu, Xinyang Lu, YIFEI LI, Jie Fu, Junxian He, and Bryan Hooi. 2024. Can LLMs Express Their Uncertainty? An Empirical Evaluation of Confidence Elicitation in LLMs. In *The Twelfth International Conference on Learning Representations*.

[43] Shi-Qi Yan, Jia-Chen Gu, Yun Zhu, and Zhen-Hua Ling. 2024. Corrective retrieval augmented generation. *arXiv preprint arXiv:2401.15884* (2024).

[44] Diji Yang, Kezhen Chen, Jinmeng Rao, Xiaoyuan Guo, Yawen Zhang, Jie Yang, and Yi Zhang. 2024. Tackling vision language tasks through learning inner monologues. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 38. 19350–19358.

[45] Diji Yang, Jinmeng Rao, Kezhen Chen, Xiaoyuan Guo, Yawen Zhang, Jie Yang, and Yi Zhang. 2024. IM-RAG: Multi-Round Retrieval-Augmented Generation

Through Learning Inner Monologues. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 730–740.

[46] Diji Yang, Linda Zeng, Kezhen Chen, and Yi Zhang. 2024. Reinforcing Thinking through Reasoning-Enhanced Reward Models. *arXiv preprint arXiv:2501.01457* (2024).

[47] Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W Cohen, Ruslan Salakhutdinov, and Christopher D Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. *arXiv preprint arXiv:1809.09600* (2018).

[48] Zijun Yao, Weijian Qi, Liangming Pan, Shulin Cao, Linmei Hu, Weichuan Liu, Lei Hou, and Juanzi Li. 2024. Seakr: Self-aware knowledge retrieval for adaptive retrieval augmented generation. *arXiv preprint arXiv:2406.19215* (2024).

[49] Tian Yu, Shaolei Zhang, and Yang Feng. 2024. Auto-RAG: Autonomous Retrieval-Augmented Generation for Large Language Models. *arXiv preprint arXiv:2411.19443* (2024).

[50] Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah Goodman. 2022. Star: Bootstrapping reasoning with reasoning. *Advances in Neural Information Processing Systems* 35 (2022), 15476–15488.

[51] Linda Zeng, Rithwik Gupta, Divij Motwani, Diji Yang, and Yi Zhang. 2025. Worse than Zero-shot? A Fact-Checking Dataset for Evaluating the Robustness of RAG Against Misleading Retrievals. *arXiv preprint arXiv:2502.16101* (2025).

[52] Dan Zhang, Sining Zhoubian, Ziniu Hu, Yisong Yue, Yuxiao Dong, and Jie Tang. 2024. Rest-mcts*: Llm self-training via process reward guided tree search. *arXiv preprint arXiv:2406.03816* (2024).

[53] Xuanwang Zhang, Yun-Ze Song, Yidong Wang, Shuyun Tang, Xinfeng Li, Zhengran Zeng, Zhen Wu, Wei Ye, Wenyuan Xu, Yue Zhang, et al. 2024. RAGLAB: A Modular and Research-Oriented Unified Framework for Retrieval-Augmented Generation. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. 408–418.