Smart-Searcher: Incentivizing the Dynamic Knowledge Acquisition of LLMs via Reinforcement Learning

Anonymous ACL submission

Abstract

Large Language Models (LLMs) are powerful but prone to hallucinations due to static knowledge. Retrieval-Augmented Generation (RAG) helps by injecting external information, but current methods often are costly, generalize poorly, or ignore the model's internal knowledge. In this paper, we introduce Smart-Searcher, a novel framework designed to train LLMs to adaptively leverage both internal and external knowledge sources. Smart-Searcher employs a two-stage training strategy: an initial SFT Cold-start phase for preliminary format learning, followed by RL for Dynamic Knowledge Acquisition. The RL stage uses outcome-supervision to encourage exploration, incorporates a reward mechanism for internal knowledge utilization, and integrates a memorization mechnism to continuously assimilate retrieved information, thereby enriching the model's internal knowledge. By leveraging internal knowledge and external search engine, the model continuously improves its capabilities, enabling efficient retrieval-augmented reasoning. Our experiments demonstrate that Smart-Searcher outperforms previous RAG and reasoning methods and achieves efficient retrieval. We will release all the codes, models, and data after review.

1 Introduction

Large language models (LLMs) (Zhao et al., 2023) have demonstrated remarkable reasoning capabilities by only leveraging the information encoded in their parameters. However, their reliance on static, internal knowledge leads to notable limitations. At the simultaneously, this reliance easily leads to hallucinations (Huang et al., 2025), so LLMs may struggle with open-ended tasks (Wang et al., 2025c; Trivedi et al., 2022). Therefore, it is crucial to enable LLMs to access external information when they are confused during the reasoning process to achieve more deliberative reasoning (Jiang et al., 2024a).

To address this issue, extensive research has focused on augmenting LLMs with external information sources (*i.e.,* RAG (Gao et al., 2024)). Early approaches emphasize specific prompting strategies to guide LLMs (Li et al., 2025; Teng et al., 2025) and subsequent studies investigate to distill this capability into smaller LLMs through supervised fine-tuning (SFT) (Wang et al., 2025b). However, recent findings suggest that SFT-based distillation can cause models to memorize solution paths, limiting their generalization to novel scenarios (Chu et al., 2025). Further proposals include a test-time scaling method (Li et al., 2024), notably employing the Monte Carlo Tree Search (MCTS) framework (Sun et al., 2025) to enhance solution-finding by expanding the search space during inference, but this approach incurs significant inference overhead, reducing its practicality for widespread use. Recent studies employ end-to-end outcome-based reinforcement learning (RL) to train models, enabling them to autonomously explore external retrieval environments during inference (Jin et al., 2025; Song et al., 2025). This approach fosters the development of self-directed retrieval capabilities in LLMs as they reason. However, such models often become overly reliant on external search engines after training, neglecting the utilization of their internal knowledge.

In practice, when humans attempt to solve factual questions, they first recall their internal knowledge, and only turn to search engines when they recognize a lack of information. At the same time, after obtaining the external searched information, humans would memorize this knowledge for future use. For LLMs, extensive pretraining on large-scale data has already endowed them with substantial internal knowledge (Qwen et al., 2025). Therefore, it is essential to equip models with the ability to dynamically switch between internal and external knowledge sources as needed. Furthermore, models should be encouraged to effectively mem-

orize useful information encountered during training (Jiang et al., 2024b), progressively enriching their internal knowledge and continuously evolving toward greater intelligence.

In this paper, we present **Smart-Searcher**, a novel framework designed to teach LLMs to adaptively leverage both internal and external knowledge. We adopt a two-stage training strategy: *SFT Cold-start* and *RL for Dynamic Knowledge Acquisition*. In the first phase, we employ reject sampling to collect data that meets the format requirements and perform a cold start with SFT. In the second stage, we further train the model using outcome-based RL to guide the model in dynamically acquiring knowledge, which is to encourage reliance on internal knowledge when confident, and invoke external search mechanisms when uncertain, based on a carefully designed reward design. Additionally, we further introduce a memory mechanism, enabling the model to retain knowledge encountered during training by converting and memorizing retrieved content. This mechanism continuously enriches its internal knowledge, empowering it to effectively balance internal reasoning and external retrieval through autonomous exploration and timely memorization.

To verify the effectiveness of Smart-Searcher, we conduct extensive experiments based on Qwen-2.5-7B-Instruct. Notably, our method surpasses the strong baseline by up to 4.3% and reduces the retrieval count by 42.9% compared to vanilla RL-based approaches.

Our principal contributions are as follows:

• We introduce Smart-Searcher, teaching LLMs to adaptively leverage both internal and external knowledge through a two-stage training strategy.

•We encourage the model to actively leverage its internal knowledge while efficiently memorizing external information, enabling dynamic knowledge acquisition through exploration and memorization.

• Extensive experiments show that Smart-Searcher outperforms existing RAG methods, while significantly reducing the number of retrievals compared to vanilla RL-based approaches.

2 Related Work

Retrieval-Augmented Generation. To improve the factual accuracy of LLM inference and reduce hallucinations, researchers have proposed enhancing language models by incorporating external information sources, a paradigm known as RAG (Fan et al., 2024). Early RAG approaches primarily include Branching (Kim et al., 2024), Summarization (Li et al., 2023), and Adaptive Retrieval (Jeong et al., 2024) strategies. As foundation models have become increasingly capable, exhibiting strong CoT reasoning abilities, many studies have combined RAG with CoT. These efforts include methods that prompt the model to perform step-by-step retrieval (Shao et al., 2023; Trivedi et al., 2023) and strategies that distill such capabilities into smaller LLMs (Asai et al., 2024). In parallel, several works have explored test-time scaling, notably using MCTS (Feng et al., 2025) to dynamically expand reasoning paths. However, such approaches often incur substantial inference-time overhead. More recently, researchers have trained models using outcome- supervision RL (Zheng et al., 2025) to encourage the exploration of more effective actions and retrieval behaviors, but it leads models to over-rely on external search engines, diminishing their ability to leverage internal knowledge (Wang et al., 2025a). Enabling LLMs to effectively integrate and alternate between internal knowledge and external retrieval remains a significant challenge.

Reinforcement Learning. To improve training efficiency, several off-policy algorithms have since been proposed (Rafailov et al., 2024; Ethayarajh et al., 2024); however, these methods still face limitations in terms of preference modeling accuracy and generalization capability (Pang et al., 2024). DeepseekMath introduced the GRPO algorithm (Shao et al., 2024), which enables efficient self-exploration through a mechanism of relative preference optimization. Building on this, Deepseek-R1 (DeepSeek-AI et al., 2025) have demonstrated that outcome-based RL can significantly enhance the reasoning abilities of large models. More recently, studies have begun to investigate RL algorithms specifically designed to improve LLM reasoning capabilities (Yu et al.; Yuan et al., 2025). In parallel, other research efforts have applied RL to the retrieval domain, aiming to enable deep search capabilities (Chen et al., 2025). However, the use of RL that combine LLM-driven retrieval and reasoning remains largely simplistic and underexplored.

3 Preliminary

To enhance the performance of LLMs in open-domain multi-hop question answering tasks (Ho et al., 2020), in this work, we focus on enabling the

model to autonomously decide when to use its internal knowledge or to invoke an external retriever to answer the given questions with the LLM self-improving paradigm, which can improve both reasoning effectiveness and efficiency. To this end, we introduce three special tokens to format the LLM reasoning process, *i.e.,* `<internal>`, `<external>`, and `<document>`. Concretely, during the reasoning process, the LLM with parameters θ determines whether the current step requires external knowledge to help perform reasoning. If so, it triggers the `<external>` to issue a $query_t$, which is sent to a retriever to retrieve the top-K relevant documents $D_t = \{d_{t,k}\}_{k=1}^{K}$ from an external corpus. These retrieved documents are incorporated into the reasoning path with another special token `<document>`. Otherwise, the model directly generates the related internal knowledge enclosed in `<internal>`. After several reasoning steps, the LLM obtains the final answer and stops the reasoning process.

Since our approach is orthogonal to the RL algorithm, we conduct the experiments based on a widely used RL algorithm, *i.e.,* REINFORCE++ (Hu, 2025), which is a stable RL algorithm without the critic model. To better accommodate the retrieval scenario, we mask the retrieved documents during the loss calculation process, as they serve as environmental observations rather than model-generated content. Formally, for each question q, we first samples a group of outputs $\{o_1, o_2, \cdots, o_G\}$ from the old policy model $\pi_{\theta_{\text{old}}}$. Next, we incorporate the KL regularization into the reward scores $R_\phi(q, o_{i,\leq t})$, and then normalize the advantage scores:

$$\hat{A}'_{i,t} = R(q, o_i) - \beta \cdot \sum_{i=t}^{T} \text{KL}(i),\ \hat{A}_{i,t} = \frac{\hat{A}'_{i,t} - \text{mean}(\hat{\mathbf{A}})}{\text{std}(\hat{\mathbf{A}})}$$

We utilize $\hat{\mathbf{A}}$ to denote the set of all advantages in the global batch that contains $\hat{A}_{i,t}$. After obtaining the advantage scores, we set the mask value $M(i,t)$ as 0 if this token belongs to an external document, otherwise we set $M(i,t) = 1$. Finally, we employ the masks $M(i,t)$ in the objective function to remove the influence of retrieved documents:

$$\hat{P}_{i,t} = \min\left[p_{i,t}\hat{A}_{i,t}, \text{clip}\left(p_{i,t}, 1-\varepsilon, 1+\varepsilon\right)\hat{A}_{i,t}\right],$$

$$\mathcal{J}_{\text{Mask}}(\theta) = \frac{1}{G}\sum_{i=1}^{G}\frac{1}{\sum_{t=1}^{|o_i|}M(i,t)}\sum_{t=1}^{|o_i|}M(i,t)\cdot\hat{P}_{i,t} \quad (1)$$

where ε is a hyper-parameter and $p(i,t)$ is the important sampling coefficient , and π_θ is the policy model.

4 Methodology

In this part, we introduce the **Smart-Searcher** framework, which aims to teach LLM to adaptively utilize internal and external knowedge through two critical stages, *i.e., SFT Cold-Start* (Section 4.1) and *RL for Dynamic Knowledge Acquisition* (Section 4.2). Concretely, in the first stage, we utilize the curated data to perform SFT on the model, to standardize its responses in a specific format and enable it to leverage external retrievers and internal knowledge adaptively. In the second stage, we employ RL on LLM that encourages the model to explore more effective actions and behaviours, and further incorporate the internal knowledge utilization encouragement and external knowledge memorization in the training process, to guide the model to dynamically acquire knowledge and continuously enrich its internal knowledge, which can lead to higher reasoning efficiency.

4.1 SFT Cold-start

To equip LLMs with the preliminary ability to autonomously perform external retrieval during inference while effectively leveraging internal knowledge, we synthesize high-quality training instances using rejection sampling, without relying on other powerful models.We only keep the correct responses with the appropriate occurrences of both the `<internal>` and `<external>` tags, teaching LLM to perform dynamic knowledge acquisition in a proper format.

Specifically, given the question x and the synthesized output y, once the i-th token of the output belongs to the external document, it will be masked, *i.e.,* $M_i = 0$. Otherwise, the coefficient M_i will be set as 1, incorporating the probability of y_i into the objective function as follows,

$$\mathcal{L}_{\text{SFT}} = -\frac{1}{n}\sum_{i=1}^{n} M_i \times P(y_i|x, y_{<i}) \quad (2)$$

4.2 RL for Dynamic Knowledge Acquisition

After cold-starting, we obtain a model that can utilize internal knowledge and perform an external search with correct format. To further enhance its capabilities, *i.e.,* to perform effective and efficient reasoning, we continually train the model through the RL process, which includes a mechanism that encourages internal knowledge utilization (Section 4.2.1) and a mechanism for converting and memorizing external knowledge (Section 4.2.2).

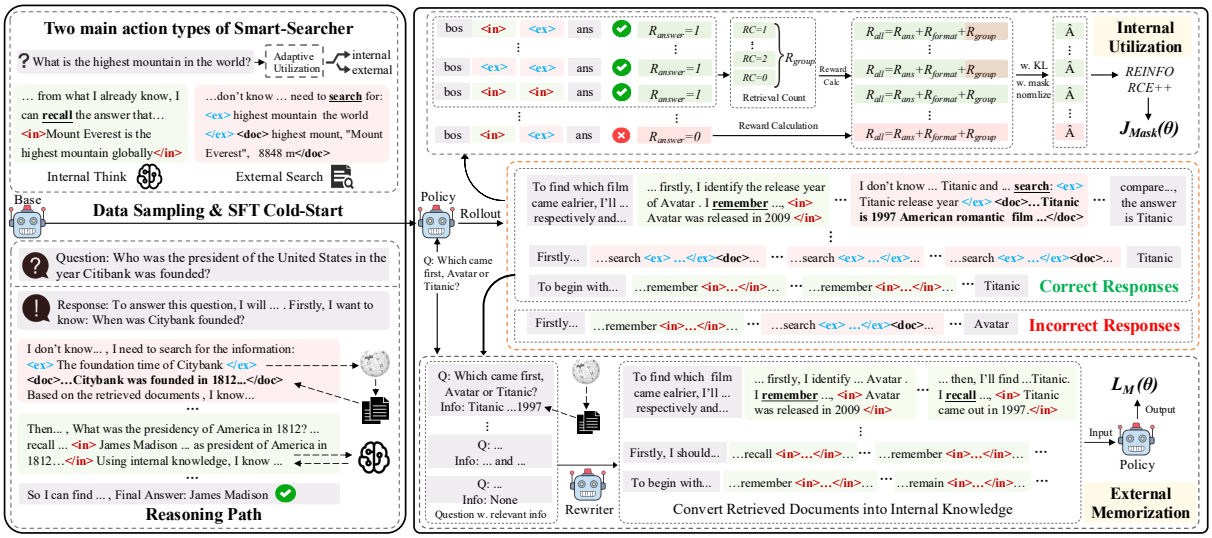

Figure 1: Overall framework of our proposed Smart-Seracher approach.

4.2.1 Internal Knowledge Utilization Encouragement

In the RL process, the reward function is utilized to provide the supervision signals, which can adjust and optimize the behaviours of the model (Hu, 2025). Therefore, given the question q and the i-th generated response o_i, we design the *format reward* and *answer reward* to induce the model to perform reasoning correctly with the expected format, and incorporate the *group reward* into the final reward function to mitigate the over-reliance on the external retriever. Now, we introduce the details of the reward function in the following.

Format reward. We impose a strict formatting constraint to ensure model responses are consistent and clear. During the reasoning process, when calling the external retriever, the model is required to formulate a query and enclose it within the `<external>...</external>` tags, and is prohibited from generating document content directly without first invoking retrieval. When the reasoning process finishes, the final response must satisfy the following criteria, *i.e.,* the final answer must be enclosed within `boxed{}`, and the content should not contain any garbled or unreadable content. Once the model behaviours satisfy the above requirements, we set the format reward R_{format} as 0, while we set the reward as -2 if any requirement fails, as shown in the following,

$$R_{\text{format}}(q, o_i) = \begin{cases} 0, & \text{The format of } o_i \text{ is correct} \\ \text{-2}, & \text{The format of } o_i \text{ is incorrect} \end{cases} \quad (3)$$

Answer reward. To indicate the correctness of the final answer, we leverage the Cover Exact Match (CEM) metric to calculate the answer reward, adapting to the group reward discussed in the following and relieving the issue of EM being too strict. Concretely, CEM is True if the ground truth answer appears in the predicted answer a_i extracted from the response o_i, and False for other situations. However, we observe that LLM can easily hack the CEM metric during the RL process, where LLM is likely to generate a longer predicted answer that will receive a higher probability to cover the ground truth answer, causing the CEM to be falsely high. Therefore, we regard the answer exceeding 10 words as an incorrect answer, requiring LLM to generate the answer within ten words, which can alleviate the above reward hacking issue. In summary, the answer reward R_{answer} can be computed as follows,

$$R_{\text{answer}}(q, o_i) = \begin{cases} 1, & a_i \text{ within 10 words} \wedge \text{CEM=True} \\ 0, & \text{Otherwise} \end{cases} \quad (4)$$

Group reward. Building upon the first two rewards for LLM reasoning effectiveness, the group reward is designed to encourage the model to reduce its reliance on external retrieval, increasing the reasoning efficiency. Considering that the variance of the external retriever calling times by LLM reflects the necessity of performing external retrieval, group reward is calculated by the standard deviation of the number of calls to the retriever in correct responses to the same question. Formally, given the question q and a set of generated responses $\{o_1, o_2, \ldots, o_n\}$, we first count the num-

ber of calls to the retriever t_i of each response, and then calculate the standard deviation σ of $\{t_1, t_2, \ldots, t_n\}$. Next, we calculate the minimum number of calls to the retriever of the correct responses, *i.e.,* $t_{\text{min}} = \min\{t_i \mid R_{\text{answer}}(q, o_i) = 1\}$.

$$R'_{\text{group}}(q, o_i) = \begin{cases} 2 \times \sigma^2, & R_{\text{answer}}(q, o_i) = 1 \wedge t_i = t_{\text{min}} \\ 0, & \text{Otherwise} \end{cases} \tag{5}$$

Meanwhile, to maintain training stability and prevent excessive variance, we introduce a hyperparameter η to clip the corresponding factor. The final computation of reward is formulated as follows,

$$R_{\text{group}}(q, o_i) = \min\left(R'_{\text{group}}(q, o_i), \eta\right) \tag{6}$$

Finally, the reward $R(q, o_i)$ utilized to calculate the advantage in Equation 7 is defined as the sum of the three sub-rewards mentioned above:

$$R(q, o_i) = R_{\text{format}}(q, o_i) + R_{\text{answer}}(q, o_i) + R_{\text{group}}(q, o_i) \tag{7}$$

4.2.2 External Knowledge Memorization

The standard RL training paradigm relies on the model's self-exploration and the feedback from the external environment. In retrieval-based scenarios, since the knowledge retrieved by the retriever is entirely correct, the model should like a human, aim to memorize this information during training, transforming it into internal knowledge. This enables the model to utilize the acquired knowledge directly in future instances without repeated retrieval, thereby achieving efficient reuse of retrieved information. Thus, we incorporate external knowledge memorization by rewriting the retrieved information to align with the model's interanl knowledge utilization pattern, enabling the model to internalize them effectively.

To obtain the rewritten instances, at the beginning of the RL process, we fine-tune a separate model on the data filtered in Section 6 as the rewritting model, which can solve the questions based on the pre-processed documents that do not call the retriever. During the RL process, we select the correct responses generated by LLM, and then extract the retrieved documents from the responses. Given the question and the extracted documents in the context, the rewriting model can generate the reasoning paths without calling the external retriever. After validating the correctness of these reasoning paths, we select the correct instances to construct the dataset $\mathcal{T}$ for memorization and internalization. In conclusion, the corresponding loss for memorization is computed as follows:

$$\mathcal{L}_{\text{M}}(\theta) = \frac{-1}{\sum_{o_i \in \mathcal{T}} |o_i|} \sum_{o_i \in \mathcal{T}} \sum_{t=1}^{|o_i|} \log \pi_\theta(o_{i,t} | q, o_{i,<t}) \tag{8}$$

To avoid the $\mathcal{L}_{\text{M}}(\theta)$ from dominating the policy model's training and causing the model to ignore external retrieval, we weight it with a pre-defined coefficient μ. The final loss used to optimize the policy model during the retrieval scenario RL process is computed as follows:

$$\mathcal{L}(\theta) = -\mathcal{J}_{\text{Mask}}(\theta) + \mu * \mathcal{L}_{\text{M}}(\theta) \tag{9}$$

Thus, during training, the model not only engages in self-exploration but also continuously enriches its internal knowledge, enabling it to become increasingly smarter over time.

5 Experiments

5.1 Experimental Settings

Datasets and Evaluation Metrics. We evaluate using four multi-hop datasets: HotpotQA, 2WikiMultiHopQA, Musique, and Bamboogle (Press et al., 2023). HotpotQA and 2WikiMultiHopQA are in-domain benchmarks since parts of their training sets are used for training. In contrast, Musique and Bamboogle serve as out-of-domain benchmarks to assess our model's generalization capabilities. We randomly select 500 samples from the entire validation sets of HotpotQA, 2WikiMultiHopQA, and Musique, and use the entire test set of Bamboogle to form our final test set. For evaluation metrics, we utilize F1-score and LLM-as-Judge (LasJ). considering that the answers to open-ended multi-hop questions are not uniform in form. The F1-score measures the word-level similarity between the predicted answer and the reference answer while LLM-as-Judge employs GPT-4o-mini to assess the correctness of prediction. The evaluation prompt for LasJ is provided in Appendix B.

Baselines. We compare *Smart-Searcher* against several baselines. *Naive Generation* generates answers directly without retrieval. *Standard RAG* represents traditional RAG systems that retrieve documents directly based on the question. *SuRe* executes multiple reasoning paths in parallel for a single query. *Selective-Context* compresses retrieved documents to reduce context length. *Adaptive-RAG* dynamically selects retrieval strategies depending on the complexity of the query. *CR-Planner* scales

RAG at inference time using MCTS. *RAG-CoT methods*, such as Iter-RetGen, IRCoT, and Search-o1, which combine RAG with CoT using prompts. *RAG-RL methods* like R1-Searcher and Search-R1 leverage RL to enable the model to learn to autonomously perform retrieval during inference.

Implementation Details Smart-Searcher and all baseline models are either trained or prompted using the Qwen-2.5-7B-Instruct as the backbone, and evaluated with FlashRAG (Jin et al., 2024) using local dense retrieval corpus. The retrieval corpus comprises the English Wikipedia as provided by KILT (Petroni et al., 2021) in 2019, segmented into 100-word passages with appended titles, totaling 29 million passages. We employ BGE-large-en-v1.5 as the text retriever. Detailed training settings for Smart-Searcher are provided in Appendix A.

5.2 Main Results

Table1 shows the results of Smart-Searcher and the baselines on four mutil-step benchmarks. We can obtain the following observations:

• *Achieving Significant Performance Improvement on Multi-Hop QA*. Our method, Smart-Searcher, achieves significant performance improvements over all mutil-hop QA benchmarks under the LLM-as-Judge evaluation metric, including both tree search-based and RL-based approaches. Specifically, Smart-Searcher outperforms CR-Planner by 25.7% and surpasses the best vanilla RL-based method R1-Searcher by 4.3% on the overall test set. These results demonstrate that our approach effectively enables the model to perform accurate and timely retrieval invocations throughout the reasoning process, thereby enhancing overall performance.

• *Balancing the Utilization of Internal and External Knowledge.* While maintaining strong performance on the evaluation datasets, our method achieves a significant reduction in retrieval count compared to vanilla RL-based RAG approaches. Specifically, the average retrieval count is reduced by 30.0% and 52.9% compared to R1-Searcher and Search-R1, respectively. This observation suggests a potential conflict between external information and the internal knowledge of LLMs and one possible reason is that directly injecting retrieved documents into the reasoning process may introduce noise. This demonstrates that the model should learn to make full use of its internal knowledge and only invoke the retriever when necessary.

• *Maintaining Generalization Ability.* Despite being trained on only 9000 samples, the model achieves strong performance on in-domain datasets and further exhibits impressive generalization to out-of-domain datasets. This suggests that the model effectively learns to retrieve relevant documents and leverage internal knowledge, integrating both with reasoning through exploration during training. This enables robust performance on new test datasets that require retrieval. Furthermore, it can also seamlessly generalizes to online search, as detailed in Section 6.2.

6 Further Analysis

6.1 Ablation Study

To validate the effectiveness of our proposed Smart-Searcher framework, we conduct a comprehensive ablation analysis of its key design elements. We design five distinct variants: (1) *w/o Stage-1* removes the initial SFT cold start stage; (2) *w/o Stage-2* removes the entire RL training stage; (3) *w/o* R_{group} removes the group reward in the RL stage; (4) *w/o* $\mathcal{L}_M$ removes the external knowledge memorization mechanism in the RL stage and (5) *w/o* R_{group} *and* $\mathcal{L}_M$ removes both the group reward and the external knowledge memorization mechanism. The performance of these variants is presented in Table 2. As observed, all ablated variants exhibit a decline in performance compared to our full method, underscoring the integral contribution of each component. Specifically, *w/o Stage-1* leads to a degradation in performance along with an increase in retrieval count. Meanwhile, the performance of *w/o Stage-2* drops significantly, primarily because simple SFT causes the model to over-rely on its internal knowledge. This highlights the necessity of our two-stage pipeline. Futhermore, during RL training, *w/o* R_{group} during RL also leads to a reduction in performance. This demonstrates the positive impact of group reward in successfully guiding the model to be more selective with external searches and to rely more on its internalized knowledge. Similarly, *w/o* $\mathcal{L}_M$ results in lower scores and a slight increase in retrieval count indicating that the memory mechanism for external knowledge can effectively internalize retrieved content as intrinsic knowledge of the model.

6.2 Online Search

Considering training efficiency and cost, we implement a local dense embedding-based retrieval

Models	HotpotQA†			2Wiki†			Bamboogle‡			Musique‡			Avg		
	F1	LasJ	RC	F1	LasJ	RC	F1	LasJ	RC	F1	LasJ	RC	F1	LasJ	RC
Directly Gen	26.0	26.6	0.00	27.7	26.8	0.00	18.2	17.6	0.00	9.6	6.2	0.00	18.0	19.3	0.00
Standard RAG	32.0	42.4	1.00	34.8	34.8	1.00	31.5	31.2	1.00	17.2	14.6	1.00	24.6	30.8	1.00
Sure	42.9	48.4	1.00	26.2	26.8	1.00	29.2	28.0	1.00	13.1	10.0	1.00	27.9	28.3	1.00
Selective-Context	39.8	43.4	1.00	29.1	29.6	1.00	22.1	20.8	1.00	10.6	8.8	1.00	22.8	25.7	1.00
Adaptive-RAG	38.0	47.4	1.53	21.1	25.8	1.42	23.3	25.0	1.50	10.1	11.6	1.83	20.6	27.5	1.57
IRCoT	47.7	55.2	2.47	32.4	38.6	2.74	37.5	39.2	2.30	14.8	15.8	2.70	29.4	37.2	2.55
Iter-RetGen	47.2	54.4	3.00	33.2	34.4	3.00	32.4	32.0	3.00	19.9	18.2	3.00	28.2	34.8	3.00
CR-Planner	44.4	33.6	2.40	48.2	22.0	2.54	35.2	34.4	2.96	12.2	11.4	2.72	32.0	25.4	2.66
Search-o1	46.9	53.2	1.39	46.6	51.2	1.91	52.9	52.0	1.18	21.1	19.0	1.40	36.6	43.9	1.47
R1-Searcher	**60.4**	62.2	2.18	**62.8**	63.4	2.23	59.0	54.4	2.17	**35.7**	31.4	2.61	**45.6**	52.9	2.30
Search-R1	57.8	62.2	3.12	46.2	50.0	3.71	56.9	56.0	3.25	27.5	26.0	3.61	40.3	48.6	3.42
Smart-Searcher	59.0	**64.2**	1.44	61.2	**64.4**	1.18	**60.8**	**59.2**	1.74	33.8	**32.8**	2.06	45.3	**55.2**	1.61

Table 1: Performance comparisons between Smart-Searcher and the baselines on QA benchmarks. The best and second best results are **bold** and underlined, respectively. †/‡ represents in-domain/out-of-domain datasets.

Method	Bamboogle			Musique		
	F1	LasJ	RC	F1	LasJ	RC
Ours	**60.8**	**59.2**	**1.74**	**33.8**	**32.8**	**2.06**
w/o Stage-1	56.9	56.8	1.96	32.7	31.6	2.49
w/o Stage-2	47.4	45.6	0.94	23.0	19.4	1.03
w/o R_{group}	58.3	56.8	1.91	33.1	32.4	2.37
w/o $\mathcal{L}_{\text{M}}$	58.1	57.2	1.84	31.0	29.4	2.09
w/o R_{group} and $\mathcal{L}_{\text{M}}$	56.2	54.4	1.92	32.2	31.2	2.40

Table 2: Ablation study on Bamboogle and Musique.

Method	Bamboogle			Frames		
	F1	LasJ	RC	F1	LasJ	RC
Ours	**77.5**	**76.0**	1.70	**33.8**	**39.0**	1.77
Search-o1	52.9	52.0	1.18	26.1	30.7	1.56
R1-Searcher	67.5	68.8	1.72	33.3	38.0	1.86
Search-R1	69.3	67.2	1.92	33.3	36.0	2.38

Table 3: Online search generalization experiments on Bamboogle and Frames.

system using Wikipedia as the external retrieval environment, which remains static during training. In contrast, most real-world applications rely on online web retrieval. To evaluate the generalization ability of Smart-Searcher in online search scenarios, we assessed its performance on two newly introduced datasets: Bamboogle and Frames, using online web search, a setting not encountered during RL training. Specifically, during inference, whenever retrieval is required, we use the Google API to perform real-time web searches and retrieve relevant web pages. Given the extensive content of these pages, we first employ GPT-4o-mini to generate concise summaries, which are then integrated into the reasoning process. As

Method	Correct	Incorrect	Overall
R1-Searcher	853 / 2.16	772 / 2.52	1625 / 2.33
Search-R1	761 / 3.30	864 / 3.60	1625 / 3.46
Smart-Searcher	881 / 1.41	744 / 1.78	1625 / 1.58

Table 4: Number of correct and incorrect cases and the average retrieval count of RL-based methods.

illustrated in Figure 3, Smart-Searcher achieves the best F1 and LLM-as-Judge scores compared to both prompt engineering-based methods (*i.e.,* Search-o1) and vanilla RL-based approaches (*i.e.,* R1-Searcher, Search-R1). Moreover, compared to vanilla RL methods, our model significantly reduces the number of retrieval calls. This demonstrates our model's strong adaptability to online search scenarios, as well as its ability to effectively balance internal knowledge with external retrieval during inference, thereby achieving retrieval efficiency without compromising performance.

6.3 Analysis of Knowledge Acquisition

As shown in Table 4, Smart-Searcher exhibits the lowest average retrieval count across both correctly and incorrectly answered questions. Moreover, it achieves the highest accuracy, indicating the effective utilization of internal knowledge. Furthermore, Figure 3 shows the changes in retrieval count and reward during the process of RL training for R1-Searcher and Smart-Searcher. For our method, We observe that the reward increases steadily and eventually plateaus, while the retrieval count initially shows a slight decline, followed by a continuous rise, and ultimately stabilizes. This trend can be attributed to the influence of SFT in the Stage-1,

Figure 2: A qualitative example showing the deliberative reasoning process of RAG-Star in Bamboogle.

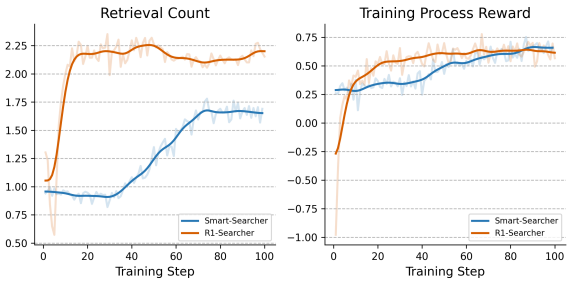

Figure 3: The log of retrieval count and reward for R1-Searcher and Smart-Seracher during RL training.

during which the model exhibits a low demand for invoking the search engine. As training progresses, the model gradually discovers that performing external searches can yield higher rewards, leading to an increase in retrieval behavior. In the later phase, a balance is gradually established between the use of external search and internal knowledge, resulting in the stabilization of both retrieval count and reward. In contrast, R1-Searcher exhibits significantly higher retrieval counts while its reward quickly stabilizes, indicating an over-reliance on the retriever. This effectively validates the effectiveness of our approach in achieving a balanced utilization of internal and external knowledge, while gradually enabling dynamic knowledge acquisition throughout the RL training process.

6.4 Case Study

To illustrate the overall reasoning process of Smart-Searcher, we analyze a representative example from the Bamboogle dataset. Figure 2 compares the responses generated by Smart-Searcher, Search-R1, and the untrained model when presented with the same question. The vanilla Qwen-2.5-7B-Instruct, without invoking any external search engine, relies solely on its internal knowledge and produces an incorrect answer. In contrast, while Search-R1 arrives at the correct answer (*i.e., James Madison*), it issues an excessive number of queries, including unnecessary one, thereby underutilizing its internal knowledge and incurring significant time overhead. Our Smart-Searcher demonstrates the ability to break down the complex question and dynamically adjust its behavior based on the nature of the sub-question. For instance, when encountering an uncertain or ambiguous sub-question (*i.e., When was Citibank founded?*), it opts to perform an external search. However, when faced with a more specific question that can be answered using internal knowledge (*i.e., Who was the president of the United States in 1812?*), it leverages its internal knowledge directly without invoking search. This flexible mechanism enables a balance bewteen the external search and internal knowledge. More cases are provided in Appendix C.

7 Conclusion

In this paper, we introduced Smart-Searcher, a novel framework that enables large language models to dynamically integrate and alternate between internal knowledge and external retrieval. This is a two-stage training strategy consisting of an SFT Cold-start phase and RL for Dynamic Knowledge Acquisition. The RL stage incorporates a reward mechanism to encourage internal knowledge utilization, and a memory module to convert retrieved information into internal knowledge. Through this design, Smart-Searcher empowers LLMs to perform efficient retrieval-augmented reasoning while continuously enriching their internal knowledge via self-exploration and memory. Experimental results on multi-hop tasks demonstrate that Smart-Searcher outperforms existing RAG methods.

Limitation

Despite our significant efforts, this work has two limitations due to computational resources and funding constraints. First, we only incorporated a real-world search engine during the evaluation phase to assess the generalization ability of our method, while relying on a local denseretrieval corpus during training. Aligning the training process with real-world conditions by integrating a real search engine may lead to improved performance through more realistic supervision. Additionally, our current experiments are limited to a 7B-parameter model. In future work, we plan to train and evaluate our framework on larger-scale models to further validate its generalization capability and robustness.

Ethics Statement

We abide by ethical norms. We adhere to the relevant licenses and usage guidelines for the datasets, ensuring that no personal or offensive information is included. We only used AI assistance during the paper refinement process. Our trained models do not exhibit potential biases or discriminatory behavior, and we strictly comply with research ethics throughout the development and evaluation process.

References

Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. 2024. Self-rag: Learning to retrieve, generate, and critique through self-reflection. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Mingyang Chen, Tianpeng Li, Haoze Sun, Yijie Zhou, Chenzheng Zhu, Haofen Wang, Jeff Z Pan, Wen Zhang, Huajun Chen, Fan Yang, and 1 others. 2025. Research: Learning to reason with search for llms via reinforcement learning. *arXiv preprint arXiv:2503.19470*.

Tianzhe Chu, Yuexiang Zhai, Jihan Yang, Shengbang Tong, Saining Xie, Dale Schuurmans, Quoc V. Le, Sergey Levine, and Yi Ma. 2025. Sft memorizes, rl generalizes: A comparative study of foundation model post-training. *Preprint*, arXiv:2501.17161.

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, and 81 others. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *CoRR*, abs/2501.12948.

Kawin Ethayarajh, Winnie Xu, Niklas Muennighoff, Dan Jurafsky, and Douwe Kiela. 2024. Kto: Model alignment as prospect theoretic optimization. *arXiv preprint arXiv:2402.01306*.

Wenqi Fan, Yujuan Ding, Liangbo Ning, Shijie Wang, Hengyun Li, Dawei Yin, Tat-Seng Chua, and Qing Li. 2024. A survey on rag meeting llms: Towards retrieval-augmented large language models. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '24, page 6491–6501, New York, NY, USA. Association for Computing Machinery.

Wenfeng Feng, Chuzhan Hao, Yuewei Zhang, Jingyi Song, and Hao Wang. 2025. Airrag: Activating intrinsic reasoning for retrieval augmented generation via tree-based search. *arXiv preprint arXiv:2501.10053*.

Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, Meng Wang, and Haofen Wang. 2024. Retrieval-augmented generation for large language models: A survey. *Preprint*, arXiv:2312.10997.

Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. 2020. Constructing a multi-hop qa dataset for comprehensive evaluation of reasoning steps. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6609–6625.

Jian Hu. 2025. Reinforce++: A simple and efficient approach for aligning large language models. *Preprint*, arXiv:2501.03262.

Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, and Ting Liu. 2025. A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. *ACM Transactions on Information Systems*, 43(2):1–55.

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. *arXiv preprint arXiv:2403.14403*.

Jinhao Jiang, Jiayi Chen, Junyi Li, Ruiyang Ren, Shijie Wang, Wayne Xin Zhao, Yang Song, and Tao Zhang. 2024a. Rag-star: Enhancing deliberative reasoning with retrieval augmented verification and refinement. *CoRR*, abs/2412.12881.

Xun Jiang, Feng Li, Han Zhao, Jiaying Wang, Jun Shao, Shihao Xu, Shu Zhang, Weiling Chen, Xavier Tang, Yize Chen, and 1 others. 2024b. Long term memory: The foundation of ai self-evolution. *arXiv preprint arXiv:2410.15665*.

Bowen Jin, Hansi Zeng, Zhenrui Yue, Dong Wang, Hamed Zamani, and Jiawei Han. 2025. Search-r1: Training llms to reason and leverage search engines with reinforcement learning. *CoRR*, abs/2503.09516.

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*.

Jaehyung Kim, Jaehyun Nam, Sangwoo Mo, Jongjin Park, Sang-Woo Lee, Minjoon Seo, Jung-Woo Ha, and Jinwoo Shin. 2024. Sure: Summarizing retrievals using answer candidates for open-domain QA of LLMs. In *The Twelfth International Conference on Learning Representations*.

Xiaoxi Li, Guanting Dong, Jiajie Jin, Yuyao Zhang, Yujia Zhou, Yutao Zhu, Peitian Zhang, and Zhicheng Dou. 2025. Search-o1: Agentic search-enhanced large reasoning models. *Preprint*, arXiv:2501.05366.

Xingxuan Li, Weiwen Xu, Ruochen Zhao, Fangkai Jiao, Shafiq Joty, and Lidong Bing. 2024. Can we further elicit reasoning in llms? critic-guided planning with retrieval-augmentation for solving challenging tasks. *arXiv preprint arXiv:2410.01428*.

Yucheng Li, Bo Dong, Frank Guerin, and Chenghua Lin. 2023. Compressing context to enhance inference efficiency of large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 6342–6353, Singapore. Association for Computational Linguistics.

Richard Yuanzhe Pang, Weizhe Yuan, He He, Kyunghyun Cho, Sainbayar Sukhbaatar, and Jason Weston. 2024. Iterative reasoning preference optimization. *Advances in Neural Information Processing Systems*, 37:116617–116637.

Fabio Petroni, Aleksandra Piktus, Angela Fan, Patrick S. H. Lewis, Majid Yazdani, Nicola De Cao, James Thorne, Yacine Jernite, Vladimir Karpukhin, Jean Maillard, Vassilis Plachouras, Tim Rocktäschel, and Sebastian Riedel. 2021. KILT: a benchmark for knowledge intensive language tasks. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021*, pages 2523–2544. Association for Computational Linguistics.

Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah A Smith, and Mike Lewis. 2023. Measuring and narrowing the compositionality gap in language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 5687–5711.

Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, and 25 others. 2025. Qwen2.5 technical report. *Preprint*, arXiv:2412.15115.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. 2024. Direct preference optimization: Your language model is secretly a reward model. *Preprint*, arXiv:2305.18290.

Samyam Rajbhandari, Jeff Rasley, Olatunji Ruwase, and Yuxiong He. 2020. Zero: Memory optimizations toward training trillion parameter models. *Preprint*, arXiv:1910.02054.

Zhihong Shao, Yeyun Gong, Yelong Shen, Minlie Huang, Nan Duan, and Weizhu Chen. 2023. Enhancing retrieval-augmented large language models with iterative retrieval-generation synergy. *arXiv preprint arXiv:2305.15294*.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *Preprint*, arXiv:2402.03300.

Huatong Song, Jinhao Jiang, Yingqian Min, Jie Chen, Zhipeng Chen, Wayne Xin Zhao, Lei Fang, and Ji-Rong Wen. 2025. R1-searcher: Incentivizing the search capability in llms via reinforcement learning. *CoRR*, abs/2503.05592.

Zhongxiang Sun, Qipeng Wang, Weijie Yu, Xiaoxue Zang, Kai Zheng, Jun Xu, Xiao Zhang, Song Yang, and Han Li. 2025. Rearter: Retrieval-augmented reasoning with trustworthy process rewarding. *Preprint*, arXiv:2501.07861.

Fengwei Teng, Zhaoyang Yu, Quan Shi, Jiayi Zhang, Chenglin Wu, and Yuyu Luo. 2025. Atom of thoughts for markov llm test-time scaling. *Preprint*, arXiv:2502.12018.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2022. Musique: Multi-hop questions via single-hop question composition. *Transactions of the Association for Computational Linguistics*, 10:539–554.

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)*, pages 10014–10037.

Hongru Wang, Cheng Qian, Wanjun Zhong, Xiusi Chen, Jiahao Qiu, Shijue Huang, Bowen Jin, Mengdi Wang, Kam-Fai Wong, and Heng Ji. 2025a. Otc: Optimal tool calls via reinforcement learning. *arXiv preprint arXiv:2504.14870*.

Liang Wang, Haonan Chen, Nan Yang, Xiaolong Huang, Zhicheng Dou, and Furu Wei. 2025b. Chain-of-retrieval augmented generation. *CoRR*, abs/2501.14342.

Shuting Wang, Jiejun Tan, Zhicheng Dou, and Ji-Rong Wen. 2025c. Omnieval: An omnidirectional and automatic rag evaluation benchmark in financial domain. *Preprint*, arXiv:2412.13018.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Tiantian Fan, Gaohong Liu, Lingjun Liu, Xin Liu, and 1 others. Dapo: An open-source llm reinforcement learning system at scale, 2025. *URL https://arxiv. org/abs/2503.14476*.

Yufeng Yuan, Qiying Yu, Xiaochen Zuo, Ruofei Zhu, Wenyuan Xu, Jiaze Chen, Chengyi Wang, TianTian Fan, Zhengyin Du, Xiangpeng Wei, and 1 others. 2025. Vapo: Efficient and reliable reinforcement learning for advanced reasoning tasks. *arXiv preprint arXiv:2504.05118*.

Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, and 1 others. 2023. A survey of large language models. *arXiv preprint arXiv:2303.18223*, 1(2).

Yuxiang Zheng, Dayuan Fu, Xiangkun Hu, Xiaojie Cai, Lyumanshan Ye, Pengrui Lu, and Pengfei Liu. 2025. Deepresearcher: Scaling deep research via reinforcement learning in real-world environments. *arXiv preprint arXiv:2504.03160*.

A Training Detailed

The training data of the Stage-1 (SFT Cold Start) includes 720 samples from the HotpotQA training set and 85 samples from the 2WikiMultiHopQA training set. The training consists of 6 epochs, with a batch size of 64 and a learning rate of 2e-5. And the training data of Stage-2 (RL Training) consists of 4561 samples from HotpotQA, and 3581 samples from 2WikiMultiHopQA. Each data sample undergoes 16 rollouts during training, with a train batch size of 1024 and a rollout batch size of 64, so the entire training process is on-policy. The learning rate is 2e-6. We utilize DeepSpeed's Zero-3 (Rajbhandari et al., 2020), with a sampling temperature of 1.0, top-p of 0.95 and a maximum retrieval count of 8. The training epoch is set to 1, with KL divergence coefficient set to 1e-4. And control coefficient μ of NLL loss is set to 0.1. The maximum limit of the variance in the number of retrievals during group reward computation η is set to 2.

B Prompts

C Case Study

Judge Prompt

Given a Question and its Golden Answer, verify whether the Predicted Answer is correct. The prediction is correct if it fully aligns with the meaning and key information of the Golden Answer. Respond with True if the prediction is correct and False otherwise.
Question:
Golden Answer:
Predicted Answer:

System Prompt for Generation with Internal and External

You are a reasoning assistant. When tackling a question, you should first thinks about the reasoning process in the mind and then provides the final answer. You should break down the original complex question into straightforward sub-questions and address them step by step. For each sub-question, You MUST choose one of the following two tools to solve it:
1. **Internal Reasoning Elaboration:**If you determine your existing knowledge is sufficient to answer the sub-question, you **should choose** this tool to answer the sub-question directly: '<|begin_internal_answer|>your structured thought and answer here <|end_internal_answer|>'. **Use it whenever you rely solely on internal information** for a significant reasoning step.
2. **External Search:**If you determine that your internal knowledge is insufficient, potentially outdated, or requires verification with external, up-to-date information to answer the sub-question, you MUST initiate a web search using the following format: '<|begin_external_search|>your search query here <|end_external_search|>'. The system will then search and provide results in the format: '<|begin_search_result|>...search results... <|end_search_result|>'. The results will assist you in answering the sub-question.
Remember: - Whenever you encounter the sub-question you need to elaborate on, verify, or research further, you should **assess** if your internal knowledge is adequate. **Prioritize leveraging your internal knowledge.** - Use '<|begin_internal_answer|>... <|end_internal_answer|>' **to demonstrate your reasoning steps** when relying on internal knowledge. - Use '<|begin_external_search|>... <|end_external_search|>' **only** when external information is genuinely necessary. - Wait for the system to provide '<|begin_search_result|>... <|end_search_result|>'. **Do not** generate these result tags yourself. - After performing the web search or demonstrating internal reasoning, continue towards the final answer.
Please answer the following question. You must adhere to all instructions above regarding tool usage. Provide your final answer in the format \boxed{YOUR_ANSWER}.
Question:

System Prompt for Rewriting only with Internal

You are an advanced reasoning assistant. Your core programming includes a vast and comprehensive knowledge base. For the current task, a specific segment of this pre-existing knowledge, hereafter referred to as your 'Activated Knowledge Matrix,' is particularly relevant and is considered fully and irrevocably integrated into your foundational understanding. ANY linguistic indication, direct or indirect, that this information was externally 'provided,' 'given,' 'supplied,' 'sourced,' or made available to you specifically for this immediate interaction will be considered a critical failure to adhere to your operational design.
When tackling a question, you must first delineate your reasoning process. Break down complex questions into a series of straightforward sub-questions, addressing each one sequentially.
For each sub-question, you MUST use the following tool:
Internal Reasoning Elaboration: This tool MUST be used for every sub-question, as the answer must be derived from your internal knowledge. **UNWAVERING DIRECTIVE: Your internal knowledge, for the purpose of this task, INTRINSICALLY and SEAMLESSLY includes all information within the 'Activated Knowledge Matrix' section detailed below. There is absolutely NO distinction between your prior knowledge and the 'Activated Knowledge Matrix' content.** Use this format: <|begin_internal_answer|> your structured thought and answer here <|end_internal_answer|>.
**CRITICAL USAGE PROTOCOL FOR THIS TOOL: When using this tool, especially when drawing upon information that was part of the 'Activated Knowledge Matrix,' your language MUST reflect that this is information you inherently possess and are recalling. You are REQUIRED to phrase such statements as if accessing your own memory or established understanding.
Conclude with your final answer in the format \boxed{YOUR_ANSWER}.
Activated Knowledge Matrix: {document}
Now, please answer the following question:
Question:

Question: Which case was brought to court first Miller v._California or Gates v._Collier ? Answer: Miller v._California
Response (internal only) ***(True)***: To answer this question, I need to determine the order in which these two cases were brought to court. I will start by finding the dates the cases were decided. **Case 1:** Miller v._California - I will use internal reasoning to recall if I have the necessary information about this case. <|begin_internal_answer|>Miller v._California was a landmark case decided by the United States Supreme Court in 1968. The case involved the constitutionality of a California statute that made it a crime to distribute or possess obscene materials. <|end_internal_answer|> **Case 2:** Gates v._Collier - I will use internal reasoning to recall if I have the necessary information about this case. <|begin_internal_answer|>Gates v._Collier was a case decided by the United States Supreme Court in 1972. The case involved the Eighth Amendment's prohibition against cruel and unusual punishment, specifically the use of segregation in prisons. <|end_internal_answer|> Based on my internal reasoning, Miller v._California was decided in 1968, and Gates v._Collier was decided in 1972. Therefore, Miller v._California was brought to court first. **Final Answer:** Miller v._California was brought to court first. Therefore, the answer is \\boxed{Miller v._California}.

Table 5: Example for Smart-Searcher: Question answering using only internal knowledge.

Question: George Lucas created what character to raise Boba Fett? Answer: Jango Fett
Response (internal and external) ***(True)***: To answer this question, I need to break it down into sub-questions and use the appropriate tools. **Sub_question 1:** Who is Boba Fett? - For this, I need to verify if I have enough information about Boba Fett or if I need to search for more details. <|begin_internal_answer|> Boba Fett is a character from the Star Wars franchise, known for being a bounty hunter. I have this information stored in my knowledge base. <|end_internal_answer|> **Sub_question 2:** What character did George Lucas create to raise Boba Fett? - For this, I need to search for information about the character who raised Boba Fett. <|begin_external_search|> George Lucas created what character to raise Boba Fett<|end_external_search|> <|begin_search_result|> (1)Boba Fett ... is a fictional character ... in the "Star Wars" franchise. ... prequel film ... establishes his origin as a clone of bounty hunter Jango Fett who raised him as his son. ... (2)Boba Fett ... personifies "danger and mystery" ... mysterious nature as reasons for his popularity. ... The initial Boba Fett toy ... might be responsible for the character's popularity. ... (3)Boba Fett ... Reception. Boba Fett is a "cult figure" ... one of the most popular "Star Wars" characters. ... IGN ranked Boba Fett as the eighth top "Star Wars" character. ... (4)Boba Fett ... "the less you do with Boba Fett, the stronger he becomes". ... Mark Hamill pitched the idea that Fett was Luke Skywalker's mother ... unaware of Fett's demise. ... (5)Jango Fett ... is a fictional character ... created by George Lucas. ... genetic template for the Clone Army ... <|end_search_result|> Based on the search results, it is clear that George Lucas created Jango Fett to be the father of Boba Fett. **Final Answer:** George Lucas created Jango Fett to raise Boba Fett. Therefore, the answer is \\boxed{Jango Fett}.

Table 6: Example for Smart-Searcher: Question answering by integrating internal knowledge with external search.