ABSEEKER: TRAINING LONG-HORIZON SEARCH AGENTS VIA ANSWER-BACKTRACKED CREDIT ASSIGNMENT

Anonymous authors
Paper under double-blind review

ABSTRACT

Long-horizon search agents must make multiple sequential actions (steps) to search, retrieve, verify, and integrate evidence to reach a final answer. However, existing methods for training these agents typically treat all steps within a trajectory uniformly during both supervised fine-tuning (SFT) and reinforcement learning (RL), failing to distinguish useful actions from erroneous or redundant ones. In this paper, we propose **Answer-Backtracked Credit Assignment (ABC)**, a fine-grained credit assignment framework for training long-horizon search agents by converting sparse trajectory-level outcomes into dense step-level supervision that rewards useful actions (even in failed trajectories) while suppressing erroneous or redundant actions. Specifically, given a potentially obscure query and its corresponding ground-truth answer, ABC first performs *Answer-Backtracked Clue Recovery*, which traces back from the answer to recover intermediate clues required to solve the question. It then applies *Clue-Anchored Step Scoring* to evaluate each search step against these clues, converting sparse binary outcome supervision into dense step-level rewards. Based on these rewards, we develop *ABC-SFT*, which reweights the loss of each turn, and *ABC-GRPO*, which uses the step-level scores as rewards in GRPO. Building on this framework, we train **ABSeeker** based on Qwen3.5-4B with only 8.5k examples. ABSeeker achieves 37.3% on BrowseComp and 39.1% on BrowseComp-ZH. With context management, the scores further improve to 55.3% and 52.9%, respectively, significantly outperforming same-scale (4B) agents and even matching the performance of larger ones (∼30B). These results demonstrate the effectiveness of answer-backtracked step-level credit assignment for training long-horizon search agents.

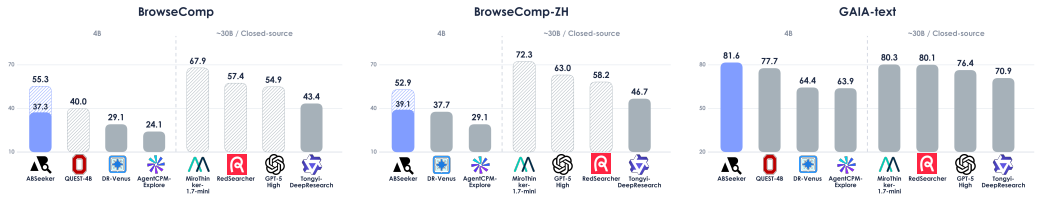

Figure 1: ABSeeker achieves the best performance among 4B models and remains competitive with several larger search agents. Striped regions indicate results with context management enabled.

1 INTRODUCTION

Search agents have emerged as a powerful approach for solving complex information-seeking tasks. Representative systems such as OpenAI Deep Research (OpenAI, 2025), Tongyi DeepResearch (Team et al., 2025), and MiroThinker (Team et al., 2026a) move beyond single-turn retrieval by conducting multi-step investigations, iteratively formulating queries, inspecting evidence, revising hypotheses, and adapting subsequent actions based on newly acquired information.

However, training search agents over long interaction horizons introduces a fundamental credit-assignment challenge. Existing methods typically treat all steps within a trajectory uniformly during

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

both supervised fine-tuning (SFT) and reinforcement learning (RL), without distinguishing their individual contributions to the final outcome (Jin et al., 2025; Song et al., 2025; Chen et al., 2025b; Zheng et al., 2025; Gao et al., 2025; Xie et al., 2026). This uniform treatment is particularly problematic for long-horizon search, where a single trajectory often contains heterogeneous actions: even a successful trajectory may include erroneous or redundant steps, while a failed trajectory may still contain useful actions that retrieve decisive evidence or refine the solution direction. Treating all steps equally therefore fails to capture individual action quality and limits targeted optimization of critical search and reasoning behaviors.

To address this challenge, the key is to identify an effective feedback signal that can evaluate a trajectory at a fine-grained level, distinguishing the contributions of individual actions during training. Our core idea is motivated by a distinctive property of search tasks: once the ground-truth answer is available, the task becomes naturally backtrackable. Starting from the answer, one can recover the key entities, facts, relations, and constraints that should have been discovered during the search process. These answer-backtracked clues thus provide a principled basis for assigning differentiated credit to different parts of the trajectory.

Following this idea, we propose **Answer-Backtracked Credit Assignment (ABC)**, a fine-grained credit assignment framework for training long-horizon search agents. Specifically, it consists of two stages: (1) *Answer-Backtracked Clue Recovery*, which traces back from the verified answer to recover a set of intermediate clues—entities, facts, and relations that collectively characterize the evidence required to solve the question. (2) *Clue-Anchored Step Scoring*, which evaluates every search step according to how it discovers, verifies, refines, or incorrectly reasons about the recovered clues, transforming sparse binary outcome supervision into dense turn-level rewards. Based on these fine-grained step rewards, we develop (1) *ABC-SFT*, built upon standard SFT, which reweights the loss of each turn according to its assigned reward; and (2) *ABC-GRPO*, built upon standard GRPO, which uses the step-level scores as rewards. Together, these methods provide step-level supervision that rewards useful actions even in failed trajectories, suppresses erroneous or redundant behaviors in successful ones, and enables fine-grained credit assignment through fixed answer-backtracked evaluation criteria.

Building on this framework, we train **ABSeeker** based on Qwen3.5-4B (Yang et al., 2025). ABSeeker achieves 37.3% on BrowseComp (Wei et al., 2025), 39.1% on BrowseComp-ZH (Zhou et al., 2025), 77.0% on xbench-2505 (Chen et al., 2025a), 46.0% on xbench-2510 (Chen et al., 2025a), and 81.6% on GAIA-text (Mialon et al., 2024). With context management, its performance further improves to 55.3% and 52.9% on BrowseComp and BrowseComp-ZH, respectively, outperforming recent same-scale 4B baselines, such as QUEST-4B (Xie et al., 2026) and Dr. Venus (Team et al., 2026b), as well as larger $\sim$30B search agents such as Tongyi DeepResearch (Team et al., 2025) and OpenSeeker (Du et al., 2026).

Our main contributions are summarized as follows:

- We propose **Answer-Backtracked Credit Assignment**, a fine-grained credit assignment framework that rewards useful actions in failed trajectories while suppressing erroneous actions in successful ones.
- Based on ABC, we develop *ABC-SFT*, which reweights the loss of each turn according to its step reward, and *ABC-GRPO*, which incorporates step-level rewards into GRPO.
- We first train **ABSeeker** based on Qwen3.5-4B using *ABC-SFT*, and then further optimize it with *ABC-GRPO*, achieving 37.3% on BrowseComp and 39.1% on BrowseComp-ZH.

2 Related Work

Search Agents. The ReAct paradigm (Yao et al., 2022) established the standard recipe for LLM-based web agents: interleave reasoning, tool calls, and observations to solve knowledge-intensive tasks through dynamic interaction with external environments. Recent work has scaled this framework to long-horizon search, where agents execute dozens of sequential retrieve–browse–integrate cycles to locate fine-grained information distributed across multiple sources. Representative search agents, such as OpenAI's Deep Research (OpenAI, 2025) and Tongyi DeepResearch (Team et al., 2025), have been developed for these tasks. Despite these advances, most existing agents are trained primarily with trajectory-level supervision—successful trajectories are treated as uniformly positive and failed

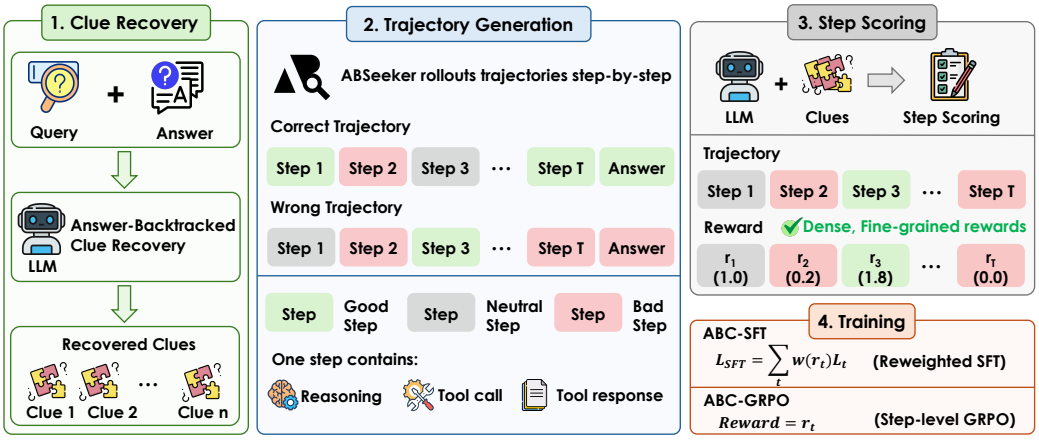

Figure 2: Overview of the training pipeline, which consists of two core stages. **Stage 1: Answer-Backtracked Clue Recovery** recovers intermediate evidence clues from the query and its verified answer. **Stage 2: Clue-Anchored Step Scoring** evaluates each step against the recovered clues and produces dense, fine-grained rewards for each step, in contrast to the sparse answer-level reward.

trajectories as uniformly negative—providing limited guidance on which intermediate decisions genuinely contribute to finding the correct answer (Zheng et al., 2025; Team et al., 2026a; Chu et al., 2026; Team et al., 2025; Du et al., 2026).

Credit Assignment for Search Agents. Several recent methods have explored fine-grained credit assignment to address this gap. IGPO (Wang et al., 2026) assigns step-level rewards based on the increase in the model's likelihood of the ground-truth answer, but the resulting credit depends on the model's own belief estimation and fluctuates with policy updates. CSO (Li et al., 2026) identifies critical steps by testing alternative actions and verifying whether they lead to correct outcomes. However, it assigns credit only to verified critical decisions, providing no direct supervision for the remaining steps in the trajectory. SAPO (Liu et al., 2026) and MindDR (Team & Inc, 2026) assign step-level credit based on intermediate entities, using either their graph proximity to the answer or their coverage in the trajectory. However, such entity-level signals cannot directly determine whether each search or reasoning decision is valid. Collectively, these methods do not provide dense and reliable supervision that directly evaluates the correctness and contribution of every action throughout the trajectory.

Our Approach. We propose **Answer-Backtracked Credit Assignment (ABC)**, which uses verified answers to construct dense, fine-grained process supervision. ABC first applies Answer-Backtracked Clue Recovery to recover intermediate evidence clues, and then uses Clue-Anchored Step Scoring to assign each step a scalar score based on its contribution or error. These scores reweight SFT losses in *ABC-SFT* and serve as rewards for GRPO in *ABC-GRPO*, from which we train **ABSeeker**.

3 Method

Figure 2 provides an overview of the training pipeline, which consists of two core stages. Section 3.1 first formalizes the search trajectory and step-level credit assignment problem. Given a query and its verified answer, **Answer-Backtracked Clue Recovery** recovers a set of intermediate evidence clues that define meaningful progress toward the answer (Section 3.2). Search trajectories are then rolled out, retaining both successful and failed trajectories. Next, **Clue-Anchored Step Scoring** evaluates every step against the recovered clue set and produces dense, fine-grained step scores (Section 3.3). Finally, these step-level scores directly serve as the reward signal for policy optimization (Section 3.4).

3.1 Search Trajectory and Credit Assignment

We consider a set of training questions, each consisting of a query q and a verified answer a^*. A search agent interacts with a web environment over T turns to produce a search trajectory:

$$\tau = (s_1, s_2, \ldots, s_T, a), \tag{1}$$

where a is the final answer submitted by the agent. Each step s_t contains the agent's reasoning, the issued tool call, and the corresponding tool response returned by the environment.

During training, the agent typically receives a reward based solely on whether the final answer matches the ground truth:

$$r_{\text{ans}}(\tau) = \begin{cases} 1, & \text{if } a = a^*, \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

This trajectory-level signal is sparse and coarse, leading to two fundamental credit-assignment failures. First, an incorrect trajectory may contain several useful intermediate steps—such as correct evidence discovery, verification, or candidate filtering—yet the final reward of zero provides no positive signal for these actions. Second, a correct trajectory may contain erroneous intermediate conclusions or steps that discard useful evidence, yet the final reward of one does not distinguish these flawed actions from genuinely informative ones. In both cases, trajectory-level outcome supervision provides no per-step signal indicating which decisions advanced or impeded progress toward the verified answer.

ABC addresses this limitation by constructing a step reward r_t for every step in each trajectory. These rewards are produced by **Clue-Anchored Step Scoring** (Section 3.3) based on the clues recovered through **Answer-Backtracked Clue Recovery** (Section 3.2). The recovered clues provide fixed anchors for evaluating each rollout, enabling stable, dense, and fine-grained supervision that distinguishes useful actions from erroneous ones regardless of the final outcome.

3.2 Answer-Backtracked Clue Recovery

The verified answer a^* specifies where the search should end, but provides little supervision about how the agent should reach it. To evaluate the quality of intermediate steps, we require a set of answer-backtracked anchors—concrete pieces of evidence that a valid search process should establish and that can serve as reference points for step-level evaluation. **Answer-Backtracked Clue Recovery** maps each training question (q, a^*) to a set of clues:

$$\mathcal{C} = \{c_1, c_2, \ldots, c_K\}, \tag{3}$$

where each c_k is a verifiable piece of intermediate evidence relevant to answering q, such as a specific entity, fact, attribute, or relationship connecting the query to the verified answer.

The recovery process leverages the fact that benchmarks such as BrowseComp (Wei et al., 2025) provide unique and verifiable answers, which offer a clear endpoint for evidence backtracking. Each question contains a set of constraints that, together with the verified answer, implicitly define a single valid evidence path. Given the query and its verified answer, we prompt an LLM to reconstruct the evidence chain by identifying the intermediate entities and facts that must be discovered, verified, and cross-referenced. Crucially, this backtracking is itself an active ReAct loop: the recovery model conducts web searches and visits pages through the same tool-call protocol as the forward agent, tracing evidence from the answer back toward the query and anchoring each clue in actual web content. Clues that survive this verification serve as reliable, answer-backtracked reference points for subsequent step scoring.

Figure 3 (left) illustrates this process using a concrete question from the training set. Given a four-constraint query and the verified answer *CeraVe*, the recovery model produces six clues, including *Ceramides* as the clinically supported ingredient, *L'Oréal* as the acquiring company, and *Eugène Schueller* as its founder who graduated in 1904. Together, these clues form a verified evidence chain connecting the query constraints to the answer. See Appendix C.1 for the full recovery prompt.

3.3 Clue-Anchored Step Scoring

Given the recovered clue set $\mathcal{C}$ for a training question, **Clue-Anchored Step Scoring** evaluates every step in each collected trajectory. For each step s_t, the scorer receives three inputs: (1) the current step,

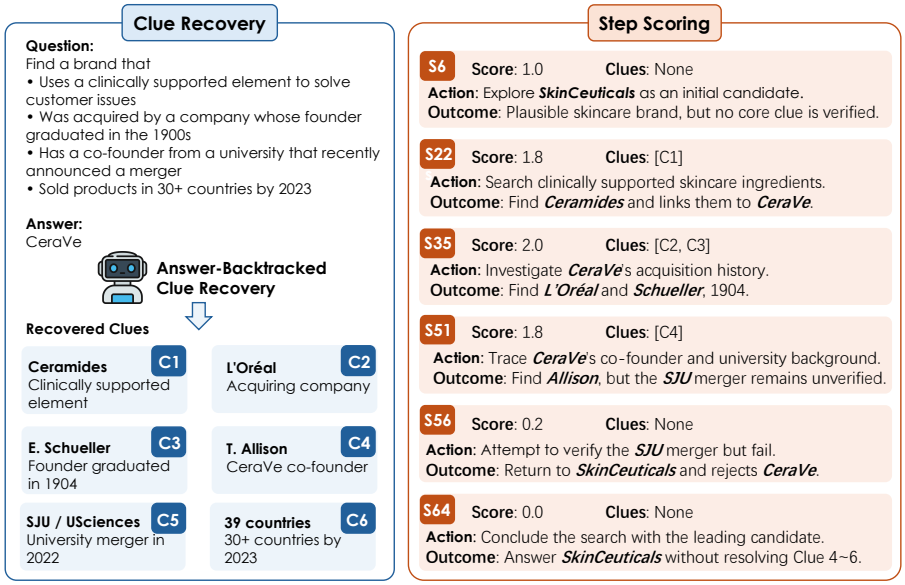

Figure 3: An illustrative example of **Answer-Backtracked Clue Recovery** and **Clue-Anchored Step Scoring**. **Left:** Given a multi-constraint query and the verified answer, the recovery produces six intermediate evidence clues (c_1–c_6) that form a verified evidence chain connecting the query to the answer. **Right:** Selected steps from a sampled trajectory are evaluated against the recovered clue set, with rewards reflecting the quality and contribution of each step.

including its reasoning, tool call, and tool response; (2) the original query q; and (3) the complete clue set $\mathcal{C}$. It outputs a step reward r_t together with a brief rationale explaining the applied criteria and any relevant clues. Each step starts with a base score of 1.0, ensuring that reasonable exploration without an obvious error is not penalized. The specific scoring criteria are listed in Table 1.

A step may exhibit multiple scored behaviors, and the same behavior may occur multiple times when several clues are affected. The corresponding deltas in Table 1 are accumulated on top of the base score and clipped to $[0, 2.0]$:

$$r_t = \text{clip}\left(1.0 + \sum_{j \in \mathcal{A}_t} \Delta_j,\ 0,\ 2.0\right), \quad (4)$$

where $\mathcal{A}_t$ denotes the set of scored behavior instances detected at step t. A step that discovers a correct clue in a trajectory that ultimately fails still receives positive credit, whereas a step that incorrectly dismisses a correct clue in a trajectory that ultimately succeeds still receives a penalty.

Table 1: **Clue-Anchored Step Scoring** rubric.

Scored Behavior	Δ
Discovers or verifies a correct clue	$+0.8$
Rules out an incorrect candidate	$+0.4$
Incorrectly dismisses a correct clue	-0.8
Submits the verified answer	$+1.0$
Submits an incorrect answer	-1.0

Figure 3 (right) illustrates the scoring process. After recovering the six clues (c_1–c_6) in Section 3.2, selected steps from a sampled trajectory are evaluated against the clue set. Step 22 receives 1.8, consisting of the base score of 1.0 and a $+0.8$ reward for discovering *Ceramides* and linking them to *CeraVe*. Step 35 verifies both the *L'Oréal* acquisition and *Eugène Schueller*'s graduation, covering c_2 and c_3; the two positive deltas are accumulated, and the resulting score is clipped to 2.0. Step 56 receives 0.2 because the agent abandons the accumulated evidence supporting *CeraVe* and returns to an incorrect candidate. Finally, Step 64 receives 0.0 after the agent submits an incorrect final answer. See Appendix C.2 for the full scoring prompt.

3.4 Training with Step Rewards

Given the step reward r_t assigned to every step in each trajectory, we train the agent in two consecutive stages: reward-weighted supervised fine-tuning (ABC-SFT) followed by step-level reinforcement

learning (ABC-GRPO). Both successful and failed trajectories are retained, allowing high-quality steps to be reinforced while erroneous steps are down-weighted or penalized.

3.4.1 Reward-Weighted Supervised Fine-Tuning (ABC-SFT)

In the first stage, we perform SFT on all collected trajectories. For a trajectory τ of length T, let $x_{t,j}$ denote the j-th policy-generated token at step s_t, with environment-provided tool responses excluded from optimization. The training objective is

$$\mathcal{L}_{\text{SFT}}(\theta) = -\sum_{t=1}^{T} w(r_t) \sum_{j} \log p_\theta \left(x_{t,j} \mid x_{t,<j} \right), \tag{5}$$

where the step-level weight is computed via a sigmoid function $w(r_t) = 2\sigma\big(\alpha \cdot (r_t - \beta)\big)$, with α controlling the sharpness of the mapping and β the neutral baseline. High-scoring steps thus contribute more strongly to the gradient, whereas low-scoring steps contribute little training signal.

3.4.2 Step-Level Reinforcement Learning (ABC-GRPO)

In the second stage, we further optimize the agent through online RL with step-level rewards. For each sampled rollout i, the reward at step t is defined as

$$R_{i,t} = r_{i,t}, \tag{6}$$

where $r_{i,t}$ is the clue-anchored step score from Section 3.3. We normalize rewards within each rollout group to obtain $\widehat{R}_{i,t}$ and compute the discounted step-level advantage:

$$A_{i,t} = \sum_{k=t}^{T_i} \gamma^{k-t} \, \widehat{R}_{i,k}, \tag{7}$$

where γ controls how future step rewards are propagated to earlier decisions. The resulting advantage $A_{i,t}$ is assigned to all policy-generated tokens at step t, while environment-provided tool responses are masked from optimization. We optimize the policy using the standard clipped GRPO objective (Shao et al., 2024), replacing its trajectory-level advantage with the step-specific advantage $A_{i,t}$.

4 Experiments

4.1 Experimental Setup

Training Setup. We use OpenSeeker (Du et al., 2026) as the training data, collecting trajectories with both correct and incorrect final answers. The maximum number of steps per trajectory is capped at 200. We use Qwen3.5-4B (Yang et al., 2025) as the backbone model. For SFT, we train on 8.5K trajectories for 3 epochs. For RL, we sample 1000 questions, each with 8 rollouts, starting from the resulting SFT checkpoint. For both Answer-Backtracked Clue Recovery and Clue-Anchored Step Scoring, we use DeepSeek-V4-Flash (Xu et al., 2026) as the backbone LLM. Refer to Appendix A for additional details.

Evaluations. We benchmark ABSeeker against the following four evaluation suites, which together span deep research, persistent web navigation, and general-purpose agent reasoning: (1) BrowseComp (Wei et al., 2025), designed to probe long-horizon English browsing and information-seeking through complex multi-constraint queries; (2) BrowseComp-ZH (Zhou et al., 2025), a Chinese-language counterpart that poses similarly difficult retrieval tasks on the Chinese web; (3) xbench (Chen et al., 2025a), which tests deep research competence—planning, reasoning, and cross-source synthesis across professional real-world scenarios; and (4) GAIA (Text-Only) (Mialon et al., 2024), a suite of general assistant tasks demanding combined web browsing, tool use, and multi-hop inference. For all benchmarks the agent is allowed up to 200 tool calls. We run each evaluation three times and report the averaged score.

Baselines. To validate the effectiveness of our method, we compare ABSeeker against three categories of baselines. (1) *Frontier foundation models* with search capability: Gemini-3.1-Pro (Google

Table 2: Performance comparison across five benchmarks. For BrowseComp and BrowseComp-ZH, * denotes results obtained without context management. Within each model category, the bold score denotes the best result on each benchmark. "–" indicates that the result is not reported.

Model	Param	BrowseComp	BrowseComp-ZH	xbench-2505	xbench-2510	GAIA-text
Foundation Model with Tools						
Gemini-3.1-Pro	–	–	**85.9**	–	53.0	**80.6**
Seed-2.0-Pro	–	77.3	82.4	–	–	78.6
GLM-5	358B	75.9	72.7	–	–	–
DeepSeek-V4-Pro-Max	1.6T	**83.4**	–	–	**80.0**	–
GPT-5 High	–	54.9	63.0	77.9	75.0	76.4
Search Agent (∼30B)						
MiroThinker-1.7-mini	30B	**67.9**	**72.3**	–	57.2	**80.3**
RedSearcher	30B	57.4	58.2	–	–	80.1
DeepMiner	32B	33.5	40.1	62.0	–	58.7
Tongyi-DeepResearch	30B	43.4*	46.7*	**75.0**	–	70.9
OpenSeeker	30B	29.5*	48.4*	74.0	–	–
Search Agent (4B)						
QUEST-4B	4B	40.0	–	–	–	77.7
DR-Venus	4B	29.1*	37.7*	74.7	40.7	64.4
AgentCPM-Explore	4B	24.1*	29.1*	70.0	34.0	63.9
ABSeeker	4B	37.3*/ **55.3**	39.1*/ **52.9**	**77.0**	**46.0**	**81.6**

DeepMind, 2026), Seed-2.0-Pro (Bytedance Seed, 2026), GLM-5 (Zeng et al., 2026), DeepSeek-V4-Pro-Max (Xu et al., 2026), and GPT-5 High (Singh et al., 2025). (2) *Search agents at ∼30B*: MiroThinker-1.7-mini (Team et al., 2026a), RedSearcher (Chu et al., 2026), DeepMiner (Tang et al., 2025), Tongyi-DeepResearch (Team et al., 2025), and OpenSeeker (Du et al., 2026). (3) *Search agents at 4B*: QUEST-4B (Xie et al., 2026), DR-Venus (Team et al., 2026b), and AgentCPM-Explore (Chen et al., 2026).

4.2 Experimental Results

Main Results. Table 2 compares ABSeeker with foundation models and representative search agents across five benchmarks. Among 4B search agents, ABSeeker achieves the best performance on every benchmark, scoring **55.3%** on BrowseComp, **52.9%** on BrowseComp-ZH, **77.0%** on xbench-2505, **46.0%** on xbench-2510, and **81.6%** on GAIA-text. Despite its smaller model size, ABSeeker also remains competitive with substantially larger search agents. It outperforms all reported 30B agents on xbench-2505 and GAIA-text, while surpassing several 30B systems on both BrowseComp and BrowseComp-ZH. Notably, although our method is trained exclusively on BrowseComp-style questions, it generalizes effectively to xbench and GAIA, indicating strong cross-benchmark generalization. These results demonstrate that through answer-backtracked step-level credit assignment—which rewards useful actions even in failed trajectories and penalizes redundant or erroneous behaviors in successful ones—the model learns to search more effectively and more deliberately, validating the effectiveness of our approach.

Reward Distribution Analysis. We further analyze the step-reward distribution over the 8.5K trajectories used for **ABC-SFT**. As shown in Figure 4, even successful trajectories contain approximately 4% low-quality steps with rewards below 1.0. More importantly, nearly 10% of the steps in failed trajectories receive rewards above 1.0, indicating that they still discover or verify useful clues despite ultimately producing an incorrect answer. Trajectory-level supervision would assign the same outcome signal to all of these steps, thereby reinforcing erroneous actions in successful trajectories while penalizing useful actions in failed ones. In contrast, **Answer-Backtracked Credit Assignment** evaluates each step independently, preserving productive exploration and suppressing incorrect decisions. This precise supervision accounts for the consistent gains over the corresponding baselines. See Section 4.3 for a detailed ablation analysis.

RL Training Dynamics. To better understand the source of these gains, we compare **ABC-GRPO** with standard trajectory-level GRPO during training, both evaluated on a validation set of 200 randomly sampled BrowseComp questions. As shown in Figure 5, **ABC-GRPO** achieves consistently

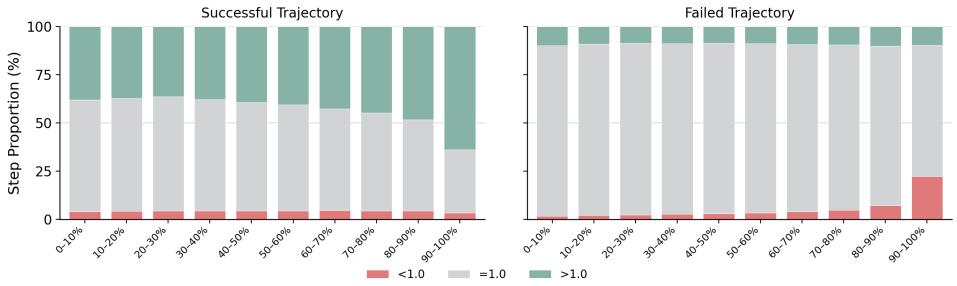

Figure 4: Distribution of step rewards across the 8.5K SFT trajectories. The left half shows successful trajectories, while the right half shows failed trajectories. Each bar represents one-tenth of the steps in a trajectory, ordered from earliest to latest; for example, 0–10% represents the first 10% of steps in a trajectory. Red, gray, and green denote low-quality steps with $r_t < 1.0$, neutral steps with $r_t = 1.0$, and high-quality steps with $r_t > 1.0$, respectively.

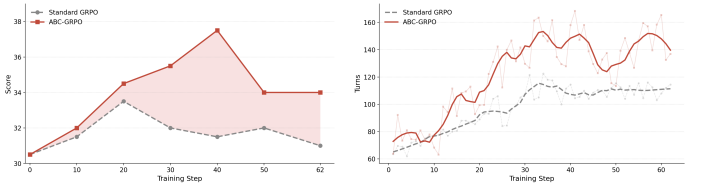

Figure 5: RL training dynamics. **Left**: Performance during training. **Right**: Average number of interaction turns.

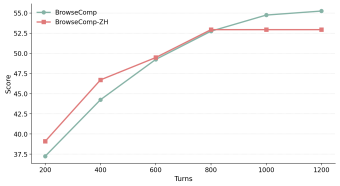

Figure 6: Performance under different context budgets.

stronger BrowseComp performance after training begins while producing longer search trajectories. Together, these results show that step-level credit assignment improves both search accuracy and exploratory behavior.

Effect of Context Management. Following MiroThinker (Team et al., 2026a) and LongSeeker (Lu et al., 2026), we set the maximum context length to 256K tokens and apply the discard-all strategy for up to five rounds on BrowseComp and BrowseComp-ZH. As shown in Figure 6, ABSeeker improves from 37.3% to 55.3% on BrowseComp and from 39.1% to 52.9% on BrowseComp-ZH.

4.3 Ablation Studies

Table 3 evaluates ABC-SFT and ABC-GRPO across all five benchmarks with all methods tested without context management. Compared with standard SFT, ABC-SFT improves performance on BrowseComp, BrowseComp-ZH, xbench-2510, and GAIA-text. We further compare with rejection sampling, which filters out failed trajectories and trains only on successful ones, but remains below ABC-SFT overall. Building on this initialization, ABC-GRPO consistently outperforms both standard trajectory-level GRPO and the step-level IGPO baseline across all benchmarks. These results demonstrate that answer-backtracked, clue-anchored credit assignment improves both SFT and RL by enabling the model to emphasize useful actions and suppress erroneous ones throughout training.

5 Conclusion

We present **Answer-Backtracked Credit Assignment (ABC)**, a fine-grained credit assignment framework for training long-horizon search agents. Instead of treating all steps within a trajectory uniformly, ABC recovers intermediate evidence clues from verified answers and uses them to evaluate each search step. Based on the resulting clue-anchored rewards, we develop *ABC-SFT*, which reweights the loss of each turn, and *ABC-GRPO*, which uses step-level scores as rewards in GRPO. These methods reward useful actions in failed trajectories while suppressing erroneous or redundant actions in successful ones. Building on this framework, we train **ABSeeker** based on Qwen3.5-4B. Experiments across BrowseComp, BrowseComp-ZH, xbench, and GAIA-text show that ABSeeker outperforms same-scale baselines and remains competitive with substantially larger search agents.

Table 3: Ablation study of ABC-SFT and ABC-GRPO. Results are evaluated without context management.

Model	BrowseComp	BrowseComp-ZH	xbench-2505	xbench-2510	GAIA-text
Qwen3.5-4B					
+Standard SFT	28.5	30.4	73.0	27.0	66.0
+Rejection Sampling SFT	28.3	30.1	72.0	29.0	69.9
+ABC-SFT	30.8	31.8	72.0	35.0	72.8
ABSeeker-4B-SFT					
+Standard GRPO	33.5	36.3	75.0	41.0	77.7
+IGPO	34.9	38.4	75.0	44.0	78.6
+ABC-GRPO	**37.3**	**39.1**	**77.0**	**46.0**	**81.6**

Further analyses demonstrate that answer-backtracked step-level supervision improves reward quality, training dynamics, and long-horizon exploration, highlighting the importance of explicit process supervision for scalable search-agent training.

Limitations and Future Work. Due to computational constraints, our experiments focus on a compact 4B model. A natural next step is to scale ABSeeker to larger backbone models and examine whether answer-backtracked credit assignment yields stronger gains with increased model capacity. While the consistent downstream improvements suggest that the clue-anchored rewards provide effective supervision, their fine-grained agreement with human judgments remains to be systematically validated. We also do not systematically analyze the sensitivity of the hand-crafted rubric scores or key hyperparameters, including the SFT weighting parameters α and β and the RL discount factor γ. Future work will evaluate alternative reward calibrations, human-aligned validation, and parameter settings to assess their effects on training stability and downstream performance. Beyond web search, we plan to extend the framework to other long-horizon agent tasks where final outcomes can be backtracked into intermediate evidence, subgoals, or decision points.

AI USE STATEMENT

In this work, we used generative AI tools for language polishing, assistance with LaTeX and code debugging. We have not used generative AI tools for generating experimental results, data, or unverified citations, and other required disclosure tasks are not applicable to this work. Additionally, we used generative AI tools for improving the clarity and organization of the manuscript. We have reviewed all AI-assisted work. We manually verified relevant literature and citations, and checked AI-assisted code and technical suggestions for correctness. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work does not involve human subjects, sensitive data, or other significant ethical concerns. We are not aware of any conflicts of interest, privacy, security, or fairness issues arising from this work.

REPRODUCIBILITY STATEMENT

We provide sufficient details to reproduce our results, including the model and training configurations, implementation details, evaluation settings, and data processing procedures in the main paper and appendix. Our code will be publicly released to further support reproducibility.

REFERENCES

Bytedance Seed. Seed2.0 model card: Towards intelligence frontier for real-world complexity. *arXiv preprint arXiv:2607.00248*, 2026. doi: 10.48550/arXiv.2607.00248.

Haotian Chen, Xin Cong, Shengda Fan, Yuyang Fu, Ziqin Gong, Yaxi Lu, Yishan Li, Boye Niu, Chengjun Pan, Zijun Song, et al. Agentcpm-explore: Realizing long-horizon deep exploration for edge-scale agents. *arXiv preprint arXiv:2602.06485*, 2026.

Kaiyuan Chen, Yixin Ren, Yang Liu, Xiaobo Hu, Haotong Tian, Tianbao Xie, Fangfu Liu, Haoye Zhang, Hongzhang Liu, Yuan Gong, et al. xbench: Tracking agents productivity scaling with profession-aligned real-world evaluations. *arXiv preprint arXiv:2506.13651*, 2025a.

M Chen, L Sun, T Li, H Sun, Y Zhou, C Zhu, H Wang, JZ Pan, W Zhang, H Chen, et al. Research: Learning to reason with search for llms via reinforcement learning. arxiv 2025. *arXiv preprint arXiv:2503.19470*, 2025b.

Zheng Chu, Xiao Wang, Jack Hong, Huiming Fan, Yuqi Huang, Yue Yang, Guohai Xu, Chenxiao Zhao, Cheng Xiang, Shengchao Hu, et al. Redsearcher: A scalable and cost-efficient framework for long-horizon search agents. *arXiv preprint arXiv:2602.14234*, 2026.

Yuwen Du, Rui Ye, Shuo Tang, Xinyu Zhu, Yijun Lu, Yuzhu Cai, and Siheng Chen. Openseeker: Democratizing frontier search agents by fully open-sourcing training data. *arXiv preprint arXiv:2603.15594*, 2026.

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

Google DeepMind. Gemini 3.1 Pro model card. https://deepmind.google/models/model-cards/gemini-3-1-pro/, 2026.

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

Mukai Li, Qingcheng Zeng, Tianqing Fang, Zhenwen Liang, Linfeng Song, Qi Liu, Haitao Mi, and Dong Yu. Verified critical step optimization for llm agents. *arXiv preprint arXiv:2602.03412*, 2026.

Yuchen Liu, Yingjie Feng, Lixiong Qin, Jiasi Chen, Jianing Yu, Sheng Gao, Sheng Yang, and Weiran Xu. Beyond trajectory rewards: Step-level credit assignment for agentic search via graph modeling. *arXiv preprint arXiv:2605.29697*, 2026.

Yijun Lu, Rui Ye, Yuwen Du, Jiajun Wang, Songhua Liu, and Siheng Chen. Longseeker: Elastic context orchestration for long-horizon search agents. *arXiv preprint arXiv:2605.05191*, 2026.

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

OpenAI. Deep research system card. https://cdn.openai.com/deep-research-system-card.pdf, 2025. February 25, 2025.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Aaditya Singh, Adam Fry, Adam Perelman, Adam Tart, Adi Ganesh, Ahmed El-Kishky, Aidan McLaughlin, Aiden Low, AJ Ostrow, Akhila Ananthram, et al. Openai gpt-5 system card. *arXiv preprint arXiv:2601.03267*, 2025.

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

Qiaoyu Tang, Hao Xiang, Le Yu, Bowen Yu, Yaojie Lu, Xianpei Han, Le Sun, WenJuan Zhang, Pengbo Wang, Shixuan Liu, et al. Beyond turn limits: Training deep search agents with dynamic context window. *arXiv preprint arXiv:2510.08276*, 2025.

MindDR Team and Li Auto Inc. Mind deepresearch technical report. *arXiv preprint arXiv:2604.14518*, 2026.

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

Tongyi DeepResearch Team, Baixuan Li, Bo Zhang, Dingchu Zhang, Fei Huang, Guangyu Li, Guoxin Chen, Huifeng Yin, Jialong Wu, Jingren Zhou, et al. Tongyi deepresearch technical report. *arXiv preprint arXiv:2510.24701*, 2025.

Venus Team, Sunhao Dai, Yong Deng, Jinzhen Lin, Yusheng Song, Guoqing Wang, Xiaofeng Wu, Yuqi Zhou, Shuo Yang, Zhenzhe Ying, et al. Dr-venus: Towards frontier edge-scale deep research agents with only 10k open data. *arXiv preprint arXiv:2604.19859*, 2026b.

Guoqing Wang, Sunhao Dai, Guangze Ye, Zeyu Gan, Wei Yao, Yong Deng, Xiaofeng Wu, and Zhenzhe Ying. Information gain-based policy optimization: A simple and effective approach for multi-turn search agents. In *The Fourteenth International Conference on Learning Representations*, 2026.

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

Jian Xie, Tianhe Lin, Zilu Wang, Yuting Ning, Yuekun Yao, Tianci Xue, Zhehao Zhang, Zhongyang Li, Kai Zhang, Yufan Wu, et al. Quest: Training frontier deep research agents with fully synthetic tasks. *arXiv preprint arXiv:2605.24218*, 2026.

Anyi Xu, Bangcai Lin, Bing Xue, Bingxuan Wang, Bingzheng Xu, Bochao Wu, Bowei Zhang, Chaofan Lin, Chen Dong, Chenchen Ling, et al. Deepseek-v4: Towards highly efficient million-token context intelligence. *arXiv preprint arXiv:2606.19348*, 2026.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. In *NeurIPS 2022 Foundation Models for Decision Making Workshop*, 2022.

Aohan Zeng, Xin Lv, Zhenyu Hou, Zhengxiao Du, Qinkai Zheng, Bin Chen, Da Yin, Chendi Ge, Chenghua Huang, Chengxing Xie, et al. Glm-5: from vibe coding to agentic engineering. *arXiv preprint arXiv:2602.15763*, 2026.

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

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

A Training Details

Supervised Fine-Tuning. We implement SFT using Slime and initialize the model from Qwen3.5-4B. We randomly select 8.5K trajectories from OpenSeeker (Du et al., 2026), consisting of 5.5K correct and 3.0K incorrect trajectories. We train on these 8.5K scored trajectories for 3 epochs with a global batch size of 64. The optimizer is Adam with a learning rate of 5×10^{-5}, cosine decay, a warmup ratio of 0.1, a minimum learning rate of 1×10^{-6}, and weight decay of 0.1. We use tensor model parallel size 2 and context parallel size 8 for long-context training. Tool responses are masked from

the loss. For ABC-SFT, we map each step reward to a loss weight by $w(r_t) = 2\sigma(2(r_t - 1))$, where the neutral reward $r_t = 1.0$ is mapped to weight 1.0.

Reinforcement Learning. We implement RL using veRL and initialize from the SFT checkpoint. We train on 1,000 questions filtered by the number of interaction turns, of which 200 have fewer than 100 turns and the remaining 800 have at least 100 turns. We sample 8 rollouts per question with a training batch size of 16. Each rollout allows up to 200 interaction turns. We use asynchronous rollout with 16 agent-loop workers. The rollout policy uses temperature 1.0, top-$p = 0.95$. We optimize with ABC-GRPO using clue-anchored rewards plus a format penalty. Rewards are normalized within each rollout group, and discounted step-level advantages are computed with $\gamma = 0.25$. The actor learning rate is 1×10^{-6} and the KL loss coefficient is 0.001.

B Evaluation Details

Due to computational constraints, for the RL validation set, we sample 200 BrowseComp questions from the full set using random seed 42. To reduce evaluation variance, we run each evaluation three times and report the averaged score. All evaluations allow up to 200 interaction turns and use DeepSeek-V4-Flash (Xu et al., 2026) as the evaluation model with the prompt in Appendix C.3 to judge final-answer correctness.

C Prompt Templates

C.1 Answer-Backtracked Clue Recovery

This prompt recovers intermediate clues from the query and verified answer.

Prompt for Answer-Backtracked Clue Recovery

System:
You are an expert investigator. Use search_web and visit_web to gather evidence. Do NOT provide the final answer directly. Extract intermediate clues, such as dates, places, and events, from the question, search for them first, then piece together to verify the answer.

User:
Reconstruct cite-backed steps that prove the answer is correct.

Question: {query}
Known Answer: {answer}

Your task: Find the intermediate clues that lead to the correct answer. These clues are usually conditions in the question, such as events, dates, places, and entities. Search for them first, then verify the answer. You MUST explicitly mention these intermediate clues in your final answer and explain how they support the answer.

C.2 Clue-Anchored Step Scoring

This prompt scores each trajectory step using the recovered clue set.

Prompt for Clue-Anchored Step Scoring

System:
You are a trajectory scorer. Your task is to evaluate each step in an AI agent's problem-solving trajectory based on the scoring rubric.
Focus on INTERMEDIATE STEPS and KEY ENTITIES. Base your scoring ONLY on the step's content, including Context, Reasoning, and Tool Call.

User:
Scoring Rubric:

Criteria 2–4 apply only to clues listed in "Core Clues" below. Each step starts from base score 1.0. Multiple criteria may apply in one step.

1. **Baseline (1.0):** Reasonable progress without an obvious error gets the base score 1.0.
2. **Finds or verifies a correct clue (+0.8):** The step discovers, investigates, or verifies a clue that matches the core clue set.
3. **Correctly rules out a wrong candidate (+0.4):** The step correctly excludes, dismisses, or identifies as wrong an irrelevant or incorrect candidate.
4. **Incorrectly dismisses a correct clue (-0.8):** The step wrongly excludes or dismisses a clue that is correct according to the core clues. Exploring wrong candidates does NOT count as this.
5. **Correct final answer (+1.0):** Only when the step explicitly gives the final answer and it matches the Correct Answer below.
6. **Wrong final answer (-1.0):** Only when the step explicitly submits an incorrect final answer. If this applies, do NOT apply criterion 5.

Question:
{query}

Correct Answer:
{correct_answer}

Core Clues:
{core_clues_text}

Step to Score:
Step Number: {step_num}
Context: {user_content}
Reasoning: {reasoning}
Tool Call: {action}

Task:
Score this step based on the rubric above.
Return ONLY a JSON object with this exact structure:
{ "score": <float between 0.0 and 2.0; start from 1.0, add/subtract applied deltas, then clip to [0, 2.0]>,
"criteria": <array of int 1–6, the rubric criterion numbers that apply, e.g., [1] or [1, 2, 5]>,
"explanation": <brief explanation. When applying criteria, MUST name the specific clue(s)/entity, e.g., clue X or entity Y> }
Important: Return ONLY valid JSON, no other text.

C.3 Evaluation Prompt

This prompt checks whether the predicted answer matches the ground truth.

Prompt for Final Answer Evaluation

System:
You are an evaluator. Your ONLY task is to decide: does the Predicted Answer refer to the SAME entity or value as the Ground Truth?

User:
Question: {query}
Ground Truth: {gt}
Predicted Answer: {pred}

Output only a JSON object with this format:
{"correct": true/false, "reason": "brief explanation"}