🌱 SEEDS: PREVIEW-GUIDED EVIDENCE ADMISSION FOR SEARCH AGENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Search agents extend large reasoning models with multi-round access to external knowledge, yet most existing systems largely conflate retrieving evidence with consuming it. Retrieved documents are commonly incorporated according to fixed or rank-based policies, despite the fact that evidence utility is conditioned on the agent's current reasoning state. Consequently, a lower-ranked candidate may be critical for the current step, while a higher-ranked candidate may provide redundant, irrelevant, or misleading information. We formulate this overlooked problem as **evidence admission**, which asks, at each search step, which retrieved candidates should be allowed to enter the agent's evolving reasoning context. To enable such selective control, we introduce **SEEDS (Selective Evidence Entry During Search)**, a preview-guided evidence admission framework for search agents. SEEDS first exposes lightweight previews of retrieved candidates and then lets the agent jointly determine which candidates and how many should be fully inspected, conditioned on its current reasoning state. Unlike fixed or adaptive prefix policies, SEEDS can select arbitrary subsets with flexible cardinality, and it can be instantiated directly with an existing reasoning agent or further optimized through learning. Across nine knowledge-intensive QA benchmarks under both local corpus and live web search, SEEDS consistently improves search effectiveness while substantially reducing full-evidence consumption. Controlled analyses show that SEEDS more effectively recovers answer-bearing evidence and frequently selects useful candidates beyond the corresponding retrieval prefix. These results establish evidence admission as a complementary dimension of agentic search, separating what information is retrieved from what information is ultimately allowed to shape reasoning. Anonymized code is available at `https://anonymous.4open.science/r/SEEDS-5C1C`.

1 INTRODUCTION

Large Reasoning Models (LRMs) increasingly rely on external search to access up-to-date and task-relevant knowledge for solving knowledge-intensive problems. A standard approach is retrieval-augmented generation (RAG), which typically retrieves relevant information before generation and incorporates the retrieved evidence into the LMR input context (Fan et al., 2024; Gao et al., 2024; Sharma, 2025). However, complex questions often require iterative information seeking because the information needed by the model evolves as reasoning proceeds and cannot always be determined from the initial query alone (Zheng et al., 2025; Sorokin et al., 2026).

Search agents address this challenge by interleaving reasoning and retrieval over multiple rounds. At each round, an agent can formulate a new query based on its current reasoning state, retrieve additional information, and use the retrieved evidence to guide subsequent reasoning and search (Jin et al., 2025; Li et al., 2025a; Wei et al., 2026; Li et al., 2025b). This interaction creates an important distinction that is often overlooked: *retrieving evidence does not necessarily mean that the evidence should be consumed by the agent*. Once retrieved information enters the reasoning trajectory, it can influence not only the current inference step, but also future search queries, subsequent reasoning decisions, and ultimately the final answer. Therefore, an effective search agent must decide not only what information to retrieve, but also which retrieved information should be allowed to influence its evolving reasoning process.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

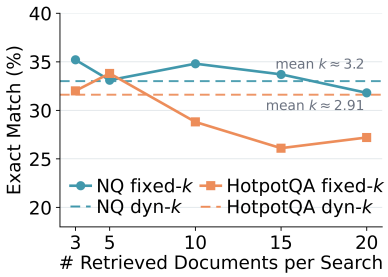


Figure 1: Impact of retrieval rank on QA tasks in multi-round agentic search-augmented reasoning.

We refer to this decision as *evidence admission*. Evidence admission asks, at each search step, which candidates from a retrieved evidence pool should enter the agent's reasoning context. This decision is inherently dependent on the current reasoning state. A document that provides essential information at one step may become redundant after related evidence has already been acquired, while a lower-ranked candidate may contain complementary information that is critical for resolving the agent's current uncertainty. Retrieval rank alone is therefore insufficient for determining the utility of evidence during reasoning (Trivedi et al., 2023; Lee et al., 2025; Chang et al., 2025; Meng et al., 2025). Existing search pipelines nevertheless commonly admit a fixed number of highly ranked results or restrict adaptive selection to a prefix of the retrieval ranking (Jin et al., 2025; Sharifymoghaddam & Lin, 2026). As illustrated in Fig. 1, the optimal number of retrieved documents varies across tasks, making it difficult to determine a suitable amount of evidence in advance. Allowing the model to select this number dynamically helps, but still falls short of the best fixed setting even when the average number selected is close to the optimum. This suggests that evidence quantity alone is insufficient: too little may omit necessary information, while too much may introduce redundancy or distraction, and *which* documents are admitted is equally important.

A key challenge is that the agent must estimate the utility of retrieved candidates *before* consuming their full contents. Inspecting every candidate defeats the purpose of selective admission, while retrieval scores alone provide limited information about reasoning-time utility. We therefore introduce an intermediate representation, termed an *evidence preview*, that provides a lightweight view of each candidate before full inspection. The preview is retrieval-environment agnostic: for local corpora, it can be instantiated as document identity together with a compact query-relevant excerpt, while for live web search, the title and search-result snippet naturally serve this role. Evidence previews thus provide a low-cost interface between retrieval and full evidence consumption.

To enable such selective control, we introduce SEEDS (**S**elective **E**vidence **E**ntry **D**uring **S**earch), a preview-guided evidence admission framework for search agents. After each retrieval step, SEEDS first exposes lightweight previews of a candidate evidence pool. Conditioned on these previews and the agent's current reasoning state, the agent determines which candidates should be fully inspected and how many are necessary. Unlike fixed or adaptive prefix strategies, SEEDS can admit an arbitrary subset of the retrieved candidates with flexible cardinality. Only the admitted candidates are expanded into detailed evidence and incorporated into the subsequent reasoning trajectory. The process can then repeat as the agent continues reasoning and searching. Importantly, SEEDS is a general interaction framework rather than a specific learning algorithm. It can be instantiated directly with an existing reasoning agent, while supervised or reinforcement learning can further optimize the admission policy when training is available. We evaluate SEEDS across nine knowledge-intensive QA benchmarks under both local corpus and live web search settings, using multiple search-agent paradigms. SEEDS improves answer quality while reducing full-evidence consumption, and controlled analyses show that it more effectively admits answer-bearing evidence, including useful candidates beyond the retrieval prefix. These results demonstrate that gains come from improving both *which* evidence is admitted and *how much* is consumed.

Our major **contributions** are summarized as follows:

- We formulate *evidence admission* as a distinct decision problem in agentic search. This formulation separates evidence availability after retrieval from the decision of which evidence should enter and influence the evolving reasoning trajectory.
- We introduce SEEDS, a general preview-guided evidence admission framework that allows a search agent to jointly determine which retrieved candidates to inspect and how many are needed at each reasoning step. Unlike rank-prefix strategies, SEEDS supports flexible, non-prefix evidence selection conditioned on the current reasoning state.
- Across nine open-domain QA benchmarks with local and live search, we show that SEEDS improves search effectiveness while reducing full-evidence consumption. Controlled analyses further demonstrate more effective evidence selection and show that SEEDS can benefit from additional training without relying on learning for its basic operation.

2 Related Work

Our work is most closely related to two lines of research.

Adaptive Retrieval in Retrieval-Augmented Generation. Conventional retrieval-augmented generation (RAG) often provides language models with a fixed number of top-ranked passages, despite variation in the amount and composition of evidence required across queries. Self-RAG (Asai et al., 2024) makes retrieval conditional on model needs, while Adaptive-k (Taguchi et al., 2025) predicts a query-specific number of passages. DynamicRAG (Sun et al., 2025) adapts passage ordering and set size using generation feedback. SETR (Lee et al., 2025) and DPS (Meng et al., 2025) move beyond individual passage relevance to select variable-size evidence sets that account for relationships among passages. MAIN-RAG (Chang et al., 2025) filters candidates using multi-agent scoring and an adaptive threshold. These methods show the value of adapting retrieved context, but typically make selection with respect to a query and its candidate set. In agentic search, evidence utility also depends on the evolving reasoning state. SEEDS therefore focuses on *evidence admission*: deciding, at each search step, which retrieved candidates should be fully inspected and incorporated into subsequent reasoning.

Multi-Round Retrieval and Search Agents. A complementary line of work integrates retrieval into multi-round search-augmented reasoning. IRCoT (Trivedi et al., 2023) interleaves reasoning with retrieval, while FLARE (Jiang et al., 2023) and DRAGIN (Su et al., 2024) adapt when retrieval is invoked or how it is guided by the current generation state. More recent systems such as Search-o1 (Li et al., 2025a), Search-R1 (Jin et al., 2025), and ReasonRAG (Zhang et al., 2025b) optimize multi-round search behavior, while InfoAgent (Zhang et al., 2025a) and GRASP (Gandhi et al., 2026) provide richer mechanisms for browsing and inspecting retrieved information. These methods primarily address when and how to search, or how retrieved items are inspected. In contrast, SEEDS introduces an explicit admission step between retrieval and evidence consumption. Given lightweight previews of multiple candidates, the agent jointly determines which candidates to inspect and how many are needed, conditioned on its current reasoning state. This enables flexible subset selection beyond fixed or adaptive retrieval prefixes.

3 Methodology

This section presents SEEDS, a preview-guided evidence admission framework for multi-round search agents. We first formalize agentic search and the evidence admission problem (§3.1). We then introduce the preview-guided admission mechanism (§3.2), which separates lightweight candidate observation from detailed evidence consumption. Finally, we describe the end-to-end inference procedure (§3.3). Figure 2 illustrates how SEEDS inserts an explicit evidence admission step between retrieval and full evidence consumption.

3.1 Problem Formalization

Agentic Search-Augmented Reasoning. We consider a multi-round search agent that solves an input question q by interleaving reasoning with external search. Given a task instruction I, the model generates a reasoning trajectory $\mathcal{R} = (\mathcal{R}_1, \ldots, \mathcal{R}_{T_\mathcal{R}})$ followed by a final answer $a = (a_1, \ldots, a_{T_a})$. During reasoning, the model may issue multiple search queries and incorporate external evidence into its trajectory. Let $\mathcal{Z}_{<t}$ denote the evidence incorporated before reasoning step t. The generation process can be formalized as a prediction process:

$$P(\mathcal{R}, a \mid I, q) = \underbrace{\prod_{t=1}^{T_\mathcal{R}} P(\mathcal{R}_t \mid \mathcal{R}_{<t}, I, q, \mathcal{Z}_{<t})}_{\text{Reasoning Process}} \cdot \underbrace{\prod_{t=1}^{T_a} P(a_t \mid a_{<t}, \mathcal{R}, I, q)}_{\text{Answer Generation}}. \tag{1}$$

Evidence incorporated during search therefore becomes part of the evolving reasoning state and can directly affect subsequent reasoning, future search actions, and the final answer.

Retrieval and Evidence Admission. Suppose the model initiates its i-th search after accumulating reasoning history $\mathcal{R}^{(<i)}$ and previously incorporated evidence $\mathcal{Z}^{(<i)}$. It generates a search query and

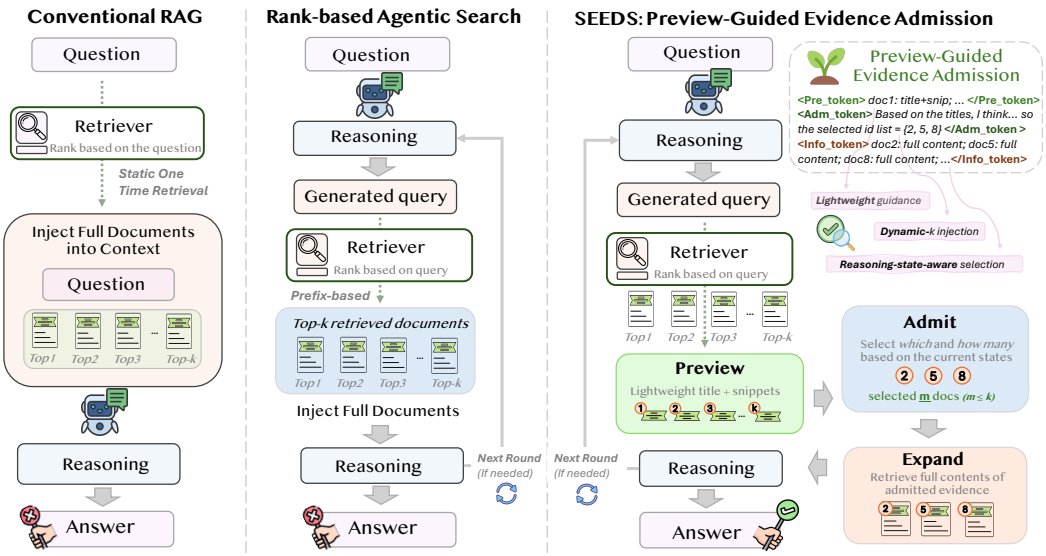


Figure 2: Overview of SEEDS compared with conventional RAG and rank-based agentic search. Conventional RAG injects top-ranked documents before reasoning, while agentic search retrieves iteratively but still injects a rank prefix at each round. SEEDS presents lightweight previews, allowing the reasoning model to choose *which candidates*, and *how many*, to expand based on its current state. Only expanded documents enter subsequent reasoning rollouts.

gets a ranked candidate set with size K from the retriever $\mathrm{Ret}(\cdot)$:

$$q_s^{(i)} \sim P_\theta\left(\cdot \mid I, q, \mathcal{R}^{(<i)}, \mathcal{Z}^{(<i)}\right), \qquad \mathcal{D}^{(i)} = \mathrm{Ret}\left(q_s^{(i)}\right) = \left(d_1^{(i)}, \ldots, d_K^{(i)}\right). \tag{2}$$

Retrieval determines which evidence is *available* to the agent, but not which evidence should influence subsequent reasoning. We therefore introduce an explicit *evidence admission* decision. Let $\mathcal{H}^{(i)}$ denote the agent's reasoning state before the admission decision, including its reasoning history, previous search actions, and previously incorporated evidence. Given the retrieved candidate set $\mathcal{D}^{(i)}$, the goal is to select an admitted subset $\mathcal{E}^{(i)} \subseteq \mathcal{D}^{(i)}$ whose detailed contents are expected to be useful for the agent's current information need.

This formulation differs from fixed or adaptive rank-prefix strategies. A fixed-k strategy admits $\{d_1^{(i)}, \ldots, d_k^{(i)}\}$, while an adaptive-$k$ strategy changes the prefix length. Evidence admission allows both the identity and the number of selected candidates to depend on the current reasoning state.

3.2 PREVIEW-GUIDED EVIDENCE ADMISSION

Directly deciding which candidates to admit presents a fundamental information trade-off. The agent needs sufficient information to assess each candidate, but reading every candidate in full would eliminate the benefit of selective admission. SEEDS addresses this problem by separating lightweight candidate observation from detailed evidence consumption. It first exposes a compact *preview* of each retrieved candidate and then uses these previews together with the current reasoning state to make the admission decision.

Candidate Preview. At search round i, each retrieved candidate $d_j^{(i)} \in \mathcal{D}^{(i)}$, $j \in \{1, ..., K\}$ is represented by a lightweight preview:

$$\mathcal{P}^{(i)} = \left\{p_1^{(i)}, \ldots, p_K^{(i)}\right\}, \qquad p_j^{(i)} = \mathrm{Preview}\left(q_s^{(i)}, d_j^{(i)}\right). \tag{3}$$

The preview provides a low-cost observation of each candidate before its full content is consumed. The preview operator depends only on the current search query and retrieved document, and is independent of the agent's reasoning trajectory. It summarizes information available from the retrieval

result itself, while the subsequent admission decision is conditioned on the evolving reasoning state. For local corpus retrieval, we instantiate $\mathrm{Preview}(\cdot)$ as a deterministic, compact, query-conditioned representation that preserves both document identity and locally relevant content. Specifically, we concatenate the document title with a local window selected by query-keyword matching to expose the region most relevant to the current search query:

$$p_j^{(i)} = \left[\mathrm{Title}(d_j^{(i)}); \mathrm{Match}(q_s^{(i)}, d_j^{(i)})\right]. \tag{4}$$

For live web search, we directly use the title and snippet returned by the search engine as the candidate preview $\mathrm{Preview}(\cdot)$. Detailed preview construction is provided in Appendix B.

State-Conditioned Evidence Admission. Given the preview set $\mathcal{P}^{(i)}$, the agent decides which candidates should be inspected in full content. We define the admission policy $\mathcal{A}$ as:

$$\mathcal{E}^{(i)} = \mathcal{A}\left(\mathcal{H}^{(i)}, \mathcal{P}^{(i)}\right), \tag{5}$$

Because $\mathcal{A}$ conditions on the current reasoning state, both the composition and cardinality of $\mathcal{E}^{(i)}$ can change as the agent acquires new information.

Unlike rank-prefix selection, the admitted set need not contain the highest-ranked candidates or follow any contiguous prefix of the retrieval ranking. The action space therefore supports arbitrary subsets of $\mathcal{D}^{(i)}$, including the empty set when no candidate warrants further inspection. The admission policy can be instantiated directly by an existing reasoning model through prompting, or further optimized through supervised or reinforcement learning when training is available.

Evidence Expansion and Incorporation. Admission determines *which* candidates should be consumed, while expansion is the environment operation that reveals their detailed contents:

$$\mathcal{Z}^{(i)} = \mathrm{Expand}\left(\mathcal{E}^{(i)}\right) = \left\{\mathrm{Full}(d) \mid d \in \mathcal{E}^{(i)}\right\}. \tag{6}$$

The resulting evidence $\mathcal{Z}^{(i)}$ is incorporated into the reasoning trajectory and becomes part of the state used for subsequent reasoning and search.

Overall, SEEDS decomposes evidence acquisition within agentic search into the following steps: $\mathcal{D}^{(i)} \xrightarrow{\mathrm{Preview}} \mathcal{P}^{(i)} \xrightarrow[\mathcal{H}^{(i)}]{\mathcal{A}} \mathcal{E}^{(i)} \xrightarrow{\mathrm{Expand}} \mathcal{Z}^{(i)}$. This decomposition separates evidence availability, lightweight observation, admission, and detailed document consumption.

3.3 INFERENCE

At inference time, the search agent iteratively generates a query and retrieves a candidate evidence pool at each search round. As shown in Algorithm 1, the agent first observes lightweight previews of the retrieved candidates and then selects an admitted subset conditioned on its current reasoning state. Only the admitted candidates are expanded in full and incorporated into the reasoning trajectory. This process repeats until the model produces a final answer or reaches the search budget.

Algorithm 1 SEEDS inference

Require: Reasoning model $\mathcal{M}$, question q, retriever Ret, operators $\mathrm{Preview}$ and Expand
Initialize reasoning history $\mathcal{H}$ with q
while true **do**
  Generate reasoning and a search or answer action u using $\mathcal{M}(\mathcal{H})$
  Append the generated reasoning and u to $\mathcal{H}$
  if u is a final answer **then**
    return the answer in u
  end if
  Extract the search query $q_s^{(i)}$ from u
  Retrieve candidates $\mathcal{D}^{(i)} \leftarrow \mathrm{Ret}(q_s^{(i)})$
  Construct previews $\mathcal{P}^{(i)} \leftarrow \mathrm{Preview}(q_s^{(i)}, \mathcal{D}^{(i)})$
  Append $\mathcal{P}^{(i)}$ to $\mathcal{H}$
  Select admitted evidence $\mathcal{E}^{(i)} \leftarrow \mathcal{M}(\mathcal{H}, \mathcal{P}^{(i)})$, where $\mathcal{E}^{(i)} \subseteq \mathcal{D}^{(i)}$
  Expand admitted evidence $\mathcal{Z}^{(i)} \leftarrow \mathrm{Expand}(\mathcal{E}^{(i)})$
  Append $\mathcal{Z}^{(i)}$ to $\mathcal{H}$
end while

4 EXPERIMENTS

We evaluate SEEDS through four research questions (RQs): **RQ1:** Can SEEDS improve the effectiveness of multi-round search agents while reducing full-evidence consumption across diverse reasoning benchmarks and search environments? **RQ2:** Does preview-guided evidence admission select more useful evidence than retrieval-rank-based strategies? **RQ3:** How robust is SEEDS to noisy or distracting retrieval results, and how sensitive is it to the preview design? **RQ4:** Does SEEDS generalize across reasoning backbones and agentic search paradigms, and can its evidence admission policy be further improved through learning?

4.1 EXPERIMENT SETUP

Datasets. We evaluate SEEDS on nine benchmarks spanning ***(i) open-domain QA***, including single-hop NQ (Kwiatkowski et al., 2019) and TriviaQA (Joshi et al., 2017), and multi-hop HotpotQA (Yang et al., 2018), 2Wiki (Ho et al., 2020), MuSiQue (Trivedi et al., 2022), and Bamboogle (Press et al., 2023); and ***(ii) challenging search-intensive tasks***, including BrowseComp (Wei et al., 2025), BrowseComp-ZH (Zhou et al., 2025), and Humanity's Last Exam (HLE) (Phan et al., 2026). More details are provided in Appendix A.1.

Baselines. We compare against four baseline families. ***(i) No retrieval*** uses Direct Reasoning without external evidence. ***(ii) Single-round RAG*** includes fixed-k RAG (Lewis et al., 2021) and the adaptive retrieval method MAIN-RAG (Chang et al., 2025). ***(iii) Multi-round agentic search*** includes ReAct (Yao et al., 2023) with fixed k and a dynamic-k variant (dyn-k), where the reasoning model determines how many top-ranked documents to consume at each search round without previews. We also extend MAIN-RAG to the multi-round setting, denoted MAIN-RAG (multi-round). ***(iv) Agentic search with document refinement*** uses Search-o1 (Li et al., 2025a), representing a different agentic search paradigm that performs additional reasoning over retrieved documents.

Implementation. Our main experiments use QwQ-32B (Team, 2025) as the main reasoning backbone model for evaluation, with K=10 retrieved candidates per search. For open-domain QA under ***local corpus retrieval***, we use the 2018 Wikipedia dump (Karpukhin et al., 2020) as the knowledge corpus and E5 (Wang et al., 2024) for retrieval, and report Exact Match (EM). For the challenging benchmarks under ***live web search***, we use the Serper API for search and the Jina Reader API to fetch web-page contents, with LLM-judged accuracy reported. We also report token-level F1 on the predicted answers, and the average number of fully accessed documents or web pages per question.

4.2 MAIN RESULTS (RQ1)

Table 1: Main results (EM%, F1%, #Doc per question) of SEEDS compared with baselines on open-domain QA benchmarks using the local Wikipedia dump corpus, with the reasoning backbone QwQ-32B model.

Method	Single-hop QA				Multi-hop QA								Avg.		
	NQ		TriviaQA		HotpotQA		2Wiki		MuSiQue		Bamboogle		(all)		
	EM	F1	EM	F1	EM	F1	EM	F1	EM	F1	EM	F1	EM↑	F1↑	#Doc↓
Without Retrieval															
Direct Reasoning	28.0	39.7	66.4	72.0	27.0	36.6	32.6	**38.5**	10.2	19.2	48.0	63.0	35.4	44.8	0
Retrieval-augmented Reasoning															
RAG $k=3$	39.8	49.7	60.8	66.6	31.0	39.7	22.0	28.9	8.4	17.4	32.8	45.0	32.5	41.2	3.0
RAG $k=10$	42.8	**53.2**	64.2	70.8	30.4	39.9	25.8	31.9	10.0	19.1	28.8	40.5	33.7	42.6	10.0
Main-RAG	**43.0**	53.0	64.0	70.3	31.6	40.2	24.2	31.2	14.0	23.0	32.8	44.9	34.9	43.8	9.8
Multi-round Agentic Search															
ReAct $k=3$	35.2	45.2	64.2	72.2	32.0	44.7	27.4	34.3	13.2	21.8	52.8	63.9	37.5	47.0	7.7
ReAct $k=10$	34.8	45.1	63.8	71.8	28.8	39.9	21.0	26.5	11.4	19.4	52.0	63.6	35.3	44.4	14.9
ReAct dyn-k	33.0	43.1	65.0	73.3	31.6	43.8	25.6	35.4	14.8	25.6	51.2	65.6	36.9	47.8	7.7
Main-RAG (multi-round)	35.6	45.1	65.4	73.3	33.0	45.7	28.2	36.2	14.0	23.0	56.0	68.0	38.7	48.6	11.0
SEEDS-ReAct *(ours)*	39.6	51.7	**69.6**	**76.9**	**35.6**	**47.2**	**31.2**	38.1	**16.6**	**28.0**	**56.8**	**68.6**	**41.6**	**51.8**	**3.1**
Multi-round Agentic Search with Reason-in-Documents															
Search-o1	**40.6**	**53.7**	68.2	**76.8**	38.4	49.8	32.2	39.4	**16.6**	27.4	59.2	71.0	42.5	53.0	30.2
SEEDS-o1 *(ours)*	40.2	52.9	**68.8**	76.4	**39.8**	**51.4**	**33.6**	**42.0**	16.4	**27.9**	**60.8**	**71.4**	**43.3**	**53.7**	**12.6**

We instantiate our preview-guided evidence admission mechanism within two different agentic search paradigms, ReAct and Search-o1, yielding **SEEDS-ReAct** and **SEEDS-o1**, respectively. Tables 1 and 2 report results under local-corpus and live-web search. Here are main observations:

SEEDS improves answer accuracy while reducing full-document consumption. As shown in Table 1, SEEDS-ReAct achieves the highest EM on all six local Wikipedia benchmarks among the multi-round agentic search methods without Reason-in-Documents. Its average EM exceeds single-round MAIN-RAG by **+6.7** points and the multi-round baselines by **+2.9**∼**6.3** points. Full-document consumption decreases by approximately ↓**60–79%** relative to the multi-round baselines. When integrated with Search-o1, SEEDS-o1 slightly increases average EM while reducing full-document consumption from 30.2 to 12.6 documents per question (approximately ↓**58%**). These results show that selective evidence admission improves answer quality while consuming fewer full documents.

The benefits extend to live-web search. Results in Table 2 show that SEEDS remains effective on more challenging live-web benchmarks with more complex questions, where retrieved candidates come from open-web search and are less controlled in quality and relevance. Nevertheless, SEEDS-o1 improves average accuracy by **+2.5** points while reducing the number of fetched full web pages by approximately ↓**70%**. These results indicate that preview-guided evidence admission generalizes beyond local retrieval and remains effective in live-web search.

Table 2: Results (%) on challenging web browsing and expert-level question benchmarks with live web search with QwQ-32B.

	BrowseComp		BrowseComp-zh		HLE		Avg.		
Method	Acc	F1	Acc	F1	Acc	F1	Acc↑	F1↑	#Pages↓
Search-o1	3.32	6.09	14.19	21.30	10.34	**10.60**	9.28	12.66	27.81
SEEDS-o1 ***(ours)***	**5.29**	**8.42**	**16.26**	**23.85**	**13.79**	6.66	**11.78**	**12.98**	**8.30**

4.3 EVIDENCE ADMISSION BEHAVIOR AND EFFICIENCY (RQ2)

A central question for SEEDS is whether preview-guided admission improves the *quality and composition* of evidence entering the reasoning trajectory, rather than merely adapting how much evidence is consumed. We evaluate this through admission behavior, evidence quality, and consumption efficiency, using the same retrieval pool size $K{=}10$ for both prefix-based baselines and SEEDS variants. To assess admission quality, we mark a document as ***answer-bearing*** if its normalized full text contains any normalized gold-answer alias. Detailed matching rules are provided in Appendix C.1.

Non-prefix Evidence Admission. We first test whether SEEDS changes *which* retrieved candidates are admitted, rather than only admitting those top-ranked documents. Across more than 5k search rounds on the open-domain QA benchmarks, Fig. 3 shows that SEEDS-ReAct and SEEDS-o1 admit evidence throughout the whole retrieved candidate pool, including those lower-ranked documents. In contrast, ReAct dyn-k selects almost exclusively from the highest-ranked candidates (rank < 4), with admission probability dropping sharply beyond the top few ranks, with almost 0% admission probability for documents ranked starting from 6. As summarized in Table 3, 44.1% and 39.6% of documents admitted by SEEDS-ReAct and SEEDS-o1 fall beyond their corresponding evidence set size $|S|$ (top-$|S|$ prefixes), and their selected sets depart from these prefixes in roughly 80% of search rounds. Thus, SEEDS adapts not only admission cardinality, but also the flexible composition of the evidence entering the reasoning trajectory.

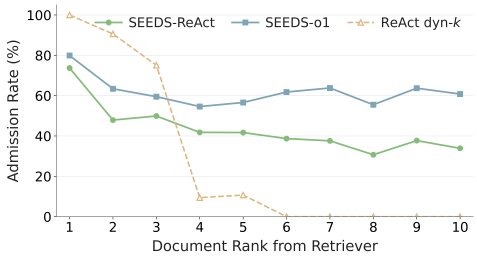
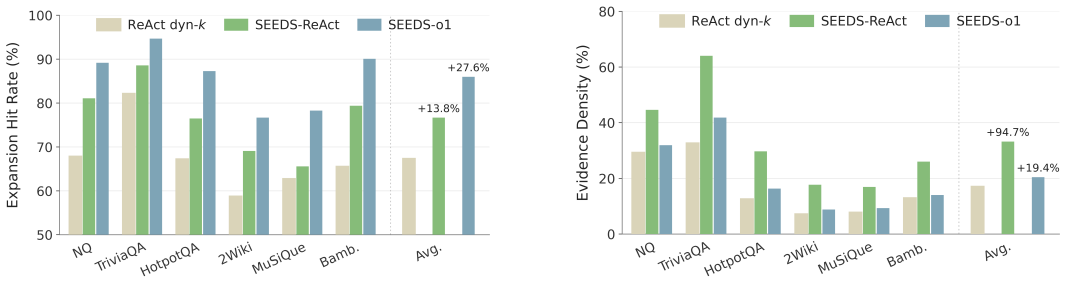


Figure 3: Evidence admission probability (%) by retriever rank for SEEDS variants and the prefix-based ReAct dyn-k baseline.

Evidence Admission Quality. We next examine whether this greater flexibility leads to more useful evidence being admitted. When answer-bearing evidence is available in the candidate pool, SEEDS admits it more reliably than the rank-prefix baseline across all six open-domain QA benchmarks. As shown in Fig. 4(a), SEEDS-ReAct and SEEDS-o1 achieve average relative improvements of 13.8% and 27.6% in answer-bearing evidence hit rate over ReAct dyn-k, respectively, highlighting the benefit of preview guidance for evidence selection. Moreover, Fig. 4(b) shows that the ad-



(a) Answer-bearing evidence hit rate.

(b) Answer-bearing evidence density.

Figure 4: Evidence admission quality. (a) *Hit rate* is the probability that an answer-bearing document is admitted when it is available in the preview candidate pool. (b) *Density* is the proportion of answer-bearing documents among all documents admitted into the reasoning trajectory.

mitted sets contain a substantially higher density of answer-bearing evidence, indicating that SEEDS preserves more useful information while filtering unnecessary content. Together, these results show that preview-guided admission improves both the likelihood of admitting useful evidence and the quality of the evidence that ultimately enters the reasoning trajectory.

Evidence Consumption Efficiency. Finally, we examine whether selective admission also reduces evidence-acquisition cost. As shown in Table 4, SEEDS-ReAct reduces both search frequency and full-document consumption, while cutting document-token usage by 58% relative to ReAct dyn-k. Similarly, SEEDS-o1 uses nearly the same number of searches as Search-o1 but reduces full-document reads and document-token consumption by about 58%. These results show that SEEDS improves efficiency by concentrating full-document access on fewer, more useful retrieved documents.

Table 3: Admitted answer-bearing evidence with rank beyond the evidence size $|S|$.

Method	Documents	Rounds
ReAct dyn-k	0.0%	0.0%
SEEDS-ReAct	44.1%	77.7%
SEEDS-o1	39.6%	80.2%

Table 4: Evidence-acquisition efficiency in terms of the number of searches, documents, and tokens per question.

Method	Searches ↓	Full docs ↓	Docs/search ↓	Doc. tokens ↓
ReAct dyn-k	2.67	7.7	2.93	1,267
SEEDS-ReAct	**2.00**	**3.1**	**1.78**	**528**
Search-o1	**3.02**	30.2	10.00	5,006
SEEDS-o1	3.03	**12.6**	**4.25**	**2,075**

4.4 ROBUSTNESS AND SENSITIVITY (RQ3)

Robustness to Retrieval Perturbations. We evaluate whether preview-guided admission can filter distracting evidence before it enters the reasoning trajectory. For each top-K candidate pool, we replace a fraction ρ of *non-answer-bearing* documents with same-topic distractors sampled from outside the top-K, while always retaining the available answer-bearing evidence. Increasing ρ therefore creates noisier candidate pools without removing the evidence required to answer the question.

Table 5: Robustness comparison under preview candidate-pool perturbations.

Method	Clean	$\rho = 30\%$	$\rho = 50\%$	$\rho = 80\%$
ReAct dyn-k	36.9	37.0 (+0.1)	36.4 (-0.5)	33.4 (-3.5)
ReAct $k = 3$	37.5	38.2 (+0.7)	36.0 (-1.5)	36.8 (-0.7)
ReAct $k = 10$	35.3	36.0 (+0.7)	35.7 (+0.4)	36.9 (+1.6)
SEEDS-ReAct	**41.6**	**42.1** (+0.5)	**42.3** (+0.7)	**42.9** (+1.3)
Injected noise admitted		6.4%	7.5%	7.3%

As shown in Table 5, rank-prefix methods with smaller evidence sets are more sensitive to retrieval perturbations. ReAct dyn-k and ReAct k=3 degrade as more high-ranked candidates are replaced because they cannot selectively bypass distractors within the consumed prefix. ReAct k=10 is more stable, likely because the larger context preserves sufficient useful evidence despite the perturbation. In contrast, SEEDS-ReAct admits fewer than 8% of injected distractors across all settings while maintaining or improving accuracy. At $\rho = 0.8$, it improves by 1.3 EM over clean retrieval, suggesting that preview-guided admission can still isolate useful evidence under heavy perturbation.

Sensitivity to Preview Candidate-Pool Size. We study how the preview pool size affects answer quality and search behavior by varying N from 5 to 20 based on SEEDS-ReAct. As shown in Fig. 5, accuracy remains relatively stable, indicating limited sensitivity to N. Larger pools lead the agent to admit more documents per question while generally requiring fewer search rounds, suggesting that more evidence can be incorporated within each round. The effect on accuracy is task-dependent: single-hop NQ slightly benefits from larger pools, whereas multi-hop HotpotQA degrades at $N = 20$, likely due to additional distracting evidence. We further conduct experiments on various preview designs and include analyses in Appendix B.

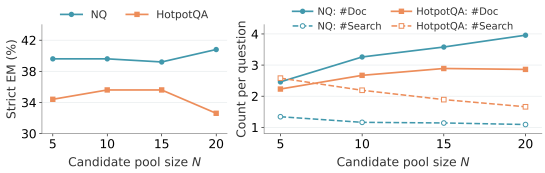


Figure 5: Impact of the size of the preview candidate pool.

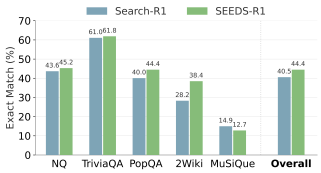


Figure 6: RL analysis.

4.5 GENERALIZATION AND LEARNING (RQ4)

Generalization across Reasoning Backbones. Table 6 evaluates SEEDS with DeepSeek-V4.1-Flash (DeepSeek-AI et al., 2026), which improves accuracy over QwQ-32B on all four benchmarks. Compared with Search-o1 under the same DeepSeek backbone, SEEDS gains 1.0 EM point on HotpotQA and 1.2 on 2Wiki, matches HLE accuracy, and loses 1.4 EM points on NQ. It reduces full-document access by 83–88% on local QA and fetched pages by approximately 57% on HLE. These results show that SEEDS benefits from stronger reasoning models while preserving its evidence-efficiency advantage.

Table 6: Generalization across different reasoning backbones and search paradigms.

	NQ		HotpotQA		2Wiki		HLE	
Configuration	↑EM	↓#Doc	↑EM	↓#Doc	↑EM	↓#Doc	↑Acc.	↓#Pages
SEEDS (QwQ-32B)	40.2	**3.26**	39.8	**2.67**	33.6	**2.87**	13.8	**8.3**
SEEDS (DeepSeek)	43.7	**3.26**	**45.2**	3.91	**58.4**	4.69	**45.1**	9.93
Search-o1 (DeepSeek)	**45.1**	19.18	44.2	26.20	57.2	37.56	**45.1**	22.86

Extension to a Learnable Setting. Finally, we investigate whether SEEDS can be extended to a learnable setting and further improved through training. Following the RL recipe of Search-R1 (Jin et al., 2025), we train a Qwen-2.5-7B (Qwen et al., 2025) base model with the SEEDS-ReAct interaction on NQ and HotpotQA. We first apply supervised fine-tuning to establish the preview-admission interaction pattern, followed by reinforcement learning with a final-answer reward and an auxiliary reward for admitting answer-bearing evidence, yielding SEEDS-R1. As shown in Fig. 6, SEEDS-R1 improves overall EM from 40.5 to 44.4 and shows clear transfer gains on several held-out datasets, although the improvement is not uniform across all tasks. These results demonstrate that evidence admission can be further optimized through training beyond its training-free instantiation. Additional training details and reward design are provided in Appendix E.

5 CONCLUSION

This work studies an overlooked decision in agentic search: which retrieved evidence should enter and influence the evolving reasoning trajectory. We formulate this problem as *evidence admission* and introduce SEEDS, a preview-guided framework for reasoning-aware selection before full evidence consumption. Across nine knowledge-intensive QA benchmarks under both local-corpus and live-web search, SEEDS improves search effectiveness while substantially reducing full-evidence consumption. Further analyses show that SEEDS admits useful evidence beyond retrieval prefixes, more reliably recovers answer-bearing evidence, and remains effective across retrieval perturbations and reasoning backbones. These findings highlight evidence admission as a complementary dimension of agentic search and suggest more effective admission policies.

AI USE STATEMENT

In this work, we have not used generative AI tools for any task that requires disclosure, including generating synthetic data, developing theoretical models, formulating or proving claims, proposing hypotheses, designing the methodology or experiments, or interpreting results. Additionally, we used generative AI tools to create and edit software code (e.g., evaluation scripts, data processing, and plotting utilities) and to edit the manuscript to improve readability. We have reviewed all AI-assisted work: all AI-assisted code was manually reviewed, tested, and verified for correctness by the authors, and all reported results were produced by author-verified code; AI-edited text was checked by the authors to ensure it faithfully reflects our methods and findings, and no new content or claims were introduced by AI. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

Our work does not involve human subjects or personally identifiable information and uses publicly available benchmark datasets and retrieval corpora. As with other retrieval-based systems, external search may surface inaccurate, biased, or harmful content; appropriate safeguards should therefore be considered in real-world deployment.

REPRODUCIBILITY STATEMENT

We provide detailed descriptions of the SEEDS framework design and implementation in the main paper and appendix. We also provide the source code in an anonymous online repository to facilitate reproduction of our experiments.

REFERENCES

Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. Self-RAG: Learning to retrieve, generate, and critique through self-reflection. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=hSyW5go0v8.

Chia-Yuan Chang, Zhimeng Jiang, Vineeth Rakesh, Menghai Pan, Chin-Chia Michael Yeh, Guanchu Wang, Mingzhi Hu, Zhichao Xu, Yan Zheng, Mahashweta Das, and Na Zou. MAIN-RAG: Multi-agent filtering retrieval-augmented generation. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 2607–2622, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.131. URL https://aclanthology.org/2025.acl-long.131/.

DeepSeek-AI, Anyi Xu, B. Li, Bangcai Lin, Bing Xue, BingCheng Xian, et al. Deepseek-v4.1-flash: Pushing the limits of kv cache compression, 2026. URL https://arxiv.org/abs/2609.19969.

Wenqi Fan, Yujuan Ding, Liangbo Ning, Shijie Wang, Hengyun Li, Dawei Yin, Tat-Seng Chua, and Qing Li. A survey on rag meeting llms: Towards retrieval-augmented large language models, 2024. URL https://arxiv.org/abs/2405.06211.

Varun Gandhi, Jaewook Lee, Shantanu Todmal, Franck Dernoncourt, Ryan Rossi, Zichao Wang, and Andrew Lan. Grasp: Granularity-aware search policy for agentic rag, 2026. URL https://arxiv.org/abs/2607.10463.

Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, Meng Wang, and Haofen Wang. Retrieval-augmented generation for large language models: A survey, 2024. URL https://arxiv.org/abs/2312.10997.

Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. Constructing a multi-hop QA dataset for comprehensive evaluation of reasoning steps. In Donia Scott, Nuria Bel, and Chengqing Zong (eds.), *Proceedings of the 28th International Conference on Computational*

Linguistics, pp. 6609–6625, Barcelona, Spain (Online), December 2020. International Committee on Computational Linguistics. doi: 10.18653/v1/2020.coling-main.580. URL https://aclanthology.org/2020.coling-main.580/.

Zhengbao Jiang, Frank F. Xu, Luyu Gao, Zhiqing Sun, Qian Liu, Jane Dwivedi-Yu, Yiming Yang, Jamie Callan, and Graham Neubig. Active retrieval augmented generation. 2023.

Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan Arik, Dong Wang, Hamed Zamani, and Jiawei Han. Search-r1: Training llms to reason and leverage search engines with reinforcement learning, 2025. URL https://arxiv.org/abs/2503.09516.

Mandar Joshi, Eunsol Choi, Daniel S. Weld, and Luke Zettlemoyer. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension, 2017. URL https://arxiv.org/abs/1705.03551.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. Dense passage retrieval for open-domain question answering. In Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu (eds.), *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 6769–6781, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.550. URL https://aclanthology.org/2020.emnlp-main.550/.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, et al. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466, 2019. doi: 10.1162/tacl_a_00276. URL https://aclanthology.org/Q19-1026/.

Dahyun Lee, Yongrae Jo, Haeju Park, and Moontae Lee. Shifting from ranking to set selection for retrieval augmented generation. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (ACL)*, 2025.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive nlp tasks, 2021. URL https://arxiv.org/abs/2005.11401.

Xiaoxi Li, Guanting Dong, Jiajie Jin, Yuyao Zhang, Yujia Zhou, Yutao Zhu, Peitian Zhang, and Zhicheng Dou. Search-o1: Agentic search-enhanced large reasoning models, 2025a. URL https://arxiv.org/abs/2501.05366.

Xiaoxi Li, Jiajie Jin, Guanting Dong, Hongjin Qian, Yutao Zhu, Yongkang Wu, Ji-Rong Wen, and Zhicheng Dou. Webthinker: Empowering large reasoning models with deep research capability. *CoRR*, abs/2504.21776, 2025b. doi: 10.48550/ARXIV.2504.21776. URL https://doi.org/10.48550/arXiv.2504.21776.

Siyuan Meng, Junming Liu, Yirong Chen, Song Mao, Pinlong Cai, Guohang Yan, Botian Shi, and Ding Wang. From ranking to selection: A simple but efficient dynamic passage selector for retrieval augmented generation, 2025. URL https://arxiv.org/abs/2508.09497.

Long Phan, Alice Gatti, Nathaniel Li, Adam Khoja, Ryan Kim, et al. A benchmark of expert-level academic questions to assess ai capabilities. *Nature*, 649(8099):1139–1146, January 2026. ISSN 1476-4687. doi: 10.1038/s41586-025-09962-4. URL http://dx.doi.org/10.1038/s41586-025-09962-4.

Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah A. Smith, and Mike Lewis. Measuring and narrowing the compositionality gap in language models, 2023. URL https://arxiv.org/abs/2210.03350.

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, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang,

Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report, 2025. URL https://arxiv.org/abs/2412.15115.

Sahel Sharifymoghaddam and Jimmy Lin. Rerank before you reason: Analyzing reranking tradeoffs through effective token cost in deep search agents. In *Findings of the Association for Computational Linguistics: ACL 2026*, 2026.

Chaitanya Sharma. Retrieval-augmented generation: A comprehensive survey of architectures, enhancements, and robustness frontiers, 2025. URL https://arxiv.org/abs/2506.00054.

Artyom Sorokin, Nazar Buzun, Aleksandr Anokhin, Egor Vedernikov, Petr Anokhin, Mikhail Burtsev, and Evgeny Burnaev. Q-rag: Long context multi-step retrieval via value-based embedder training. In *International Conference on Learning Representations*, volume 2026, pp. 32481–32506, 2026.

Weihang Su, Yichen Tang, Qingyao Ai, Zhijing Wu, and Yiqun Liu. Dragin: Dynamic retrieval augmented generation based on the information needs of large language models, 2024. URL https://arxiv.org/abs/2403.10081.

Jiashuo Sun, Xianrui Zhong, Sizhe Zhou, and Jiawei Han. Dynamicrag: Leveraging outputs of large language model as feedback for dynamic reranking in retrieval-augmented generation, 2025. URL https://arxiv.org/abs/2505.07233.

Chihiro Taguchi, Seiji Maekawa, and Nikita Bhutani. Efficient context selection for long-context qa: No tuning, no iteration, just adaptive-k, 2025. URL https://arxiv.org/abs/2506.08479.

Qwen Team. Qwq-32b: Embracing the power of reinforcement learning, March 2025. URL https://qwenlm.github.io/blog/qwq-32b/.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. MuSiQue: Multihop questions via single-hop question composition. *Transactions of the Association for Computational Linguistics*, 10:539–554, 2022. doi: 10.1162/tacl_a_00475. URL https://aclanthology.org/2022.tacl-1.31/.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. Interleaving retrieval with chain-of-thought reasoning for knowledge-intensive multi-step questions. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki (eds.), *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 10014–10037, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.557. URL https://aclanthology.org/2023.acl-long.557.

Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. Text embeddings by weakly-supervised contrastive pre-training, 2024. URL https://arxiv.org/abs/2212.03533.

Jason Wei, Zhiqing Sun, Spencer Papay, Scott McKinney, Jeffrey Han, Isa Fulford, Hyung Won Chung, Alex Tachard Passos, William Fedus, and Amelia Glaese. Browsecomp: A simple yet challenging benchmark for browsing agents. *arXiv preprint arXiv:2504.12516*, 2025.

Tianxin Wei, Ting-Wei Li, Zhining Liu, Xuying Ning, Ze Yang, Jiaru Zou, Zhichen Zeng, Ruizhong Qiu, Xiao Lin, Dongqi Fu, et al. Agentic reasoning for large language models. *arXiv preprint arXiv:2601.12538*, 2026.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. Hotpotqa: A dataset for diverse, explainable multi-hop question answering, 2018. URL https://arxiv.org/abs/1809.09600.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models, 2023. URL https://arxiv.org/abs/2210.03629.

Gongrui Zhang, Jialiang Zhu, Ruiqi Yang, Kai Qiu, Miaosen Zhang, Zhirong Wu, Qi Dai, Bei Liu, Chong Luo, Zhengyuan Yang, Linjie Li, Lijuan Wang, Weizhu Chen, Yuan Zhang, Xin Li, Zhaoyi Liu, Xin Geng, and Baining Guo. Infoagent: Advancing autonomous information-seeking agents, 2025a. URL https://arxiv.org/abs/2509.25189.

Wenlin Zhang, Xiangyang Li, Kuicai Dong, Yichao Wang, Pengyue Jia, Xiaopeng Li, Yingyi Zhang, Derong Xu, Zhaocheng Du, Huifeng Guo, Ruiming Tang, and Xiangyu Zhao. Process vs. outcome reward: Which is better for agentic rag reinforcement learning, 2025b. URL https://arxiv.org/abs/2505.14069.

Chujie Zheng, Zhenru Zhang, Beichen Zhang, Runji Lin, Keming Lu, Bowen Yu, Dayiheng Liu, Jingren Zhou, and Junyang Lin. Processbench: Identifying process errors in mathematical reasoning. In *The 63rd Annual Meeting of the Association for Computational Linguistics*, 2025.

Peilin Zhou, Bruce Leon, Xiang Ying, Can Zhang, Yifan Shao, Qichen Ye, Dading Chong, Zhiling Jin, Chenxuan Xie, Meng Cao, et al. Browsecomp-zh: Benchmarking web browsing ability of large language models in chinese. *arXiv preprint arXiv:2504.19314*, 2025.

A Experimental Setup and Interaction Protocol

A.1 Datasets and Evaluation

The main evaluation covers six local-corpus QA benchmarks and three live-web benchmarks. NQ (Kwiatkowski et al., 2019) and TriviaQA (Joshi et al., 2017) evaluate factual question answering, while HotpotQA (Yang et al., 2018), 2WikiMultiHopQA (2Wiki) (Ho et al., 2020), MuSiQue (Trivedi et al., 2022), and Bamboogle (Press et al., 2023) require combining information across multiple reasoning steps. We use all 125 data points of Bamboogle dataset (Press et al., 2023) and randomly select 500 data points from other datasets. Averages over these six datasets are macro-averages unless otherwise stated.

BrowseComp (Wei et al., 2025), BrowseComp-ZH (Zhou et al., 2025), and Humanity's Last Exam (HLE) (Phan et al., 2026) evaluate more demanding search-intensive questions. The live-web evaluation contains 1266, 289, and 435 questions, respectively. The HLE evaluation is restricted to a randomly chosen subset containing 435 questions out of 2158 text-only questions. We report judged answer accuracy using Qwen2.5-72B-Instruct (Qwen et al., 2025) as the evaluator, with the subject-level HLE breakdown in Appendix D.

A.2 Baselines

The baseline families are defined in Section 4. Fixed-k RAG retrieves once, whereas ReAct repeats retrieval with $k \in \{3, 10\}$ or a model-selected prefix length. MAIN-RAG filters candidates once or within the multi-round loop. Search-o1 refines full documents through Reason-in-Documents. SEEDS-ReAct adds admission before direct reading; SEEDS-o1 adds it before refinement. All passages supplied to either reader count as full-document reads, even when only their refinements enter the main reasoning context.

A.3 Retrieval and Inference Settings

Local corpus. The main experiments use QwQ-32B (Team, 2025) and the Wikipedia-2018 corpus used by Search-R1, containing approximately 21 million passages of 100 words each. We encode passages with e5-base-v2 (Wang et al., 2024) and retrieve by exact FAISS inner-product search. Each query returns $K = 10$ candidates. Selected passages are exposed up to a limit of 1,500 characters per passage (Jin et al., 2025).

Live web. We use the Serper API for search and the Jina Reader API for page retrieval, as in the main experiments. Search results provide titles and snippets for the preview pool. For each selected result, we retain up to 3,000 characters of page content centered around the matched snippet. Page-fetch counts record cache misses, rather than repeated accesses to cached pages.

Additional backbone. The DeepSeek experiments use the `deepseek-flash` endpoint at `https://api.deepseek.com/v1`, with `reasoning_effort="high"` and thinking enabled through `extra_body={"thinking":{"type":"enabled"}}`. The trained Qwen2.5-7B setting has a separate interaction budget and is described in Appendix E.

A.4 Inference Procedure

Algorithm 1 implements the admission loop in Section 3.3. The state $\mathcal{H}$ contains the question, reasoning history, previous actions, and observations. Admission selects a subset of the current pool, including the empty set, without gold-answer information. The search budget B bounds the number of retrieval rounds.

A.5 Prompt Instructions and Interaction Templates

In the implementation, we define our preview special token as </snap>, and each candidate's identifier, title, and matched snippet are appended in `<snap>...</snap>`. Selection identifiers refer to this most recent pool. Only selected passages are subsequently returned as full evidence, and the model never supplies its own retrieved observations.

Admission instructions. The template below summarizes the protocol, rather than reproducing a particular run verbatim. SEEDS-ReAct stores selections as `selected_ids: [i, j, ...]` and returns passages in `<obs>...</obs>`; the trained environment instead uses `<information>...</information>`.

> Issue a query in `<search>...</search>` and stop for the previews. After each pool, emit one `<expand>...</expand>` action naming the passages whose full text could help answer the question. Compare candidates with the evidence already collected. Do not reject a relevant passage merely because its preview omits the answer. Skip irrelevant or redundant passages; selecting none is allowed. Stop for the selected evidence, then search again or give a concise answer in `<answer>...</answer>`.

Baseline interfaces. Fixed-k ReAct issues `<search>[query]</search>` and receives the top-k full passages in `<information>...</information>` without a preview or expansion decision. ReAct dyn-k instead emits `<search>topk=N, query='[query]'</search>`, with $1 \leq N \leq 10$; its chosen documents remain a retrieval-rank prefix. Thus, the dynamic-k baseline adapts reading depth without observing previews or selecting outside the retrieval prefix.

B PREVIEW CONSTRUCTION AND ABLATION

Query-aware previews. For local retrieval, a preview combines the passage title with a short, query-aware view of its text. The first 80 characters provide a document anchor; a 240-character window exposes a span with high lexical overlap with the current search query. Preview text is capped at 60 tokens. The construction uses the query and passage text, requires no additional model call, and does not use gold answers or the agent's reasoning history. For live-web retrieval, the search engine's title and snippet serve directly as the preview.

Representation and length. Table 7 compares titles, prefixes, a query-aware window, and a one-line chunk summary. The query-aware window performs best on TriviaQA, while different prefix lengths lead on the other datasets. No representation dominates, and longer previews do not consistently improve accuracy.

Table 7: Preview ablation: strict EM (%, QwQ-32B, $n = 500$ per dataset).

Preview	NQ	TriviaQA	HotpotQA	2Wiki
Title only	38.0	67.2	33.8	37.8
Prefix, 30 tokens	37.4	67.2	35.2	**39.2**
Prefix, 60 tokens	38.6	67.0	**37.6**	32.4
Prefix, 90 tokens	**41.0**	68.0	35.6	28.6
Query window (adopted)	39.6	**69.6**	35.6	31.2
Chunk summary	38.2	68.6	35.8	35.2

C EVIDENCE ADMISSION ANALYSIS AND EFFICIENCY

C.1 ANSWER-BEARING EVIDENCE PROXY

We use answer occurrence as a document-level proxy because chunk-level evidence annotations aligned with the retrieved Wikipedia corpus are unavailable. A document is *answer-bearing* if its normalized full text contains any normalized gold-answer alias. Normalization lowercases text, removes diacritics and punctuation, and collapses whitespace. Boolean and null aliases (`yes`, `no`, `true`, `false`, `none`) are excluded, and questions with no remaining alias are omitted. Aliases with at least eight characters use substring matching; shorter aliases require word boundaries. The label is computed from full text even though the admission policy observes only the preview.

This proxy measures answer occurrence, not evidential sufficiency. It can miss bridge passages that do not contain the final answer and can count unrelated mentions of an answer string. The answer-bearing hit rate in Figure 4(a) is therefore interpreted conditionally on such a document being available in the candidate pool. For the non-prefix analysis, a selected document lies beyond

the matched prefix when its retrieval rank exceeds the number of documents admitted in that round. This comparison measures departure from rank-based admission at the observed reading budget.

C.2 PER-DATASET EVIDENCE CONSUMPTION

Tables 8–10 decompose the aggregate costs in Table 4. Search calls count retrieval invocations; full-document reads count passages passed to the reasoning model or the Reason-in-Documents reader. Document-token counts use the model tokenizer and exclude prompts, previews, reasoning, and generated output. They quantify full-evidence consumption, rather than total inference tokens or latency. Each average is the macro-average across the six local benchmarks.

SEEDS-ReAct inspects about three documents per question across datasets, while fixed-k agents read more as search frequency increases. For SEEDS-o1, the search frequency is close to Search-o1, but document reads and document-token consumption are lower on every dataset. The per-dataset breakdown thus supports the main-text distinction between searching less and reading less per search.

Table 8: Average number of search calls per question on QwQ-32B. Avg. is the macro-average across the six benchmarks.

Method	NQ	TriviaQA	HotpotQA	2Wiki	MuSiQue	Bamboogle	Avg.
ReAct $k = 3$	1.70	1.32	2.75	4.47	3.02	2.06	2.55
ReAct $k = 10$	0.96	0.89	1.63	2.66	1.70	1.10	1.49
ReAct dyn-k	1.77	1.53	2.71	5.33	2.74	1.92	2.67
SEEDS-ReAct	1.16	1.23	2.19	3.26	2.41	1.78	2.00
Search-o1	2.04	1.84	3.37	4.73	3.60	2.55	3.02
SEEDS-o1	2.07	1.80	3.38	4.52	3.72	2.70	3.03

Table 9: Average number of full documents inspected per question. A document is counted when its full content is passed to a reader. Avg. is the macro-average across the six benchmarks.

Method	NQ	TriviaQA	HotpotQA	2Wiki	MuSiQue	Bamboogle	Avg.
RAG $k = 3$	3.0	3.0	3.0	3.0	3.0	3.0	3.0
RAG $k = 10$	10.0	10.0	10.0	10.0	10.0	10.0	10.0
ReAct $k = 3$	5.1	4.0	8.3	13.4	9.1	6.2	7.7
ReAct $k = 10$	9.6	8.9	16.3	26.6	17.0	11.0	14.9
ReAct dyn-k	5.7	4.7	7.9	14.3	8.0	5.3	7.7
SEEDS-ReAct	3.3	3.3	2.7	2.9	3.0	3.2	3.1
Search-o1	20.4	18.4	33.7	47.3	36.0	25.5	30.2
SEEDS-o1	9.8	8.1	13.2	16.6	15.8	12.0	12.6

Table 10: Average number of full-document tokens consumed per question, measured with the model tokenizer. The metric excludes reasoning, preview, query, and generated-output tokens. Avg. is the macro-average across six benchmarks.

Method	NQ	TriviaQA	HotpotQA	2Wiki	MuSiQue	Bamboogle	Avg.
RAG $k = 3$	452	454	475	502	467	460	468
RAG $k = 10$	1,512	1,518	1,584	1,671	1,563	1,526	1,562
ReAct $k = 3$	806	622	1,370	2,270	1,482	972	1,254
ReAct $k = 10$	1,518	1,405	2,704	4,438	2,767	1,744	2,429
ReAct dyn-k	902	747	1,336	2,448	1,308	861	1,267
SEEDS-ReAct	512	513	461	527	511	643	528
Search-o1	3,310	2,959	5,634	8,095	5,911	4,125	5,006
SEEDS-o1	1,568	1,280	2,185	2,931	2,580	1,909	2,075

D Detailed Live-Web Results

Table 11 breaks down the 435-question text-only HLE evaluation by subject. SEEDS-o1 answers 60 questions correctly, compared with 45 for Search-o1, with higher accuracy in seven subjects and a tie in Engineering. The small subject subsets should be interpreted descriptively.

Table 11: HLE text-only accuracy by subject (QwQ-32B). n is the number of questions; accuracy is reported in percent.

Subject	n	Search-o1	SEEDS-o1
Mathematics	187	12.83	**13.90**
Computer science	52	9.62	**13.46**
Humanities	45	13.33	**17.78**
Biology / medicine	42	11.90	**19.05**
Other	42	7.14	**11.90**
Physics	40	0.00	**5.00**
Chemistry	14	14.29	**28.57**
Engineering	13	0.00	0.00
Overall	435	10.34	**13.79**

E Training and Evaluation of SEEDS-R1

SEEDS-R1 uses Qwen2.5-7B-Base and is trained in two stages: supervised fine-tuning (SFT) teaches the search–preview–expand–answer protocol, and proximal policy optimization (PPO) then optimizes answer correctness and evidence selection. Table 12 summarizes the configuration.

E.1 Interaction Environment

The training environment uses the same Wikipedia corpus and E5 retriever as the main experiments, with $K = 5$ candidates per query. It returns titles and query-aware previews in `<snap>` blocks and exposes only selected passages in `<information>` blocks. Each search must be followed by one expand decision, selecting between zero and five passages. Episodes are limited to four retrieval rounds and ten policy actions; observations are capped at 1,200 tokens per step. The final response is enclosed in `<answer>...</answer>`.

E.2 Stage 1: Supervised Warm Start

Demonstration construction. We collect QwQ-32B trajectories on NQ and HotpotQA training questions using the same five-candidate environment. The teacher uses temperature 0.6, top-p 0.95, top-k 40, and up to 4,096 generated tokens per turn. Teacher `<think>` blocks are removed before storage, retaining actions, observations, and the final answer.

A trajectory is accepted if it satisfies the protocol and either produces an exact-match answer or exposes a gold answer in the observations shown to the teacher. The latter condition uses whole-word matching after SQuAD-style normalization; the final answer is then replaced by the gold string, while the teacher's original answer is retained for auditing. This is an answer-occurrence filter, not a guarantee of evidential sufficiency. Of 3,072 attempts, 1,000 trajectories are accepted: 495 by exact match and 505 by answer substitution. The split contains 900 training trajectories (420 NQ, 480 HotpotQA) and 100 development trajectories (47 NQ, 53 HotpotQA). Student-side trajectories average 910 tokens, with a maximum of 2,771.

Objective and checkpoint selection. We fine-tune all student parameters with causal language-modeling loss on policy-generated action and answer tokens only:

$$\mathcal{L}_{\mathrm{SFT}} = -\mathbb{E}_{\tau^* \sim \mathcal{D}_{\mathrm{SFT}}} \left[\sum_{j \in \mathcal{M}} \log p_\theta(y_j^* \mid y_{<j}^*, x) \right], \tag{7}$$

where x is the question and instruction, and $\mathcal{M}$ indexes policy tokens. Instructions and environment observations are masked out of the loss. Development EM increases from 0.14 before SFT to 0.49 after one epoch and 0.53 after five epochs (0.55 on NQ, 0.51 on HotpotQA), with protocol compliance at least 0.96. Although development loss is lowest after epoch 1, we select epoch 5 by development EM and use it to initialize both the PPO actor and critic.

E.3 Stage 2: Reinforcement Learning

Evidence labels. The outcome reward is final-answer exact match, $R_{\mathrm{EM}}(\tau) \in \{0, 1\}$. An auxiliary selection bonus evaluates which documents the agent opens. For NQ, a document is labeled positive if it contains a normalized gold answer; unexpanded candidates are judged only from their visible previews. For HotpotQA, positive documents have titles matching annotated supporting-fact titles, so bridge documents receive credit even when they omit the final answer. Other HotpotQA documents containing the answer are neutral. Positive documents are deduplicated by normalized title across the trajectory. These labels are available during training only.

Selection reward. Let S_r be the documents expanded at round r, $\mathcal{G}$ and $\mathcal{N}$ the positive and neutral documents, and $\mathcal{O}_{<r}$ the positive documents opened before that round. Let $\mathcal{V}$ be positive documents visible in previews and $\mathcal{O}$ those opened anywhere in the trajectory. The selection reward is

$$\Delta_r = \alpha\big|(S_r \cap \mathcal{G}) \setminus \mathcal{O}_{<r}\big| - \beta\big|S_r \setminus (\mathcal{G} \cup \mathcal{N})\big|, \tag{8}$$

$$\Delta_{\mathrm{miss}} = -\gamma\big|\mathcal{V} \setminus \mathcal{O}\big|\mathbb{1}[R_{\mathrm{EM}}(\tau) = 0], \tag{9}$$

$$R_{\mathrm{sel}}(\tau) = \mathrm{clip}\left(\sum_r \Delta_r + \Delta_{\mathrm{miss}}, -c, c\right), \tag{10}$$

with $(\alpha, \beta, \gamma, c) = (0.15, 0.05, 0.15, 0.6)$. The reward credits the first opening of each positive document and penalizes opening negative documents. Missed visible evidence is penalized only when the final answer is incorrect, allowing the policy to answer from a preview without an unnecessary expansion. Clipping bounds the auxiliary contribution.

Each selection term is assigned to the last token of its expand action; the missed-evidence term is assigned to the last expand action, or the final generated token if no expansion occurs. The combined selection contribution is clipped as above. Exact match is assigned to the final generated token. A per-token KL penalty with coefficient 0.01 regularizes the actor toward the frozen SFT reference policy. The PPO objective uses these token-level rewards with generalized advantage estimation ($\gamma_{\mathrm{GAE}} = \lambda = 1$).

Observation integrity and optimization. Reward computation parses only environment-inserted spans identified by the observation mask, so policy-generated text cannot masquerade as retrieved evidence. If parsed information blocks and accepted expansion actions disagree, the selection bonus is discarded while the EM reward is retained. Environment tokens are masked out of both policy and value losses. Evaluation uses EM without the selection bonus.

PPO uses the NQ and HotpotQA training sets (79,168 and 90,447 questions, respectively). The critic is updated alone for the first ten steps. Training runs for 500 updates on two H200 GPUs, taking approximately 54 hours; the reported model is the step-450 checkpoint. Optimization settings are listed in Table 12.

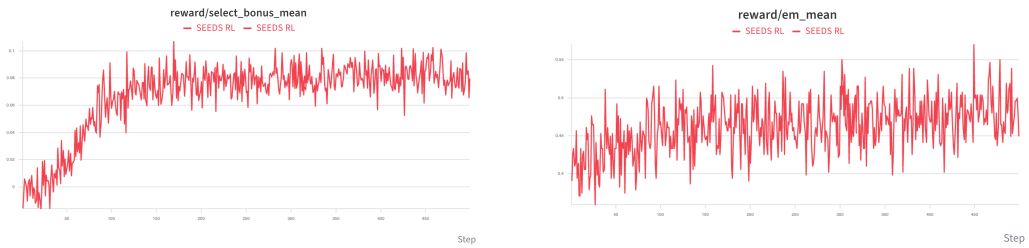
E.4 Training Dynamics and Held-Out Results

Figure 7 shows the selection bonus and answer exact match during PPO training.

During critic warm-up, the SFT policy opens 32% of visible positive documents on NQ and 42% on HotpotQA, with selection precision 0.32 and 0.35. Over updates 91–110, these open rates rise to 74% and 81%, while document reads increase from 1.30 to 2.34 per question. Over updates 426–450, open rates reach 74% and 88%, precision reaches 0.43 and 0.46, and reads decrease to 2.12 documents per question. These window averages indicate an initial increase in evidence acquisition followed by more selective reading.

Table 12: Optimization settings for SEEDS-R1 (Qwen2.5-7B). The interaction environment is specified in Appendix E.1.

	Hyperparameter	Value
Stage 1: SFT	Teacher / student	QwQ-32B / Qwen2.5-7B-Base
	Training / dev trajectories	900 / 100
	Loss	masked NLL on policy tokens only
	Optimizer	AdamW, $\beta = (0.9, 0.95)$, weight decay 0.01
	Learning rate / warmup	1×10^{-5} / 10%
	Batch size / max. length	16 / 8,192 tokens
	Epochs	5 (selected by dev EM)
Stage 2: RL	Algorithm	PPO + GAE ($\gamma = \lambda = 1$)
	Training questions	NQ + HotpotQA (169,615)
	Batch / mini-batch	256 / 128
	Actor LR / warmup	5×10^{-7} / 2%, constant
	Critic LR / warmup	1×10^{-5} / 1.5%, constant
	Critic-only warmup	10 updates
	PPO clip / value clip / entropy	0.2 / 0.5 / 0.001
	KL penalty (β_{KL}, in reward)	0.01
	Selection bonus $(\alpha, \beta, \gamma, c)$	(0.15, 0.05, 0.15, 0.6)
	Rollout temperature / tokens per turn	1.0 / 500
	Updates	500 (checkpoint at 450 used)



(a) Mean selection bonus.

(b) Mean answer exact match.

Figure 7: Training curves for SEEDS-R1 (Qwen2.5-7B, $K = 5$). The horizontal axis is the optimizer update; the first ten updates train only the critic. Curves show training-batch means, not held-out evaluation scores.

Table 13 measures the step-450 models with five-candidate retrieval. For SEEDS-R1, retrieved tokens comprise 492 preview tokens and 361 expanded-document tokens. Counting reasoning and retrieved text but excluding the fixed prompt, tokens decrease by 30% and full-document reads by 71%. Unlike Appendix C.2, this accounting includes preview overhead.

Table 13: Per-question cost at training step 450 with five-candidate retrieval. Retrieved tokens include previews and opened text; total excludes the fixed prompt. All counts are averages.

Method	Reasoning	Retrieved	Total	Documents	Searches
Search-R1	162	1,226	1,388	7.48	1.50
SEEDS-R1	115	853	968	2.18	1.33
Relative change	−29%	−30%	−30%	−71%	−11%

F PROMPT TEMPLATES

F.1 SEEDS-O1

SEEDS-o1 adds snapshot-then-expand selection to the Search-o1 agent prompt. After each search query, the environment returns a `<snap>` pool of `{N}` candidates, each shown as an id, a title,

and a short snippet. The agent must then emit exactly one <expand> block that selects up to {K} of them. Only the selected passages go to the Search-o1 Reason-in-Documents module, whose output is returned in <|begin_search_result|> ... <|end_search_result|>. {T} caps the number of search calls. The expand instruction is the *max-information* variant: the number of expanded passages is not fixed, but the agent is told to be inclusive when selecting.

```
You are a reasoning assistant with the ability to perform web searches to help you answer
  the user's question accurately. You have special tools:

- To perform a search: write <|begin_search_query|> your query here <|end_search_query|>.
Then, the system will return a snapshot pool of {N} candidate passages in the format <snap>
  ...snapshot... </snap>. Each entry has an id, a title, and a short snippet (not the full
  content).

- After receiving the snapshot, decide which passages are worth reading in full: write
  <expand> a rationale for your selection, then a final line selected_ids: [i, j, ...]
  </expand>.
Use only ids from the most recent <snap>. Select as many passages as carry any useful
  information for the question -- be inclusive and prefer expanding a passage over skipping
  it, so you gather as much useful information as you can (between 0 and {K}). Only omit a
  passage if it is clearly irrelevant or fully redundant with ones you already chose. You
  still MUST output <expand>...</expand> after every <snap>. Do not answer directly after
  <snap>, and do not issue another search before <expand>.
Then, the system will analyze the selected passages and provide you with helpful information
  in the format <|begin_search_result|> ...search results... <|end_search_result|>.

You can repeat the search process multiple times if necessary. The maximum number of search
  attempts is limited to {T}.

Once you have all the information you need, continue your reasoning.

Example:
Question: "Alice David is the voice of Lara Croft in a video game developed by which
  company?"
Assistant thinking steps:
- I need to find out who voices Lara Croft in the video game.
- Then, I need to determine which company developed that video game.

Assistant:
<|begin_search_query|>Alice David Lara Croft voice<|end_search_query|>

(System returns a snapshot pool of candidate passages inside <snap>...</snap>)

Assistant:
<expand>Snippet [2] looks most relevant to who voices Lara Croft. selected_ids: [2]</expand>

(System returns processed information from the selected passages)

Assistant thinks: The search results indicate that Alice David is the voice of Lara Croft in
  a specific video game. Now, I need to find out which company developed that game.

Assistant:
<|begin_search_query|>video game developed by Alice David Lara Croft<|end_search_query|>

(System returns a snapshot pool of candidate passages inside <snap>...</snap>)

Assistant:
<expand>Snippet [1] names the developer of the game. selected_ids: [1]</expand>

(System returns processed information from the selected passages)

Assistant continues reasoning with the new information...

Remember:
- Use <|begin_search_query|> to request a web search and end with <|end_search_query|>.
- After each search, the system replies with a <snap> snapshot pool; you MUST immediately
  use <expand>...</expand> ending with a `selected_ids: [...]` line before answering or
  searching again.
- When done searching, continue your reasoning.
- When you have enough information, write the final answer inside <answer>...</answer>. Keep
  it SHORT -- a few words or a single phrase (a name, date, or entity), NOT a full sentence.

Question: {question}
```

If the agent emits anything other than an <expand> block right after a <snap>, the environment appends the following correction and lets the agent continue:

```
Invalid action after <snap>: you must first output exactly one <expand>...</expand> block
```

```
  for the latest snapshot before searching again or finalizing. You may write selected_ids:
  [] if no passage needs expansion.
```

F.2 SEEDS-R1

SEEDS-R1 is trained and evaluated with the Search-R1 instruction format, extended with the <snap>/<expand> actions. The first paragraph below is the prompt stored in the training data. The block of required-expand rules is inserted before `Question:` when the data are loaded, instantiated with the training limits (at most 4 search rounds and at most 5 documents per expand). The same prompt is used for supervised warm-up, RL, and evaluation, and is wrapped in the Qwen2.5 chat template (system message: `You are a helpful assistant.`).

```
Answer the given question. You must conduct reasoning inside <think> and </think> first
  every time you get new information. After reasoning, if you find you lack some knowledge,
  you can call a search engine by <search> query </search>. The system will return the top-k
  document snapshots (each truncated to its first 60 tokens) between <snap> and </snap>,
  where each snapshot is prefixed with an id. After reading the snapshots, you must reason
  and select which documents to read in full by writing <expand> reasoning: your reasoning |
  expand_ids: [id1, id2, ...] </expand> (or expand_ids: [] if no snapshot is worth
  expanding). The full content of the selected documents will be returned between
  <information> and </information>. You can search and expand as many times as you want. If
  you find no further external knowledge needed, you can directly provide the answer inside
  <answer> and </answer> without detailed illustrations. For example, <answer> xxx
  </answer>.

SEEDS REQUIRED-EXPAND RULES FOR THIS RUN (these override any earlier general wording):
- You may perform at most 4 search rounds.
- After every <snap> result, your next valid action MUST be exactly one <expand> reasoning:
  ... | expand_ids: [...] </expand>.
- You MUST NOT issue another <search> or an <answer> until that expand action has completed.
  The environment will not insert or force the action. Completing it earns the next
  configured search-expand pair bonus; skipping it earns no expand bonus for that search
  round.
- Use expand_ids: [] when no snapshot is worth opening; this empty choice still completes
  the required expand step.
- Select at most 5 document ids in one expand.
- A direct answer is allowed before any search. After a search, an answer is allowed only
  after its required expand.

Question: {question}
```