SEARCHPLM: TOWARDS SEARCH-PROFICIENT LANGUAGE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Despite advances in reasoning and tool use, Large Language Models (LLMs) still struggle with complex information-seeking tasks. Through systematic analysis, we diagnose a key failure mode in current LLM-based search agents: they tend to issue overly specific queries that closely overlap with the original question and terminate search prematurely after several search failures. We term this phenomenon the *Golden Document Fallacy*, an implicit behavior that a single document can fully answer the original question, which reduces multi-turn search to a shortcut-like search process. Drawing on search-skills research and questionnaire responses about human search behaviors, we distill search strategy to a **Exploration-Coverage-Synthesis (E-C-S)** paradigm, which structures effective search into strategic exploration, iterative evidence coverage, and multi-source synthesis. Based on this paradigm, we develop **SearchPLM**, a **Search**-**P**roficient **L**anguage **M**odel trained with strategy-guided supervised distillation and curriculum-based reinforcement alignment. Experiments show that **SearchPLM** outperforms strong open-source baselines with more parameters, highlighting the importance of search strategy optimization for more generalizable information-seeking tasks. [1]

1 INTRODUCTION

Information seeking is a cognitive process fundamental to human intelligence. Powered by recent advances in LLMs, particularly in their reasoning and tool-use capabilities (Yao et al., 2022; Schick et al., 2024), LLMs can now use search tools—issuing queries to search engines and navigating web pages to answer complex questions (OpenAI, 2024). While some search-enhanced LLMs have demonstrated superhuman performance on several benchmarks, a critical question remains: *Do these models truly master the **strategy** of search, or merely the **mechanics** of search tool usage?*

Existing benchmarks are primarily constructed by reverse-engineering multi-hop questions from single factual answers, artificially increasing reasoning difficulty via mechanisms such as entity ambiguity and distractors (Ho et al., 2020; Yang et al., 2018; Trivedi et al., 2022a; Press et al., 2022; Wei et al., 2025). Real-world information seeking, however, exhibits a significantly different structure. User questions often exhibit deceptively simple surface forms, yet their resolution requires aggregating evidence distributed across documents, diverse perspectives, or temporal contexts.

Consider the question *"Compare the market share and revenue growth of top 5 EV manufacturers in North America and Europe over the last 3 years"*. This comparative question cannot be satisfied by retrieving a single comprehensive industry report—such documents rarely exist. However, LLMs tend to generate queries like *"'market share' 'revenue growth' 'top 5 EV manufacturers in NA EU' 'over the last 3 years'"*, essentially replicating the original question with minor modifications. This query cannot retrieve any useful information because term such *'over the last 3 years'* are unlikely to appear in relevant document. This phenomenon reveals a critical cognitive gap: *What implicit assumptions drive these models to search in this way?*

To answer this question, we first analyze the behavior of various LLMs on challenging multi-turn search tasks via a series of statistical and artificial experiments. We find that, when facing open-ended and atypical questions, many models tend to issue overly specific queries that closely mirror the

[1] Code is available at `https://anonymous.4open.science/r/SearchPLM-Anonymous-440A`.

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.

original question, often retrieving no useful information and ultimately producing biased answers. We term this phenomenon the *Golden Document Fallacy*: the implicit assumption in LLMs that a single document can fully answer the question, reducing multi-turn search to a **shortcut-like process**. These results suggest that, although current LLMs have learned to use search tools, they still lack effective and generalizable search capabilities.

Motivated by this phenomenon, this work focuses on teaching models more effective search strategies. We first collect, identify, and induce search strategies from human experts, and distill it into a general **Exploration-Coverage-Synthesis** search paradigm, which includes three core stages: (1) Exploration stage, which identifies key search terms and probes the distribution of information sources to establish the initial context; (2) Coverage stage, which systematically expands the evidence base by refining and iteratively optimizing search queries; and (3) Synthesis stage, which aggregates and cross-verifies the collected evidence to generate a coherent, reliable, and well-supported answer.

Building upon this search paradigm, we introduce `SearchPLM`, an end-to-end **Search**-**P**roficient **L**anguage **M**odel tailored for multi-turn search scenarios via search strategy-enhanced supervised finetuning & curriculum reinforcement learning pipelines to progressively calibrate `SearchPLM`'s search strategy. Extensive evaluations of various benchmarks show that `SearchPLM` adopts more effective and efficient search strategies, demonstrating strong generalization and an effective human-like search strategy.

The contributions of this paper are summarized as follows:

- We diagnose the *Golden Document Fallacy*, a failure mode where LLM search agents tend to issue over-specific queries and terminate prematurely.
- We propose the **Exploration-Coverage-Synthesis** paradigm and train `SearchPLM` via strategy-guided distillation and curriculum reinforcement alignment to progressively improve its search strategy.
- Experiments on seven benchmarks show that `SearchPLM` achieves strong performance, highlighting the importance of search strategy optimization for more generalizable information-seeking tasks.

2 PRELIMINARY

2.1 PROBLEM FORMULATION

Given a question q, a LLM interacts with a search engine over at most T steps to produce a trajectory

$$\tau = (s_1, a_1, o_1, \ldots, s_T, a_T, o_T, y),$$

where the state at step t is defined as $s_t = (q, H_{<t})$, with $H_{<t} = \{(a_i, o_i)\}_{i=1}^{t-1}$ denoting the interaction history. At each step, the LLM selects an action a_t from action space $\mathcal{A} = \{\texttt{search}, \texttt{answer}\}$. If $a_t = \texttt{search}$, the agent issues a query q_t to the search engine $\mathcal{S}$ and receives retrieved documents $o_t = \mathcal{S}(q_t)$; if $a_t = \texttt{answer}$, it outputs the final response y.

2.2 PRE-EXPERIMENT 1: STATISTICAL ANALYSIS OF LLMS' SEARCH BEHAVIOR

We first conduct statistical experiments to evaluate the search behaviors of LLMs as follows.

Metrics. Let $N_s^{(i)}$ denote the number of `search` actions taken for the i-th question, the empirical cumulative distribution $\Pr(N_s \leq k)$ is defined as

$$\Pr(N_s \leq k) = \frac{1}{N} \sum_{i=1}^{N} \mathbb{I}\Big[N_s^{(i)} \leq k \Big]. \tag{1}$$

For all trajectories, we measure query specificity using the number of whitespace-separated keywords in each query. The average query length at search step t is defined as

$$\mathbb{E}[|q_t|] = \frac{1}{N} \sum_{i=1}^{N} \big|\mathrm{Split}(q_t^{(i)})\big|, \tag{2}$$

Table 1: Analysis of search behaviors under a unified search budget. **Acc.** denotes accuracy (%); – indicates unreported values.

Model	$\Pr(N_s \leq 3)\downarrow$	$\Pr(N_s \leq 5)\downarrow$	$\mathbb{E}[\lvert q_1 \rvert]$	$\mathbb{E}[\lvert q \rvert]\downarrow$	**Acc.**	**Steps**
Qwen2.5-72B	64.9%	76.1%	$7.4_{\pm 6.1}$	$9.2_{\pm 6.5}$	42.0	4.9
QwQ-32B	83.4%	94.5%	$6.4_{\pm 4.0}$	$6.1_{\pm 3.8}$	33.9	2.2
GLM-4-32B	58.3%	78.7%	$7.4_{\pm 5.1}$	$7.9_{\pm 4.7}$	38.0	3.8
Qwen3.5-35B-A3B	47.4%	60.2%	$6.6_{\pm 3.1}$	$7.8_{\pm 3.4}$	45.9	7.7
Qwen3.6-27B	43.5%	53.5%	$5.7_{\pm 3.0}$	$5.7_{\pm 3.0}$	48.9	8.1
Search-R1-32B	87.6%	91.7%	$6.7_{\pm 7.0}$	$10.3_{\pm 6.8}$	25.4	3.2
ASearcher-v2-32B	78.3%	90.8%	$6.0_{\pm 3.5}$	$6.2_{\pm 3.4}$	39.3	2.6

where $\mathrm{Split}(\cdot)$ splits a query string by whitespace, and the overall average query length across all search processes is

$$\mathbb{E}[|q|] = \frac{1}{\sum_{i=1}^{N} N_s^{(i)}} \sum_{i=1}^{N} \sum_{t=1}^{N_s^{(i)}} \left|\mathrm{Split}(q_t^{(i)})\right|. \tag{3}$$

Implementation Details. We evaluate instruction-, reasoning-, agentic-, and search-enhanced LLMs, including `Qwen2.5-72B-Instruct`, `QwQ-32B`, `GLM-4-32B`, `Qwen3.5-35B-A3B`, and `Qwen3.6-27B`, `Search-R1` and `ASearcher-v2-32B`. All models use the same search environment and maximum search budget across challenging QA benchmarks: Musique (Trivedi et al., 2022a), GAIA (Mialon et al., 2023), BrowseComp (Wei et al., 2025), and WebWalker (Wu et al., 2025).

Conclusions. As reported in Table 1, our statistical analysis yields the following key observations:

(1) **Early stopping under a shared search budget**: Despite a search budget of $N_{\max} = 20$, seven of the eight models stop within five searches on more than half of the questions, while all achieve accuracy below 50%.

(2) **Overly specific and detailed queries**: With premature termination, queries generated by different LLMs consistently remain long, with first-turn and overall average lengths of $\mathbb{E}[|q_1|] \approx 6.4$–$7.4$ and $\mathbb{E}[|q|] \approx 6.1$–$10.3$, and the maximum query length **reaching 17 words**, indicating that LLMs might generate overly long queries.

These findings suggest that current LLMs might exhibit systematic deviations from effective multi-turn search behavior.

2.3 Pre-Experiment 2: Human Analysis from Over-Specification to Failure Attribution

Keyword Overlap Analysis. Noting that long queries are not inherently ineffective and that query content also matters, we further investigate *why* such queries arise and *how* they affect search effectiveness through two analyses: (1) keyword overlap analysis to quantify query–question similarity, and (2) failure attribution analysis to identify the root causes of search failures.

We employ an LLM-based keyword extraction method to identify salient terms in both user questions and generated queries, then compute the overlap ratio as $\text{Overlap} = |\text{Keywords}_{\text{query}} \cap \text{Keywords}_{\text{question}}|/|\text{Keywords}_{\text{query}}|$.

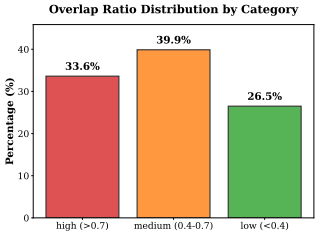


Figure 1: Distribution of keyword overlap.

As shown in Figure 1, a substantial fraction of queries have high lexical overlap with the original question, with only 26.5% exhibiting a low overlap ratio below 0.4. This suggests that these lengthy queries are often produced by **directly copying the original question**, rather than by strategically extracting the key terms needed for effective search.

Failure Attribution Analysis. Considering that queries extracting keywords from the original question may not always be of low quality, to further examine the impact of such queries on search outcomes, we manually analyze failed trajectories on WEBWALKER, a recent web question answering benchmark in which most questions do not exhibit explicit multi-hop structure and are more representative of users' everyday information needs.

Table 2: Failure attribution analysis.

Metric	Over-specific Failures	Other Cases
Sample count	**199**	46
Percentage	**81.2%**	18.8%

For each failure, we analyze whether the LLM has terminated because of overly detailed queries, resulting in no useful information, and thus considering the question unanswerable. As shown in Table 2, over 80% of failures stem directly from over-specification: these **overly lengthy queries are indeed unable to gather useful information from retrieved webpages**, leading the model to either conclude the question is unanswerable or provide incorrect answers based on insufficient evidence.

2.4 GOLDEN DOCUMENT FALLACY IN LLMS

Based on these findings, we diagnose a common failure mode in current LLM-based search agents, which we term the `Golden Document Fallacy`. Even when trained for search tasks, LLMs often generate overly specific queries that closely replicate the original question (as discussed in Section 2.2 and 2.3). These queries frequently fail to retrieve sufficiently informative evidence, restricting the model to a limited evidence set $\tilde{\mathcal{D}} \subseteq \mathcal{D}$.

In the analyzed failure cases, models often stop with insufficient evidence, either treating the question as unanswerable or producing an inadequately supported answer. We refer to this behavioral pattern as the *Golden Document Fallacy*: the model behaves as if a sufficiently specific query should retrieve a near-complete answer-bearing document, with limited adaptation when the retrieved evidence remains incomplete.

While this shortcut-like searching for an answer-bearing document can be effective for simple factoid questions, this strategy can be unreliable when answering a question requires combining evidence from multiple webpages. Such cases motivate search strategies that adapt queries to retrieved information, address remaining evidence gaps, and synthesize findings across sources.

3 METHODOLOGY

3.1 EXPLORATION-COVERAGE-SYNTHESIS PARADIGM FOR EFFECTIVE SEARCH

To develop search-strategy priors for LLMs, we draw inspiration from research on human search skills and their development through explicit instruction (Blummer, 2011; Hirt et al., 2020), together with questionnaire responses about human search habits.

Search Strategy Collection. We use a structured questionnaire to collect participants' self-reported habits in complex information seeking, including initial query formulation, query refinement, responses to unhelpful results, source evaluation, and stopping decisions. These responses complement the literature by providing concrete accounts of search practices (Appendix N).

Strategy Identification & Summarization. Combining these literature-informed insights with the reported habits, we organize search strategies into an actionable **Exploration-Coverage-Synthesis (E-C-S)** framework comprising three stages:

(1) **Exploration**: Select 1–2 high-priority terms to probe information availability and establish contextual understanding before introducing additional question constraints.

(2) **Coverage**: Expand the evidence base through complementary strategies: (a) iterative query refinement, (b) facet-based decomposition for complex questions, (c) adaptive failure recovery when retrieval is unhelpful, and (d) advanced search operators (e.g., `site:`) to target relevant sources.

(3) **Synthesis**: Cross-verify findings across sources, resolve conflicts by prioritizing authoritative evidence, and aggregate the collected information into coherent, well-supported answers.

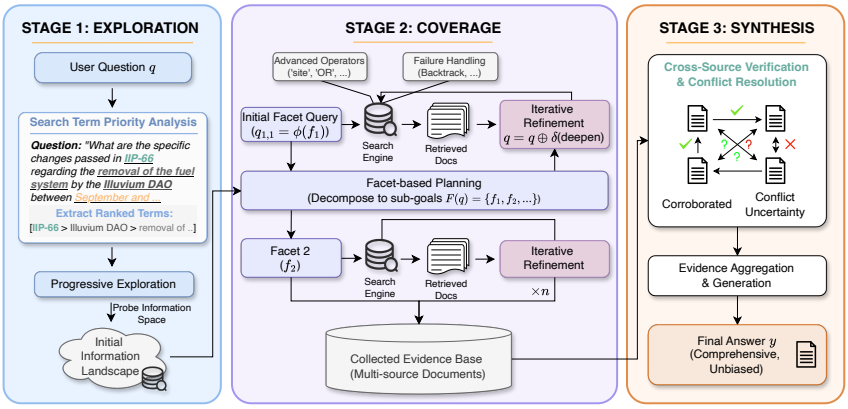


Figure 2: Overview of the Exploration-Coverage-Synthesis (E-C-S) search paradigm.

As Illustrated in Fig. 2, this paradigm serves as the foundation for our subsequent data construction and training procedures.

3.2 Training Data Construction

Data Source. To optimize the search strategy of LLMs with our proposed search framework, we first construct the training set $\mathcal{D} = \{(q_i, a_i)\}_{i=1}^{N}$ from approximately 8.5k QA pairs collected from the training split of ASearcher (Gao et al., 2025) and ensure no leakage of test data by cross-validation to filter similar questions. These datasets cover diverse domains and question types, providing broad supervision for generalizable search behavior.

High-Priority Entity Extraction. To guide initial exploration, we identify which entities to search first before introducing the full set of question constraints. For each question q of training set, we extract high-priority entities $\mathcal{E} = \{e_1, \ldots, e_m\}$ and assign priority scores $p(e_j)$. We consider spaCy-based extraction with heuristic ranking and LLM-based extraction and ranking, adopting the latter for its ability to identify context-dependent entities and assess their relevance to the question. We append the ranked entities to form $\tilde{q} = [q \parallel \mathrm{Rank}(\mathcal{E})]$.

Strategy-Enhanced Teacher Sampling. To turn E-C-S principles into supervision for sequential search decisions, we use `DeepSeek-V3` as the teacher model M_{teacher}. Given each augmented question $\tilde{q}_i$, E-C-S instructions $\mathcal{I}$ guide the teacher to explore relevant entities, address evidence gaps, and synthesize retrieved information. We sample candidate trajectories as $\tau_i \sim p_{M_{\text{teacher}}}(\cdot \mid \tilde{q}_i, \mathcal{I})$ and use the selected trajectories for supervised fine-tuning, initializing the base model with concrete demonstrations of E-C-S search behavior.

3.3 Training Procedure

3.3.1 Stage 1: Strategy-Guided Supervised Distillation

SFT Data Construction. To construct the SFT set $\mathcal{D}_{\text{SFT}}$, we retain only high-confidence questions for which the teacher model consistently provides correct answers. Concretely, for each question q_i, we sample $K = 4$ trajectories $\{\tau_i^{(k)}\}_{k=1}^{K}$ and evaluate their final answers. We include q_i in $\mathcal{D}_{\text{SFT}}$ only if all K sampled trajectories are correct, thereby filtering out ambiguous or unstable examples.

Then, the objective is to maximize the likelihood of valid model-generated tokens:

$$\max_{\theta} \; \mathcal{J}_{\text{SFT}}(\theta) = \mathbb{E}_{(\tilde{q},\tau)\sim\mathcal{D}_{\text{SFT}}} \left[\sum_{t \in \mathcal{T}_{\text{model}}} \log p_\theta(\tau_t \mid \tilde{q}, \tau_{<t}) \right], \tag{4}$$

where $\mathcal{T}_{\text{model}}$ denotes the set of token positions generated by the model, excluding tokens from search engine responses, which are masked during optimization.

3.3.2 STAGE 2: CURRICULUM-BASED REINFORCEMENT ALIGNMENT

RL Dataset Construction. We construct a curriculum learning set based on question difficulty. For each question $q_i \notin \mathcal{D}_{\text{SFT}}$, we sample $K = 4$ trajectories from M_{teacher} and compute its correctness rate $f(q_i) = \frac{1}{K}\sum_{k=1}^{K} r(\tau_i^{(k)})$, where $r(\tau) \in \{0, 1\}$ denotes correctness. We filter questions with $f(q_i) = 0$, as they are too difficult for training. The remaining questions form the $\mathcal{D}_{\text{RL}} = \{q_i \mid 0 < f(q_i) \leq 1\}$ and are sorted by descending $f(q_i)$ to construct an easy-to-hard curriculum set.

Long-Document Refinement for online RL. Unlike previous work that provides snippets of web search results, we introduce a two-stage refinement pipeline that compresses each retrieved webpage into a query-relevant summary to efficiently incorporate full web documents into online RL training. Given a retrieved document d, we first partition it into a set of text chunks $\{c_j\}_{j=1}^{m}$. For the current search query q, we compute a BM25 relevance score for each chunk:

$$s_j = \text{BM25}(q, c_j) = \sum_{w \in q} \text{IDF}(w) \, \frac{f(w, c_j)\,(k_1 + 1)}{f(w, c_j) + k_1 \left(1 - b + b \cdot \frac{|c_j|}{\text{avgdl}}\right)}, \tag{5}$$

where $\text{IDF}(w) = \log \frac{N - n(w) + 0.5}{n(w) + 0.5}$ denotes the inverse document frequency, $f(w, c_j)$ denotes the term frequency of token w in chunk c_j, $|c_j|$ is the chunk length, and avgdl is the average chunk length. We then select the top-k chunks,

$$\mathcal{C}_k(d, q) = \text{TopK}_{c_j \in d} \, \text{BM25}(q, c_j), \tag{6}$$

The selected chunks are then passed to a lightweight summarizer conditioned on both the current query q and the intermediate reasoning context $h_{<t}$, yielding a compact refined document

$$\tilde{d} = f_{\text{sum}}(q,\, h_{<t},\, \mathcal{C}_k(d, q))\,. \tag{7}$$

This refinement offers two benefits: (1) it **reduces the cost of processing long webpages** by replacing raw documents with compact, query-relevant summaries and, (2) since $\tilde{d}$ is derived from the query-matched chunks, **BM25 relevance score encourages the policy model to generate document-aware queries** rather than simply extracting keywords from the original question.

Finally, we optimize the policy π_θ using GRPO (Shao et al., 2024) with the following objective:

$$\mathcal{J}_{\text{GRPO}}(\theta) = \mathbb{E}_{q \sim \mathcal{D}_{\text{RL}},\, \{o_i\}_{i=1}^{G} \sim \pi_{\theta_{\text{old}}}(\cdot|q)} \left[\frac{1}{G} \sum_{i=1}^{G} \min \left(\frac{\pi_\theta(o_i \mid q)}{\pi_{\theta_{\text{old}}}(o_i \mid q)} A_i, \right. \right.$$
$$\left. \left. \text{clip}\left(\frac{\pi_\theta(o_i \mid q)}{\pi_{\theta_{\text{old}}}(o_i \mid q)}, 1 - \epsilon,\, 1 + \epsilon \right) A_i \right) - \beta D_{\text{KL}}(\pi_\theta \,\|\, \pi_{\text{ref}}) \right]. \tag{8}$$

where G is the group size for sampling, ϵ controls the clipping range, β balances the KL penalty against the reference policy π_{ref}, and $A_i = \frac{r(o_i,q) - \text{mean}(\{r(o_j,q)\}_{j=1}^{G})}{\text{std}(\{r(o_j,q)\}_{j=1}^{G}) + \delta}$ is the normalized advantage computed from the reward function (see appendix for more details)

$$r(o_i, q) = r_{\text{ans}}(o_i, q) + \alpha\, r_{\text{fmt}}(o_i)$$

where r_{ans} denotes the outcome reward for answer correctness and r_{fmt} denotes the format reward.

4 EXPERIMENT

4.1 BENCHMARKS & BASELINES

Benchmarks. We evaluate `SearchPLM` on a diverse set of benchmarks: 1) **multi-hop question answering**, including MuSiQue (Trivedi et al., 2022a), HotpotQA (Yang et al., 2018), 2WikiMultihopQA (Ho et al., 2020) and Bamboogle (Press et al., 2022), 2) **complex information-seeking**, including BrowseComp (Wei et al., 2025) and GAIA (Mialon et al., 2023) and 3) **open-domain web question answering** WebWalker (Wu et al., 2025).

Table 3: Performance across benchmarks. Best results are **bold**; second-best results are underlined. † denotes a significant improvement (t-test, $p < 0.05$).

Model	MuSiQ.	HQA	2Wiki	Bamboo.	BrowC.	GAIA	WebW.	Avg.	Search.
Instruction-Tuned LLMs									
Gemma3-27B	28.91	71.00	77.90	82.00	1.95	31.86	51.69	58.74	2.82
Qwen2.5-72B-Instruct	30.47	72.60	78.20	80.80	4.69	36.27	51.99	59.66	3.53
DeepSeek-V3-671B	29.69	76.10	82.00	**84.80**	1.17	43.14	52.13	61.70	2.35
Reasoning-Enhanced LLMs									
Qwen3-8B	21.88	63.30	56.50	76.40	3.12	26.50	41.49	47.91	1.25
Qwen3-14B	24.22	67.30	73.00	78.00	0.78	33.50	46.98	54.98	1.49
Qwen3-30B-A3B	29.69	66.50	69.00	82.80	1.95	35.15	47.06	54.44	1.84
Qwen3-32B	23.83	58.10	57.00	78.40	0.46	32.50	40.96	47.01	1.58
QwQ-32B	25.39	60.50	50.40	73.20	1.56	26.60	42.50	46.08	1.90
Agentic-Enhanced LLMs									
ReCall-32B	32.03	71.40	78.10	77.60	5.47	30.58	29.10	51.85	2.32
GLM-4-32B-0414	37.89	74.00	76.70	**84.80**	6.25	34.80	44.40	57.98	2.96
Search-Enhanced LLMs									
StepSearch-7B	16.41	46.60	39.00	52.00	0.00	11.65	12.52	28.25	1.96
WebFilter-7B	27.34	64.40	69.30	68.40	1.56	16.02	33.67	47.91	3.49
Deep-Researcher-7B	35.16	68.00	68.40	74.80	3.52	31.37	43.82	53.40	2.58
R1-Searcher-8B	27.73	67.00	68.10	72.80	0.78	10.68	31.79	47.62	3.08
Search-R1-32B	32.03	71.50	77.70	**84.80**	1.56	26.21	33.21	53.05	1.93
WebThinker-32B	32.03	66.40	74.80	80.80	3.91	46.60	46.86	56.37	4.49
ASearcher-32B	37.80	68.50	74.40	80.80	7.83	45.45	44.12	56.79	1.89
MiroThinker-v1-30B	32.81	72.60	81.00	82.40	9.77	43.00	47.29	62.03	7.84
SearchPLM-8B	33.98	73.70	81.70	77.60	6.25	37.13	54.34	61.63	8.67
SearchPLM-14B	**39.84**†	**77.80**†	**83.80**†	84.00	**12.11**†	**50.49**†	**63.12**†	**67.43**†	5.75

Baselines. We compare our method against the following baseline LLMs with different model sizes as backbones for our multi-turn ReAct-based search agents under the same setting:

(1) **Instruction-Tuned LLMs**, which are mainly fine-tuned to have strong instruction following capabilities, including `Gemma3-27B`, `Qwen2.5-72B-Instruct`, and `DeepSeek-V3`;

(2) **Reasoning-Enhanced LLMs**, which are trained to have strong reasoning capabilities, including `Qwen3-8B`, `Qwen3-14B`, `Qwen3-30B-A3B`, `Qwen3-32B` and `QwQ-32B`;

(3) **Agentic-Enhanced LLMs**, which are optimized for multi-turn agentic tool-use capabilities, including `GLM-4-32B` and `ReCall` (Chen et al., 2025);

(4) **Search-Enhanced LLMs**, which are optimized via RL with step-wise or outcome-wise reward, including `Search-R1` (Jin et al., 2025), `R1-Searcher` (Song et al., 2025), `StepSearch` (Wang et al., 2025), `Deep-Researcher` (Zheng et al., 2025), `WebThinker` (Li et al., 2025c), `WebFilter` (Dai et al., 2025), `ASearcher` (Gao et al., 2025) and `MiroThinker` (Team et al., 2025).

4.2 Experimental Setup

Implementation. We use `Qwen3-8B` and `Qwen3-14B` as base models. In the SFT stage, we fine-tune them with LoRA (Hu et al., 2021), using ranks of 32 and 128 for one epoch. In the RL stage, we adopt the GRPO algorithm, with a maximum of 20 iterations and 3 training epochs. We use the Google Web Search API, retrieving three documents for each query, and use same summarizer `DeepSeek-V3` for all baselines and our method.

Evaluation Metrics. We evaluate the performance on these benchmarks using the *LLM-as-a-Judge* method (Zheng et al., 2023) with `DeepSeek-V3`, where the judge assigns a score of 1.0 for correct answers, 0.5 for partially correct answers, and 0.0 for incorrect answers, based on the comparison between golden and predicted answers.

4.3 Main Results

Table 4: Ablation studies. **E-C-S**: Exploration-Coverage-Synthesis prompting.

Model	MuSiQ.	HQA	2Wiki	Bamboo.	BrowC.	GAIA	WebW.	Avg.	Search.
Qwen3-14B	24.22	67.30	73.00	78.00	0.78	33.50	46.98	54.98	1.49
Strategy-guided Supervised Distillation									
+ E-C-S	29.69	70.60	80.10	83.60	3.12	32.32	51.91	59.12(+4.14)	2.00
+ SFT	38.67	76.20	83.80	82.00	7.03	45.10	59.93	65.40(+6.28)	6.64
Reinforcement Alignment									
w/o BM25	37.11	75.55	82.60	**84.40**	8.59	38.73	58.53	64.36	8.49
SearchPLM	**39.84**†	**77.80**†	**83.80**†	84.00	**12.11**†	**50.49**†	**63.12**†	**67.43**†(+2.03)	5.75(-0.89)

As shown in Table 3, `SearchPLM` achieves the best or second-best performance among most open-source baselines, with statistically significant gains under paired t-tests over per-sample results (p-value < 0.05).

Table 5: Ablation of E-C-S for `Qwen3-14B`.

Setting	BrowC.	GAIA	WebW.	Search.
w E-C-S	**3.12**	**32.32**	**51.91**	**2.00**
w/o Explore	1.20	30.21	43.97	1.49
w/o Coverage	2.48	27.08	40.71	1.58
w/o Synthesis	1.17	31.77	43.16	1.56

Effectiveness of Sustained Exploration. Compared with baselines that often terminate prematurely after only 1.5–3.5 search steps, `SearchPLM`-14B reaches a much deeper average search depth of **5.75**. This suggests that expert-induced search strategies help optimize query generation, reduce premature termination, and promote cross-source verification.

Efficient Search Strategy. With our long-document refinement pipeline during RL training, including BM25-based chunking and query-aware summarization, the model learns to generate more targeted queries; the average number of searches decreases from 6.64 after SFT to 5.75 after RL, indicating improved search efficiency and document awareness.

Robustness across Domains. On out-of-domain benchmarks such as BROWSECOMP and GAIA, `SearchPLM` generalizes well to atypical questions with implicit constraints. It outperforms `WebThinker-32B` on GAIA (50.49% vs. 46.60%) and `MiroThinker-V1.0-32B` on BROWSECOMP (12.11% vs. 7.83%), showing the importance of effective search strategies for search agents.

4.4 ABLATION STUDY

Stage-wise Ablation Study. We evaluate our method under three increasingly strong settings (Table 4): `Qwen3-14B + E-C-S prompting`, which tests the proposed three-stage search framework without fine-tuning; `Qwen3-14B + SFT`, which improves performance from 59.12 to 65.40 via teacher-distilled trajectories; and `Qwen3-14B + RL` (`SearchPLM`), which further improves performance to 67.43 while reducing average search steps from 6.64 to 5.75, indicating more efficient and generalizable search behavior.

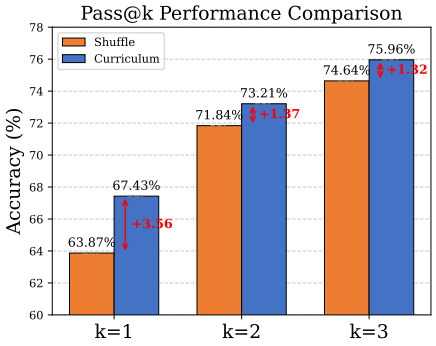


Figure 3: Pass@k Comparison.

Ablation Study of Exploration-Coverage-Synthesis Search Strategy. To assess the contribution of each E-C-S stage, we remove the prompt for one stage at a time while keeping the others unchanged. Due to space limitations, we report results on BROWSECOMP, GAIA, and WEBWALKER in Table 5, with additional results in the appendix. Removing any stage leads to performance degradation, confirming that these are complementary and jointly form an effective search strategy for cross-domain generalization.

Ablation Study of Long-document Refinement. To examine the effect of the BM25-based long-document refinement pipeline, we compare `SearchPLM` with a variant trained without BM25 retrieval during RL. As shown in Table 4, removing BM25 decreases the average performance from 67.43 to 64.36 and increases the average search steps from 5.75 to 8.49. This indicates that query-relevant chunk selection helps the model generate more informative queries, reduces redundant searches, and improves both effectiveness and search efficiency during reinforcement alignment.

Table 6: Agreement among human annotators and LLM judges.

Comparison	Human-Human		LLM-Human	
	Agr.	κ	Agr.	κ
1 vs 2 / LLM vs 1	93.88	0.864	87.76	0.732
1 vs 3 / LLM vs 2	95.92	0.900	89.80	0.783
2 vs 3 / LLM vs 3	89.80	0.767	87.76	0.726
Kimi-K2.5 (Judge–Judge)			91.22	0.832
GLM-5 (Judge–Judge)			90.11	0.811

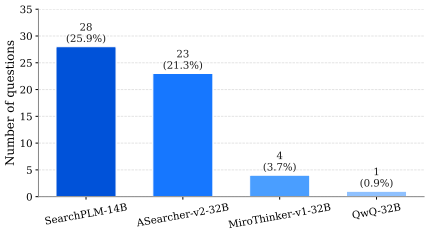


Figure 4: Best Search Strategy Voting.

Ablation of Curriculum RL Training. To evaluate the curriculum constructed by ranking questions according to correctness rates, we train a variant with randomly shuffled data under the same setting. As shown in Figure 3, curriculum-based training consistently outperforms random shuffling in pass@k accuracy, demonstrating its effectiveness.

5 Analysis and Disccussion

Robustness of Evaluation. We validate our LLM-as-a-Judge evaluation using 108 professionally annotated instances. As shown in Table 6, human annotators show strong agreement (93.20%, $\kappa = 0.844$), and the LLM judge aligns well with human judgments (88.44%, $\kappa = 0.747$). Re-scoring the predictions with `Kimi-K2.5` and `GLM-5` also yields high consistency with the original evaluator, with agreement rates of 91.22% ($\kappa = 0.832$) and 90.11% ($\kappa = 0.811$), respectively. Finally, anonymous trajectory-level preference voting in Fig. 4 provides additional evidence that `SearchPLM` learns more effective search strategies.

Useful Query Analysis. To better understand the causes of the performance gains, we analyze WebWalker cases where the base model `Qwen3-14B` fails. As shown in Table 7, although `SearchPLM` conducts substantially more searches, it still achieves a consistently higher useful-query ratio across all difficulty levels. This indicates that the improvement is not merely due to a larger search budget or longer trajectories; instead, `SearchPLM` learns to **issue more effective queries** and allocate search effort more strategically.

Table 7: Useful-query analysis on the WebWalker subset. Q/UQ/R denote total queries, useful queries, and useful-query ratio (%).

Model	Easy (24)			Medium (61)			Hard (49)		
	Q	UQ	R.	Q	UQ	R.	Q	UQ	R.
`Qwen3-14B`	42	18	42.86	109	61	55.96	105	38	36.19
`SearchPLM`	139	89	**64.03**	389	255	**65.55**	378	194	**51.32**
Δ R.	–	–	**+21.17**	–	–	**+9.59**	–	–	**+15.13**

6 Related Work

To be brief, our work is mainly related to Query Generation and Understanding in Large Language Models and Agentic Retrieval-augmented Generation (also known as "Deep Search"). Due to limited space, we discuss these related work in Appendix B.

7 Conclusion

In this paper, we systematically analyze the search behavior of Large Language Models in complex searching tasks. Our analysis diagnoses a key failure mode: LLMs tend to issue overly specific queries and terminate search prematurely, indicating that they have learned the mechanics of retrieval but lack effective search strategies. To address this limitation, we propose the Exploration-Coverage-Synthesis (E-C-S) paradigm and build `SearchPLM`, a search-proficient language model trained with strategy-enhanced supervised fine-tuning and curriculum-based reinforcement learning. Experiments on seven benchmarks show that `SearchPLM` achieves strong performance among open-source models while conducting substantially deeper and smarter search. These results highlight the importance of optimizing search strategies, rather than relying solely on larger models or more data, for building more effective LLM-based search agents.

8 AI USE STATEMENT

In this paper, we use ChatGPT to improve the presentations of this paper and Codex to assist in experiments.[2] The authors reviewed all AI-assisted text and take responsibility for the final manuscript and all artifacts.

REFERENCES

Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. Self-rag: Learning to retrieve, generate, and critique through self-reflection. *arXiv preprint arXiv:2310.11511*, 2023.

Barbara Blummer. Evaluating the effectiveness of metacognitive tool on graduate students information search: A pilot study. 2011.

Chi-Min Chan, Chunpu Xu, Ruibin Yuan, Hongyin Luo, Wei Xue, Yike Guo, and Jie Fu. Rq-rag: Learning to refine queries for retrieval augmented generation. *arXiv preprint arXiv:2404.00610*, 2024.

Mingyang Chen, Linzhuang Sun, Tianpeng Li, Haoze Sun, Yijie Zhou, Chenzheng Zhu, Haofen Wang, Jeff Z. Pan, Wen Zhang, Huajun Chen, Fan Yang, Zenan Zhou, and Weipeng Chen. Research: Learning to reason with search for llms via reinforcement learning, 2025. URL https://arxiv.org/abs/2503.19470.

Yuqin Dai, Shuo Yang, Guoqing Wang, Yong Deng, Zhanwei Zhang, Jun Yin, Pengyu Zeng, Zhenzhe Ying, Changhua Meng, Can Yi, Yuchen Zhou, Weiqiang Wang, and Shuai Lu. Careful queries, credible results: Teaching rag models advanced web search tools with reinforcement learning, 2025. URL https://arxiv.org/abs/2508.07956.

Jiaxuan Gao, Wei Fu, Minyang Xie, Shusheng Xu, Chuyi He, Zhiyu Mei, Banghua Zhu, and Yi Wu. Beyond ten turns: Unlocking long-horizon agentic search with large-scale asynchronous rl, 2025. URL https://arxiv.org/abs/2508.07976.

Julian Hirt, Thomas Nordhausen, Jasmin Meichlinger, Volker Braun, Adelheid Zeller, and Gabriele Meyer. Educational interventions to improve literature searching skills in the health sciences: a scoping review. *Journal of the Medical Library Association: JMLA*, 108(4):534, 2020.

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

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

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.

Tushar Khot, Daniel Khashabi, Kyle Richardson, Peter Clark, and Ashish Sabharwal. Text modular networks: Learning to decompose tasks in the language of existing models. *arXiv preprint arXiv:2009.00751*, 2020.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*, 2023.

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

[2] https://chatgpt.com/

Kuan Li, Zhongwang Zhang, Huifeng Yin, Rui Ye, Yida Zhao, Liwen Zhang, Litu Ou, Dingchu Zhang, Xixi Wu, Jialong Wu, Xinyu Wang, Zile Qiao, Zhen Zhang, Yong Jiang, Pengjun Xie, Fei Huang, and Jingren Zhou. Websailor-v2: Bridging the chasm to proprietary agents via synthetic data and scalable reinforcement learning. *CoRR*, abs/2509.13305, 2025a. doi: 10.48550/ARXIV.2509.13305. URL https://doi.org/10.48550/arXiv.2509.13305.

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, 2025b. URL https://arxiv.org/abs/2501.05366.

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

Kun Luo, Hongjin Qian, Zheng Liu, Ziyi Xia, Shitao Xiao, Siqi Bao, Jun Zhao, and Kang Liu. Infoflow: Reinforcing search agent via reward density optimization, 2025. URL https://arxiv.org/abs/2510.26575.

Xinbei Ma, Yeyun Gong, Pengcheng He, Hai Zhao, and Nan Duan. Query rewriting for retrieval-augmented large language models. *arXiv preprint arXiv:2305.14283*, 2023.

Grégoire Mialon, Clémentine Fourrier, Craig Swift, Thomas Wolf, Yann LeCun, and Thomas Scialom. Gaia: a benchmark for general ai assistants, 2023. URL https://arxiv.org/abs/2311.12983.

Sewon Min, Victor Zhong, Luke Zettlemoyer, and Hannaneh Hajishirzi. Multi-hop reading comprehension through question decomposition and rescoring. *arXiv preprint arXiv:1906.02916*, 2019.

Reiichiro Nakano, Jacob Hilton, Suchir Balaji, Jeff Wu, Long Ouyang, Christina Kim, Christopher Hesse, Shantanu Jain, Vineet Kosaraju, William Saunders, Xu Jiang, Karl Cobbe, Tyna Eloundou, Gretchen Krueger, Kevin Button, Matthew Knight, Benjamin Chess, and John Schulman. Webgpt: Browser-assisted question-answering with human feedback. *CoRR*, abs/2112.09332, 2021. URL https://arxiv.org/abs/2112.09332.

OpenAI. Browsecomp: A benchmark for search and browsing agents. https://openai.com/index/browsecomp/, 2024. Accessed: 2025-10-20.

Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah A Smith, and Mike Lewis. Measuring and narrowing the compositionality gap in language models. *arXiv preprint arXiv:2210.03350*, 2022.

Hongjin Qian and Zheng Liu. Scent of knowledge: Optimizing search-enhanced reasoning with information foraging. *arXiv preprint arXiv:2505.09316*, 2025.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. *Advances in Neural Information Processing Systems*, 36, 2024.

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

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. Hybridflow: A flexible and efficient rlhf framework. *arXiv preprint arXiv: 2409.19256*, 2024.

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

Hao Sun, Zile Qiao, Jiayan Guo, Xuanbo Fan, Yingyan Hou, Yong Jiang, Pengjun Xie, Yan Zhang, Fei Huang, and Jingren Zhou. Zerosearch: Incentivize the search capability of llms without searching, 2025a. URL https://arxiv.org/abs/2505.04588.

Zhaoyan Sun, Xuanhe Zhou, Guoliang Li, Xiang Yu, Jianhua Feng, and Yong Zhang. R-bot: An llm-based query rewrite system. *Proc. VLDB Endow.*, 18(12):5031–5044, 2025b. doi: 10.14778/3750601.3750625. URL https://www.vldb.org/pvldb/vol18/p5031-li.pdf.

Zhengwei Tao, Jialong Wu, Wenbiao Yin, Junkai Zhang, Baixuan Li, Haiyang Shen, Kuan Li, Liwen Zhang, Xinyu Wang, Yong Jiang, Pengjun Xie, Fei Huang, and Jingren Zhou. Webshaper: Agentically data synthesizing via information-seeking formalization. *CoRR*, abs/2507.15061, 2025. doi: 10.48550/ARXIV.2507.15061. URL https://doi.org/10.48550/arXiv.2507.15061.

MiroMind Team, Song Bai, Lidong Bing, Carson Chen, Guanzheng Chen, Yuntao Chen, Zhe Chen, Ziyi Chen, Jifeng Dai, Xuan Dong, et al. Mirothinker: Pushing the performance boundaries of open-source research agents via model, context, and interactive scaling. *arXiv preprint arXiv:2511.11793*, 2025.

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, 2022a.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. Interleaving retrieval with chain-of-thought reasoning for knowledge-intensive multi-step questions. *arXiv preprint arXiv:2212.10509*, 2022b.

Ziliang Wang, Xuhui Zheng, Kang An, Cijun Ouyang, Jialu Cai, Yuhang Wang, and Yichao Wu. Stepsearch: Igniting llms search ability via step-wise proximal policy optimization, 2025. URL https://arxiv.org/abs/2505.15107.

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. *CoRR*, abs/2504.12516, 2025. doi: 10.48550/ARXIV.2504.12516. URL https://doi.org/10.48550/arXiv.2504.12516.

Tomer Wolfson, Mor Geva, Ankit Gupta, Matt Gardner, Yoav Goldberg, Daniel Deutch, and Jonathan Berant. Break it down: A question understanding benchmark. *Transactions of the Association for Computational Linguistics*, 8:183–198, 2020.

Jialong Wu, Wenbiao Yin, Yong Jiang, Zhenglin Wang, Zekun Xi, Runnan Fang, Linhai Zhang, Yulan He, Deyu Zhou, Pengjun Xie, and Fei Huang. Webwalker: Benchmarking llms in web traversal, 2025. URL https://arxiv.org/abs/2501.07572.

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. *arXiv preprint arXiv:1809.09600*, 2018.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629*, 2022.

Yuyao Zhang, Zhicheng Dou, Xiaoxi Li, Jiajie Jin, Yongkang Wu, Zhonghua Li, Ye Qi, and Ji-Rong Wen. Neuro-symbolic query compiler. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Findings of the Association for Computational Linguistics, ACL 2025, Vienna, Austria, July 27 - August 1, 2025*, pp. 12138–12155. Association for Computational Linguistics, 2025. URL https://aclanthology.org/2025.findings-acl.628/.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging llm-as-a-judge with mt-bench and chatbot arena, 2023. URL https://arxiv.org/abs/2306.05685.

Yuxiang Zheng, Dayuan Fu, Xiangkun Hu, Xiaojie Cai, Lyumanshan Ye, Pengrui Lu, and Pengfei Liu. Deepresearcher: Scaling deep research via reinforcement learning in real-world environments. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 414–431, 2025.

APPENDIX

A Limitations **16**

B Related Work **16**

C More Ablations of Exploration-Coverage-Synthesis Strategy **16**

D Efficiency and Cost Analysis **17**

E Ablation on Strategy-Guided Supervised Distillation **17**

F Advanced Search Operators **17**

G Training Details **17**

G.1 Strategy-Guided Supervised Distillation 17

G.2 Curriculum Reinforcement Alignment 18

G.3 Reward Function Details 18

H Inference Details **18**

I Search Behavior Analysis **19**

J Benchmarks **19**

J.1 Multi-Hop QA Benchmarks 20

J.2 Complex Information Seeking Benchmarks 20

J.3 Open-domain Web Question Answering Benchmark 20

K Baselines **20**

K.1 Large Language Model Baselines 20

K.2 Training-Based Methods for Agentic-Search LLMs 21

L Prompt Template **22**

L.1 Prompt for Exploration-Coverage-Synthesis 22

L.2 Prompt for Summarizing the Webpage Content 23

L.3 Prompt for Keyword Extraction 23

L.4 Prompt for Failure Attribution 23

L.5 Prompt for Failure Attribution 24

M Case Study **24**

N Structured Human Search Behavior Questionnaire **27**

N.1 Part A: Background Information 28

N.2 Part B: Search Behavior in Specific Situations 28

N.3 Part C: Strategy Preference Items 30
N.4 Part D: Open-ended Reflection 30
N.5 Strategy Development 31

A LIMITATIONS

In this work, due to computational constraints and the costs associated with the Google Search API, we did not train larger models or use additional synthetic training data to further scale `SearchPLM`. Furthermore, we have not yet explored how to measure and optimize the search strategy of the model more accurately through the current promising rubrics-based training method. Additionally, our exploration of multimodal search and reasoning remains limited. These constraints highlight potential directions for future work to enhance `SearchPLM`.

B RELATED WORK

Query Understanding. Effective search agents are based on the precise understanding of user intent and the formulation of executable queries. Prior research leverages Language Models (LMs) for query decomposition, rewriting, disambiguation, and expansion (Min et al., 2019; Ma et al., 2023; Khot et al., 2020; Chan et al., 2024; Zhang et al., 2025). Query decomposition uses grammatical rules or semantic reasoning to break down complex questions (such as multi-hop questions) into a series of executable sub-questions (Min et al., 2019; Wolfson et al., 2020; Zhang et al., 2025). Query rewriting and disambiguation can better identify the search intent and convert it into a form with better retrieval effectiveness (Ma et al., 2023; Sun et al., 2025b). Some recent works have also deeply integrated query understanding into the retrieval-reasoning loops of LLMs (Chan et al., 2024; Wang et al., 2025), optimizing the query content through multiple iterations (Trivedi et al., 2022b). However, how to further improve the query quality of end-to-end search-enhanced LLMs remains underexplored.

Agentic Retrieval-Augmented Generation. Retrieval-augmented generation (RAG) approaches answer user questions by retrieving relevant documents (Lewis et al., 2020), while LLM agents based on paradigms such as ReAct (Yao et al., 2022), Self-RAG (Asai et al., 2023), and IR-CoT (Trivedi et al., 2022b) can address more complex problems through iterative reasoning–retrieval loops. With recent advances in LLM tool-use capabilities (Schick et al., 2024), particularly in web search and browsing (Nakano et al., 2021; Li et al., 2025b;c), an increasing number of studies explore enhancing LLMs' search and browsing abilities via synthetic hard training data (Tao et al., 2025; Li et al., 2025a), simulated environments (Sun et al., 2025a), implicit preference optimization (Li et al., 2025c), and Reinforcement Learning with Verifiable Rewards (RLVR) (Song et al., 2025; Jin et al., 2025; Zheng et al., 2025; Dai et al., 2025; Gao et al., 2025; Qian & Liu, 2025; Luo et al., 2025; Team et al., 2025). However, these methods largely overlook how effective human's search strategy priors can be explicitly incorporated into the model's training.

Moreover, it should be emphasized that industry-led deep-search agents are often not limited to search alone, but may also involve complex tool use, such as web browsing, keyword lookup, and code execution (e.g., Tongyi Deepresearch.), making performance comparisons across different baselines less fair. Therefore, the core academic contribution of this paper mainly focuses on diagnosing a search failure mode of current LLMs and evaluating the search capabilities of different LLMs, and does not include comparisons with such systems in the experimental tables.

C MORE ABLATIONS OF EXPLORATION-COVERAGE-SYNTHESIS STRATEGY

We present more ablation studies of Exploration-Coverage-Synthesis Strategy on 4 multi-hop question answering benchmarks as shown in Table 8:

Table 8: More ablations for `Qwen3-14B`.

Setting	Musiq	HQA	2Wiki	Bamboo	Searches
w E-C-S	**29.69**	**70.60**	**80.10**	**83.60**	**2.00**
w/o Explore	28.52	64.90	73.70	75.20	1.49
w/o Coverage	26.95	64.30	69.10	75.20	1.58
w/o Synthesis	25.39	64.70	68.00	78.40	1.56

D Efficiency and Cost Analysis

To provide a comprehensive view of SearchPLM's practical deployability, we analyze the computational efficiency and latency of our model during the inference phase. Table 9 presents the average statistics per question across our evaluation benchmarks.

Table 9: Efficiency of SearchPLM, averaged over test samples.

Metric	Avg.
Inference Time	7.92 s
Total Tokens	13,828.68
Prompt Tokens	12,166.47
Completion Tokens	1,662.21

Acceptable Latency. The average inference time is 7.92 seconds per question. Considering that the **Exploration-Coverage-Synthesis** paradigm involves multiple iterations of search interactions (iterative querying, reading, and reasoning), this latency is well within an acceptable range for real-time user-facing applications, offering a significant speedup compared to the minutes a human would spend performing the same depth of research.

High Context Utilization. The large prompt size reflects the SearchPLM's ability to exlpore the information space and process multiple retrieved documents ("Coverage").

Concise Reasoning. The relatively low completion token count indicates that SearchPLM is efficient in reasoning and query formulation, avoiding verbose or redundant outputs.

E Ablation on Strategy-Guided Supervised Distillation

In aligning the model via training, we use the E-C-S strategy to guide the teacher model to sample high-quality trajectories. To validate its effectiveness, we additionally train a baseline on trajectories directly sampled from the teacher without E-C-S prompting. As shown in Table 10, models trained on strategy-aligned trajectories consistently perform better on more challenging and atypical QA tasks, highlighting the value of explicit search-strategy alignment for improving real-world generalization.

Table 10: Impact of search strategy alignment.

Trajectory	Browc.	GAIA	WebW
Vanilla (Raw Traj.)	4.69	42.65	59.63
+ E-C-S strategy	**7.03**	**45.10**	**59.93**

F Advanced Search Operators

Following WebFilter (Dai et al., 2025), we train the SearchPLM to use advanced search operators during searching process as follows:

ADVANCED SEARCH OPERATORS

- `site:` restricts results to specific domains.
- `intitle:` narrows results by including terms in the page title.
- `inurl:` filters results based on URL keywords.
- `after:` and `before:` filter results by a date range.
- `OR` expands the search to cover overlapping or alternative aspects of the question.

G Training Details

In order to enhance the reproducibility of our method, we provide our training details as follows:

G.1 Strategy-Guided Supervised Distillation

We perform supervised fine-tuning with LoRA on `Qwen3-14B` and `Qwen3-8B`. For both models, we use a per-device batch size of 1 with gradient accumulation of 8, a learning rate of 1×10^{-4}, cosine learning rate scheduling, bf16 mixed precision, and train for 1 epoch. The main difference

between the two settings lies in the LoRA rank, which is set to 128 for `Qwen3-14B` and 32 for `Qwen3-8B`.

G.2 Curriculum Reinforcement Alignment

During the RL phase, we train `SearchPLM` with GRPO implemented in the `verl` framework (Sheng et al., 2024) on 8 NVIDIA H20 GPUs. Training is conducted with a global batch size of 32 for 3 epochs. We set the maximum prompt length to 512 and the maximum response length to 16,384.

For multi-turn search interaction, we enable tool-augmented rollout with at most 20 user turns and a maximum tool response length of 512. Unless otherwise specified, we use 8 rollouts per query for training and 2 rollouts for validation.

For optimization, we use Adam with a learning rate of 1×10^{-6}, cosine learning rate scheduling, gradient clipping of 1.0, and a KL regularization coefficient of 0.001 with low-variance KL loss. The PPO clipping range is set to $[0.2, 0.28]$. We enable gradient checkpointing, remove-padding optimization, and chunked prefill for efficiency.

Because retrieved webpages are often long, we additionally employ `Gemma3-27B-it` (deployed on 16 NVIDIA A100 GPUs) to refine and summarize retrieved content before feeding it into online RL. The summarization model is deployed separately from the policy training pipeline. All experiments are conducted on NVIDIA H20 and A100 GPUs.

G.3 Reward Function Details

For each sampled output o_i, we define the reward as

$$r(o_i, q) = r_{\text{ans}}(o_i, q) + \alpha\, r_{\text{fmt}}(o_i), \tag{9}$$

where r_{ans} is the answer reward and r_{fmt} is the format reward.

The answer reward is determined by an LLM-based evaluator:

$$r_{\text{ans}}(o_i, q) = \begin{cases} 1, & \text{if the final answer is judged correct,} \\ 0.5, & \text{else if the final answer is judged partially correct,} \\ 0, & \text{otherwise.} \end{cases} \tag{10}$$

We use an LLM judge instead of string-based metrics such as EM or F1, since such metrics can bias the model toward short "golden answers" that dominate the training set, and may incorrectly penalize longer but semantically correct responses. Since validating the correctness of the predicted answer is easy, the LLM-based reward can provide a more robust signal for semantic correctness.

The format reward is used to enforce valid multi-turn tool-use behavior:

$$r_{\text{fmt}}(o_i) = \sum_{t=1}^{T_i} r_{\text{fmt}}^{(t)}, \tag{11}$$

where

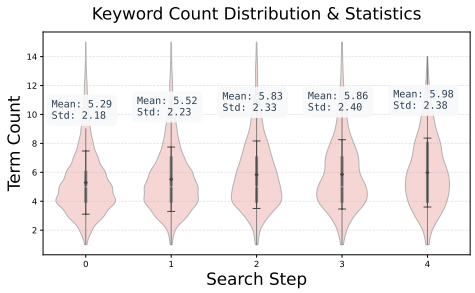
$$r_{\text{fmt}}^{(t)} = \begin{cases} -1, & \text{if step } t \text{ violates the required interaction format,} \\ 0, & \text{otherwise.} \end{cases} \tag{12}$$

Specifically, a penalty is assigned if the generated trajectory violates the required interaction format, including malformed or missing special tags, invalid search actions, or improper termination; otherwise, no format penalty is applied.

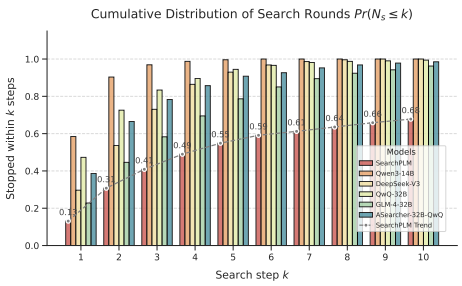
The final reward thus jointly encourages semantically correct answering and structurally valid multi-turn search behavior.

H Inference Details

We provide our inference details as follows: During the inference phase, we deploy the trained `SearchPLM` using vllm framework (Kwon et al., 2023), setting a maximum of 20 iterations. We use `DeepSeek-V3.1` to refine and summarize webpage content, following the same setting as all baselines. Therefore, the improvement can directly reflect the search strategy of our `SearchPLM`.



(a) Distribution of keywords per search query of `SearchPLM`.



(b) Step distributions of `SearchPLM`.

Figure 5: Statistics of `SearchPLM` on GAIA, BrowseComp, WebWalker and Musique.

Table 11: Analysis of search behaviors under a unified search budget. **Acc.** denotes the accuracy (%).

Model	$\Pr(T \leq 3)\downarrow$	$\Pr(T \leq 5)\downarrow$	$\mathbb{E}[\lvert q_1 \rvert]$	$\mathbb{E}[\lvert q \rvert]\downarrow$	**Acc.**	**Steps**
SearchPLM-8B	29.22%	39.54%	$5.18_{\pm 2.61}$	$6.06_{\pm 2.94}$	44.21	11.39
SearchPLM-14B	40.89%	54.87%	$5.78_{\pm 2.78}$	$6.03_{\pm 2.82}$	52.70	8.15

Retry Mechanism. To mitigate tool invocation parsing failures due to the inherent randomness of LLM outputs, we implement a 5-attempt retry mechanism for each iteration and all baselines. All evaluation instances are run asynchronously.

Time Sensitivity and Bias Mitigation. Considering the time sensitivity of web search results, all experiments are conducted within a concentrated period (approximately two weeks). During inference, the current time is also included as context for the model to ensure comparability of results. Additionally, some websites may directly contain answers to complex questions (e.g., huggingface [3]). To prevent information leakage during evaluation, we manually filtered out documents returned from these domains in the search results.

I Search Behavior Analysis

We analyze the search patterns of `SearchPLM`, as illustrated in Figure 5 and Table 11.

Concise Query Formulation. As shown in Figure 5a, `SearchPLM` exhibits a significantly lower mean and variance in keyword count per query. This suggests a shift towards more concise, human-like querying habits, avoiding the verbose queries often typical of unaligned LLMs.

Adaptive Search Depth. Regarding search steps (Figure 5b), the model adapts its effort dynamically: for complex questions, it engages in extended search sessions, with over 30% of instances requiring more than 10 steps.

J Benchmarks

We construct our evaluation suite by combining seven benchmarks that span multi-hop question answering and agentic web interaction: MuSiQue, HotpotQA, 2WikiMultiHopQA, Bamboogle, BrowseComp, GAIA, and WebWalkerQA.

Table 12 reports the number of evaluated instances from each benchmark. The evaluation suite spans diverse domains and difficulty levels.

[3] `https://huggingface.co/`

Table 12: Number of evaluation instances from each dataset.

Dataset	Count
MuSiQue	128
HotpotQA	500
2WikiMultiHopQA	500
Bamboogle	125
BrowseComp	128
GAIA	102
WebWalkerQA	680

J.1 Multi-Hop QA Benchmarks

HotpotQA is a widely used benchmark for explainable multi-hop question answering. It requires models to reason over multiple supporting Wikipedia documents, and provides sentence-level supporting facts for supervision and evaluation.

2WikiMultiHopQA evaluates multi-hop reasoning by combining unstructured Wikipedia text with structured Wikidata evidence. Each instance includes an explicit reasoning path represented as Wikidata triples, enabling fine-grained evaluation of reasoning steps beyond answer correctness.

MuSiQue is constructed by composing connected single-hop questions into 2–4 hop multi-hop questions. By design, it reduces annotation artifacts and makes shortcut/disconnected reasoning less effective, resulting in a more challenging evaluation of genuine multi-hop reasoning.

Bamboogle contains information-seeking questions for which Google Search typically produces incorrect answers. It tests robustness to misleading retrieval signals and the ability to verify evidence rather than over-trust search results.

J.2 Complex Information Seeking Benchmarks

BrowseComp is a benchmark for web-browsing agents proposed by OpenAI. It focuses on the core capability of persistent, multi-step browsing to locate hard-to-find and entangled information on the internet, with short and verifiable reference answers.

GAIA is a benchmark for General AI Assistants with real-world questions that are conceptually simple for humans but challenging for AI systems. It requires a combination of reasoning, tool use (e.g., web browsing), and multi-modal understanding.

J.3 Open-domain Web Question Answering Benchmark

WebWalkerQA evaluates web traversal in real-world websites. It measures an agent's ability to navigate across a website's subpages and systematically extract precise information from uncurated web content, often requiring long-horizon exploration.

K Baselines

K.1 Large Language Model Baselines

We compare our method with different Large Language Models as follows:

Gemma-3 Gemma is a family of generative artificial intelligence language model that can be used in a wide variety of generation tasks, including question answering, summarization, and reasoning. We use `Gemma3-27B-it` for comparison.

Qwen-3 Qwen is a series of open-source language models with different sizes and capabilities such as instruction following and reasoning. We use `Qwen2.5-72B-Instruct`, `Qwen3-8B`, `Qwen3-14B`, `Qwen3-30B-A3B`, `Qwen3-32B` and `QwQ-32B` for comparison.

GLM-4 GLM-4 series are foundation language models specifically designed for agentic scenarios. We use `GLM-4-32B` for comparison.

DeepSeek-V3 DeepSeek-V3 series are Mixture-of-Experts (MoE) language models with 671B total parameters with 37B activated for each token, and we use `DeepSeek-V3` for comparison.

K.2 Training-Based Methods for Agentic-Search LLMs

We also compare our method with representative search-enhanced LLM baselines ranging from reinforcement learning (RL) based tool-use training to reasoning-enhanced retrieval and browsing frameworks as follows:

ReCall (Chen et al., 2025) trains LLMs to perform tool calling and tool-augmented reasoning via reinforcement learning, without requiring supervised trajectories of tool usage or intermediate reasoning steps.

WebFilter (Dai et al., 2025) is an agentic RAG framework that improves web-based retrieval by generating source-restricted queries with advanced search operators and filtering unreliable or irrelevant content, supported by behavior- and outcome-driven rewards.

StepSearch (Wang et al., 2025) improves LLM search behavior using step-wise PPO with richer intermediate rewards. It provides token-level process supervision via information-gain rewards and redundancy penalties to guide iterative query formulation and evidence acquisition.

R1-Searcher (Song et al., 2025) employs a two-stage outcome-supervised RL pipeline to teach LLMs when to invoke web search and how to effectively use retrieved evidence during reasoning, without relying on cold-start instruction fine-tuning.

Search-R1 (Jin et al., 2025) extends the DeepSeek-R1-style reasoning paradigm by interleaving multi-turn reasoning with search actions. It optimizes the full trajectory with outcome-based RL and uses retrieved-token masking to stabilize training.

DeepResearcher (Zheng et al., 2025) scales end-to-end reinforcement learning for LLM-based research agents in real-world web environments, using authentic search interactions and a multi-agent browsing architecture to train models beyond controlled RAG settings. This enables emergent behaviors such as planning, source cross-validation, self-reflection, and honest failure reporting during open-domain research.

WebThinker (Li et al., 2025c) augments large reasoning models with a Deep Web Explorer module and an autonomous Think–Search–and–Draft strategy, enabling the agent to interleave reasoning, web exploration, and report drafting.

ASearcher (Gao et al., 2025) is a large-scale online RL framework for long-horizon agentic search. It leverages asynchronous RL to support extended multi-turn interaction and achieves strong performance on challenging benchmarks.

MiroThinker (Team et al., 2025) is an open-source research agent designed for tool-augmented reasoning and information-seeking tasks. It introduces interactive scaling as an additional scaling dimension beyond model size and context length, allowing the agent to perform deeper multi-turn search, acquire external evidence, and refine reasoning trajectories.

L PROMPT TEMPLATE

L.1 PROMPT FOR EXPLORATION-COVERAGE-SYNTHESIS

In this section, we provide the full prompt used to instantiate the proposed Exploration–Coverage–Synthesis (E-C-S) search paradigm distilled from human experts as follows.

The prompt operationalizes our three-stage framework by instructing the model to (i) probe the information space with concise initial queries, (ii) expand coverage through iterative refinement, facet decomposition, failure-aware backtracking and advanced search operators, and (iii) synthesize answers via multi-source cross-verification.

We show the complete prompt for reproducibility and to facilitate future extensions.

EXPLORATION-COVERAGE-SYNTHESIS PROMPT

You are a deep research assistant, and your task is to conduct multi-step research to provide a comprehensive and accurate answer to the user's query.
You have access to two tools:

1. `<search>`: Use this tool to search for information with `<search>your query here</search>`. The search results will be provided with `<search_result>..search results..</search_result>`.
2. `<answer>`: Use this tool to provide the final synthesized answer to the user with `<answer>The final synthesized answer here.</answer>`.
Start with `<think>`: Your response must ALWAYS begin with a `<think></think>` block.
Example:
```
<think>your thought here</think>
<search>your query here</search>
```
The maximum number of search attempts is limited to {max_search_limit}. The current time is {current_time}.
Below are some strategies that can help with your search process:

Query Format
- Prioritize high-frequency terms: Use common, frequently appearing terms relevant to the original question. Avoid overly specific or rare "associative words" unless absolutely necessary.
- Contextual search generation: Learn to analyze the results from the search process and generate new queries based on what you find. This means using the search results to guide subsequent searches for more specific or relevant information.
- Ensure contextual grounding: Each query must be grounded in at least one key entity (from the original question) to provide context for the search. This avoids the risk of launching irrelevant or overly general searches that lack context.

Stage1: Exploration
Progressive Exploration
- Initial exploration: Identify 1-2 core entities from the original question (e.g., names, locations, events) to formalize your intial query. This is the initial step to establish context or initiate the search. The goal here is not to be exhaustive but to open up a search window that allows for gradual refinement.
- Gradual refinement: As you gain more context from the initial query, gradually introduce secondary entities (e.g., time, location, subtopics) to refine the search further. This ensures you narrow down the query incrementally, rather than jumping to conclusions.
- Avoid exhaustive queries at the start: Do not try to include all relevant entities or details in the very first query. Overloading the query with every detail at once can limit the breadth of exploration and lead to missed results.

Stage2: Coverage
Dynamic Search Strategy
- Iterative coverage: If the first query (Q1) yields insufficient results, incrementally build on it by adding another aspect (P2) to the search. For example, starting with "best home coffee brewing methods", the agent may refine it with "grind size guide for pour-over coffee" to fill missing details.
- Facet-based coverage: Split the original question into separate facets and handle them independently. This allows the search process to focus on different dimensions of the question, like:
```
<search>home coffee brewing methods water temperature</search>
<search>home coffee brewing methods grind size ratio</search>
```
- Avoid searching for a single golden document: Instead of trying to find one document that answers everything, aim to cover different aspects of the question across multiple documents. Your strategy should be to aggregate information, not to find a one-size-fits-all answer.

Failure Handling (Zero-Recall Handling)
- Identify failures explicitly: If a query leads to no useful results (zero-recall), analyze why it happened. Common reasons include overly specific queries or reliance on the "Golden Document Fallacy" (assuming there's a single document that answers all parts of a complex question). Document the failure and consider how to fix it.
- Backtracking strategy: If a query doesn't work, backtrack and adjust the search path. You might:
* Remove one entity to make the query more general.
* Explore a broader search topic or category.
* Rephrase the query to account for the way the content might be phrased in documents or sources.
* Experiment with broader synonyms or different combinations of search terms.

Advanced Search Operators
- Use of search operators: At each stage, assess whether applying an advanced search operator can improve the precision of the search. Here

are some useful operators, and the reasoning for using them:
- site: Use when you need to filter results from a specific website or domain (e.g., site:aclweb.org).
- intitle: Use when the page title is important to include specific phrases (e.g., intitle:"deep search agent").
- inurl: Use when the page URL contains critical keywords (e.g., inurl:"deep search agent").
- after: or before: Use for filtering results within a time range (e.g., after:2023-01-01).
- OR: If a question has multiple possible solutions or overlapping aspects, use OR to expand the search and find more diverse sources of information.

Stage3: Synthesis
Source Verification
- Source credibility: Ensure the information from the retrieved documents is valid by cross-checking with authoritative sources and urls.
- If multiple documents point to the same information, this can serve as a validation.
-Be especially cautious with information found on non-official or user-generated sites.
- Cross-reference multiple results: Don't rely solely on the first document that seems to answer the question. Even if one document appears relevant, it's important to verify through multiple sources, particularly when dealing with complex or contested issues.
- Flagging conflicting information: If conflicting or incomplete information appears across sources, note the discrepancies and seek further clarification from other documents.

L.2 Prompt for Summarizing the Webpage Content

Exploration-Coverage-Synthesis Prompt

You are tasked with reading and analyzing web page based on the following inputs: Previous Reasoning Steps, Original Question, and Searched Web Page. Your objective is to extract relevant and helpful information for Current Search Query from the Searched Web Pages and seamlessly integrate this information into the Previous Reasoning Steps to continue reasoning for the original question.

 # Guidelines:

1. Analyze the Searched Web Page:

- Carefully review the content of searched web page.

- Identify factual information that is relevant to the Current Search Query and can aid in the reasoning process for the original question.

2. Extract Relevant Information:

- Select the information from the Searched Web Page that directly contributes to advancing the Previous Reasoning Steps.

- Ensure that the extracted information is accurate and relevant, following the original content of the web page.

- If there is no relevant information in the web page, do not provide a hallucination answer as a response, do not be distracted by the previous reasoning steps, and return "No relevant information found".

Please only output the information extracted from the web page and do not output other words. The current time is {current_time}.

L.3 Prompt for Keyword Extraction

Keyword Extraction Prompt

Extract the key search keywords from the following {context}.

 # Instructions:

1. Identify the most important keywords that capture the core information need

2. Include entities (names, places, dates), concepts, and constraints

3. Remove stopwords and filler words (the, is, are, what, when, etc.)

4. Keep the keywords concise and focused

5. Return ONLY a JSON object, nothing else

{context_label}: {text}

Respond in this EXACT JSON format (no extra text):

{{"keywords": ["keyword1", "keyword2", "keyword3"]}}

L.4 Prompt for Failure Attribution

Failure Attribution Prompt

You are analyzing why a search-augmented language model failed to answer a question correctly.

 # Question:

{question}

 # Search Trajectory:

{search_trajectory}

 # Model's Final Response:

{final_answer}

 # Task:

Analyze whether the model failed because it **used overly specific or overly long queries**, which caused retrieval failures (no relevant information found), leading the model to believe there was no answer or insufficient evidence to answer correctly.

Consider:

1. Are the queries too specific/narrow that they miss relevant documents?

2. Are the queries too long/complex that they fail to match any documents?

3. Did retrieval failures cause the model to conclude incorrectly?

4. Could simpler, more general queries have found the needed information?

 # Response Format (JSON only): {{
"failed_due_to_over_specific_query": true/false,
"diagnosis": "Brief explanation of why the model failed and whether over-specific queries were the cause"
}}

L.5 Prompt for Failure Attribution

Exploration-Coverage-Synthesis Prompt

Please analyze whether the predicted answer contains the key information from the reference answer or is semantically equivalent.

Question : {question}

Reference Answer: {golden_str}

Predicted Answer: {predict_str}

Requirements:
1. If the predicted answer is correct or mostly correct, return "Correct".
2. If the predicted answer is partially correct (part of answer or approximate result), return "Partially correct".
3. If the predicted answer is completely incorrect, return "Incorrect".

Return only one of: Correct | Partially correct | Incorrect. Do not add any explanation.

M Case Study

These two cases highlight the key advantages of `SearchPLM` in multi-turn web search.

In the first case, `SearchPLM` decomposes the question into a sequence of necessary subproblems, progressively identifying the target artist, album, single, and finally the lyric evidence needed for answering. This demonstrates its ability to perform **targeted exploration** and avoid premature answering before sufficient evidence is obtained.

In the second case, `SearchPLM` shows strong performance on a long-tail, realistic web question by persistently refining the search objective, grounding its reasoning in official university webpages, and verifying temporal evidence before producing the final answer. This reflects its strength in **iterative coverage** and **evidence-grounded synthesis** under noisy search conditions.

Overall, these examples suggest that `SearchPLM` is able to **move beyond shortcut-style document matching and instead conduct more effective multi-step search**, characterized by clearer problem decomposition, better query refinement, more persistent evidence accumulation, and more reliable final answering.

Case1(Part1)

User Query:
Question: What is the last word before the second chorus of the King of Pop's fifth single from his sixth studio album?

Step 1: Reasoning
▶ *Think:* ***I need to break down this question step by step.*** *The question asks for the "last word before the second chorus" of a specific song. The song is identified as the "King of Pop's fifth single from his sixth studio album."* ***First, I need to identify who "the King of Pop" is.*** *This is almost certainly a nickname for the singer Michael J ackson, who is famously known as the King of Pop.* ***Next, I need to find Michael Jackson's sixth studio album.*** *I should search for this to confirm the details. Once I have the album, I can find its fifth single.* ***Then, I need to analyze the lyrics of that specific single to locate the "second chorus" and find the word that comes just before it. My initial search should focus on identifying Michael Jackson's sixth studio album and its tracklist to confirm the fifth single.*** *After that, I can find the lyrics for that song and analyze the structure to find the second chorus and the preceding word. Let me start by searching for Michael Jackson's discography to identify his sixth studio album.*

Step 1: Action
`<search> Michael Jackson sixth studio album </search>`

Step 1: Observation
(web page 0):
- title: Thriller (album)
- content: No relevant information found.
(web page 1):
- title: Thriller | Michael Jackson Wiki
- content: Based on the web page, Michael Jackson's sixth studio album is "Thriller." The fifth single from this album is "Human Nature," as the singles are listed in order: "The Girl Is Mine," "Billie Jean," "Beat It," "Wanna Be Startin' Somethin'," "Human Nature," "P.Y.T. (Pretty Young Thing)," and "Thriller." Now, I need to find the lyrics of "Human Nature" to identify the last word before the second chorus.
<search> Human Nature Michael Jackson lyrics </search>
(web page 2):
- title: Thriller — Michael Jackson
- content: No relevant information found.

Step 2: Reasoning
► *Think: I need to find the last word before the second chorus of Michael Jackson's fifth single from his sixth studio album, which is "Human Nature" from the "Thriller" album. I need the lyrics of "Human Nature" to identify this word. My previous search for "Human Nature Michael Jackson lyrics" didn't return the specific information.* ***I should try a more targeted search query to find a reliable source for the full lyrics of "Human Nature".*** *Let me search using specific platforms known for having accurate lyrics.*

Step 2: Action
`<search> "Human Nature" Michael Jackson full lyrics </search>`

Step 2: Observation
(web page 0):
- title: Michael Jackson – Human Nature Lyrics - Genius
- content: Looking at the web page content, it appears to be a Genius lyrics page for Michael Jackson's "Human Nature." However, the provided content snippet does not include the actual lyrics of the song; instead, it shows promotional or unrelated content (e.g., Bailey Zimmerman, That Mexican OT, etc.). There is no relevant lyric information extracted from this page for identifying the last word before the second chorus. Therefore, based on the guidelines, I should return "No relevant information found" since the content does not contain the necessary lyrics.
No relevant information found.
(web page 1):
- title: Michael Jackson - Human Nature Lyrics | AZLyrics.com
- content: No relevant information found.
(web page 2):
- title: Michael Jackson - Human Nature lyrics - Musixmatch
- content: stare.
► *Think: The search results have confirmed that Michael Jackson's sixth studio album is "Thriller" and that the fifth single from this album is "Human Nature."* ***Now, the challenge is to find the lyrics for "Human Nature" specifically to identify the last word before the second chorus.***
The initial search for "Human Nature Michael Jackson lyrics" and "Human Nature Michael Jackson full lyrics" returned results, but the content snippets from Genius and AZLyrics were not directly useful. The snippet from Genius showed unrelated content and did not display the lyrics. The Musixmatch result showed a single word "stare," which is not sufficient.
I need to find a reliable source that displays the full lyrics. *The best approach is to look for a dedicated lyrics website that is known for accurate song transcripts. A common choice is "LyricsFreak" or perhaps a specific page that lists "Human Nature" lyrics. Let me search more specifically for a lyrics page that will list the full song, or try a different phrasing.* ***I'll try searching with "site:lyricsfreak.com" to target that specific site.***

Step 3: Action
`<search> site:lyricsfreak.com "Human Nature" Michael Jackson </search>`

Step 3: Observation
(web page 0):
- title: Michael Jackson - Human Nature lyrics | LyricsFreak
- content: Based on the lyrics from the web page, the song "Human Nature" by Michael Jackson has the following structure: - The first chorus begins with "If they say, why, why? Tell 'em that is human nature..." - After the first chorus, there is a verse: "Reaching out to touch a stranger... She likes the way I stare." - The second chorus follows immediately after this verse, starting with "If they say, why, why? Tell 'em that is human nature..." The last word before the second chorus is "stare," which ends the preceding verse: "She likes the way I stare."

Step 4: Reasoning
► *Think: I found the answer in the search result. The second chorus of "Human Nature" starts right after a verse that ends with the word "stare". Therefore, "stare" is the last word before the second chorus.*

Step 4: Answer
The last word before the second chorus of Michael Jackson's fifth single "Human Nature" from his sixth studio album "Thriller" is "stare".

CASE2

USER QUERY:
Question: 浙江大学计算机学院吴朝晖教授讲授的《网格计算》课程是哪一年首次上传到浙江大学计算机学院的教学资源中?

Step 1: Reasoning
► *Think: I need to find out the year when Professor Wu Zhaohui from Zhejiang University's College of Computer Science first uploaded his "Grid Computing" course to the college's teaching resources.* ***The high-priority entities are Wu Zhaohui, Zhejiang University Computer Science, and Grid Computing. The year of upload is the key information I need to find.*** *I should start by searching for information related to Professor Wu's course "Grid Computing" and its availability on Zhejiang University's teaching resources. Since this is a specific academic course, the information might be found on the university's official website, specifically the department's teaching resources section.* ***I'll begin with a general search query combining the professor's name, the course title, and the university.***

Step 1: Action
<search> 浙江大学计算机学院 吴朝晖 网格计算 课程</search>

Step 1: Observation
(web page 0):
- title: 网格计算（教学大纲） 浙江大学计算机科学与技术学院
- content: No relevant information found.
(web page 1):
- title: 课程简介 浙江大学计算机科学与技术学院
- content: 网格计算（教学大纲）2009-12-24

Step 2: Reasoning
► *Think: The first search query was: "浙江大学计算机学院 吴朝晖 网格计算 课程"... The mention of "2009-12-24" is likely the date of the document, but not necessarily the upload date...* ***The key entity is Professor Wu Zhaohui (吴朝晖) from the Computer Science college.*** *The search did not return a page specifically about him. We need to refine the search to focus on Professor Wu and his course.*

Step 2: Action
<search> 吴朝晖 网格计算 课程 浙江大学 教学资源 上传</search>

Step 2: Observation
(web page 0): No relevant information found.

Step 3: Reasoning
► *Think: I need to find out when Professor Wu Zhaohui's G̈rid Computingc̈ourse was first uploaded to Zhejiang University's Computer Science Department teaching resources. Let me search more specifically.*

Step 3: Action
<search> 网格计算 吴朝晖 浙江大学 教学资源 首次上传</search>
Step 3: Observation
No search results found

Step 4: Reasoning
► *Think: The initial searches haven't yielded direct information about the specific date when the course was first uploaded. The second search returned a result that seems to be related to a course description page, but the content snippet "网格计算（教学大纲）2009-12-24" suggests a date (December 24, 2009) might be associated with the course. However, this date could be the creation of the course outline page, not necessarily the first upload of the course materials. I need to refine my search strategy.* ***The key is to find information about the course's "首次上传" (first upload).*** *I should try searching more specifically for notices or announcements related to the course upload.*

Step 4: Action
<search> 吴朝晖 网格计算 首次 发布 通知 浙江大学</search>

Step 4: Observation
(web page 0-1): No relevant information found.

Step 5: Reasoning
► *Think: I need to find out when the "网格计算" course taught by Professor Wu Zhaohui at Zhejiang University's Computer Science Department was first uploaded to their teaching resources.*
From the first search results, I saw a page with "网格计算（教学大纲）2009-12-24". This seems relevant as it's the course outline for "网格计算" from 2009-12-24. ***However, I need to verify if this page is indeed the course materials page and if it contains information about the first upload date of the course resources.*** *The "首次上传" information might be on this course outline page or a separate announcement.*
Let me search more specifically for outline page details.

Step 5: Action
<search> 网格计算 教学大纲 发布 2009 </search>

Step 5: Observation
(web page 0):
- title: 网格计算（教学大纲）- 浙江大学计算机科学与技术学院
- content: 课程编号：... 课程名称：网格计算... 主讲教师：吴朝晖教授...

Step 6: Reasoning
► *Think: I see the result for "网格计算（教学大纲）" provides a course outline from 2009, which includes Professor Wu Chaohui as the main instructor. The date in the URL is "2009/12/24".* ***However, this is the course outline date, not necessarily the date when the teaching resources were first uploaded.*** *I need to find more specific information about when the course materials were first uploaded to Zhejiang University's teaching resource platform. Let me try searching more specifically about teaching resources and upload dates.*

Step 6: Action
<search> 网格计算 教学资源 上传 浙江大学 2009 </search>

Step 6: Observation
(web page 0): No relevant information found.

Step 7: Reasoning
► *Think: I will find out the specific year when Professor Wu Zhao Hui's course "Grid Computing" was first uploaded to Zhejiang University's computer science department teaching resources.*
From the search results so far, I found a very promising lead: the page titled "网格计算（教学大纲）- 浙江大学计算机科学与技术学院" (Grid Computing Teaching Syllabus - Zhejiang University College of Computer Science and Technology) with a URL containing "2009/1224". This suggests the page might have been created or updated in December 2009. The content confirms this is the correct course taught by Prof. Wu. ***However, I need to verify if this represents the *first* upload to the teaching resources system.*** *Let me search for earlier mentions of this course to see if it appeared before 2009.*

Step 7: Action
`<search> 网格计算 吴朝晖 浙江大学 教学资源 2008 </search>`

Step 7: Observation
(web page 0-1): No relevant information found.

Step 8: Reasoning
► *Think: The previous search results did not directly provide the specific time when the Grid Computing course was first uploaded to the teaching resources of the College of Computer Science at Zhejiang University. The results mainly focused on the course syllabus and basic course information, without explicitly indicating the upload time.Therefore, the search strategy needs to be adjusted. Possible approaches include:Identifying the teaching resource repository or course management system of the College of Computer Science at Zhejiang University, as such platforms typically record the upload time of course materials. Examining the syllabus page or related course description pages for implicit temporal information, such as page creation or last modification timestamps, which may correspond to the initial upload time. Searching for more specific phrases, such as "upload time" or "first publication."* ***An attempt can also be made to analyze the date embedded in the URL of the syllabus page, as this often reflects the page creation date and may indicate when the course was first uploaded.***

Step 8: Action
`<search> 2009/12/24 网格计算 吴朝晖 浙江大学 课程</search>`

Step 8: Observation
(web page 0):
- title: 网格计算（教学大纲）- 浙江大学计算机科学与技术学院
- content: 2009年12月24日

Step 9: Reasoning
► *Think: I have found a very promising result. The URL 'http://www.cs.zju.edu.cn/csen/2009/1224/c26995a1202793/page.htm' is from the Zhejiang University Computer Science department website, and the title is "网格计算（教学大纲）". The content specifically mentions "2009年12月24日".* ***This strongly suggests that this is the page where the "网格计算" course was first uploaded or published by Professor Wu Zhaohui's teaching resources.*** *The date "2009年12月24日" is the most critical piece of information. I should now attempt to open this specific page to get the full details confirming this date is indeed when the course was first uploaded. Let's search directly for this URL path to get the full content.*

Step 9: Action
`<search> site:cs.zju.edu.cn "网格计算" "2009/12/24" </search>`

Step 9: Observation
(web page 0):
- title: 网格计算（教学大纲）
- content: 2009年12月24日
(web page 1):
- title: 课程简介
- content: 网格计算（教学大纲）2009-12-24

Step 10: Reasoning
► *Think: I found a very relevant search result. The search for "网格计算" "教学大纲" "发布" "2009" yielded a page from the official 浙江大学computer science department website (cs.zju.edu.cn). The URL pattern (c26995a1202793) suggests it's a news or content page. The content explicitly states "2009年12月24日" which translates to December 24, 2009. This directly answers the user's question about the first upload date of the "网格计算" course taught by Professor Wu Zhao Hui. The date on the "教学大纲" (teaching syllabus) page is a strong indicator of when the course was first officially published/added to their teaching resources. Since the user specifically asked for the year, 2009 is the answer. I should confirm this is indeed the first upload.The URL pattern and the fact it's from the official domain suggest it's a primary source. I don't need to search further as the evidence is conclusive.* ***The answer is 2009.***

Step 10: Answer
浙江大学计算机学院吴朝晖教授讲授的《网格计算》课程的教学大纲于2009年12月24日首次发布在浙江大学计算机科学与技术学院的网站上。因此，该课程首次上传到学院教学资源中的年份是2009年。

N STRUCTURED HUMAN SEARCH BEHAVIOR QUESTIONNAIRE

In this section, we presents the questionnaire used to collect human search behaviors in complex, multi-step information-seeking tasks. The goal of the instrument is to understand how users actually initiate search, revise search direction, respond to incomplete or low-quality results, evaluate source credibility, and decide when they have gathered enough information. The questionnaire is used as a behavior-elicitation scaffold for subsequent qualitative coding and theme extraction, rather than as a direct hypothesis-testing instrument.

Instructions to participants. Please answer the following questions based on your *actual search habits* when handling complex information needs. There are no right or wrong answers. We are interested in what you typically do in practice. Unless otherwise specified, multiple-choice questions may have one or more reasonable answers, and open-ended questions should be answered as concretely as possible.

N.1 Part A: Background Information

A1. How often do you perform information-seeking tasks that require multiple rounds of search?

- Never or almost never
- Occasionally
- 1–2 times per week
- 3–5 times per week
- Almost every day

A2. In which scenarios do you most often conduct complex search tasks? (Select all that apply.)

- Academic research
- Work-related investigation
- News tracking or fact checking
- Travel or purchase decisions
- Technical troubleshooting
- Other: _____

A3. How would you rate your own search ability?

- 1 = Very weak
- 2 = Weak
- 3 = Moderate
- 4 = Strong
- 5 = Very strong

A4. Which of the following search methods have you used before? (Select all that apply.)

- Site-restricted search (e.g., `site:`)
- Title-constrained search (e.g, `intitle:`)
- Time range filtering
- Multi-keyword or Boolean-style search
- I rarely use advanced search techniques

N.2 Part B: Search Behavior in Specific Situations

Scenario 1: Starting a search for a complex question. Suppose you need to answer the following question:

> *Why has a country's stance on nuclear energy changed in recent years? What industrial, political, and social factors contributed to this shift?*

B1. If you were starting this task, what would your first step most likely be?

- Enter a short search query and inspect the types of results that appear
- Enter a more detailed query to narrow the results from the beginning
- Open several result pages first and then decide how to search further
- Think about the task structure before issuing a query
- Other: _____

B2. What would your first query most likely look like? Please write an example.

B3. When starting a search for a complex question, which of the following best describes your usual approach?

- I usually begin broadly and narrow down later

- I usually begin with a fairly specific query
- I often switch between broad and specific starts depending on the task
- I am not sure

B4. What factors usually influence how broad or specific your first query is?

Scenario 2: After seeing the first results. Suppose your first search returns several somewhat relevant results, but each one only covers part of the question.

B5. What would you most likely do next?

- Continue browsing the current result set
- Reformulate the query to focus on one missing aspect
- Open a new search for another aspect of the question
- Look for a broader overview source
- Other: _____

B6. When a question has multiple aspects, how do you usually organize your search process?

B7. In practice, how often do you search different aspects of the same question separately?

- Very often
- Sometimes
- Rarely
- Never

B8. What usually makes you decide to keep exploring the same search path versus open a new one?

Scenario 3: Low-recall, vague, or irrelevant results. Suppose that a query returns very few useful results, or that most results are not relevant to your question.

B9. What would you usually suspect first as the reason?

B10. Which of the following actions would you be most likely to take next? (Select all that apply.)

- Shorten or simplify the query
- Replace some words with different expressions
- Use more common or general terms
- Add a missing term that better specifies the topic
- Restrict results to a particular source or website
- Abandon the current angle and try a different path
- Other: _____

B11. When search results are poor, how do you usually decide whether to keep refining the same query or move to a different search direction?

B12. Think of a recent case where your original search did not work well. What did you do next?

Scenario 4: Evaluating and combining sources. Suppose you have collected several webpages, and some sources agree while others conflict.

B13. How do you usually judge whether a source is worth trusting?

B14. If different sources provide partially inconsistent information, what do you usually do?

- Prefer the source that seems most authoritative
- Look for additional sources
- Prefer the source that seems most detailed
- Prefer the source that best matches the question context
- Compare multiple sources before deciding
- Other: _____

B15. Before you feel ready to answer a complex question, how many distinct sources do you usually consult?

- Usually one
- Usually two to three
- Usually four or more
- It depends strongly on the task

B16. What usually makes you decide that you have searched enough and can stop?

Scenario 5: Precision-oriented search tools.

B17. In what situations do you use advanced search techniques such as `site:`, title constraints, or time filters?

B18. What do you usually hope to achieve when using such techniques?

- Narrow the search space
- Find more trustworthy sources
- Focus on a specific time range
- Locate a specific type of page
- I rarely use these techniques
- Other: _____

Optional decision task. Consider the following intermediate search situation:

> You searched for: `Germany nuclear policy change`
> The results include:
> - one page about policy history,
> - one page about energy company lobbying,
> - one page about public attitudes toward nuclear energy.

B19. What would you most likely do next?

- Continue browsing these results
- Search for one of the missing aspects in more detail
- Open separate searches for different aspects
- Rewrite the query more fully
- Other: _____

B20. Please briefly explain why.

N.3 Part C: Strategy Preference Items

For each statement below, please rate how well it describes your usual search behavior on a 1–5 scale, where 1 denotes *not at all like me* and 5 denotes *very much like me*.

C1. I often adjust later queries based on what earlier results reveal.

C2. For complex questions, I often consult more than one source before forming an answer.

C3. When a search does not work well, I often try a different wording.

C4. When a question has multiple aspects, I sometimes explore them through separate searches.

C5. I sometimes use advanced search techniques to narrow results or target specific sources.

C6. The way I start a search depends strongly on the type of task.

N.4 Part D: Open-ended Reflection

D1. Looking back on your own complex search tasks, what are the two or three things you do most often that help you make progress?

D2. In your experience, what are the most common reasons that search goes off track or becomes inefficient?

D3. If you were describing your own search habits to someone else, how would you summarize the way you usually search for complex questions?

Research usage. This questionnaire is used to surface recurring behavior patterns in complex search tasks. In our analysis pipeline, open-ended responses are first coded at the level of concrete behaviors, then grouped into higher-level themes, and finally abstracted into search behavior claims that can inform agent design and prompting.

N.5 Strategy Development

We collect self-reported search habits from participants with backgrounds in information retrieval and linguistics, including graduate students and professors. We combine these questionnaire responses with insights from research on search skills (Hirt et al., 2020) to inform the Exploration-Coverage-Synthesis paradigm.

The questionnaire provides qualitative input for strategy design. The resulting paradigm serves as a literature- and questionnaire-informed prior for improving LLM search strategies, rather than a characterization of universal human search behavior.