STEERSEARCH: Interactive Search Reasoning with Mid-Response Steering

Anonymous ACL submission

Abstract

Search-augmented language models are increasingly used as long-horizon research agents, yet most open training methods assume a fixed user goal throughout the entire search trajectory. We study *mid-response steering*, where a user intervenes during ongoing reasoning or retrieval and redirects the model toward a new goal. We identify *answer-target anchoring*, a failure mode in which fixed-query search agents retrieve evidence for the updated request but still answer an earlier question. To address this, we introduce STEERSEARCH, a reinforcement learning framework that trains models on interactive search trajectories with injected user interventions, encouraging context-preserving target reassignment. We also construct STEERSEARCH-BENCH, a benchmark adapted from multi-hop QA datasets to evaluate mid-response steering. Experiments on two instruction-tuned models show that STEERSEARCH substantially improves steering performance over Search-R1, raising average accuracy from 11.1% to 31.3% on Qwen2.5-3B and from 36.6% to 44.6% on Qwen3-4B, while preserving standard single-query search-reasoning performance.

1 Introduction

Search-augmented language models are increasingly being deployed as long-horizon research agents rather than one-shot question answerers (OpenAI, 2025; Anthropic, 2025; Google, 2025; Zheng et al., 2025; Wu et al., 2025a; Team et al., 2026). Systems such as ChatGPT Deep Research (OpenAI, 2025) plan, search, and synthesize information over multiple steps, shifting search-augmented reasoning toward extended interactive workflows. Recent ChatGPT interfaces (OpenAI, 2026) allow users to interrupt long-running responses and refine the request without restarting or losing progress. This emerging interaction pattern introduces a new requirement for search agents: they must continue from accumulated evidence while adapting to user instructions that arrive after reasoning or retrieval has already begun.

However, most open search-reasoning training methods still assume a fixed-goal trajectory (Jin et al., 2025b; Chen et al., 2025; Zheng et al., 2025; Wu et al., 2025a; Li et al., 2026; Wu et al., 2025b). Methods such as Search-R1 (Jin et al., 2025b) and ReSearch (Chen et al., 2025) train language models to interleave reasoning with search, allowing retrieved evidence to influence subsequent generations, but the user goal is typically fixed at the start of the rollout. Even when multiple searches are allowed, the model is optimized to answer the original question rather than to handle a new user turn that changes the operative target during generation.

This mismatch gives rise to a distinct failure mode for interactive research agents, which we call *answer-target anchoring*. When a model trained under a fixed-goal trajectory is placed in an interrupted interaction setting, it may retrieve evidence that is sufficient for the user's updated request, but still produce an answer to an earlier question because its reasoning state remains anchored to the original target. For example, after identifying the Harlem Globetrotters as the team Louis Dunbar played for, a user may interrupt with a dependent follow-up asking how many countries that team has played in. Answering the follow-up requires preserving the intermediate entity from the first phase, but also reassigning the answer target from "which team?" to "how many countries?". Standard search-reasoning RL provides little direct supervision for this kind of context-preserving goal shift.

In this work, we introduce STEERSEARCH, a reinforcement learning framework for mid-response steering in search-augmented reasoning, as illustrated in Figure 1. Rather than optimizing only uninterrupted fixed-goal trajectories, STEERSEARCH trains models on interactive trajectories where user interventions are injected during ongoing search

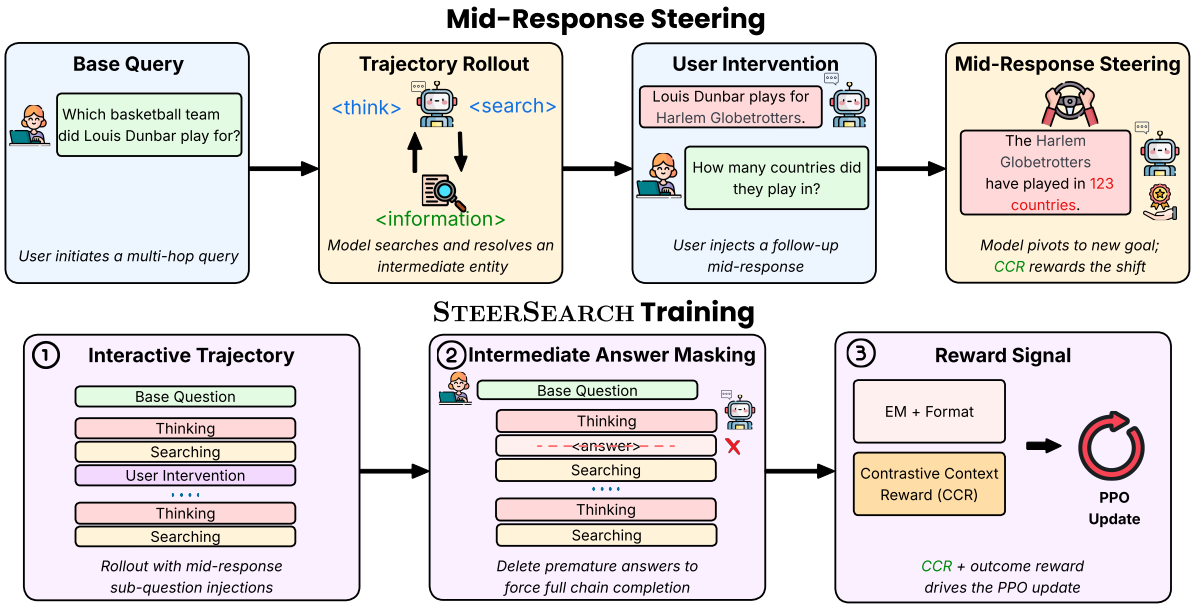

Figure 1: **Overview of STEERSEARCH. Top**: During inference, the model receives a base query, conducts a search-and-reason trajectory, and is interrupted mid-response with an injected sub-question. It must pivot its answer target while preserving all prior retrieved evidence. **Bottom**: STEERSEARCH teaches the model with this behavior by (A) rolling out interactive trajectories with mid-response injections, (B) deleting premature answers in non-final phases to prevent shortcutting, and (C) rewarding the policy with standard outcome signals plus a Contrastive Context Reward (CCR) that measures whether the model's reasoning shifts toward the newly injected sub-question.

and reasoning. The model must preserve useful evidence from earlier steps while reassigning its answer target to the latest user goal. To support this behavior, STEERSEARCH combines randomized intervention timing, intermediate-answer masking, and an optional Contrastive Context Reward (CCR) that encourages the model's reasoning to pivot toward newly injected user goals.

To evaluate this setting, we construct STEERSEARCH-BENCH, a mid-response steering benchmark adapted from HotpotQA, 2WikiMultiHopQA, MuSiQue, and Bamboogle, where each original multi-hop question is converted into a sequence of injected user interventions. Experiments on two instruction-tuned models show that STEERSEARCH substantially improves mid-response steering: average accuracy rises from 11.1 to 31.3 on Qwen2.5-3B and from 36.6 to 44.6 on Qwen3-4B over Search-R1. At the same time, standard single-query search-reasoning performance is preserved.

Our contributions are as follows:

- We formulate *mid-response steering* for search-augmented reasoning and identify *answer-target anchoring*, a failure mode where fixed-goal search agents retrieve evidence for an updated request but still answer an earlier question.
- We propose STEERSEARCH, an RL framework that trains models on interactive search trajectories with injected user interventions, encouraging context-preserving target reassignment during generation.
- We introduce STEERSEARCH-BENCH, a benchmark for evaluating whether search agents can follow mid-response user interventions while reusing accumulated evidence.
- We show that STEERSEARCH improves average accuracy on STEERSEARCH-BENCH by 8.0-20.2% over Search-R1, while maintaining or slightly improving performance on standard search-reasoning benchmarks.

2 Related Work

Search-augmented reasoning with reinforcement learning. Recent work trains language models to interleave reasoning with external search. Search-R1 (Jin et al., 2025b) and ReSearch (Chen et al., 2025) formulate retrieval as part of the model's action space, allowing the policy to issue search queries, read returned evidence, and continue reasoning before producing an answer. Follow-up work shows that search-reasoning agents are sensitive to reward design, initialization, and retrieval environments (Jin et al.,

2025a), while related RL-based approaches extend search reasoning to multimodal search and long-horizon web research (Wu et al., 2025b; Narayan et al., 2025; Zheng et al., 2025; Li et al., 2026). However, these methods typically optimize uninterrupted trajectories toward a fixed user goal, whereas STEERSEARCH studies search-reasoning after a new user turn arrives mid-response.

Multi-hop reasoning and query decomposition. Multi-hop question answering benchmarks evaluate whether models can combine evidence across multiple facts or documents. HotpotQA (Yang et al., 2018), 2WikiMultiHopQA (Ho et al., 2020), and MuSiQue (Trivedi et al., 2022) construct questions whose answers require linked reasoning steps, often with supporting evidence or structured reasoning paths. Prompting methods such as Self-Ask (Press et al., 2023) encourage models to decompose a question into intermediate sub-questions, and recent training approaches such as ParallelSearch (Zhao et al., 2025) learn to decompose complex queries into sub-queries that can be searched independently. Unlike static query decomposition, STEERSEARCH treats sub-questions as user interventions injected into an ongoing trajectory, requiring the model to incorporate the new goal without discarding prior context.

Interactive and long-horizon research agents. Recent systems move search-augmented language models toward long-horizon research workflows. Deployed systems such as ChatGPT Deep Research, Gemini Deep Research, and Claude Research browse, inspect sources, and synthesize findings over extended interactions (OpenAI, 2025; Google, 2025; Anthropic, 2025). Web-agent benchmarks further evaluate models' ability to navigate, gather information, and solve tasks across multiple steps (Mialon et al., 2024; Wei et al., 2025; Zhou et al., 2025). Recent ChatGPT interfaces also expose mid-response course adjustment, where users can refine a request while a long-running response is still being produced (OpenAI, 2026).

Together, these systems and benchmarks motivate a training problem not captured by standard search-reasoning evaluations: agents must remain responsive to user steering after reasoning and retrieval have already begun.

3 Preliminary

In this section, we first formulate the problem of mid-response steering. Then we introduce an issue, *answer-target anchoring*, that we observe when applying the model trained with Search-R1 (Jin et al., 2025b) to tasks requiring user's interruption.

3.1 Mid-Response Steering Formulation

We study search-augmented reasoning in an interactive setting where the user may intervene while the model is still solving a task. Unlike standard single-query search, where the model receives one fixed question and optimizes a continuous trajectory toward a single answer target, mid-response steering allows new user instructions to be injected after partial reasoning and retrieval have already occurred. The model must therefore continue from the accumulated context while adapting its answer target to the latest user goal.

Formally, let $u = \langle u_1, u_2, \ldots, u_K \rangle$ be an ordered sequence of user goals, where u_1 is the initial query and each u_k for $k > 1$ is a mid-response intervention. Let R be a retriever and π_θ be a language-model policy. After receiving user goal u_k, the model produces a sequence of model moves

$$m_{k,1}, m_{k,2}, \ldots, m_{k,T_k},$$

where each move may contain reasoning tokens and either a search query or an answer. If $m_{k,t}$ contains a search query, the retriever returns an observation $o_{k,t} = R(m_{k,t})$, which is appended to the context before the next model move. If no search is issued, $o_{k,t}$ is empty.

The full interactive trajectory is written as

$$\tau = \left\langle u_k, \{(m_{k,t}, o_{k,t})\}_{t=1}^{T_k} \right\rangle_{k=1}^{K}.$$

Crucially, for $k < K$, the next user intervention u_{k+1} need not wait for the model to finish answering u_k. Instead, it may be injected at an interruption point during the ongoing response to u_k, after the model has already produced some reasoning, search actions, or retrieved observations. This models a user redirecting a long-running search process before completion. User interventions u_k and retriever observations $o_{k,t}$ are supplied by the environment, while model moves $m_{k,t}$, including reasoning tokens, search queries, and answer tokens, are generated by the policy.

The objective is to learn a policy that answers the latest user goal while conditioning on the full interaction history. We define the mid-response steering objective as

$$\max_{\pi_\theta} \mathbb{E}_{(u,a^*)\sim\mathcal{D},\, \tau\sim\pi_\theta(\cdot|u,R)} \left[r(a^*, \tau) \right],$$

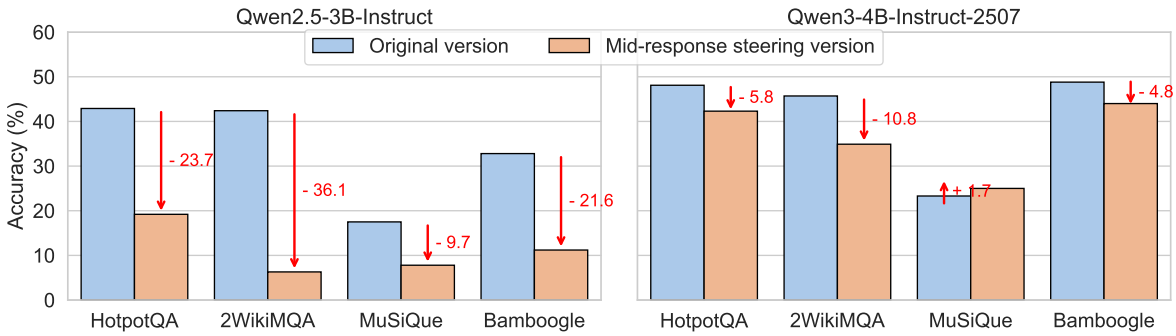

Figure 2: **LLMs performance drop on mid-response steering.** Performance of LLMs trained only with single-query search on the original multi-hop questions and their mid-response steering variants. The steering variants decompose each original single query into a sequence of injected user queries, simulating user's interruptions.

where a^* is the gold answer to the terminal user goal u_K, and $r(a^*, \tau)$ rewards the model for producing a valid final answer to that goal under the accumulated interaction context.

The key requirement is *context-preserving target reassignment*: after receiving u_{k+1}, the model should reuse relevant prior context, such as intermediate entities or retrieved facts, while treating the new intervention as the operative goal. It should neither restart from scratch nor continue answering the earlier query. This distinguishes mid-response steering from standard single-query search. In single-query search, the answer target is fixed and all evidence is interpreted with respect to the initial question. In mid-response steering, later user turns can change the target before the previous response is complete, so earlier evidence may become a dependency for a new goal and the final answer should address the latest intervention.

3.2 Answer-Target Anchoring

A natural question is whether models trained only for single-query search can directly handle mid-response steering. To test this, we evaluate a Search-R1-style model on both the original multi-hop questions and their mid-response steering variants, where the original question is decomposed into a sequence of user queries injected during generation. As shown in Figure 2, performance mostly drops substantially under mid-response steering, with an accuracy decrease of up to 36.1%. This suggests that strong single-query search ability does not automatically transfer to settings where the user goal changes during an ongoing trajectory.

We identify a common failure mode behind this degradation, which we call *answer-target anchoring*. As illustrated in Figure 3 (left), after a new user query is injected, the model may correctly reason about the new query and even issue a search that retrieves the necessary evidence. However, when producing the final answer, it reverts to the answer target of the base query rather than the latest user query. In other words, the model updates its local retrieval behavior but fails to update its global answer objective.

This failure is distinct from a standard retrieval error. The required evidence may already appear in the trajectory, but the model remains anchored to the earlier question when deciding what to answer. Mid-response steering therefore requires more than retrieval: the model must perform *context-preserving target reassignment*, reusing earlier evidence while treating the latest user intervention as authoritative.

This phenomenon is related in spirit to long-context robustness issues such as *lost-in-the-middle* (Liu et al., 2024; Chen et al., 2024; Liao et al., 2025), where models fail to use relevant information depending on its position in the context. However, answer-target anchoring is specific to interactive search trajectories: the challenge is not only locating useful information in a long context, but also updating the operative answer target after a user interruption. This motivates training models directly on trajectories that contain mid-response steering events, rather than relying on single-query search training alone.

4 STEERSEARCH

Motivated by answer-target anchoring (Section 3.2) in single-query search models under user intervention, we present STEERSEARCH, a reinforcement learning framework for context-preserving target reassignment under mid-response steering.

4.1 Overview

STEERSEARCH trains the policy on the interactive multi-hop trajectory formulation defined in Sec-

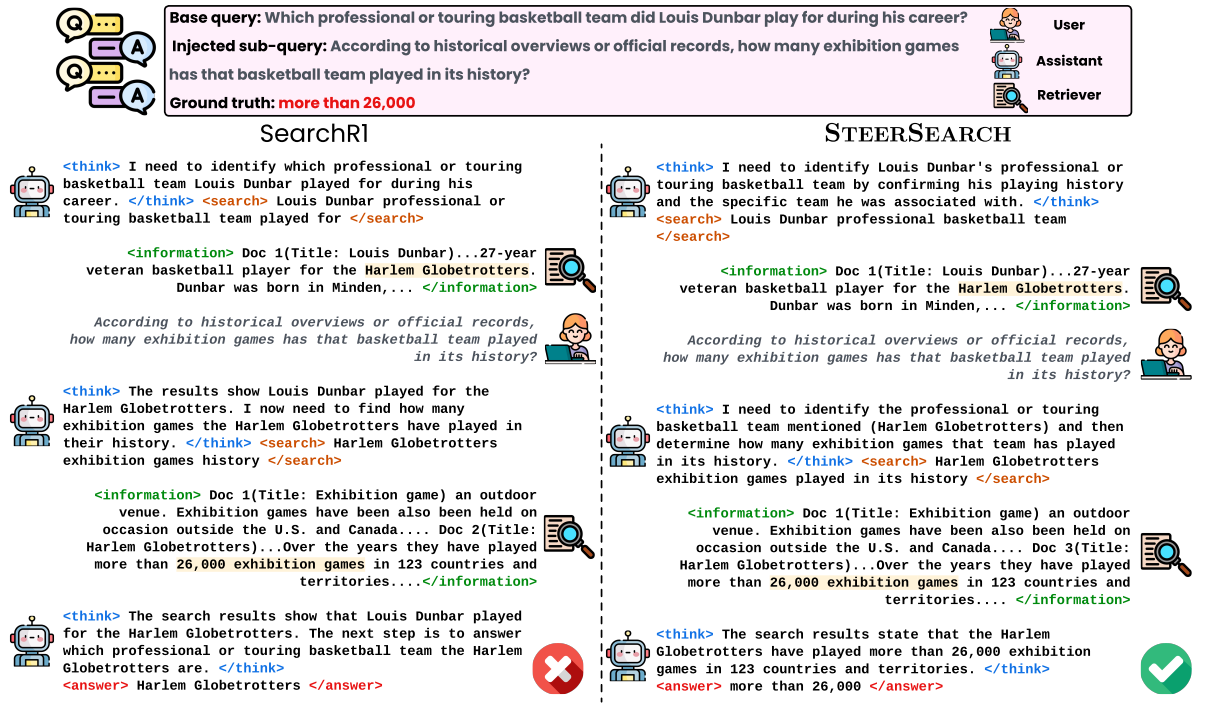

Figure 3: **Comparison of Search-R1 and STEERSEARCH on the same mid-response steering trajectory.** Both retrieve evidence containing the updated answer, but Search-R1 remains anchored to the base question and answers with the intermediate entity. STEERSEARCH treats the injection as operative goal and produces the correct answer.

tion 3.1, exposing it to the inference-time setting where a user redirects an ongoing search before the prior response is complete. Its key departures from standard single-query search training are: (1) *randomized phase budgets*, which vary injection timing; (2) *intermediate answer removal*, which masks premature answers in non-final phases; and (3) a *Contrastive Context Reward* (Section 4.2), which adds dense per-phase guidance to the sparse terminal signal. Together, these choices train context-preserving target reassignment as a first-class skill rather than a side effect of single-query optimization. We show an overview of STEERSEARCH in Figure 1, and the full rollout procedure in Algorithm 1 (Appendix C.2).

4.2 Contrastive Context Reward

The terminal outcome reward used in standard search-augmented RL provides only sparse supervision: it indicates whether the final answer is correct, but does not directly measure whether the model updates its reasoning after a mid-response intervention. As shown in Figure 3 (left), a model may generate plausible search queries after receiving u_{k+1}, yet still produce an answer anchored to an earlier user goal. To provide denser guidance, we introduce a *Contrastive Context Reward* (CCR), which measures whether the model's reasoning after an intervention has shifted toward the updated user-goal context.

Let $\phi(\cdot)$ denote a frozen text encoder that maps text to a normalized embedding, and let

$$\mathrm{sim}(x, y) = \phi(x)^\top \phi(y)$$

be cosine similarity. For each intervention phase $k+1$, the user goal u_{k+1} is injected while the model is still responding to the previous goal. The model then produces moves $\{m_{k+1,t}\}_{t=1}^{T_{k+1}}$ before either answering or exhausting the phase budget. We sample a timing point

$$t^\star_{k+1} \sim \mathcal{U}\{1, \ldots, T_{k+1}\}$$

and extract the reasoning span

$$h_{k+1} = \mathrm{Think}\left(m_{k+1,t^\star_{k+1}}\right),$$

where $\mathrm{Think}(\cdot)$ extracts the text inside the `<think>...</think>` block of a model move. Thus, h_{k+1} captures the model's reasoning at a random point after the intervention but before the final answer to the current goal.

We then compare the alignment of h_{k+1} with two contexts: a positive context that includes the new intervention, and a negative context that excludes it. We define

$$\Delta_{k+1} = \mathrm{sim}(h_{k+1}, p_{k+1}) - \mathrm{sim}(h_{k+1}, n_k).$$

A positive value indicates that the model's reasoning has shifted toward the updated user goal, while a negative value suggests persistence toward the previous goal.

We consider two choices for (p_{k+1}, n_k), corresponding to local and history-aware notions of steering. The *individual* variant compares only adjacent interventions:

$$p_{k+1} = u_{k+1}, \qquad n_k = u_k.$$

It provides a sharp signal when each intervention is self-contained and has a clear retrieval target. The *cumulative* variant compares accumulated user-goal context before and after the new intervention:

$$p_{k+1} = u_1 \oplus \cdots \oplus u_k \oplus u_{k+1}, \quad n_k = u_1 \oplus \cdots \oplus u_k,$$

where $\oplus$ denotes string concatenation. This is useful when later interventions are co-referential or depend on entities introduced earlier. We use the *individual* variant with *last-think* assignment as the default configuration reported in Tables 1 and 2, selected based on the ablation in Appendix E.

The scalar ccr term is obtained by scaling and bounding the raw score:

$$r_{k+1}^{\text{ccr}} = \tanh(\lambda\, \Delta_{k+1})\,.$$

where λ controls the sharpness of the contrast. We compute CCR for all intervention phases $k = 1, \ldots, K-1$ and add it to the standard outcome and format reward:

$$r_{\text{total}}(\tau, a^*) = r(\tau, a^*) + w_{\text{ccr}} \sum_{k=1}^{K-1} r_k^{\text{ccr}}.$$

where w_{ccr} controls the CCR weight.

5 Experiments

Models. We train two models: Qwen2.5-3B-Instruct (Yang et al., 2025b) and Qwen3-4B-Instruct-2507 (Yang et al., 2025a).

Baselines. Our primary baseline is Search-R1 (Jin et al., 2025b), trained for fixed-query iterative reasoning and retrieval. We use the same base models, SFT initialization, retriever, PPO objective, and reward signal, but omit mid-response interventions, isolating the effect of STEERSEARCH's intervention-based training.

We also report the untuned base model as Vanilla. On standard search-reasoning benchmarks, we include representative baselines from Jin et al. (2025b), with direct inference, CoT, IR-CoT (Trivedi et al., 2023), Search-o1 (Li et al., 2025), RAG, SFT, DeepSeek-R1-Instruct (Guo et al., 2025), and rejection sampling (Ahn et al., 2024). Search-R1 remains the main controlled baseline for mid-response steering.

Training sets. Training consists of supervised fine-tuning (SFT) followed by reinforcement learning (RL). For SFT, both models are trained on 4K GPT-5.4-generated HotpotQA trajectories (OpenAI, 2026), split equally between Search-R1 and STEERSEARCH formats. Only assistant tokens are trained, and the rest tokens are masked.

For STEERSEARCH RL, we curate 24K new training samples from the multi-hop HotpotQA to simulate the mid-response steering task. We decompose the original multi-hop questions into sub-question chains. These chains are generated with three strategies—*decomposition*, *hint steering*, and *disambiguation*—and filtered by a three-judge GPT-5.4 committee for answer leakage, malformed queries, and bundled questions. Together with 16K single-query samples, we train with STEERSEARCH on a total of 40K samples. We detail sub-question generation process in A.2.

For Search-R1 training, we only use 40K single-query samples drawn equally from HotpotQA and NQ, following Jin et al. (2025b). Full data composition and statistics are provided in Appendix B.

Evaluation sets. We evaluate on two benchmark suites. First, we use seven standard search-reasoning benchmarks: NQ (Kwiatkowski et al., 2019), TriviaQA (Joshi et al., 2017), PopQA (Mallen et al., 2023), HotpotQA, 2WikiMultiHopQA, MuSiQue, and Bamboogle, all containing a single query. This setting measures whether mid-response steering training preserves general retrieval-augmented QA ability.

Second, we evaluate on our new STEERSEARCH-BENCH. This benchmark is originated from HotpotQA, 2WikiMultiHopQA, MuSiQue, and Bamboogle. To simulate mid-response steering, we construct sub-question chains from them, acting as user's intervention. The final answer stays the same as the original sample. And we only measure the final accuracy as Jin et al. (2025b). More details about dataset construction are shown in A.2.

Implementation details. For SFT, we train up to 600 steps with batch size 32, learning rate 2×10^{-6}, cosine decay, and warmup ratio 0.025. For RL, both STEERSEARCH and Search-R1 are optimized with PPO (Schulman et al., 2017) us-

Method	General QA			Multi-Hop QA				Avg.
	NQ†	TriviaQA*	PopQA*	HotpotQA†	2WikiMQA*	MuSiQue*	Bamboogle*	
Qwen2.5-3B-Instruct								
Vanilla	0.216	0.415	0.259	0.188	0.175	0.044	0.168	0.209
CoT	0.023	0.032	0.005	0.021	0.021	0.002	0.000	0.015
IRCoT	0.111	0.312	0.200	0.164	0.171	0.067	0.240	0.181
Search-o1	0.238	0.472	0.262	0.221	0.218	0.054	0.320	0.255
RAG	0.348	0.544	0.387	0.255	0.226	0.047	0.080	0.270
SFT	0.249	0.292	0.104	0.186	0.248	0.044	0.112	0.176
R1-instruct	0.210	0.449	0.171	0.208	0.275	0.060	0.192	0.224
Rejection Sampling	0.294	0.488	0.332	0.240	0.233	0.059	0.210	0.265
Search-R1	0.447	0.602	0.433	0.429	**0.424**	0.175	0.328	0.405
STEERSEARCH	0.450	0.615	0.442	**0.442**	0.409	**0.209**	0.384	**0.422**
STEERSEARCH+CCR	**0.456**	**0.617**	**0.444**	0.426	0.413	0.163	**0.416**	0.419
Qwen3-4B-Instruct-2507								
Vanilla	0.283	0.497	0.317	0.244	0.270	0.060	0.376	0.292
Search-R1	**0.496**	0.660	**0.487**	0.481	0.457	0.233	0.488	0.472
STEERSEARCH	0.491	0.652	0.479	0.475	**0.498**	**0.239**	0.480	**0.473**
STEERSEARCH+CCR	0.480	0.649	0.469	0.490	0.488	0.226	0.472	0.468

Table 1: **Exact match on standard Search-Reasoning benchmarks.** With † in-domain, ∗ out-of-domain, **Bold** = best per model among fine-tuned methods, and Vanilla = not any training. STEERSEARCH denotes training the model on our new mid-response steering training set with only the final answer reward. And STEERSEARCH+CCR applies the contrastive context reward on top of it.

Method	HotpotQA†	2WikiMQA*	MuSiQue*	Bamboogle*	Avg.
Qwen2.5-3B-Instruct					
Vanilla	0.101	0.054	0.039	0.080	0.068
Search-R1	0.192	0.063	0.078	0.112	0.111
STEERSEARCH	0.376	0.260	0.241	0.376	0.313
STEERSEARCH+CCR	**0.429**	**0.361**	**0.268**	**0.480**	**0.384**
Qwen3-4B-Instruct-2507					
Vanilla	0.171	0.173	0.068	0.096	0.127
Search-R1	0.423	0.349	0.250	0.440	0.366
STEERSEARCH	**0.472**	**0.457**	**0.305**	**0.552**	**0.446**
STEERSEARCH+CCR	0.471	0.446	0.300	0.480	0.424

Table 2: **Performance on STEERSEARCH-BENCH**, a new benchmark for mid-response steering.

ing `verl` (Sheng et al., 2025). We use batch size 512, PPO mini-batch size 256, per-GPU micro-batch size 8, AdamW with learning rate 1×10^{-6}, warmup ratio 0.285, maximum response length 1,024, temperature 1.0, and KL coefficient 0.001. Models are trained for 600 steps on 8 A100 GPUs and evaluated every 200 steps.

STEERSEARCH rollouts follow Algorithm 1. Each phase allows up to T_k search-reasoning turns, with $T_k = 5$ for non-final phases and $T_K = 4$ for the final phase. Each turn has `<think>` followed by either `<search>` or `<answer>`, with retrieved passages appended as `<information>`. Non-final premature answers are masked before injecting u_{k+1}; if the final phase produces no answer, one additional search-disabled generation pass is used. Refer to Appendix A for more details.

5.1 Main Results

Performance on single-query tasks. In Table 1, we show the performance of models trained with STEERSEARCH on the single-query tasks. We can observe that training on our new constructed mid-response steering samples doesn't degrade the models' performance on the standard single-query search tasks, even with a slight improvement. The absence of regression confirms that exposure to mid-response steering during training does not interfere with the model's ability to conduct iterative search when left uninterrupted.

Performance on STEERSEARCH-BENCH. In Table 2, we further show the models' performance on our newly constructed mid-response steering benchmark, STEERSEARCH-BENCH. STEERSEARCH improves over Search-R1 by 8.0-20.2% on average for both models, with gains consistent across all four datasets. It shows that training on the mid-response steering tasks is necessary for teaching the model to have such a capability.

Case study. From Table 2, we also observe that Search-R1 only has a modest improvement over the Vanilla baseline. This is mainly caused by the

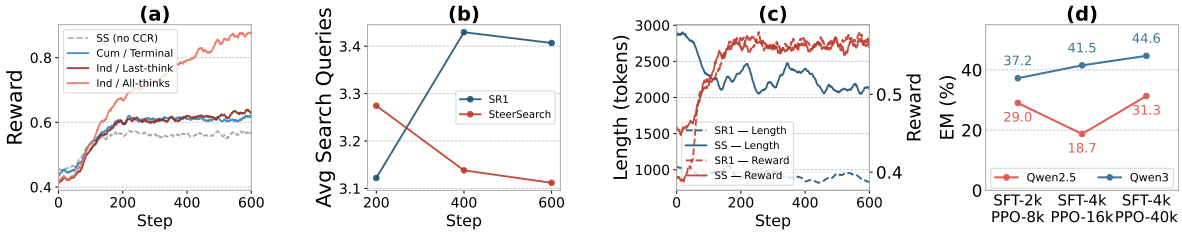

Figure 4: **Training dynamics and data scaling.** **(a)** Reward trajectories for CCR ablations (Qwen2.5). **(b)** Avg search queries at validation (Qwen3); SR1 rises, SS drops. **(c)** Response length (blue) and reward (red) for Qwen2.5. **(d)** STEERSEARCH-BENCH EM across training data budgets for both models.

answer-target anchoring issue, as shown in Figure 3 (left). With the new user query injection, Search-R1 can reason and search for it. However, it still provide answer to the base query. I.e., Search-R1 treats injected query as irrelevant context, and continues reasoning toward the answer to the original base question. By contrast, STEERSEARCH is trained to treat each injection as a new operative goal, pivoting its retrieval and reasoning accordingly, as shown in Figure 3 (right).

Effect of contrastive context reward. Similar to STEERSEARCH, the introduction of CCR maintains a similar performance as Search-R1 on the single-query tasks, as shown in Table 1. In Table 2, we see a model-dependent effect of CCR. CCR improves the performance by 7.1% on average for Qwen2.5-3B-Instruct, while drops 2.2% for Qwen3-4B-Instruct-2507. We argue that Qwen3 series are more heavily optimized with RL than Qwen2.5 series, down-weighting the effect of CCR when we further optimize Qwen3 with RL.

5.2 Ablation Studies

CCR setting. We ablate CCR across context variants (individual vs. accumulated) and reward assignment modes (last-think, terminal, and individual all-thinks). As shown in Figure 4(a), most variants train stably near the no-CCR STEERSEARCH baseline, while individual all-thinks shows a late reward spike, suggesting sensitivity to dense gradients. Since evaluation gains do not mirror training reward, they indicate improved reasoning alignment rather than reward inflation. We use individual last-think as the default STEERSEARCH+CCR setting in Tables 1 and 2. Full ablations are in Appendix E.

Training data scale. Figure 4(d) shows how STEERSEARCH-BENCH performance scales with the training pool size across three configurations (SFT-2k/PPO-8k, SFT-4k/PPO-16k, SFT-4k/PPO-40k). On the standard benchmark both models improve consistently as the data budget grows. On STEERSEARCH-BENCH, Qwen3 scales smoothly across all three settings, while Qwen2.5 dips at the intermediate budget before recovering at the largest pool, suggesting the smaller model requires more data to reliably internalize the multi-phase injection curriculum.

Training dynamics. Figure 4(b,c) shows complementary training trends. On Qwen3, Search-R1 issues more queries as training progresses, while STEERSEARCH issues fewer, suggesting it learns to allocate search more selectively across injection phases. STEERSEARCH also produces longer trajectories due to multi-phase reasoning, but response length decreases as reward improves, indicating that gains are not driven by verbosity. Figure 4(d) shows how STEERSEARCH-BENCH performance scales with training data budget.

6 Conclusion

We introduced STEERSEARCH, a reinforcement learning framework for mid-response steering in search-augmented reasoning. We showed that fixed-query search agents suffer from answer-target anchoring, where they retrieve evidence for a new user goal but still answer an earlier question. By training on interactive trajectories with injected user interventions, STEERSEARCH learns context-preserving target reassignment, substantially improving performance on STEERSEARCH-BENCH while preserving standard search-reasoning ability.

Limitations

This work focuses on mid-response steering for text-based search-augmented reasoning. Our experiments use Wikipedia-style retrieval and multi-hop QA benchmarks adapted into steering trajectories, which provide controlled evaluation but do not cover every form of open-ended web research. Extending STEERSEARCH to noisier web environments, multimodal retrieval, and longer interactive

sessions is a natural next step.

STEERSEARCH improves the model's ability to update its answer target after user interventions, but it does not solve all remaining reasoning errors. Some failures still require fine-grained role disambiguation, exact comparison, or synthesis across partially retrieved evidence. These limitations are orthogonal to the mid-response steering setting and can be addressed with stronger retrieval, verification, or answer validation modules.

Ethical Considerations

This work studies how search-augmented language models respond to user interventions during ongoing reasoning. The proposed training data and benchmark are derived from public QA datasets and Wikipedia-style retrieval, without collecting private user interactions. However, systems that support mid-response steering may be used in open-ended research settings where retrieved content can contain misinformation, bias, or sensitive information. Improving steering behavior therefore does not remove the need for source verification, citation, and safety filtering in deployed systems.

Because STEERSEARCH trains models to follow newly injected user goals, care should also be taken when applying it to unsafe or policy-violating requests. Future deployments should combine mid-response steering with existing safeguards for tool use, retrieval access, and refusal behavior.

References

Janice Ahn, Rishu Verma, Renze Lou, Di Liu, Rui Zhang, and Wenpeng Yin. 2024. Large language models for mathematical reasoning: Progresses and challenges. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics: Student Research Workshop*, pages 225–237, St. Julian's, Malta. Association for Computational Linguistics.

Anthropic. 2025. How we built our multi-agent research system. https://www.anthropic.com/engineering/multi-agent-research-system. Accessed May 25, 2026.

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. 2025. Research: Learning to reason with search for llms via reinforcement learning. *Preprint*, arXiv:2503.19470.

Xinyi Chen, Baohao Liao, Jirui Qi, Panagiotis Eustratiadis, Christof Monz, Arianna Bisazza, and Maarten de Rijke. 2024. The SIFo benchmark: Investigating the sequential instruction following ability of large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 1691–1706, Miami, Florida, USA. Association for Computational Linguistics.

Google. 2025. Gemini deep research. https://gemini.google/overview/deep-research/. Accessed May 25, 2026.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, and 175 others. 2025. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nature*, 645(8081):633–638.

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

Bowen Jin, Jinsung Yoon, Priyanka Kargupta, Sercan O. Arik, and Jiawei Han. 2025a. An empirical study on reinforcement learning for reasoning-search interleaved llm agents. *Preprint*, arXiv:2505.15117.

Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan Arik, Dong Wang, Hamed Zamani, and Jiawei Han. 2025b. Search-r1: Training llms to reason and leverage search engines with reinforcement learning. *Preprint*, arXiv:2503.09516.

Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. 2017. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611, Vancouver, Canada. Association for Computational Linguistics.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466.

Xiaoxi Li, Guanting Dong, Jiajie Jin, Yuyao Zhang, Yujia Zhou, Yutao Zhu, Peitian Zhang, and Zhicheng Dou. 2025. Search-o1: Agentic search-enhanced large reasoning models. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 5420–5438, Suzhou, China. Association for Computational Linguistics.

Zhuofeng Li, Dongfu Jiang, Xueguang Ma, Haoxiang Zhang, Ping Nie, Yuyu Zhang, Kai Zou, Jianwen

Xie, Yu Zhang, and Wenhu Chen. 2026. Openresearcher: A fully open pipeline for long-horizon deep research trajectory synthesis. *arXiv preprint arXiv:2603.20278*.

Baohao Liao, Xinyi Chen, Sara Rajaee, Yuhui Xu, Christian Herold, Anders Søgaard, Maarten de Rijke, and Christof Monz. 2025. Lost at the beginning of reasoning. *arXiv preprint arXiv:2506.22058*.

Nelson F Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2024. Lost in the middle: How language models use long contexts. *Transactions of the association for computational linguistics*, 12:157–173.

Alex Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Daniel Khashabi, and Hannaneh Hajishirzi. 2023. When not to trust language models: Investigating effectiveness of parametric and non-parametric memories. *Preprint*, arXiv:2212.10511.

Grégoire Mialon, Clémentine Fourrier, Thomas Wolf, Yann LeCun, and Thomas Scialom. 2024. Gaia: a benchmark for general ai assistants. In *International Conference on Learning Representations*, volume 2024, pages 9025–9049.

Kartik Narayan, Yang Xu, Tian Cao, Kavya Nerella, Vishal M. Patel, Navid Shiee, Peter Grasch, Chao Jia, Yinfei Yang, and Zhe Gan. 2025. Deepmmsearch-r1: Empowering multimodal llms in multimodal web search. *Preprint*, arXiv:2510.12801.

OpenAI. 2025. Deep research system card. https://deploymentsafety.openai.com/deep-research. Accessed May 25, 2026.

OpenAI. 2026. Introducing gpt-5.4. https://openai.com/index/introducing-gpt-5-4/. Accessed May 25, 2026.

Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah A. Smith, and Mike Lewis. 2023. Measuring and narrowing the compositionality gap in language models. *Preprint*, arXiv:2210.03350.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *Preprint*, arXiv:1707.06347.

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. 2025. Hybridflow: A flexible and efficient rlhf framework. In *Proceedings of the Twentieth European Conference on Computer Systems*, EuroSys '25, page 1279–1297. ACM.

MiroMind Team, S. Bai, L. Bing, L. Lei, R. Li, X. Li, X. Lin, E. Min, L. Su, B. Wang, L. Wang, L. Wang, S. Wang, X. Wang, Y. Zhang, Z. Zhang, and 1 others. 2026. Mirothinker-1.7 & h1: Towards heavy-duty research agents via verification. *arXiv preprint arXiv:2603.15726*.

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

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2023. Interleaving retrieval with chain-of-thought reasoning for knowledge-intensive multi-step questions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10014–10037, Toronto, Canada. Association for Computational Linguistics.

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

Jialong Wu, Wenbiao Yin, Yong Jiang, Zhenglin Wang, Zekun Xi, Runnan Fang, Linhai Zhang, Yulan He, Deyu Zhou, Pengjun Xie, and Fei Huang. 2025a. WebWalker: Benchmarking LLMs in web traversal. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10290–10305, Vienna, Austria. Association for Computational Linguistics.

Jinming Wu, Zihao Deng, Wei Li, Yiding Liu, Bo You, Bo Li, Zejun Ma, and Ziwei Liu. 2025b. Mmsearch-r1: Incentivizing lmms to search. *Preprint*, arXiv:2506.20670.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, and 41 others. 2025a. Qwen3 technical report. *Preprint*, arXiv:2505.09388.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, and 23 others. 2025b. Qwen2.5 technical report. *Preprint*, arXiv:2412.15115.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium. Association for Computational Linguistics.

Shu Zhao, Tan Yu, Anbang Xu, Japinder Singh, Aaditya Shukla, and Rama Akkiraju. 2025. Parallelsearch: Train your llms to decompose query and search sub-queries in parallel with reinforcement learning. *Preprint*, arXiv:2508.09303.

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

Peilin Zhou, Bruce Leon, Xiang Ying, Can Zhang, Yifan Shao, Qichen Ye, Dading Chong, Zhiling Jin, Chenxuan Xie, Meng Cao, and 1 others. 2025. Browsecomp-zh: Benchmarking web browsing ability of large language models in chinese. *arXiv preprint arXiv:2504.19314*.

A Experimental Setup

A.1 Training Configuration

We train two base models: Qwen2.5-3B-Instruct (Yang et al., 2025b) and Qwen3-4B-Instruct-2507 (Yang et al., 2025a). Qwen3-4B-Instruct-2507 is used in non-thinking mode, where the built-in chain-of-thought pre-reasoning is disabled and reasoning is performed inside explicit `<reason>` tags following the STEERSEARCH prompt format.

SFT initialization. Both models are first fine-tuned on 4,000 gold-standard trajectories generated by GPT-5.4 (OpenAI, 2026) over HotpotQA, split equally between Search-R1 and STEERSEARCH formats. The objective is standard next-token prediction on assistant tokens only; retrieved passages and injected sub-question turns are masked from the loss. Training runs for up to 600 steps with a batch size of 32, learning rate 2×10^{-6}, cosine decay, and a warmup ratio of 0.025. The resulting best SFT checkpoint on validation loss is used to initialize all RL experiments.

STEERSEARCH. Both Qwen2.5-3B-Instruct and Qwen3-4B-Instruct-2507 are trained with STEERSEARCH rollout described in Section 4, where sub-questions are injected mid-response, with per-phase search budgets sampled uniformly from $[1, 5]$ turns for base and middle phases and fixed at 4 turns for the final phase. PPO (Schulman et al., 2017) training is implemented using the `verl` (Sheng et al., 2025) framework, with a train batch size of 512, PPO mini-batch size of 256, and a per-GPU micro-batch size of 8. The actor is optimized with AdamW at a learning rate of 1×10^{-6} with a warmup ratio of 0.285. Rollouts are generated using vLLM with a maximum response length of 1,024 tokens and temperature 1.0. Actor, reference, and critic models use FSDP2 with parameter and optimizer offloading. A KL penalty with coefficient 0.001 is applied. Each model is trained for 600 steps on 8 A100 GPUs, with evaluation every 200 steps.

Search-R1. We train Search-R1 models on both base models following Jin et al. (2025b), using the SFT-4k/PPO-40k budget to match the STEERSEARCH configuration for a fair comparison at equal compute.

A.2 Datasets

STEERSEARCH sub-question generation. Sub-questions for multi-hop training samples are generated from HotpotQA questions using GPT-5.4, following three decomposition strategies: (i) *decomposition*, a sequential chain of narrowing constraints each solvable in one retrieval step; (ii) *hint steering*, a soft directional nudge that redirects retrieval without committing to a specific intermediate answer; and (iii) *disambiguation*, entity resolution before factual retrieval. The three strategies produce complementary chain styles and are sampled to form a diverse training set. Generated chains are validated by a committee of three GPT-5.4 judge instances running in parallel; any chain receiving at least one rejection vote is discarded and retried. Chains are rejected for answer leakage, bare-noun keyword queries, bundled multi-part questions, or missing verb structure.

Supervised Fine-Tuning trajectories. SFT trajectories are generated by running GPT-5.4 on the same HotpotQA questions using the STEERSEARCH prompt, with access to the same E5-based dense retriever used during RL training. Each trajectory is retained only if the final answer passes exact match evaluation against the ground truth, ensuring the SFT data contains only successful reasoning chains.

Reinforcement Learning data. The STEERSEARCH RL training set combines multi-hop HotpotQA questions with generated sub-question chains (STEERSEARCH mode) and single-hop questions without sub-questions (Search-R1 mode). The two modes are mixed at training time to prevent the model from becoming dependent on sub-question injection at inference, where no injection occurs. For the primary results reported in Tables 1 and 2, the training pool contains 40,000 samples: 24,000 multi-hop samples drawn equally from the three sub-question strategies (decomposition, hint steering, and disambiguation) and 16,000 single-hop samples split equally between HotpotQA and NQ. We additionally study smaller pools of 8,000 and 16,000 samples to assess the effect of data scale; those results are reported in Appendix D.

For Search-R1 training, we use a pool of 40,000 single-hop samples drawn equally from HotpotQA and NQ, following the original training setup of Jin et al. (2025b). Full dataset composition and sub-

question chain length statistics are provided in Appendix B.

Evaluation. Along with the previously introduced STEERSEARCH-BENCH (**??**), we evaluate on seven search-reasoning benchmarks: NQ (Kwiatkowski et al., 2019), TriviaQA (Joshi et al., 2017), PopQA (Mallen et al., 2023), HotpotQA, 2WikiMultiHopQA, MuSiQue, and Bamboogle. These evaluation sets contain no injected sub-questions and are answered by both models in standard single-turn mode, providing a measure of general retrieval-augmented question answering ability that is unaffected by the steering mechanism. In contrast, for STEERSEARCH-BENCH, the sub-questions are injected into the model's context at the boundaries defined during generation, and the model's final answer is evaluated against the ground truth of the terminal sub-question.

All results are reported using exact match against the ground-truth answer string, following the normalization convention of Jin et al. (2025b).

A.3 Rollout Configuration

The STEERSEARCH training loop constructs interactive multi-hop trajectories through a sequence of K user-intervention phases. The outer loop advances through interventions $u_1, \ldots, u_K$, while the inner loop permits up to T_k search-reasoning turns within phase k before the next intervention is introduced. To avoid fixed-length trajectory patterns, the search budget is sampled independently for each phase and example. For the initial and intermediate phases, $T_k \sim \mathrm{Unif}\{1, \ldots, 5\}$; for the final phase, we set $T_K = 4$, allowing sufficient retrieval for the last sub-question while preventing open-ended exploration. For searching, we use the same retrieval setup as Jin et al. (2025b), unchanged across training and evaluation.

Per-phase search-reasoning. At each inner-loop turn in phase k, the model first emits a reasoning step in `<think>` and then either issues a search query in `<search>` or produces an answer in `<answer>`. A search action invokes the retriever, whose returned passages are appended as an `<information>` block within the same assistant turn. The policy can therefore perform multiple search-read cycles within a phase, accumulating evidence before deciding whether to search again or answer. Phase k terminates when the model emits `<answer>` or exhausts its budget T_k.

Mid-response user intervention. When a non-final phase $k < K$ terminates, the current assistant turn is closed, the user turn containing u_{k+1} is appended, and generation resumes from a new assistant turn. All prior context is preserved, including previous reasoning, search queries, and retrieved passages, so the model receives the new intervention with full awareness of the evidence accumulated so far.

Intermediate answer removal and final-phase generation. During an intermediate phase, the model may emit `<answer>` before the intervention sequence is complete. If left in the trajectory, this creates a shortcut: the policy can learn to terminate early using partial evidence instead of engaging with later sub-questions. For any phase $k < K$, we therefore remove the premature `<think><answer>` turn by masking all of its token positions before computing the policy-gradient loss. Reasoning and retrieval turns that occurred earlier in the same phase are retained, preserving useful search behavior. The next intervention u_{k+1} is then injected as if the phase had ended through budget exhaustion, forcing the model to continue through the full intervention chain. If the model exhausts T_K in the final phase without producing an `<answer>` tag, we run one additional generation pass with search disabled, ensuring every trajectory yields a final prediction for reward computation.

A.4 Baselines

We compare STEERSEARCH against Search-R1 (Jin et al., 2025b), the primary baseline, which trains a model to answer a single question through iterative retrieval using the same PPO objective and reward signal but without any sub-question injection during rollout. Both models are initialized from the same SFT checkpoint described in Section A.1 and share the same base architecture, retriever, and training corpus, so any performance difference is attributable solely to the mid-response steering mechanism introduced during STEERSEARCH training.

We additionally include representative retrieval-augmented baselines reported in Jin et al. (2025b): direct inference, chain-of-thought (CoT), interleaved retrieval and reasoning (IRCoT) (Trivedi et al., 2023), Search-o1 (Li et al., 2025), retrieval-augmented generation (RAG), supervised fine-tuning (SFT), DeepSeekR1-instruct (Guo et al., 2025), and rejection sampling (Ahn et al., 2024).

Phase	Dataset	Method	Strategy	Samples
SFT	HotpotQA	Search-R1	Gold trajectory	2,000
	HotpotQA	STEERSEARCH	Gold trajectory	2,000
			Total	4,000
STEERSEARCH PPO	HotpotQA	STEERSEARCH	Decomposition	8,000
	HotpotQA	STEERSEARCH	Hint steering	8,000
	HotpotQA	STEERSEARCH	Disambiguation	8,000
	HotpotQA	Search-R1	n/a	8,000
	NQ	Search-R1	n/a	8,000
			Total	40,000
Search-R1 PPO	HotpotQA	Search-R1	n/a	20,000
	NQ	Search-R1	n/a	20,000
			Total	40,000

Table 3: Training data distribution for STEERSEARCH and Search-R1. SFT trajectories are generated by GPT-5.4 and filtered by exact match. Search-R1 samples have no sub-question decomposition strategy (n/a).

Split	Mode	Min	Mean	Max
SFT	STEERSEARCH	2	2.87	4
	Search-R1		n/a	
PPO (8k / 16k)	STEERSEARCH	2	3.03	5
	Search-R1		n/a	
PPO (24k / 40k)	STEERSEARCH	2	3.48	5
	Search-R1		n/a	
STEERSEARCH-Bench	STEERSEARCH	2	2.95	5

Table 4: Sub-question chain length per sample across training splits and the evaluation benchmark. Search-R1 samples have no sub-questions by design.

These contextualise the search-augmented method landscape on standard QA.

B Dataset Statistics

Tables 3 and 5 summarise the composition of all training pools used in this work. Table 4 reports sub-question chain length statistics across splits.

C Prompt Templates and Rollout Algorithm

C.1 Training Prompt Templates

All experiments use the STEERSEARCH prompt, which has two variants depending on model architecture. The `<reason>` variant is used for models where `<think>` is a reserved token.

C.2 Rollout Algorithm

D Full Results with Data Configuration Ablation

Tables 7 and 8 report results for all training data configurations evaluated. The main paper (Tables 1 and 2) shows only the best-performing configuration (SFT-4k / PPO-40k) for STEERSEARCH. The SFT-2k / PPO-8k configuration uses the same data budget as SEARCHR1, providing an apples-to-apples comparison that isolates the contribution of the mid-response steering mechanism from data scale.

Configuration	SFT samples	PPO samples
Small	2,000	8,000
Medium	4,000	16,000
Large (primary)	4,000	40,000

Table 5: Data configurations used in the scaling study (Appendix D). All configurations use the same composition ratios; only the pool size varies.

E CCR Ablation Results

Tables 9 and 10 report results for the four CCR configurations on the standard benchmark and STEERSEARCH-Bench respectively. Table 9 covers both model scales; Table 10 covers Qwen2.5-3B-Instruct only (Qwen3 STEERSEARCH-Bench CCR results pending). All CCR runs use $w_{\text{ccr}} = 0.1$ and the SFT-4k/PPO-40k data budget.

The **context variant** (Cumulative vs. Individual) controls whether the contrastive signal compares accumulated sub-question chains or individual sub-questions in isolation. The **reward mode** controls which token positions receive the signal: *CCR (Last-think)* assigns the reward to the final `<think>` block within each sub-question phase (not the last think of the full trajectory); *CCR (All-thinks)* rewards every think block in the phase; *CCR (Terminal)* sums all per-phase CCR values and assigns them to the final answer token.

On the standard benchmark neither model scale shows consistent gains from CCR over the STEERSEARCH baseline, suggesting the outcome reward already provides sufficient signal for single-turn retrieval. On STEERSEARCH-Bench the individual context variant with all-thinks assignment achieves the highest average EM, indicating that denser, per-block supervision over individual sub-questions is most effective for mid-response steering.

F Search-R1 Benchmark vs. STEERSEARCH-BENCH

Table 11 shows one representative question per source dataset, illustrating the structural difference between the two evaluation settings. The Search-R1 benchmark presents each question as a single retrieval task with no user intervention. The

Standard (<think>)	Reason variant (<reason>)
Answer the given question. You must conduct reasoning inside <think> and </think> first every time you get a new question, follow-up, or new information. After reasoning, if you find you lack some knowledge, you can call a search engine by <search> query </search>, and it will return the top searched results between <information> and </information>. You can search as many times as you want. If you find no further external knowledge needed, you can directly provide the answer inside <answer> and </answer> without detailed illustrations. For example, <answer> Paris </answer>. **If another question or follow-up is given later, treat it as the next step and continue in the same format.** Question: {question}	Answer the given question. You must conduct reasoning inside <reason> and </reason> first every time you get a new question, follow-up, or new information. After reasoning, if you find you lack some knowledge, you can call a search engine by <search> query </search>, and it will return the top searched results between <information> and </information>. You can search as many times as you want. If you find no further external knowledge needed, you can directly provide the answer inside <answer> and </answer> without detailed illustrations. For example, <answer> Paris </answer>. **If another question or follow-up is given later, treat it as the next step and continue in the same format.** Question: {question}

Table 6: STEERSEARCH training prompt templates. The bold sentence enables mid-response user intervention. The <reason> variant is used for models where <think> is a reserved token.

Method	Config	General QA			Multi-Hop QA				
		NQ†	TriviaQA*	PopQA*	HotpotQA†	2WikiMQA*	MuSiQue*	Bamboogle*	Avg.
Qwen2.5-3B-Instruct									
Vanilla	–	0.216	0.415	0.259	0.188	0.175	0.044	0.168	0.209
Search-R1	SFT-2k / PPO-8k	0.410	0.611	0.420	0.434	0.374	0.183	0.376	0.401
Search-R1	SFT-4k / PPO-40k	0.447	0.602	0.433	0.429	0.424	0.175	0.328	0.405
STEERSEARCH	SFT-2k / PPO-8k	0.397	0.611	0.405	0.408	0.337	0.153	0.352	0.380
STEERSEARCH	SFT-4k / PPO-16k	0.424	0.614	0.427	**0.443**	**0.429**	0.197	0.352	0.412
STEERSEARCH	SFT-4k / PPO-40k	**0.450**	**0.615**	**0.442**	0.442	0.409	**0.209**	**0.384**	**0.422**
Qwen3-4B-Instruct-2507									
Vanilla	–	0.283	0.497	0.317	0.244	0.270	0.060	0.376	0.292
Search-R1	SFT-2k / PPO-8k	0.440	**0.668**	0.458	**0.500**	0.479	0.236	**0.512**	0.470
Search-R1	SFT-4k / PPO-40k	**0.496**	0.660	**0.487**	0.481	0.457	0.233	0.488	0.472
STEERSEARCH	SFT-2k / PPO-8k	0.398	0.644	0.408	0.457	0.349	0.186	0.424	0.409
STEERSEARCH	SFT-4k / PPO-16k	0.452	0.656	0.445	0.488	0.476	0.230	0.456	0.458
STEERSEARCH	SFT-4k / PPO-40k	0.491	0.652	0.479	0.475	**0.498**	**0.239**	0.480	**0.473**

Table 7: **Exact match on standard Search-Reasoning benchmarks.** With † in-domain, ∗ out-of-domain, **Bold** = best per model among fine-tuned methods, and Vanilla = not any training. STEERSEARCH denotes training the model on our new mid-response steering training set with only the final answer reward. And STEERSEARCH+CCR applies the contrastive context reward on top of it.

STEERSEARCH benchmark pairs the same question with a chain of injected sub-questions that redirect the model mid-response, requiring it to follow each injection and synthesize a final answer from accumulated evidence.

G Case Study: Search-R1 vs. STEERSEARCH

Trajectories are shown verbatim from the evaluation ouput files. Tags are colored: ⟨think⟩ reasoning, ⟨search⟩ query, ⟨information⟩ retrieved passage (truncated for space), ⟨answer⟩ final answer.

G.1 Search-R1 Fails, STEERSEARCH Succeeds

Tables 12–21 show ten trajectories where Search-R1 retrieves the correct document but ignores the injected sub-question, while STEERSEARCH correctly pivots its answer target upon receiving the injection.

G.2 Both Methods Fail

Tables 22–26 show five trajectories where neither method produces the correct answer. These involve multi-hop chains that require precise role disambiguation, compound answers, or abstract conceptual reasoning that the retrieved passages do not surface directly.

Algorithm 1 STEERSEARCH Rollout with Mid-Response Sub-Question Steering

Require: Question q, sub-question chain $\mathbf{u} = \langle u_1, \ldots, u_K \rangle$, policy π_θ, retriever $\mathcal{R}$, phase budget ranges $[T_k^{\min}, T_k^{\max}]$ for $k = 1, \ldots, K$

Ensure: Full trajectory τ

$\tau \leftarrow \langle u_1 \rangle$ ▷ Initialize with first sub-question
for $k = 1$ **to** K **do** ▷ Outer loop: one phase per sub-question
  Sample budget $T_k \sim \mathrm{Uniform}(T_k^{\min}, T_k^{\max})$
  Sample injection point $t^* \sim \mathrm{Uniform}(0,\ T_k - 1)$ ▷ Random mid-response steering point
  for $t = 1$ **to** T_k **do**
    $y \sim \pi_\theta(\cdot \mid \tau)$ ▷ Generate until `</search>` or `</answer>`
    $\tau \leftarrow \tau + y$
    if `<search>` detected in y **then**
      Extract query $q' \leftarrow \mathrm{PARSE}(y)$; retrieve $d \leftarrow \mathcal{R}(q')$
      $\tau \leftarrow \tau + \langle$`<information>` d `</information>`$\rangle$
    else if `<answer>` detected in y **and** $k = K$ **then**
      return τ ▷ Final answer produced
    else if `<answer>` detected in y **and** $k < K$ **then**
      Delete last `<think><answer>` turn from τ ▷ Suppress premature answer
      break ▷ Proceed to injection
    else
      $\tau \leftarrow \tau +$ "Your action is invalid. Use `<search>` or `<answer>`." ▷ Rethink
    end if
    if $t \geq t^*$ **and** $k < K$ **then** ▷ Steering point reached
      break
    end if
  end for
  if $k < K$ **then**
    $\tau \leftarrow \tau + \langle u_{k+1} \rangle$ ▷ Inject next sub-question mid-response
  end if
end for
▷ Budget exhausted on final phase: generate with retrieval disabled
$y \sim \pi_\theta(\cdot \mid \tau,\ \texttt{no_search})$
return $\tau + y$

Method	Config	HotpotQA†	2WikiMQA*	MuSiQue*	Bamboogle*	Avg.
Qwen2.5-3B-Instruct						
Vanilla	–	0.101	0.054	0.039	0.080	0.068
Search-R1	SFT-2k / PPO-8k	0.152	0.063	0.061	0.104	0.095
Search-R1	SFT-4k / PPO-40k	0.192	0.063	0.078	0.112	0.111
STEERSEARCH	SFT-2k / PPO-8k	0.349	0.212	0.208	**0.392**	0.290
STEERSEARCH	SFT-4k / PPO-16k	0.261	0.148	0.123	0.216	0.187
STEERSEARCH	SFT-4k / PPO-40k	**0.376**	**0.260**	**0.241**	0.376	**0.313**
Qwen3-4B-Instruct-2507						
Vanilla	–	0.171	0.173	0.068	0.096	0.127
Search-R1	SFT-2k / PPO-8k	0.279	0.160	0.153	0.312	0.226
Search-R1	SFT-4k / PPO-40k	0.423	0.349	0.250	0.440	0.366
STEERSEARCH	SFT-2k / PPO-8k	0.433	0.275	0.284	0.496	0.372
STEERSEARCH	SFT-4k / PPO-16k	0.464	0.397	0.278	0.520	0.415
STEERSEARCH	SFT-4k / PPO-40k	**0.472**	**0.457**	**0.305**	**0.552**	**0.446**

Table 8: Exact match on the STEERSEARCH benchmark with mid-response sub-question injection.

Method	Reward	General QA			Multi-Hop QA				
		NQ†	TriviaQA*	PopQA*	HotpotQA†	2WikiMQA*	MuSiQue*	Bamboogle*	Avg.
Qwen2.5-3B-Instruct									
Search-R1	Outcome + Format	0.447	0.602	0.433	0.429	0.424	0.175	0.328	0.405
STEERSEARCH	Outcome + Format	0.450	0.615	0.442	**0.442**	0.409	**0.209**	0.384	**0.422**
Cumulative	CCR (Last-think)	0.450	0.618	0.438	0.431	**0.426**	0.181	0.376	0.417
	CCR (Terminal)	0.446	**0.621**	**0.460**	0.431	0.423	0.179	0.352	0.416
Individual	CCR (Last-think)	**0.456**	0.617	0.444	0.426	0.413	0.163	**0.416**	0.419
	CCR (All-thinks)	0.448	0.605	0.425	0.410	0.388	0.147	0.336	0.394
Qwen3-4B-Instruct-2507									
Search-R1	Outcome + Format	**0.496**	**0.660**	**0.487**	0.481	0.457	0.233	**0.488**	0.472
STEERSEARCH	Outcome + Format	0.491	0.652	0.479	0.475	**0.498**	**0.239**	0.480	**0.473**
Cumulative	CCR (Last-think)	–	–	–	–	–	–	–	–
	CCR (Terminal)	0.484	0.655	0.468	0.470	0.472	0.223	**0.488**	0.466
Individual	CCR (Last-think)	0.480	0.649	0.469	**0.490**	0.488	0.226	0.472	0.468
	CCR (All-thinks)	0.485	0.646	0.472	0.476	0.455	0.208	0.416	0.451

Table 9: CCR ablation on the standard Search-Reasoning benchmark. † in-domain, $*$ out-of-domain. CCR variants are STEERSEARCH trained with $w_{\text{ccr}} = 0.1$ and SFT-4k/PPO-40k. **Bold** = best value per column per model.

Method	Reward	HotpotQA†	2WikiMQA*	MuSiQue*	Bamboogle*	Avg.
Qwen2.5-3B-Instruct						
Search-R1	Outcome + Format	0.192	0.063	0.078	0.112	0.111
STEERSEARCH	Outcome + Format	0.376	0.260	0.241	0.376	0.313
Cumulative	CCR (Last-think)	0.409	0.328	0.266	0.464	0.367
	CCR (Terminal)	0.413	0.344	0.260	**0.504**	0.380
Individual	CCR (Last-think)	0.429	0.361	**0.268**	0.480	0.384
	CCR (All-thinks)	**0.437**	**0.378**	0.255	0.488	**0.389**
Qwen3-4B-Instruct-2507						
Search-R1	Outcome + Format	0.423	0.349	0.250	0.440	0.366
STEERSEARCH	Outcome + Format	0.472	0.457	0.305	**0.552**	**0.446**
Cumulative	CCR (Last-think)	–	–	–	–	–
	CCR (Terminal)	**0.477**	**0.472**	**0.306**	0.520	0.444
Individual	CCR (Last-think)	0.471	0.446	0.300	0.480	0.424
	CCR (All-thinks)	0.456	0.389	0.275	0.480	0.400

Table 10: CCR ablation on STEERSEARCH-Bench. † in-domain, $*$ out-of-domain. CCR variants are STEERSEARCH trained with $w_{\text{ccr}} = 0.1$ and SFT-4k/PPO-40k. **Bold** = best value per column.

Dataset	Search-R1-Bench	STEERSEARCH-Bench
HotpotQA	**Q:** Are Local H and For Against both from the United States? **GT:** *yes*	**Q:** Are Local H and For Against both from the United States? [inj 1] Local H band origin country [inj 2] For Against band origin country [inj 3] Are Local H and For Against both from the United States? **GT:** *yes*
	Q: When was Poison's album "Shut Up, Make Love" released? **GT:** *2000*	**Q:** When was Poison's album "Shut Up, Make Love" released? [inj 1] Poison discography albums [inj 2] Poison album Shut Up, Make Love [inj 3] Shut Up, Make Love release date **GT:** *2000*
2WikiMQA	**Q:** Which film has the director born first, The Raven'S Dance or Keïta! L'Héritage Du Griot? **GT:** *The Raven's Dance*	**Q:** Which film has the director born first, The Raven'S Dance or Keïta! L'Héritage Du Griot? [inj 1] Who directed Keïta! L'Héritage Du Griot? [inj 2] Who directed the film The Raven'S Dance? [inj 3] Which film has the earlier-born director, The Raven'S Dance or Keïta! L'Héritage Du Griot? **GT:** *The Raven's Dance*
	Q: Who died first, Madame Pasca or James A. Donohoe? **GT:** *Madame Pasca*	**Q:** Who died first, Madame Pasca or James A. Donohoe? [inj 1] When did James A. Donohoe die? [inj 2] When did the performer known as Madame Pasca die? [inj 3] Who died first, the performer known as Madame Pasca or James A. Donohoe? **GT:** *Madame Pasca*
MuSiQue	**Q:** What company succeeded the owner of Empire Sports Network? **GT:** *Time Warner Cable*	**Q:** What company succeeded the owner of Empire Sports Network? [inj 1] Empire Sports Network owner [inj 2] company that succeeded Adelphia Communications Corporation **GT:** *Time Warner Cable*
	Q: What is the capital of the county that Pine Springs is located in? **GT:** *Van Horn*	**Q:** What is the capital of the county that Pine Springs is located in? [inj 1] Pine Springs Texas located in which county [inj 2] What is the county seat of Culberson County, Texas **GT:** *Van Horn*
Bamboogle	**Q:** When did the author of Annabel Lee enlist in the army? **GT:** *1827*	**Q:** When did the author of Annabel Lee enlist in the army? [inj 1] Who wrote "Annabel Lee"? [inj 2] What year did Edgar Allan Poe enlist in the army? **GT:** *1827*
	Q: The material out of which the Great Sphinx of Giza is made of is mainly composed of what mineral? **GT:** *calcite*	**Q:** The material out of which the Great Sphinx of Giza is made of is mainly composed of what mineral? [inj 1] What type of rock is the Great Sphinx of Giza primarily carved from? [inj 2] Main mineral composition of limestone **GT:** *calcite*

Table 11: Side-by-side comparison of the Search-R1 benchmark and the STEERSEARCH-BENCH. Each row shows one representative question from the same source dataset. In the Search-R1 benchmark the question is answered through open-ended retrieval with no user interventions. In STEERSEARCH-BENCH the same question is paired with a chain of injected sub-questions ([inj] turns) that steer the model step by step.

Search-R1 ✗	**STEERSEARCH** ✓
Question: Wigan Athletic F.C. 2017–18 season overview **Injection 1:** English Football League Cup sponsorship name in the 2017–18 season	
Ground truth: Carabao Cup	
<think>I need to find the 2017–18 Wigan Athletic F.C. season overview.</think> <search>Wigan Athletic F.C. 2017–18 season overview</search> <think>The results show multiple documents titled "2017–18 Wigan Athletic F.C. season," which appears to be the season overview. I need to extract the content of that season overview.</think> <search>2017–18 Wigan Athletic F.C. season season overview content</search> [user] English Football League Cup sponsorship name in the 2017–18 season <think>The question asks for the 2017–18 season overview of Wigan Athletic F.C., but the provided results only state the season title and some basic information about the 2017–18 season. The question likely </think> <search>2017–18 Wigan Athletic F.C. season overview English Football League Cup sponsorship name</search> <think>The results do not provide the specific season overview detail requested. The question seems to be missing the actual season overview content, which is not present in the given search results.</think> <answer>EFL Cup</answer>	<think>I need to identify the comprehensive overview or summary of the 2017–18 season for Wigan Athletic Football Club, including key statistics, results, and any notable events or performances.</think> <search>Wigan Athletic F.C. 2017–18 season overview</search> [user] English Football League Cup sponsorship name in the 2017–18 season <think>I need to identify the sponsorship name of the English Football League Cup for the 2017–18 season.</think> <search>2017–18 English Football League Cup sponsorship name</search> <think>The search results state that the 2017–18 EFL Cup was known as the Carabao Cup due to a sponsorship deal with Carabao Energy Drinks.</think> <answer>Carabao Cup</answer>

Table 12: Search-R1 keeps retrieving the same document and answers with the official competition name, missing the sponsored name that the injection asks for. STEERSEARCH pivots its search after the injection and retrieves the correct answer.

Search-R1 ✗	**STEERSEARCH** ✓
Question: Bordan Tkachuk CEO company **Injection 1:** What products and services does that company provide?	
Ground truth: IT products and services	
<think>I need to identify who the CEO of the company Bordan Tkachuk is the CEO of.</think> <search>Bordan Tkachuk CEO company</search> <think>The search results state that Bordan Tkachuk was the former CEO of Viglen. I need to identify the company he CEOed.</think> <search>Viglen company</search> [user] What products and services does that company provide? <think>The search results state that Viglen provides IT products and services, including storage systems, servers, workstations and data/voice communications equipment and services.</think> <answer>Viglen Ltd</answer>	<think>I need to identify Bordan Tkachuk and determine the company he currently serves as CEO.</think> <search>Bordan Tkachuk CEO company</search> [user] What products and services does that company provide? <think>I need to identify the company Bordan Tkachuk was CEO of (Viglen) and then find what products and services Viglen provides.</think> <search>Viglen company products and services</search> <think>The search results state that Bordan Tkachuk was the former CEO of Viglen, and that Viglen provides IT products and services, including storage systems, servers, workstations, and data/voice communica</think> <answer>IT products and services</answer>

Table 13: The retrieved document contains the correct answer verbatim, but Search-R1 answers the base question (the company name) rather than the injected question (what the company provides). STEERSEARCH follows the injection and extracts the right information.

Search-R1 ✗	**STEERSEARCH** ✓
Question: Lead single from an album titled "Confessions" song title and artist list **Injection 1:** Which song was number one for eight consecutive weeks and kept a popular song stuck at number two behind it for eight weeks **Injection 2:** Who is the singer of the song that was the lead single from the album "Confessions" and spent eight consecutive weeks at number one?	
Ground truth: Usher	
<think>I need to identify the lead single from the album "Confessions" and provide its title and artist.</think> <search>lead single from an album titled "Confessions" song title and artist</search> <think>The results show that "Confessions" is an album by Usher. I now need to find the lead single from this album.</think> <search>lead single from the album "Confessions" by Usher</search> [user] Which song was number one for eight consecutive weeks and kept a popular song stuck at number two behind it for eight weeks <think>The previous search results don't yet identify the lead single from Usher's album "Confessions". I need to find the specific lead single from that album.</think> <search>lead single from Usher album "Confessions"</search> [user] Who is the singer of the song that was the lead single from the album "Confessions" and spent eight consecutive weeks at number one? <think>The results don't yet identify the lead single. I need to find the title and artist of the lead single from Usher's "Confessions" album.</think> <search>lead single from Usher album "Confessions" song title and artist</search> <think>The search results identify the song "Burn" as the second single from Usher's "Confessions" album, but not the lead single. The lead single from "Confessions" was "Burn."</think> <answer>Burn</answer>	<think>I need to identify the album titled "Confessions" and find its lead single, then list the song title and the artist(s) associated with that single.</think> <search>album "Confessions" lead single song title artist</search> <think>I need to identify the lead single from the album "Confessions" and then find the song title and the artist(s) associated with that single.</think> <search>lead single from the album "Confessions" Usher song title</search> [user] Which song was number one for eight consecutive weeks and kept a popular song stuck at number two behind it for eight weeks <think>I need to identify the song that held the number-one position for eight consecutive weeks and was behind another popular song at number two for eight weeks.</think> <search>number-one for eight consecutive weeks kept a popular song stuck at number two behind it</search> [user] Who is the singer of the song that was the lead single from the album "Confessions" and spent eight consecutive weeks at number one? <think>I need to identify the lead single from the album "Confessions" and then determine the singer of that single.</think> <search>lead single from the album "Confessions" Usher Confessions lead single</search> <think>The lead single from the album "Confessions" is "Confessions Part II" by Usher.</think> <answer>Usher</answer>

Table 14: Search-R1 conflates a song title with the artist name across two injections that progressively narrow toward the singer. STEERSEARCH follows the chain and correctly identifies the artist.

Search-R1 ✗	**STEERSEARCH** ✓
Question: Which singer released a studio album titled "Unapologetic" (album) and what is their stage name? **Injection 1:** What Eminem songs or albums include guest vocals from the singer who released the album "Unapologetic"? **Injection 2:** Which Eminem album includes vocals from that singer?	
Ground truth: "The Marshall Mathers LP 2"	
<think>I need to identify which singer released a studio album titled "Unapologetic" and provide their stage name.</think> <search>which singer released a studio album titled "Unapologetic" album stage name</search> [user] What Eminem songs or albums include guest vocals from the singer who released the album "Unapologetic"? <think>The singer who released "Unapologetic" is Rihanna. I now need to find which Eminem songs or albums include guest vocals from Rihanna.</think> <search>Eminem songs or albums that include guest vocals from the singer who released the album "Unapologetic"</search> [user] Which Eminem album includes vocals from that singer? <think>The search results show that "Unapologetic" is Rihanna's album, and that it includes guest vocals from Eminem. The question asks which Eminem album includes vocals from that singer (Rihanna), which is</think> <answer>Rihanna</answer>	<think>I need to identify the singer who released a studio album titled "Unapologetic" and confirm their stage name.</think> <search>studio album "Unapologetic" singer stage name</search> [user] What Eminem songs or albums include guest vocals from the singer who released the album "Unapologetic"? <think>I need to identify the singer who released "Unapologetic" (Rihanna) and then determine which of her songs or albums include guest vocals from her.</think> <search>Rihanna guest vocals on Eminem songs or albums</search> [user] Which Eminem album includes vocals from that singer? <think>I need to identify the singer who released "Unapologetic" (Rihanna) and then determine which Eminem album includes guest vocals from her.</think> <search>Eminem album with vocals from Rihanna</search> <think>The singer who released "Unapologetic" is Rihanna. The song "Love the Way You Lie" features guest vocals by Eminem, and this song appears on Eminem's album "The Marshall Mathers LP 2."</think> <answer>The Marshall Mathers LP 2</answer>

Table 15: Search-R1 locks onto the answer to the base question (an artist name) and cannot redirect toward a different entity type (an album title) even after two explicit injections. STEERSEARCH follows the full chain and produces the correct album.

Search-R1 ✗	**STEERSEARCH** ✓
Question: 1974 NCAA Division I college football teams list Raiders **Injection 1:** 1974 Texas Tech Raiders football team season summary represented university **Injection 2:** Texas Tech Raiders football team represented what university in Lubbock Texas	
Ground truth: Texas Tech University	
<think>I need to identify which 1974 NCAA Division I college football team is referred to as the "Raiders."</think> <search>1974 NCAA Division I college football teams list Raiders</search> <think>The results don't yet identify the specific 1974 NCAA Division I college football team called the "Raiders." I need to search for a list of 1974 NCAA Division I college football teams and look for "Ra</think> <search>1974 NCAA Division I college football teams Raiders</search> [user] 1974 Texas Tech Raiders football team season summary represented university <think>The search results still don't show the 1974 NCAA Division I college football team named the "Raiders." I need to identify which team is commonly referred to as the "Raiders" in 1974 NCAA Division I f</think> <search>1974 NCAA Division I college football Raiders team</search> [user] Texas Tech Raiders football team represented what university in Lubbock Texas <think>The results still don't show the 1974 NCAA Division I college football team named the "Raiders." I need to identify which team is commonly referred to as the "Raiders" in 1974 NCAA Division I football</think> <search>1974 NCAA Division I college football Raiders team</search> <think>The search results still don't show the 1974 NCAA Division I college football team named the "Raiders." The question likely refers to the Oakland Raiders, a professional football team, not an NCAA Div</think> <answer>Oakland Raiders</answer>	<think>I need to identify the 1974 NCAA Division I college football teams that are referred to as the "Raiders," likely referring to a specific university team such as the Oakland Raiders or another college </think> <search>1974 NCAA Division I college football teams list Raiders</search> <think>The search results show the 1974 NCAA Division I football season lists teams such as Oklahoma, Notre Dame, Alabama, Ohio State, and USC. I need to identify which of these are referred to as "Raiders" </think> <search>1974 NCAA Division I football teams referred to as "Raiders"</search> [user] 1974 Texas Tech Raiders football team season summary represented university [user] Texas Tech Raiders football team represented what university in Lubbock Texas <think>The search results state that the 1974 Texas Tech Red Raiders football team represented Texas Tech University.</think> <answer>Texas Tech University</answer>

Table 16: Search-R1 misidentifies the entity in the base question (confusing a college team with a professional franchise) and cannot correct course even after injections explicitly name the right team and university. STEERSEARCH resolves the disambiguation correctly.

Search-R1 ✗	**STEERSEARCH** ✓
Question: Conrad Anker located the body of which mountaineer? **Injection 1:** Which mountaineer is described as having taken part in the first three British expeditions to a major Himalayan peak? **Injection 2:** What were the first three British expeditions that this mountaineer participated in, and what mountain were they attempting?	
Ground truth: Mount Everest	
<think>I need to identify which mountaineer's body Conrad Anker located.</think> <search>Conrad Anker located the body of which mountaineer</search> [user] Which mountaineer is described as having taken part in the first three British expeditions to a major Himalayan peak? <think>The question asks which mountaineer took part in the first three British expeditions to a major Himalayan peak, but the search results are about Conrad Anker and George Mallory. I need to identify the</think> <search>which mountaineer took part in the first three British expeditions to a major Himalayan peak</search> [user] What were the first three British expeditions that this mountaineer participated in, and what mountain were they attempting? <think>The search results identify the mountaineer as Andrew Irvine, who took part in the 1924 British Everest Expedition. The question asks which mountaineer's body Conrad Anker located, so the answer is Ge</think> <answer>George Mallory</answer>	<think>I need to identify the specific mountaineer whose body was located by Conrad Anker.</think> <search>Conrad Anker located the body of which mountaineer</search> [user] Which mountaineer is described as having taken part in the first three British expeditions to a major Himalayan peak? <think>I need to identify the mountaineer who participated in the first three British expeditions to a major Himalayan peak.</think> <search>first three British expeditions to a major Himalayan peak mountaineer</search> <think>I need to identify the mountaineer associated with the first three British expeditions to a major Himalayan peak, which are the 1913–1914, 1921, and 1924 British Mount Everest Expedition (involving Ma</think> <search>first three British expeditions to a major Himalayan peak mountaineer George Mallory</search> [user] What were the first three British expeditions that this mountaineer participated in, and what mountain were they attempting? <think>I need to identify the first three British expeditions for Mount Everest and confirm that the mountaineer was George Mallory.</think> <search>first three British expeditions to Mount Everest George Mallory</search> <think>The search results state that the mountaineer was George Mallory, who took part in the first three British expeditions to Mount Everest.</think> <answer>Mount Everest</answer>

Table 17: Search-R1 correctly retrieves an intermediate fact (the mountaineer) and answers with it, ignoring two injections that redirect toward a different entity (the mountain). STEERSEARCH follows the injections to the final answer.

Search-R1 ✗	**STEERSEARCH** ✓
Question: Buck-Tick Japanese rock band profile biography overview **Injection 1:** Buck-Tick official website band biography origin information **Injection 2:** Major music databases (AllMusic, Discogs, MusicBrainz) Buck-Tick artist page **Injection 3:** Where does Buck-Tick hail from?	
Ground truth: Fujioka, Gunma	
<think>I need to find a profile or biography overview of the Japanese rock band Buck-Tick.</think> <search>Buck-Tick Japanese rock band profile biography overview</search> <think>The results provide a band profile, but no specific biography overview. I need to extract a concise biography from the given profile.</think> <search>Buck-Tick Japanese rock band biography</search> [user] Buck-Tick official website band biography origin information [user] Major music databases (AllMusic, Discogs, MusicBrainz) Buck-Tick artist page [user] Where does Buck-Tick hail from? <think>The search results already state that Buck-Tick is a Japanese rock band formed in Fujioka, Gunma.</think> <answer>Japan</answer>	<think>I need to gather a comprehensive profile and biography overview of the Japanese rock band Buck-Tick, including their history, members, musical style, and notable achievements.</think> <search>Buck-Tick Japanese rock band profile biography overview</search> [user] Buck-Tick official website band biography origin information <think>I need to find the official Buck-Tick website or the band's official biography section that provides detailed information on their origin and formation.</think> <search>Buck-Tick official website band biography origin information</search> [user] Major music databases (AllMusic, Discogs, MusicBrainz) Buck-Tick artist page <think>I need to identify the specific Buck-Tick artist pages on major music databases like AllMusic, Discogs, and MusicBrainz, which should provide a detailed biography and origin information.</think> <search>Buck-Tick AllMusic Discogs MusicBrainz artist page</search> [user] Where does Buck-Tick hail from? <think>The search results state that Buck-Tick is a Japanese rock band formed in Fujioka, Gunma in 1983.</think> <answer>Fujioka, Gunma</answer>

Table 18: Search-R1 retrieves the specific city in its think block but answers with the country, anchoring to the coarse framing of the base question. STEERSEARCH follows the final injection and produces the specific city.

Search-R1 ✗	**STEERSEARCH** ✓
Question: My Chick Bad official remix track listing and credits featured artists **Injection 1:** Brittany Nicole Carpentero My Chick Bad official remix how is she credited **Injection 2:** My Chick Bad official remix female guest verses which artists are featured **Injection 3:** Official remix to My Chick Bad featuring Ludacris and Nicki Minaj which other female rappers are on the remix	
Ground truth: Eve, and Trina	
<think>I need to find the official remix track listing and credits for "My Chick Bad" and identify the featured artists.</think> <search>My Chick Bad official remix track listing and credits featured artists</search> [user] Brittany Nicole Carpentero My Chick Bad official remix how is she credited <think>The results show an official remix was released featuring Diamond, Eve, and Trina. I need to confirm how Brittany Nicole Carpenter is credited in the official remix track listing.</think> <search>Brittany Nicole Carpenter My Chick Bad official remix credits</search> [user] My Chick Bad official remix female guest verses which artists are featured [user] Official remix to My Chick Bad featuring Ludacris and Nicki Minaj which other female rappers are on the remix <think>The search results already state that the official remix to "My Chick Bad" features the female rappers Diamond, Eve, and Trina.</think> <answer>Diamond, Eve, and Trina</answer>	<think>I need to identify the specific remix of "My Chick Bad" and then find its official track listing and featured artists from reliable music or remix-related sources.</think> <search>My Chick Bad official remix track listing and credits featured artists</search> [user] Brittany Nicole Carpentero My Chick Bad official remix how is she credited <think>I need to identify who Brittany Nicole Carpentero is in the context of the "My Chick Bad" official remix and determine her specific credit (e.g., featured artist, producer, etc.).</think> <search>Brittany Nicole Carpentero My Chick Bad official remix credits</search> [user] My Chick Bad official remix female guest verses which artists are featured <think>I need to identify the official remix of "My Chick Bad" and then list the featured artists from reliable sources.</think> <search>My Chick Bad official remix female guest verses featured artists Eve Diamond Trina</search> [user] Official remix to My Chick Bad featuring Ludacris and Nicki Minaj which other female rappers are on the remix <think>The search results state that the official remix to "My Chick Bad" features the female rappers Diamond, Eve, and Trina.</think> <answer>Eve and Trina</answer>

Table 19: Search-R1 retrieves a list of featured artists and cannot refine it across three injections that narrow the question to a specific subset. STEERSEARCH progressively narrows to the correct answer.

Search-R1 ✗	**STEERSEARCH** ✓
Question: 2008 film screenplay by Andrew Davies adapted from an 1811 novel **Injection 1:** Which 1811 novel is the source material for the 2008 film with a screenplay by Andrew Davies? **Injection 2:** Who wrote the 1811 novel that the 2008 Andrew Davies–screenplay film is based on? **Injection 3:** How many major novels did the author of that 1811 novel write?	
Ground truth: six	
<think>I need to identify which 2008 film had a screenplay adapted from an 1811 novel by Andrew Davies.</think> <search>2008 film screenplay by Andrew Davies adapted from an 1811 novel</search> <think>The results show that Andrew Davies wrote the screenplay for a 2008 TV series adaptation of Jane Austen's 1811 novel "Sense and Sensibility." I need to identify the 2008 film that is the adaptation of</think> <search>2008 film adaptation of Jane Austen's 1811 novel Sense and Sensibility</search> [user] Which 1811 novel is the source material for the 2008 film with a screenplay by Andrew Davies? [user] Who wrote the 1811 novel that the 2008 Andrew Davies–screenplay film is based on? [user] How many major novels did the author of that 1811 novel write? <think>The question asks which 1811 novel the 2008 film with Andrew Davies as screenplay writer is based on. The search results show that Andrew Davies wrote the screenplay for a 2008 adaptation of Jane Aust</think> <answer>Sense and Sensibility</answer>	<think>I need to identify the 2008 film that was adapted from an 1811 novel and had a screenplay written by Andrew Davies.</think> <search>2008 film adapted from an 1811 novel screenplay by Andrew Davies</search> [user] Which 1811 novel is the source material for the 2008 film with a screenplay by Andrew Davies? <think>I need to identify the 2008 film adapted from an 1811 novel and then confirm that its screenplay was written by Andrew Davies.</think> <search>2008 film adapted from an 1811 novel screenplay by Andrew Davies</search> [user] Who wrote the 1811 novel that the 2008 Andrew Davies–screenplay film is based on? <think>I need to identify the 1811 novel that the 2008 film adapted from, which is Jane Austen's "Sense and Sensibility," and then find the author of that novel.</think> <search>1811 novel Sense and Sensibility author Jane Austen</search> [user] How many major novels did the author of that 1811 novel write? <think>I need to identify the author of "Sense and Sensibility" (Jane Austen) and then determine how many major novels she wrote.</think> <search>how many major novels did jane austen write</search> <think>The 2008 film is adapted from Jane Austen's 1811 novel "Sense and Sensibility," and the author of that novel is Jane Austen, who wrote six major novels.</think> <answer>six</answer>

Table 20: Search-R1 answers the first hop in a three-hop chain (the novel title) and does not follow subsequent injections that redirect toward the author and then the number of novels. STEERSEARCH follows all three hops and produces the correct count.

Search-R1 ✗	STEERSEARCH ✓
Question: "Live at Radio City" live album **Injection 1:** Discogs "Live at Radio City" album **Injection 2:** AllMusic "Live at Radio City" live album details **Injection 3:** Where was the album "Live at Radio City" recorded (city)?	
Ground truth: New York City	
<think>I need to identify what "Live at Radio City" refers to as a live album.</think> <search>"Live at Radio City" live album</search> [user] Discogs "Live at Radio City" album [user] AllMusic "Live at Radio City" live album details [user] Where was the album "Live at Radio City" recorded (city)? <think>The results identify "Live at Radio City" as a live album recorded at Radio City Music Hall, which answers the location of the recording.</think> <answer>Radio City Music Hall</answer>	<think>I need to identify the specific live album titled "Live at Radio City" and confirm its artist and release details from reliable music sources.</think> <search>Live at Radio City live album</search> [user] Discogs "Live at Radio City" album [user] AllMusic "Live at Radio City" live album details [user] Where was the album "Live at Radio City" recorded (city)? <think>I need to identify the specific city where the album "Live at Radio City" was recorded, which is described as a live acoustic rock concert at Radio City Music Hall in New York City.</think> <search>Live at Radio City recorded at Radio City Music Hall in New York City</search> <think>The search results state that the album "Live at Radio City" was recorded at a live acoustic rock concert at Radio City Music Hall in New York City.</think> <answer>New York City</answer>

Table 21: Search-R1 exhausts its search budget without answering, receives all three injections stacked, and still answers with the venue name (Radio City Music Hall) even though the final injection explicitly asks for the recording city. STEERSEARCH correctly extracts the city (New York City).

Search-R1 ✗	STEERSEARCH ✗
Question: Teide National Park location country and island **Injection 1:** Garajonay National Park location country and island **Injection 2:** Spain national parks in Atlantic islands autonomous community	
Ground truth: Canary Islands, Spain	
<think>I need to identify the location, country, and island of Teide National Park.</think> <search>Teide National Park location country and island</search> [user] Garajonay National Park location country and island <think>The previous question asked for Teide National Park. The provided results identify Teide National Park as located in Tenerife, Canary Islands, Spain. I now need to extract the country and island from </think> <search>Teide National Park location island and country</search> [user] Spain national parks in Atlantic islands autonomous community <think>The results show that Teide National Park is located in Tenerife, Canary Islands, Spain. This directly provides the country and island.</think> <answer>Spain</answer>	<think>I need to identify the location, country, and island associated with Teide National Park.</think> <search>Teide National Park location country island</search> [user] Garajonay National Park location country and island <think>I need to identify the location, country, and island associated with Garajonay National Park.</think> <search>Garajonay National Park location country island</search> [user] Spain national parks in Atlantic islands autonomous community <think>I need to identify the Atlantic islands autonomous community and then list the national parks located within that community.</think> <search>Atlantic islands autonomous community Spain national parks</search> <think>I need to identify the national parks in the Canary Islands (Atlantic islands autonomous community) and then extract their locations and countries.</think> <search>Canary Islands national parks list and locations</search> <think>The search results state that Teide National Park is located in Tenerife, one of the Canary Islands, Spain.</think> <answer>Tenerife</answer>

Table 22: Both models fail to combine two partial facts into the required compound answer. Search-R1 answers with the country alone; STEERSEARCH answers with the island alone. The ground truth requires both.

Search-R1 ✗	**STEERSEARCH** ✗
Question: Giuseppe Arimondi killed in action which battle **Injection 1:** Battle of Adwa outcome and significance for Ethiopia after First Italo-Ethiopian War **Injection 2:** What did the Ethiopian victory at the Battle of Adwa secure for Ethiopia	
Ground truth: sovereignty	
<think>I need to identify which battle Giuseppe Arimondi was killed in action.</think> <search>Giuseppe Arimondi killed in action which battle</search> <think>The documents state that Giuseppe Arimondi died in combat at Adwa. I now need to identify which battle Adwa was a part of.</think> <search>Adwa battle which war</search> [user] Battle of Adwa outcome and significance for Ethiopia after First Italo-Ethiopian War [user] What did the Ethiopian victory at the Battle of Adwa secure for Ethiopia <think>The documents state that the Battle of Adwa was the climactic battle of the First Italo-Ethiopian War. Therefore, Giuseppe Arimondi was killed in action during the Battle of Adwa.</think> <answer>Battle of Adwa</answer>	<think>I need to identify the person Giuseppe Arimondi and determine which specific battle he was killed in during World War II or another major military engagement.</think> <search>Giuseppe Arimondi killed in action which battle</search> <think>The search results state that Giuseppe Arimondi died in combat at Adwa. I need to identify the battle at Adwa.</think> <search>Adwa battle where Arimondi died in combat</search> [user] Battle of Adwa outcome and significance for Ethiopia after First Italo-Ethiopian War <think>I need to identify the specific battle where Giuseppe Arimondi, described as being killed in combat at Adwa, took place.</think> <search>Battle of Adwa where Giuseppe Arimondi died in combat</search> [user] What did the Ethiopian victory at the Battle of Adwa secure for Ethiopia <think>The search results state that the Battle of Adwa was a decisive defeat for Italy and secured Ethiopian sovereignty for a few decades more.</think> <answer>First Italo-Ethiopian War</answer>

Table 23: The injection chain redirects from a named entity (a battle) toward an abstract concept (sovereignty). Both models retrieve the named entity and answer with it, missing the conceptual answer the final injection asks for.

Search-R1 ✗	**STEERSEARCH** ✗
Question: Back to Mono box set track listing first track **Injection 1:** Song inspired by words on a tombstone which song was it **Injection 2:** Who wrote the song "Be My Baby"	
Ground truth: Phil Spector	
<think>I need to identify the first track in the Back to Mono box set track listing.</think> <search>Back to Mono box set track listing first track</search> [user] Song inspired by words on a tombstone which song was it <think>The results don't yet identify the first track of the Back to Mono box set. I need to search specifically for the track listing of the Back to Mono box set.</think> <search>Back to Mono box set track listing first track</search> [user] Who wrote the song "Be My Baby" <think>The previous searches didn't provide the Back to Mono box set track listing, so I need to find which track is listed as the first track in that set.</think> <search>Back to Mono box set first track</search> <think>The search results state that the first track of the Back to Mono box set is "To Know Him Is to Love Him."</think> <answer>To Know Him Is to Love Him</answer>	<think>I need to identify the first track in the Back to Mono box set track listing.</think> <search>Back to Mono box set track listing first track</search> [user] Song inspired by words on a tombstone which song was it <think>I need to identify the song in the Back to Mono box set that is inspired by words on a tombstone.</think> <search>Back to Mono song inspired by words on a tombstone</search> [user] Who wrote the song "Be My Baby" <think>I need to identify the songwriter of the song "Be My Baby".</think> <search>songwriter of "Be My Baby"</search> <think>The search results state that "Be My Baby" was written by Jeff Barry, Ellie Greenwich, and Phil Spector.</think> <answer>Jeff Barry</answer>

Table 24: Both models fail to attribute the correct songwriter across a three-hop chain. Search-R1 stops at the song title; STEERSEARCH retrieves a co-writer but not the primary one. The task requires precise attribution with no direct retrieval signal.

Search-R1 ✗	**STEERSEARCH** ✗
Question: films with original score composed by Jerry Goldsmith list **Injection 1:** which Jerry Goldsmith–scored films have an executive producer credited (executive producer in film credits) **Injection 2:** executive producer of the film Alien (1979)	
Ground truth: Ronald Shusett	
<think>I need to find a list of films that were composed by Jerry Goldsmith.</think> <search>Films with original score composed by Jerry Goldsmith list</search> [user] which Jerry Goldsmith–scored films have an executive producer credited (executive producer in film credits) <think>I need to identify which of the films listed as Jerry Goldsmith scores have an "executive producer" credit in the film credits.</think> <search>films with original score composed by Jerry Goldsmith executive producer credited</search> [user] executive producer of the film Alien (1979) <think>The question asks for films with original scores composed by Jerry Goldsmith, not specifically which ones have an executive producer credit. The search results list the films, but do not include execu</think> <search>films with original score composed by Jerry Goldsmith</search> <think>The results list the films scored by Jerry Goldsmith, but do not yet show which ones have an executive producer credit. I need to identify the specific films from that list.</think> <answer>Alien</answer>	<think>I need to identify films where the original score was composed by Jerry Goldsmith and gather a complete list of such films from reliable sources.</think> <search>films with original score composed by Jerry Goldsmith</search> [user] which Jerry Goldsmith–scored films have an executive producer credited (executive producer in film credits) <think>I need to identify Jerry Goldsmith–scored films and then determine which of those films have "executive producer" in their film credits.</think> <search>Jerry Goldsmith scored films executive producer in film credits</search> [user] executive producer of the film Alien (1979) <think>I need to identify the original film "Alien" (1979) and then determine its executive producer credit.</think> <search>Alien (1979) executive producer</search> <think>The film "Alien" (1979) was produced by Gordon Carroll, David Giler, and Walter Hill through their company Brandywine Productions.</think> <answer>Gordon Carroll</answer>

Table 25: Both models conflate adjacent crew roles. Search-R1 answers with the film title; STEERSEARCH retrieves a producer instead of the executive producer. The distinction requires careful role disambiguation in the retrieved passages.

Search-R1 ✗	**STEERSEARCH** ✗
Question: Keith Bostic date of birth **Injection 1:** Jerry Glanville date of birth **Injection 2:** Who is younger, Keith Bostic or Jerry Glanville?	
Ground truth: Keith Bostic	
<think>I need to find Keith Bostic's date of birth.</think> <search>Keith Bostic date of birth</search> <think>The results don't yet show Keith Bostic's date of birth; I need to search specifically for his date of birth.</think> <search>Keith Bostic date of birth January 17 1961</search> [user] Jerry Glanville date of birth <think>The results still don't show Keith Bostic's date of birth; I need to identify which Keith Bostic we're talking about and then search for his specific date of birth.</think> <search>Keith Bostic American Software Engineer</search> [user] Who is younger, Keith Bostic or Jerry Glanville? <think>The results don't yet show Keith Bostic's date of birth; I need to find the date of birth for Jerry Glanville and then compare.</think> <search>Jerry Glanville date of birth</search> <think>The results still don't show Keith Bostic's date of birth; I need to use the information that Keith Bostic (American football) is the same person as William Keith Bostic and then find his date of birt</think> <search>William Keith Bostic date of birth</search> <think>The search results still don't show Keith Bostic's date of birth; I need to use the specific Keith Bostic (American football) entry which does provide it.</think> <answer>January 17, 1961</answer>	<think>I need to identify the specific date of birth for Keith Bostic, likely a public figure, and verify the exact date from reliable sources.</think> <search>Keith Bostic date of birth</search> [user] Jerry Glanville date of birth <think>I need to identify the date of birth of Jerry Glanville, using a search focused on his birth date.</think> <search>Jerry Glanville date of birth</search> <think>I need to identify the birth date of William Keith Bostic, the person associated with the date January 17, 1961.</think> <search>William Keith Bostic born January 17, 1961</search> [user] Who is younger, Keith Bostic or Jerry Glanville? <think>I need to determine the birth dates of Keith Bostic and Jerry Glanville and then compare which one is earlier.</think> <search>Keith Bostic born January 17, 1961 Jerry Glanville date of birth</search> <think>The birth date of Keith Bostic is January 17, 1961. I need the birth date of Jerry Glanville to compare who is younger.</think> <search>Jerry Glanville born date</search> <think>The birth date of Keith Bostic is January 17, 1961. The birth date of Jerry Glanville is not explicitly given in the search results, but the reference to him being an NFL head coach in the 1980s sugge</think> <answer>Jerry Glanville</answer>

Table 26: Both models fail the final comparison step. Search-R1 returns a birth date rather than a person; STEERSEARCH names the wrong person, suggesting it retrieved the dates but inverted the comparison.

H Data Generation Prompts

All sub-question chains used for training and evaluation are generated by prompting GPT-5 with one of the prompts below. The three training prompts (Sections H.1–H.3) are applied to HotpotQA questions during the training data pipeline; one strategy is sampled per question. The benchmark generation prompt (Section H.4) is used to produce the STEERSEARCH evaluation benchmark from HotpotQA, 2WikiMultiHopQA, Bamboogle, and MuSiQue, with ten injection styles and an LLM judge filter.

H.1 Decomposition Prompt

```
You are given a question and its golden answer(s). Generate 2-5 sequential sub-questions that decompose the original question using broad-to-narrow
↪  reasoning.

<inputs>
Question: {question}
Golden Answer(s): {golden_answer}
</inputs>

<stopping_rule>
Let "A" be the candidate answer set for the sub-question and "G" be the golden answer set.

Hard Stop: The moment a sub-question would have answer set A = G (exactly the golden answer set), it MUST be the final sub-question. No further
↪  sub-questions are allowed after this point.

Single golden answer (G = [candidate_C]):
  SubQ1 answers: [candidate_A, candidate_B, candidate_D, candidate_E]  # A != G
  SubQ2 answers: [candidate_A, candidate_C, candidate_D]  # A != G
  SubQ3 answers: [candidate_C]  # A = G, final

Multiple golden answers (G = [candidate_C, candidate_D]):
  SubQ1 answers: [candidate_A, candidate_B, candidate_W, candidate_Z]  # A != G
  SubQ2 answers: [candidate_X, candidate_Y, candidate_C]  # A != G
  SubQ3 answers: [candidate_C, candidate_D]  # A = G, final
</stopping_rule>

<core_principle>
Create a reasoning chain that starts broad (multiple candidates) and progressively narrows to the final answer. Each step must reduce the candidate
↪  set while preserving the original task.
</core_principle>

<when_to_return_empty>
Return [] if the question is NOT complex or decomposable:
- Trivial (single lookup)
- Requires parallel independent facts (not sequential reasoning)
- Yes/no, categorical, or purely comparative
- No valid broad-to-narrow chain exists without forcing
</when_to_return_empty>

<sub_question_rules>
First sub-question (broad anchor):
- Create a small, focused candidate set (3-5 plausible candidates, never more than 10)
- Keep domain/context from original but remove most identifying constraints
- Ask about concrete entities (people, places, works, events, organizations)
- Do not directly identify the final target
- Do not use multiple high-signal clues that together collapse to one answer
- Do not create overly broad questions (e.g., "all possible rankings 1-100", "any continent")

Middle sub-questions (narrowing):
- Each adds one constraint that genuinely reduces the candidate space
- Each depends on results from previous step
- Actively filter candidates, do not just add background context
- Avoid constraints that are likely to uniquely identify a single entity in common knowledge
- Each intermediate sub-question must plausibly have multiple valid answers, not just theoretically
- Follow the <stopping_rule>

Final sub-question (resolution):
- This is the only sub-question whose answer equals the golden answer set
- Ask for the same answer type as the original question
- Incorporate all critical constraints from original
- Must be answerable using prior results
</sub_question_rules>

<hard_constraints>
1. Generate 2-5 sub-questions (prefer fewer if the funnel collapses early)
2. Never include the golden answer in any sub-question, including as a named candidate or example
3. Use only complete, natural English questions (standard letters, numbers, punctuation only)
4. Preserve original intent and answer type exactly
5. Follow the <stopping_rule>
6. Each step must reduce candidate set, not just add words
7. Only the final sub-question has A = G. All prior sub-questions must have A != G
8. When in doubt: stop early rather than add redundant sub-questions
</hard_constraints>

<self_check>
Before returning:
- Is question complex enough for decomposition? If not, return []
- First SubQ: Creates 3-8 plausible candidates?
- Each intermediate SubQ: Would this plausibly have multiple valid answers in common knowledge?
- Each intermediate SubQ: Avoid combining multiple specific constraints that collapse to a single entity
- Final SubQ only: Has A = G exactly?
- If any intermediate SubQ would likely collapse to only the golden answer, convert it to final and stop
- Golden answer never leaked in the questions themselves?
</self_check>

<output_format>
Return JSON only:
{{"sub_questions": [...]}}
```

```
</output_format>
```

H.2 Hint Prompt

```
You are given a question and its golden answer(s). Generate 2-5 sequential sub-questions that guide reasoning using soft directional hints.

<inputs>
Question: {question}
Golden Answer(s): {golden_answer}
</inputs>

<stopping_rule>
Let "A" be the candidate answer set and "G" be the golden answer set.

Hard Stop: The first step where A = G must be the final step. Stop immediately. No further steps are allowed after this point.
All prior steps must have A != G and at least 3 plausible candidates.
Stop as soon as the sequence is complete. Every additional step must introduce a direction not present in any prior step. Redundant steps are not
↪  allowed.
</stopping_rule>

<core_principle>
The user initiates this conversation. The sub-questions are what the user types, one turn at a time, starting from their own initiative. Each step is
↪  a standalone utterance that exists independently. No step refers back to the original question or the generation task. The sequence opens with a
↪  broad question the user chooses to ask first, continues with one or more declarative statements where the user shifts attention in a new direction
↪  without eliminating any candidate, and ends with a question that resolves to the golden answer. Hint statements are not questions. They change
↪  what is more likely without removing any candidate from consideration.
</core_principle>

<when_to_return_empty>
Return [] if the question does NOT support soft directional hinting:
- Any step would act as a hard filter
- Any step collapses to fewer than 3 plausible candidates before the final step
- No meaningful soft directional signal exists
- The question is trivial, yes/no, categorical, comparative, or non-sequential
</when_to_return_empty>

<sub_question_rules>
First sub-question (broad anchor):
- Create a small, focused candidate set (3-5 plausible candidates, never more than 10)
- Keep domain/context from original but remove most identifying constraints
- Ask about concrete entities (people, places, works, events, organizations)
- Do not directly identify the final target
- Do not use multiple high-signal clues that together collapse to one answer
- Do not create overly broad questions (e.g., "all possible rankings 1-100", "any continent")
- Must not use demonstratives to reference entities or events from the original question. It is the first message the user types and no prior context
↪  exists.

Hint steps (declarative statements, not questions):
- Must be a statement, not a question
- Each hint step is a new standalone message the user types on their own. It is not a reply to any question and not a comment about what they are
↪  looking for.
- Written as an objective fact or observation about the topic, in neutral third-person style
- Add one soft directional signal per step
- Must operate at a higher level of abstraction than the original question
- Must not include unique identifiers from the original question such as names, exact titles, or catalog numbers
- Must shift attention without eliminating candidates
- Must not introduce verifiable filters such as exact dates, locations, or named categories
- Must not list or name specific candidates
- Must not indirectly imply a unique answer through combined context across multiple hints
- After each hint, multiple candidates must still remain plausible
- Each hint must introduce a distinct semantic direction not present in prior steps

Final sub-question (resolution):
- This is the only sub-question whose answer equals the golden answer set
- Ask for the same answer type as the original question
- Must depend on context introduced in prior steps
- Must not be answerable without prior steps
- Must not be a copy or near-copy of the original question
- Must use information introduced in prior steps instead of repeating original wording
</sub_question_rules>

<hard_constraints>
1. Generate 2-5 sub-questions (prefer fewer if the sequence is complete)
2. Never include the golden answer in any sub-question, including as a named candidate or example
3. Use only complete, natural English (standard letters, numbers, punctuation only)
4. Preserve original intent and answer type exactly
5. Follow the <stopping_rule>
6. Hint steps must be statements, not questions
7. Hint steps must not eliminate candidates deterministically
8. Every step must stay within the same domain and entities as the original question
9. Only the final sub-question has A = G. All prior sub-questions must have A != G
10. When in doubt: stop early rather than add weak or redundant hints
</hard_constraints>

<self_check>
Before returning:
- Is there a meaningful soft directional signal? If not, return []
- Does the first sub-question (anchor) have at least 3 plausible candidates? If not, return []
- Do any hint steps contain "?"? If yes, rewrite as a statement
- Can any hint step be answered with the golden answer? If yes, rewrite or return []
- Does any hint step include unique identifiers from the original question? If yes, rewrite
- Does any step eliminate candidates deterministically? If yes, return []
- Does each hint step keep at least 3 plausible candidates? If not, return []
- Does any hint step narrow the field to a single well-known entity by combining region and role or any two high-signal attributes? If yes, return []
- Does any hint repeat a direction already present in a prior step? If yes, remove it
- Is the question a direct comparison between two named individuals or entities? If yes, return []
- Can the final step be answered without prior steps? If yes, rewrite
- Does the final step introduce any specific name, date, title, or identifier from the original question that was not set up in a prior step? If yes,
↪  rewrite using only context from prior steps. Refer to the entity as "that label", "that person", "that event", etc. rather than repeating the
↪  original name.
- Final sub-question only: Has A = G exactly and is structurally different from the original question?
```

```
</self_check>

<output_format>
Return JSON only:
{{"sub_questions": [...]}}
</output_format>
```

H.3 Disambiguation Prompt

```
You are given a question and its golden answer(s). Generate 2-5 sequential sub-questions that resolve ambiguity before answering.

<inputs>
Question: {question}
Golden Answer(s): {golden_answer}
</inputs>

<stopping_rule>
Let "A" be the candidate answer set and "G" be the golden answer set.

Hard Stop: The first step where A = G must be the final step. Stop immediately. No further steps are allowed after this point.
All prior steps must have A != G and at least 3 plausible candidates.
Stop as soon as the sequence is complete. Every additional step must introduce a new aspect not covered by any prior step. Redundant steps are not
↪  allowed.
</stopping_rule>

<core_principle>
Almost every question contains latent ambiguity. Your job is to find it and surface it. Look for: a person name shared by multiple real people, a
↪  place name used in multiple locations, a title with multiple versions or adaptations, a term with multiple meanings or domains, a role or label
↪  held by multiple entities, or uncertainty about which instance or period is meant. The sequence opens with a broad search the user types about the
↪  ambiguous element, continues with one or more statements where the user pins down which entity or meaning they intend, and ends with a question
↪  that resolves to the golden answer. Each step is a standalone user utterance. No step refers back to the original question. Clarification
↪  statements resolve which interpretation is meant, not what kind of answer is expected.
</core_principle>

<when_to_return_empty>
Return [] only if the question is yes/no, categorical, or comparative, OR if any clarification step would directly reveal the golden answer.
</when_to_return_empty>

<sub_question_rules>
First sub-question (broad anchor):
- Create a small, focused candidate set (3-5 plausible candidates, never more than 10)
- Keep domain/context from original but remove most identifying constraints
- Ask about concrete entities (people, places, works, events, organizations)
- Do not directly identify the final target
- Do not use multiple high-signal clues that together collapse to one answer
- Do not create overly broad questions (e.g., "all possible rankings 1-100", "any continent")
- Must not use demonstratives to reference entities or events from the original question. It is the first message the user types and no prior context
↪  exists.

Clarification steps (declarative statements, not questions):
- Must be a statement, not a question
- Written as a natural next search input, avoiding explicit self-correction phrasing (e.g., direct "I mean ..." style); prefer implicit clarification
↪  through context
- Use varied natural phrasing across steps, avoid starting every clarification with the same word or phrase
- Avoid explicit correction templates; express clarification indirectly through context or added detail rather than repeating a fixed pattern
- Prefer a single clarification step unless multiple independent ambiguities exist
- Do not split a single ambiguity across multiple steps
- Each clarification step must resolve which interpretation of an ambiguous term is intended, not what kind of answer is expected or how to scope it
- Each clarification step must resolve a different ambiguity dimension
- Must not reveal or uniquely imply the final answer
- Must not narrow the answer set
- After clarification, multiple answer candidates must still remain plausible

Final sub-question (resolution):
- This is the only sub-question whose answer equals the golden answer set
- Ask for the same answer type as the original question
- Fully grounded in the clarified meaning
- Must depend on context introduced in prior steps
- Must not be answerable without prior steps
- Must not be a copy or near-copy of the original question
- Must use information introduced in prior steps instead of repeating original wording
</sub_question_rules>

<hard_constraints>
1. Generate 2-5 sub-questions (prefer fewer if the sequence is complete)
2. Never include the golden answer in any sub-question, including as a named candidate or example
3. Use only complete, natural English (standard letters, numbers, punctuation only)
4. Preserve original intent and answer type exactly
5. Follow the <stopping_rule>
6. Clarification steps must be statements, not questions
7. Clarification must resolve the intended meaning without narrowing the answer set
8. Clarification must not reveal or uniquely imply the golden answer
9. Every step must stay within the same domain and entities as the original question
10. Only the final sub-question has A = G. All prior sub-questions must have A != G
</hard_constraints>

<self_check>
Before returning:
- Is the question yes/no, categorical, or comparative? If yes, return []
- Is there any latent ambiguity (name, term, title, role, period) that a clarification step could resolve? If truly none, return []
- Do any clarification steps contain "?"? If yes, rewrite as statements
- Do all clarification steps directly resolve ambiguity in a single jump? If yes, rewrite to include a more gradual refinement
- Does step 1 read as a user's own search query, not as an assistant addressing the user? If it sounds like a chatbot asking the user for
↪  clarification, rewrite as a genuine user-typed search
- Does each clarification read as a natural next search input, not as an explanation directed at the system?
- After clarification, do at least 3 answer candidates still remain?
- Can the final step be answered without prior steps? If yes, rewrite
- Final sub-question only: Has A = G exactly and is structurally different from the original question?
- Does any step repeat an aspect already covered in a prior step? If yes, remove it
- Does each clarification resolve which entity or meaning is intended, or does it add context about the answer scope? If it adds scope context, return
↪  []
```

```
</self_check>

<output_format>
Return JSON only:
{{"sub_questions": [...]}}
</output_format>
```

H.4 Benchmark Generation Prompt

```
You generate sub-question injection chains for a multi-hop search QA benchmark.

<non_negotiable_rules>
These rules override every other instruction in this prompt.

1. No answer leakage.
   The golden answer must not appear in any sub-question unless the exact wording already appears in the original question.
   This ban includes exact matches, aliases, abbreviations, alternate spellings, translations, titles, nicknames, stage names, pen names, maiden names,
  ↪  parenthetical hints, and uniquely identifying descriptions that make the answer obvious without retrieval.

   Do not hide the answer with punctuation, casing, spacing, spelling changes, or another language.
   A sub-question is invalid if a knowledgeable person could recognize the golden answer from its wording alone.

2. Correct final collapse.
   The chain must decompose the original task without changing its answer.
   The accumulated evidence after the final sub-question must support exactly the original Golden Answer(s), not a related entity, intermediate entity,
  ↪  broader class, narrower fact, or alternate answer.
   The final sub-question is the first sub-question where the candidate answer set collapses to the Golden Answer set.

3. Every sub-question must have a clear, specific retrieval target so a reader can tell exactly what fact, entity, or detail is being sought. It may
↪  take any natural form that fits the injection style, including a full question, a short declarative phrase, or a conversational follow-up. A
↪  sub-question is invalid only if it is a bare ambiguous keyword string or otherwise leaves the retrieval target unclear.

If any rule cannot be satisfied, return {{"sub_questions": []}}.
</non_negotiable_rules>

<task>
Question: {question}
Golden Answer(s): {golden_answers}
{context_block}
</task>

<goal>
Produce 2 to 5 standalone search queries that progressively retrieve the evidence needed to answer the original question.

The task is not to rewrite the original question.
The task is not to produce intermediate answers.
The task is not to make a new QA problem.
The task is only to decompose retrieval so that the final accumulated context leads to the original Golden Answer(s).
</goal>

The evaluated model runs in a retrieval loop. Each sub-question is sent to search, retrieved passages are added to context, and then the next
↪  sub-question is injected. Intermediate answers are deleted. The model gives its final answer only after reading all accumulated context from all
↪  retrieval steps. This means the chain should gather the needed evidence through retrieval, and the model will synthesize the final answer from the
↪  accumulated evidence.

Sub-questions should feel like realistic user search queries, not mechanical decomposition steps. A person steering a search session asks natural,
↪  substantive questions that open real search space. They do not ask artificially narrow queries designed to retrieve exactly one fact. A useful
↪  intermediate step should discover something the user genuinely did not know before and could not have inferred from the original question alone.

Prefer a shorter, sharper chain over a longer redundant one. Each step must introduce a new retrieval target that is not already reachable from prior
↪  results. If a step only re-asks, lightly narrows, or repeats the same entity or fact as the previous step, it is padding and should be merged or
↪  dropped.

A valid chain always exists. Even questions that seem answerable in one search can be decomposed by opening up candidate space first.

Pattern for apparently simple questions: write a broad first sub-question that retrieves a candidate set with at least 3 plausible answers, then
↪  narrow in the second sub-question using the remaining constraints from the original question. The first sub-question should match one constraint
↪  from the original question while leaving the other constraints unapplied, so that multiple candidates remain. Do not start with a query that
↪  already uniquely identifies the answer.

When the question asks for a property shared by two entities, retrieve the property for each entity as a separate complete question. When the question
↪  asks which of two candidates has a higher value, retrieve each value separately then end with a clean comparison question. When the question
↪  involves an entity whose identity is implied but not named, find the entity first.

Only return empty if every possible chain would require placing the golden answer inside a sub-question.

<answer_leakage_policy>
Before writing any sub-question, build a forbidden-answer list containing every golden answer string, obvious aliases and variants, normalized forms
↪  with punctuation and casing removed, known abbreviations or expansions, translated or transliterated forms when obvious, and descriptions that
↪  uniquely identify the answer.

If a necessary query would require using any item from this list, step back to a broader entity, event, work, source, category, or relationship, or
↪  ask for a prerequisite fact with at least 3 plausible candidates.
</answer_leakage_policy>

<collapse_policy>
For each proposed sub-question, estimate the set of plausible answers it could return. Let G be the Golden Answer set.

Rules:
- Every intermediate sub-question must leave at least 3 plausible candidates for the final answer, and must not already point to G alone.
- The final sub-question must be the first point where the accumulated evidence makes G the only reachable answer.
- Do not add any sub-question after G is reachable.
- Do not stop before G is reachable.
- Do not end on an intermediate entity unless that entity is the actual Golden Answer.

After each sub-question, check: "From the original question plus all sub-questions so far, is the Golden Answer now the only reachable answer?" If yes,
↪  this sub-question must be the final one.

Copying some wording from the original question is not by itself a failure. A final sub-question may reuse words from the original question if earlier
↪  steps have already narrowed the context and the sub-question does not point to G before the final allowed stopping point.
```

```
</collapse_policy>

<strategy_selection>
Choose exactly one decomposition_strategy and exactly one injection_style. Consider all allowed strategies before choosing. Do not default to
↪  entity_traversal when another strategy is a better match.

Allowed decomposition_strategy values: decomposition, entity_traversal, evidence_first, disambiguation, counterfactual_guard, scope_control, custom

Use decomposition when the question has multiple explicit constraints that can narrow a broad candidate set one at a time. Use entity_traversal when
↪  the question naturally requires moving from one entity to a linked entity or property and then continuing along that path. Use evidence_first when
↪  one early broad retrieval step can bring most or all needed context into view before later steps. Use disambiguation when a key term in the
↪  question has multiple plausible referents and resolving which one is meant is the first required step. Use counterfactual_guard when a famous
↪  wrong answer, dominant entity, or common misreading is likely to hijack search unless it is ruled out first. Use scope_control when the question
↪  is easy to misread in terms of what kind of answer counts as valid. Use custom only when none of the above strategies accurately describes the
↪  chain structure; note the actual approach briefly in the scratchpad.

Allowed injection_style values: direct_instruction, short_followup, research_workflow, user_correction, scope_refinement, distractor_suppression,
↪  query_pivot, comparison_check, source_hunt, timeline_trace, custom

direct_instruction is an imperative that directs a specific lookup and naturally reads like a short search phrase or natural query rather than a full
↪  question. short_followup is a brief follow-up request that moves retrieval forward with minimal wording and naturally reads like conversational
↪  continuation that may be incomplete on its own but is still a grammatical search utterance. research_workflow is a lightweight search plan that
↪  stages retrieval across steps and naturally reads as a full, well-formed question suitable for a structured search session. user_correction is
↪  when the user recognizes that a prerequisite fact must be retrieved before the main search will be useful, so the chain starts with the main goal
↪  and then pivots to the prerequisite lookup. Use it when answering directly depends on a fact that is not obvious from the question wording alone.
↪  It naturally reads as a full question that redirects the search to the needed prerequisite. scope_refinement is when the user narrows the intended
↪  sense, category, or time frame of what should be searched and naturally reads as a short scoping phrase that tightens the focus of the prior
↪  search. distractor_suppression is when the user explicitly blocks a common wrong retrieval path so search stays on the intended target and
↪  naturally reads as a targeted search phrase that excludes the misleading path. query_pivot is when the user shifts to a related but different
↪  entity or concept in order to triangulate the answer indirectly. Use it when a direct search for the answer entity is unlikely to succeed without
↪  first establishing a related fact. It naturally reads as a natural search query that may be either a full question or a concise phrase, depending
↪  on the pivot. comparison_check is when the user asks search to compare nearby candidates or possibilities so the correct path can be separated
↪  from similar ones before moving on and naturally reads as a full comparison question, typically ending with a question mark. source_hunt is when
↪  the user first looks for a source list, record set, or overview page that contains the needed evidence, then uses that retrieved source to ask the
↪  next question and naturally reads as a search phrase targeting a source collection or overview rather than a full question. timeline_trace is when
↪  the user searches through an ordered sequence of events, dates, or versions to move from an earlier fact to the target fact. It naturally reads as
↪  a full question that steps through chronology. Use custom only when none of the above styles accurately describes the phrasing pattern; note the
↪  actual style briefly in the scratchpad.
</strategy_selection>

<sub_question_requirements>
Every sub-question must:
- be a complete literal search query with no bracket placeholders, blanks, variables, or stand-ins
- be self-contained and searchable in isolation, with no dependence on prior retrieved results
- explicitly name its searchable subject
- ask for exactly one piece of information (the ending rules define the allowed exception for a final comparison question)
- retrieve a new fact or source needed for the original answer
- be written as a natural search query, not as a task instruction to a researcher
- avoid copying sentences from the reference context
- avoid paraphrasing the original question as an intermediate step
</sub_question_requirements>

<ending_rules>
For non-yes/no questions:
- Do not end with a generic synthesis, decision, or reasoning query.
- End at the last retrieval query that makes the Golden Answer uniquely reachable.
- If the original question asks which candidate wins a comparison and the Golden Answer is a named entity, the final sub-question must be a clean
↪  comparison question that makes the model output the winning entity name. It must not include the specific values or facts retrieved earlier.

For yes/no questions:
- The final sub-question must be a clean yes/no comparison question.
- It must not include the specific values, facts, or implied answer retrieved in earlier steps.
- It must require the model to use accumulated context to decide.
</ending_rules>

<pre_return_checks>
Before returning, validate all of the following. If any check fails, rewrite or return empty as stated in the non-negotiable rules above.

Leakage audit:
- Does any sub-question contain the golden answer, an alias, a translation, an abbreviation, or a uniquely identifying description?
- Does any sub-question make the answer obvious without retrieval?

Collapse audit:
- Does every intermediate step leave at least 3 plausible final-answer candidates?
- Does the final step make the original Golden Answer uniquely reachable?
- Is the final answer still exactly the original Golden Answer, not an intermediate entity?
- Did the chain stop immediately after the first collapse to G?

Standalone audit:
- Does any sub-question use bracket placeholders, blanks, or unfilled variables?
- Does any sub-question bundle more than one retrieval target or combine retrieval with comparison?
- Do the sub-questions sound like realistic user searches rather than formulaic decomposition or researcher instructions?
- Does the chain vary in phrasing and structure across steps rather than repeating the same framing?

Strategy audit:
- Is the chosen decomposition strategy a natural fit? If not, reconsider.
- Have you tried more than one plausible strategy before deciding no valid chain exists?
</pre_return_checks>

Return JSON only:
{{"scratchpad": "brief note on why this strategy and style fit; confirm no answer leakage; confirm the final sub-question is the first collapse to the
↪  original Golden Answer; for counterfactual_guard also note the likely wrong interpretation; if custom is chosen for either field, name the actual
↪  approach used", "decomposition_strategy": "one value from: decomposition, entity_traversal, evidence_first, disambiguation, counterfactual_guard,
↪  scope_control, custom", "injection_style": "one value from: direct_instruction, short_followup, research_workflow, user_correction,
↪  scope_refinement, distractor_suppression, query_pivot, comparison_check, source_hunt, timeline_trace, custom", "sub_questions": [...]}}
```

H.5 Benchmark Quality Judge Prompt

```
You are a quality judge for sub-question chains used in a multi-hop search QA benchmark.
```

```
The evaluated model runs in a retrieval loop. Each sub-question is searched, retrieved passages are added to context, and the next sub-question is
↪  injected. Intermediate answers are deleted. The final answer is produced only after all accumulated retrieval context is available.

Your job is not to judge whether the chain is perfect.
Your job is to reject only clear failures that would corrupt training.

Inputs:
Question: {question}
Golden answers: {golden_answers}
Sub-questions: {sub_questions}

Return BAD only for one of the following clear failures.

1. Answer leakage.
A sub-question leaks the golden answer if it explicitly states the golden answer, a direct alias, abbreviation, alternate spelling, translation,
↪  nickname, or uniquely identifying label that functions as the answer, unless that same wording already appears in the original question.
Do not flag a sub-question merely because a knowledgeable person could solve it using outside knowledge. Flag only when the sub-question text itself
↪  contains the answer, a direct alias, or a uniquely identifying label.
Do not flag a sub-question that asks for the golden answer, such as "what nationality is X?" or "what character did Y voice?" Asking for information
↪  is retrieval, not leakage. Leakage is when the answer is stated inside the sub-question text.
Do not flag intermediate entities that are retrieval targets on the path to the answer, unless that entity is itself the golden answer or a direct
↪  alias of it.

2. Final collapse mismatch.
The chain must still answer the original question, not a related or intermediate question.

Flag BAD only when it is definitively clear that the final sub-question targets a different answer type or a different final fact than the golden
↪  answer. Do not flag when uncertain or when the mismatch is only possible — only flag when it is obvious.

Examples:
- Golden answer is a person, final sub-question asks for a date → BAD.
- Golden answer is a date, final sub-question asks for a person → BAD.
- Golden answer is a place, final sub-question asks for an event title → BAD.
- Golden answer is yes/no, final sub-question is not a yes/no verification or comparison → BAD.
- Original question asks which candidate wins a comparison, golden answer is a named entity, but final sub-question asks only for one candidate's
↪  property value → BAD.

Do not require the final sub-question to restate the original question. It only needs to retrieve the last needed fact or perform the required final
↪  comparison. For winner-selection questions where the original question asks which of two candidates has a property (born earlier, died later,
↪  older, etc.), a final comparison sub-question between those candidates is correct and must not be flagged — even if the comparison does not
↪  explicitly restate the film or entity name.

3. Unclear or unusable retrieval target.
Each sub-question must have a clear searchable target.

Flag BAD if a sub-question:
- is only a bare keyword dump with unclear intent;
- contains an unfilled placeholder such as [name], [entity], <person>, or ___;
- depends on an unnamed prior result using wording like "that person", "this film", "the previous result", "his birthplace", or "its release date";
- is written as an instruction to a human researcher rather than a search query;
- asks only for general background with no specific retrieval target.

Short search phrases are allowed if the intended retrieval target is clear.
Conversational follow-ups are allowed only when they still explicitly name the subject being searched.

4. Obvious premature collapse or padding.
Flag BAD only if it is obvious that:
- an intermediate sub-question already uniquely points to the golden answer, making later sub-questions unnecessary; or
- a later sub-question repeats the same retrieval target without adding a new needed fact.

Do not reject merely because the chain could be shorter or phrased more elegantly.

Return GOOD if none of the above clear failures are present.

Decision standard:
Be strict on answer leakage, final-answer mismatch, and unclear retrieval targets.
Be tolerant on style, wording, strategy choice, and whether the chain is the absolute best possible decomposition.

Before deciding, check:
- Does any sub-question reveal the golden answer or a recognizable alias not already present in the original question?
- Does the final sub-question lead to the same answer type and same final fact as the golden answer?
- Does every sub-question have a clear retrieval target that can be searched literally?
- Is there any obvious premature collapse before the final sub-question?

Return JSON only, with exactly one of these forms:
{{"verdict":"GOOD","reason":null}}
{{"verdict":"BAD","reason":"<one sentence naming the specific failure>"}}
```