SELECTING WHAT TO LEARN AND CORRECTING POLICY GRADIENTS FOR SEARCH AGENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Search agents extend language models to information-seeking tasks that require repeated retrieval and reasoning across sources. Learning from rewards based on final-answer correctness remains difficult. Questions that are too easy or too hard provide little reward contrast, while sampled trajectories produce noisy gradients. We introduce an adaptive training framework that couples question selection with gradient correction. A frozen generator constructs questions from retrieved evidence, and independent probes estimate the current agent's success rate on each question. The framework then routes questions to supervised fine-tuning, reinforcement learning, or later reassessment. For reinforcement learning, we propose Gradient-Corrected Group Relative Policy Optimization (GC-GRPO) to reduce sampling noise in policy gradients. GC-GRPO constructs zero-mean reference gradients from current trajectories and combines them using weights learned from earlier batches. Our analysis shows when this correction reduces mean squared estimation error while preserving the expected GRPO gradient at the sampling policy. Across five search benchmarks, the trained 4B search agent achieves 85.44% accuracy on GAIA, 48.66% on BrowseComp, 47.75% on BrowseComp-ZH, 47.75% on SEAL-0, and 82.00% on XBench-DS.

1 INTRODUCTION

Search agents address information needs that require repeated retrieval and reasoning across sources. They must choose queries, inspect evidence, and decide when they have enough information to answer (Yao et al., 2023; MiroMind Team, 2026). Reinforcement learning (RL) from final-answer rewards can train this behavior (Jin et al., 2025), while question generation from retrieved evidence expands the supply of training experience (Liu et al., 2025a). The generated questions may still provide little useful feedback for the current agent, and gradient estimates from sampled trajectories can remain noisy.

A question's usefulness changes with the agent's ability to solve it. In Group Relative Policy Optimization (GRPO), all-success or all-failure groups have zero task advantages, whereas mixed groups provide reward contrast (Shao et al., 2024). Successful trajectories can still supply demonstrations for supervised fine-tuning (SFT), and questions with no observed success may become useful after further training. Prior work combines SFT with RL and revisits questions as the policy changes (Zhang et al., 2025; Coelho et al., 2026). These observations motivate deciding which learning objective a question should serve at the current stage, alongside whether to use it at all.

Reward contrast alone does not eliminate sampling noise in a policy update. Different sampled trajectories yield different gradient estimates, even for the same policy and questions. Control variates can reduce this variability by subtracting a correlated, zero-mean reference from the gradient estimate (Greensmith et al., 2004; Zhong et al., 2021). Applying this principle to GRPO requires references that track its sampling fluctuations and correction weights that can be estimated during training. The objective is to reduce gradient estimation error while retaining the expected GRPO gradient at the sampling policy.

We address these two problems through an adaptive training framework and Gradient-Corrected GRPO (GC-GRPO). A frozen generator constructs questions from retrieved evidence. Independent probes estimate the current agent's success rate and route questions to SFT, RL, or later reassessment. Successful demonstrations support SFT; the subsequent RL stage draws fresh trajectories

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

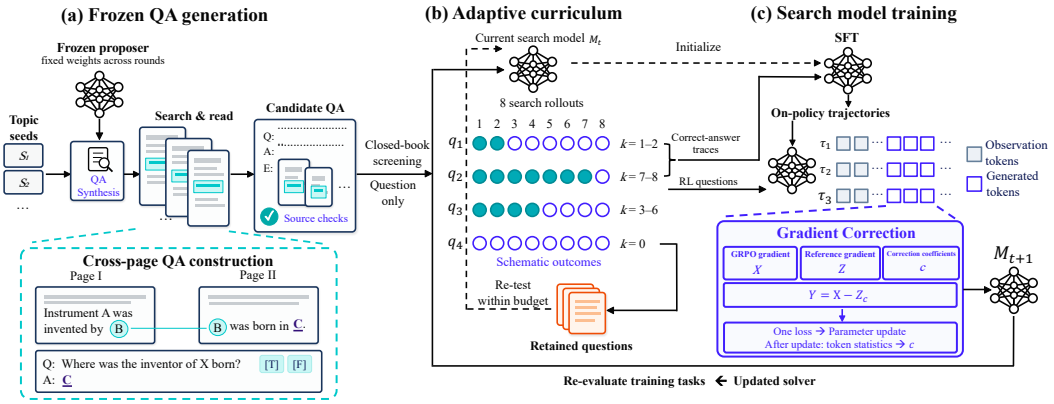


Figure 1: The adaptive training framework comprises (a) frozen question generation, (b) adaptive curriculum, and (c) search agent training.

for questions selected to favor reward contrast. GC-GRPO corrects their gradient estimates using reference gradients constructed from information preceding each sampled token and rewards from other trajectories on the same question. The references come from current trajectories, while their combination weights are learned from earlier batches.

We make three contributions.

- We propose an adaptive training framework that combines question generation with SFT and RL, tailoring the use of training questions to the search agent's evolving capabilities.
- We propose GC-GRPO to correct sampling fluctuations in GRPO gradients using token-prefix information and rewards from other trajectories on the same question. It reuses current rollouts, learns correction coefficients from earlier batches, and requires one backward pass.
- We introduce Apt, a 4B search agent trained with the adaptive framework and GC-GRPO. Across five search benchmarks, Apt improves over its pretrained base model by 14.57–19.12 percentage points in accuracy.

2 Adaptive Training Framework for Search Agents

Figure 1 shows how our framework adapts training to the search agent's current capabilities. A frozen question generator constructs questions from retrieved evidence. The generated questions are routed to SFT or RL according to their difficulty for the current agent, while those with no observed success are retained for reassessment. Each round applies SFT followed by RL on newly sampled trajectories. The updated agent guides question routing in the next round.

2.1 Constructing Questions from Retrieved Evidence

The question generator starts from a topic seed and searches for connected facts, following the exploration-based approach to task synthesis studied by Liu et al. (2025a). An entity found on one page can become the next query, allowing the generator to connect sources without a predefined evidence graph. *For example, one page links an instrument A to its inventor B, another identifies B's birthplace C, and a third names the river that flows through C. Asking which river flows through the birthplace of the inventor of A hides the intermediate entities while retaining an answer supported by the visited sources.*

Each submission contains a question, a short answer, ordered reasoning steps, and source URLs with supporting quotations. We check the required fields and literal answer leakage, verify URLs against the recorded searches and page visits, and match quotations to returned page text. Each question must draw on at least three and at most ten distinct source pages. The generated questions are then assessed for their difficulty relative to the current agent.

2.2 Routing by Current Solvability

Data probing combines closed-book screening, search rollouts, success-based routing, and reassessment of deferred questions into one stage. Its purpose is to identify questions suited to the agent's current learning needs.

In each round t, we first remove questions answered correctly without tools, then assess the remaining questions with $G = 8$ search attempts. Let $K_t(q)$ be the number of successful attempts for question q. Rare successes provide demonstrations of behavior the agent has not yet mastered, while successes on easier questions reinforce reliable search behavior. We use the following routing rule.

$$\mathrm{route}_t(q) = \begin{cases} \text{reassess later}, & K_t(q) = 0, \\ \text{successful-trajectory SFT}, & K_t(q) \in \{1, 2, 7, 8\}, \\ \text{RL with new trajectories}, & K_t(q) \in \{3, 4, 5, 6\}. \end{cases} \tag{1}$$

Questions with no observed success remain available for reassessment in later rounds. Policy improvement or further sampling may reveal successful trajectories and change their training role.

Selecting questions for Reinforcement Learning. Our RL stage uses a GRPO-based method that retains the group-relative task advantage $A_i = (R_i - \bar{R})/(s_R + \epsilon)$ for each trajectory (Shao et al., 2024). Here $R_i \in \{0, 1\}$ is the terminal reward, $\bar{R}$ and s_R are the group reward mean and sample standard deviation, and $\epsilon = 10^{-6}$. An all-success or all-failure group has $A_i = 0$ for every trajectory and supplies no task-reward gradient. For a fixed policy and question with success probability p, G independent attempts yield nonzero task advantages with probability

$$\Pr(\exists i : A_i \neq 0 \mid p) = m_G(p) = 1 - p^G - (1 - p)^G. \tag{2}$$

This probability is maximized at $p = 1/2$. Questions of intermediate solvability are therefore most likely to yield both positive and negative task advantages (Coelho et al., 2026). The RL route in Eq. 1 uses the probe success count as a heuristic for targeting this regime.

3 GC-GRPO: Gradient Correction for GRPO

The RL route favors questions likely to provide reward contrast. We next seek to make the resulting update more reliably reflect this reward-guided direction. Our method uses likelihood gradients from the same trajectories as references to predict and correct sampling fluctuations in the GRPO gradient.

3.1 Reference Gradients from the Same Trajectories

At each training step, the policy π_θ generates G trajectories for each of n questions. GRPO converts their terminal rewards into advantages A_i within each question group. Write $N = nG$, let L_i count the generated tokens in trajectory i, and denote its t-th token and preceding context by a_{it} and h_{it}. Prompts and tool observations enter the context but are excluded from the token index t. We construct K reference gradients by assigning each token a feature vector $\boldsymbol{w}_{it} \in \mathbb{R}^K$, whose scalar component $w_{it,j}$ weights its contribution to reference Z_j. The original GRPO ascent gradient X and the references, evaluated at the policy that generated the trajectories, are

$$\begin{aligned} X &= \frac{1}{N} \sum_{i=1}^{N} \frac{A_i}{L_i} \sum_{t=1}^{L_i} \nabla_\theta \log \pi_\theta(a_{it} \mid h_{it}) - \nabla_\theta \mathcal{L}_{\mathrm{KL}}, \\ Z_j &= \frac{1}{N} \sum_{i=1}^{N} \sum_{t=1}^{L_i} m_{it} w_{it,j} \nabla_\theta \log \pi_\theta(a_{it} \mid h_{it}), \qquad j = 1, \ldots, K. \end{aligned} \tag{3}$$

Here $m_{it} = 1$ for tokens sampled from the policy and $m_{it} = 0$ for tokens produced by a forced final-answer step without stochastic sampling. In the search agent, this step uses greedy decoding after retrieval ends or reaches its budget limit. Its output tokens still contribute to the original GRPO loss and its gradient X, but are excluded from the reference gradients Z_j.

To let the reference weights vary with generation progress and the current sampling distribution, we use five prefix features $\phi_{it} = (1, u_{it}, u_{it}^2, p_{\mathrm{end},it}, H_{it})^\top \in \mathbb{R}^5$. We define $u_{it} = s_{it}/B_{\mathrm{gen}}$, where s_{it}

counts generated tokens before the current token across all model calls in the trajectory and B_{gen} is a fixed trajectory-level generation budget chosen before sampling. It is not the realized trajectory length or the response limit for one model call. The constant gives an unweighted reference, while u_{it} and u_{it}^2 allow linear and quadratic dependence on progress. The probability $p_{\text{end},it}$ of ending the current assistant message and the Shannon entropy H_{it} of $\pi_\theta(\cdot \mid h_{it})$ distinguish prefixes with different stopping tendencies and uncertainty at the same progress.

To incorporate success on the same question, let $\bar{R}_{-i}$ be the mean reward of the other $G-1$ trajectories. We concatenate the prefix features with a second copy scaled by the centered peer success rate, giving

$$\boldsymbol{w}_{it} = (w_{it,1}, \ldots, w_{it,K})^\top = \left[\phi_{it}^\top,\ (\bar{R}_{-i} - \tfrac{1}{2})\phi_{it}^\top\right]^\top \in \mathbb{R}^K, \qquad K = 10. \tag{4}$$

The first five components use prefix information alone, and the remaining five additionally depend on peer success.

The construction follows the zero-mean identity used by score-function policy gradients and baselines (Williams, 1992; Greensmith et al., 2004; Zhong et al., 2021). The prefix features are known before the current token is drawn, and the peer success rate excludes the trajectory's own reward. With conditionally independent trajectories and these weights held constant during differentiation, the likelihood-score identity gives $\mathbb{E}[Z_j] = 0$ for every reference. The proof covers adaptive stopping and tool observations (Appendix A). The references can therefore track sampling fluctuations without adding an expected update of their own.

3.2 Using Reference Gradients to Correct GRPO

Let $\mu = \mathbb{E}[X]$ be the mean GRPO gradient, with expectations conditioned on the current policy, question batch, and statistics available before generation. The deviation $X - \mu$ is the sampling error we aim to reduce. Collect the references into $\mathcal{Z} = [Z_1, \ldots, Z_K]$, and let $\boldsymbol{c} \in \mathbb{R}^K$ assign one coefficient to each reference. We subtract their weighted combination to correct the part of the error they can predict. The coefficients are learned from preceding batches and held fixed for the current batch. Since each reference has zero conditional mean, this gives

$$Y = X - \mathcal{Z}\boldsymbol{c}, \qquad \mathbb{E}[Y] = \mathbb{E}[X] = \mu. \tag{5}$$

Figure 2 illustrates this correction, with $\widehat{e} = \mathcal{Z}\boldsymbol{c}_k$ denoting the error prediction at step k.

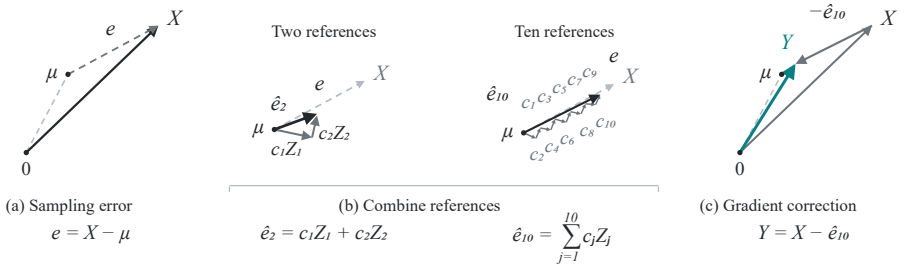


Figure 2: **Schematic of gradient correction in GC-GRPO.** (a) A sampled gradient X deviates from its conditional mean μ. (b) Two- and ten-reference examples show how weighted reference gradients predict this deviation. (c) Subtracting the ten-reference prediction $\widehat{e}_{10} = \mathcal{Z}\boldsymbol{c}_k$ gives the corrected gradient $Y = X - \widehat{e}_{10}$.

Choosing the correction coefficients. Mean preservation alone does not make the gradient estimate more precise. We seek coefficients that minimize the mean squared error (MSE), $\mathbb{E}\|Y - \mu\|^2$. Two quantities determine this fit: $\Gamma = \mathbb{E}[\mathcal{Z}^\top \mathcal{Z}]$ measures the magnitudes and overlap of the references, and $\boldsymbol{b} = \mathbb{E}[\mathcal{Z}^\top (X - \mu)]$ measures their association with the sampling error. Because the references have zero mean, $\boldsymbol{b} = \mathbb{E}[\mathcal{Z}^\top X]$ does not require knowing μ. An optimal coefficient vector is $\boldsymbol{c}^* = \Gamma^\dagger \boldsymbol{b}$, where the Moore–Penrose pseudoinverse $\Gamma^\dagger$ also accommodates redundant references. For any coefficient vector fixed before sampling, the MSE decomposes as (Greensmith et al., 2004; Zhong et al., 2021)

$$\mathbb{E}\|Y - \mu\|^2 = \mathbb{E}\|X - \mu\|^2 - \underbrace{\boldsymbol{b}^\top \Gamma^\dagger \boldsymbol{b}}_{\text{removable error}} + \underbrace{(\boldsymbol{c} - \boldsymbol{c}^*)^\top \Gamma (\boldsymbol{c} - \boldsymbol{c}^*)}_{\text{coefficient error}}. \tag{6}$$

Equation 6 separates the original estimation error, the reduction achievable with ideal coefficients, and the penalty for imperfect coefficients. The correction improves precision when the achievable reduction exceeds this penalty. Thus the remaining task is to estimate useful coefficients during training. These statements concern the gradient at the sampling policy, before gradient-norm clipping and the optimizer update (Appendix A).

Estimating the coefficients from token weights. Directly estimating Γ and $\boldsymbol{b}$ requires inner products involving separate, model-sized reference gradients. We approximate this fit using scalar token weights. The connection is that X and every Z_j in Eq. 3 sum the same token likelihood gradients with different weights. At the sampling policy, combining the advantage and sampled KL contributions gives a token's weight in X:

$$\alpha_{it} = \frac{1}{L_i}\left[A_i + \eta_{\mathrm{KL}}\left(\frac{\pi_{\mathrm{ref}}(a_{it} \mid h_{it})}{\pi_\theta(a_{it} \mid h_{it})} - 1\right)\right], \tag{7}$$

where π_{ref} is the frozen reference policy and η_{KL} is the KL coefficient. The two terms account for the task advantage and KL regularization. Subtracting $\mathcal{Z}\boldsymbol{c}$ changes this token weight from α_{it} to $\alpha_{it} - m_{it}\boldsymbol{w}_{it}^\top \boldsymbol{c}$. This suggests a tractable approximation: fit $\boldsymbol{w}_{it}^\top \boldsymbol{c}$ to α_{it} at stochastic tokens by weighted least squares. We weight each squared residual by ν_{it}, the squared norm of the token's output-layer likelihood gradient, computed from token probabilities and the final hidden state. Tokens with larger output-layer gradient norms therefore have more influence on the fit. For batch k, the required regression statistics are

$$\widetilde{\Gamma}_k = \frac{1}{N}\sum_{i,t} m_{it}\nu_{it}\boldsymbol{w}_{it}\boldsymbol{w}_{it}^\top, \qquad \widetilde{\boldsymbol{b}}_k = \frac{1}{N}\sum_{i,t} m_{it}\nu_{it}\alpha_{it}\boldsymbol{w}_{it}. \tag{8}$$

These statistics form a $K \times K$ matrix and a K-dimensional vector; they do not require storing the reference gradients. They approximate the ideal fit using same-token output-layer contributions, omitting cross-token interactions and hidden-layer derivatives. The resulting coefficients are therefore an approximation to $\boldsymbol{c}^*$, not the exact full-gradient optimum.

To track the changing policy, we maintain bias-corrected exponential averages $\widehat{\Gamma}_k$ and $\widehat{\boldsymbol{b}}_k$ of the batch statistics. A regularized regression gives the coefficients for the *next* batch:

$$\boldsymbol{c}_{k+1} = (\widehat{\Gamma}_k + \lambda D_k)^{-1}\widehat{\boldsymbol{b}}_k, \qquad D_k = \mathrm{diag}(\widehat{\Gamma}_k), \tag{9}$$

where D_k scales the ridge regularization by the feature second moments, with a positive numerical floor, and $\lambda > 0$ controls its strength. Batch k thus learns $\boldsymbol{c}_{k+1}$ while using the previously determined $\boldsymbol{c}_k$ for its own correction. Approximation and regularization affect the coefficient-error term in Eq. 6; this update order preserves the mean identity in Eq. 5.

Applying the correction in one backward pass. We implement the corrected gradient by adding a scalar term to the original GRPO loss. At step k, we generate trajectories, compute their original rewards and advantages, and use $\boldsymbol{c}_k$ in

$$\mathcal{L}_{\mathrm{corr}} = \mathcal{L}_{\mathrm{GRPO}} + \frac{1}{N}\sum_{i,t} m_{it}\,\mathrm{sg}(\boldsymbol{w}_{it}^\top \boldsymbol{c}_k)\log \pi_\theta(a_{it} \mid h_{it}), \tag{10}$$

where sg holds the scalar correction weight constant during differentiation. One backward pass produces the loss gradient $-Y$, followed by gradient-norm clipping and the optimizer update. Only after a successful update do we incorporate the batch statistics and compute $\boldsymbol{c}_{k+1}$. Warmup uses zero coefficients while collecting these statistics. The controller retains only running moments and coefficients, without storing past gradients. Appendix B gives the derivation and update settings.

4 Experiments

After describing the experimental setup, we compare with existing work (Section 4.2), then assess the adaptive training framework (Section 4.3), GC-GRPO within this framework (Section 4.4), and the contribution of individual components (Section 4.5).

Table 1: Apt and selected search agents on five benchmarks. Scores are accuracy. Further comparisons with general-purpose models are provided in Table 7. Green and yellow mark the best and second-best listed scores.

Model / system	Size	GAIA	BrowseComp	BrowseComp-ZH	SEAL-0	XBench-DS
AgentCPM-Explore	4B	63.9	24.1	29.1	40.5	70.0
LiteResearcher	4B	71.3	–	–	41.8	78.0
QUEST-SFT	4B	77.7	40.0	–	–	–
C-GRPO	4B	50.2	17.5	24.7	–	54.0
DR-Venus-RL	4B	64.4	29.1	37.7	–	74.7
ABSeeker	4B	81.6	**55.3**	**52.9**	–	77.0
RE-TRAC	4B	70.4	30.0	36.1	–	76.6
NestBrowse	4B	68.9	22.4	28.4	–	74.0
WebExplorer	8B	50.0	15.7	32.0	–	53.7
MiroThinker	8B	66.4	31.1	40.2	40.4	60.6
C-GRPO	30B-A3B	56.3	24.8	33.3	–	57.7
WebSailor-V2	30B-A3B	74.1	35.3	44.1	–	73.7
IterResearch	30B-A3B	72.8	37.3	45.2	39.6	–
WebLeaper	30B-A3B	73.2	38.8	–	**48.6**	72.0
Tongyi DeepResearch	30.5B-A3.3B	70.9	43.4	46.7	–	75.0
ASearcher-QwQ v2	32B	58.7	–	–	–	51.1
WebDancer-QwQ	32B	51.5	3.8	18.0	–	38.3
Qwen3.5 (initial)	4B	70.87	29.54	32.18	31.53	66.00
Apt (ours)	4B	**85.44**	48.66	47.75	47.75	**82.00**

4.1 Experimental Setup

Model and benchmarks. Training is based on Qwen3.5-4B. Evaluation covers five benchmarks: the full BrowseComp (1,266 questions) (Wei et al., 2025) and BrowseComp-ZH (289) (Zhou et al., 2025) sets; the 103 text-only GAIA validation questions (Mialon et al., 2024); the complete 111-question SEAL-0 subset of SealQA (Pham et al., 2026); and the original 100-question XBench-DS release (xbench Team, 2025). Dataset provenance and version details are provided in Appendix C.

Agent harness and tools. We use the native DeepSeek Harness v0.1.1-rc.2 (DeepSeek-AI, 2026), with stage-specific tools registered through its MCP interface. The question generator receives `search`, `read_page`, and `submit_qa`, together with the native `todo_write` planning tool. During training and evaluation, the search agent uses `search`, `read_page`, and `PythonInterpreter`. Training primarily uses our local search engine over 21.5 million pages (41.6 million text chunks), with Serper-style search and Jina-style page-reading interfaces.[1] Benchmark evaluation is conducted on the live web. In both environments, a separate Qwen3.8-27B service extracts page evidence for the requested goal. Appendix C.3 details the local corpus, tool interfaces, and runtime settings.

Training and evaluation. Training trajectories are limited to 102,400 tokens and 150 tool-use turns. The adaptive training schedule is configured for up to 50 rounds. Question generation is performed by a frozen model, instantiated as DeepSeek-v4-flash in our experiments. Each round introduces 128 new questions and reassesses retained ones. After removing questions answered correctly without tools, we use the success count over $G = 8$ search trajectories per question to assign questions to SFT, RL, or later reassessment (Section 2.2). Each round uses five SFT epochs on the selected successful trajectories, followed by one GRPO epoch on the RL question pool. Each RL batch comprises eight questions with eight trajectories per question. The terminal RL reward is 1 if the normalized answer matches a reference or accepted alias and at least one page has been successfully retrieved, and 0 otherwise. Further training settings are given in Appendix D.

For evaluation, the context window is extended to 256K and the interaction budget to 300 tool-use turns. A separate Qwen3.8-27B judge assesses terminal answers against the question and reference answers, accepting semantically equivalent, complete answers and rejecting contradictions or missing required information. Incomplete trajectories and empty answers are counted as incorrect.

[1] Serper: `https://serper.dev/`; Jina Reader: `https://jina.ai/reader/`.

4.2 Comparison with Existing Work

Table 1 compares **Apt**, our trained 4B search agent, with existing search agents and includes its Qwen3.5-4B initialization as a reference. Apt improves over its initialization across all five benchmarks, with absolute gains of 14.57–19.11 percentage points. The largest gain occurs on BrowseComp, where accuracy increases from 29.56% to 48.66%. Among the compared search agents, Apt achieves the highest scores on GAIA (85.44%) and XBench-DS (82.00%), ahead of ABSeeker (81.6%) and LiteResearcher (78.0%), respectively. It ranks second on BrowseComp, BrowseComp-ZH, and SEAL-0, with its SEAL-0 score 0.85 points below WebLeaper.

4.3 Direct RL versus the Adaptive Training Framework

In this section, we experimentally assess the effectiveness of our proposed adaptive training framework. Both settings use the same 1,024 generated QA pairs and GC-GRPO: direct RL trains on all pairs, whereas the adaptive framework routes them through probing, SFT, and a selected RL pool (Section 2). With eight questions and eight trajectories per RL step (8×8), direct RL takes 128 steps and the adaptive route takes 82. At step 82, adaptive training reward is 0.567 versus 0.409 for direct RL; direct RL first reaches 0.567 at step 122 (Figure 3). This faster, less volatile rise benefits from excluding questions with little estimated RL utility from the RL pool.

Although final training rewards are similar, the final checkpoints differ on the five benchmarks (Table 2). The adaptive checkpoint scores higher on four benchmarks, gaining 6.00 points on XBench-DS and 4.76 on BrowseComp, but trails direct RL by 2.70 points on SEAL-0. These gains may also reflect SFT on difficult questions with rare successful trajectories, which provide direct supervision when RL feedback is sparse.

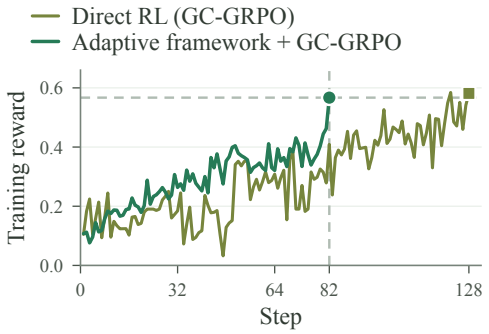


Figure 3: Training reward over RL steps for direct RL and the adaptive training framework.

Table 2: Accuracy (%) of the final checkpoints. Direct RL uses 128 steps; adaptive training uses SFT and 82 RL steps. Green marks the higher score.

Benchmark	Direct RL	Adaptive framework
GAIA	77.67	**80.58**
BrowseComp	31.51	**36.27**
BrowseComp-ZH	32.87	**35.99**
SEAL-0	**36.03**	33.33
XBench-DS	71.00	**77.00**

4.4 Effectiveness of GC-GRPO

In this section, we assess the effectiveness of GC-GRPO within the adaptive training framework. We compare it with GRPO, GSPO (Zheng et al., 2025), and Dr. GRPO (Liu et al., 2025b), while keeping question generation, data probing, SFT, and benchmark evaluation the same across methods. This comparison examines the RL algorithm after the framework has selected what to learn.

The four algorithms intervene at different points in the policy update. GRPO estimates critic-free advantages from rewards within each question group but still relies on a sampled gradient (Shao et al., 2024). GSPO moves importance weighting and clipping from tokens to whole trajectories to stabilize policy-ratio updates (Zheng et al., 2025). Dr. GRPO removes response-length and group-standard-deviation normalization, changing how trajectories and questions are weighted (Liu et al., 2025b). GC-GRPO instead constructs zero-mean reference gradients from token-prefix and peer-success features and fits their coefficients on preceding batches (Section 3.2). In Table 3, GC-GRPO leads on four of five benchmarks, trailing Dr. GRPO only on BrowseComp. This pattern is consistent with the intended effect of subtracting predictable sampling fluctuations from the GRPO gradient while preserving its expected direction at the sampling policy.

Table 3: Accuracy (%) of four RL algorithms within the adaptive training framework on five benchmarks. The best and second-best scores in each column are highlighted in green and pale yellow, respectively.

RL algorithm	GAIA	BrowseComp	BrowseComp-ZH	SEAL-0	XBench-DS
GRPO	77.67	45.73	35.29	35.14	79.00
GSPO	76.70	46.68	33.56	42.34	77.00
Dr. GRPO	83.50	**49.84**	43.94	40.54	75.00
GC-GRPO (ours)	**85.44**	48.66	**47.75**	**47.75**	**82.00**

To further test whether GC-GRPO reduces gradient sampling error, we directly measure gradients from training states. At each state, we fix the policy, reference model, and correction coefficients, repeatedly sample trajectories for the same questions, and compute the GRPO and GC-GRPO gradients on the same trajectories. Under the mean-preservation property in Section 3.2, the expected difference between their squared norms equals the reduction in gradient-estimation MSE, allowing us to estimate the error reduction achieved by the actual correction through repeated sampling.

As shown in Figure 4, despite batch-level fluctuations, the estimated mean MSE reduction is positive at all 20 model states, with each state's 95% confidence interval lying entirely above zero. These results show that, in this controlled setting, correction coefficients fitted to historical batch statistics effectively reduce gradient sampling error, providing direct empirical support for the gradient correction mechanism of GC-GRPO.



Figure 4: Gradient sampling error reduction with GC-GRPO at model states from 20 training rounds. Dots show the squared-norm difference $d = \|X\|^2 - \|Y\|^2$ between the GRPO gradient X and corrected gradient Y computed on the same trajectory batch. Under mean preservation, the expectation of d equals the MSE reduction; positive values indicate reduced error. Short horizontal lines and shaded bands show the sample mean at each state and its 95% confidence interval obtained by paired batch-level bootstrap, respectively.

4.5 Component Ablations

In this section, we isolate the roles of data probing, SFT, and the peer-success features in GC-GRPO. Starting from the initial Qwen3.5-4B model, we add Full GC-GRPO, data probing, and SFT in sequence. Probing denotes the complete screening and rollout-based routing stage. The final configuration keeps probing and SFT fixed but replaces Full GC-GRPO with a prefix-only variant: it retains the first five token-prefix features in Eq. 4 and omits the five corresponding peer-success-weighted features.

Table 4 shows that direct RL yields modest gains over the initial model on BrowseComp (+1.97 points) and BrowseComp-ZH (+0.69). Probing then raises these scores by 16.99 and 11.77 points, respectively, consistent with directing questions with useful reward variation to RL. SFT further improves GAIA by 4.86 points and BrowseComp-ZH by 3.11, using successful trajectories on difficult questions as supervision. With probing and SFT fixed, Full GC-GRPO exceeds the prefix-only variant on all five benchmarks (by 0.72–2.92 points), supporting the value of peer-success information.

Table 4: Component ablation accuracy (%). The all-cross row is the initial model before our training; checkmarks indicate active components. Green marks the best score in each benchmark.

Training components				Benchmark accuracy (%)				
Data probing	**SFT**	Prefix GC-GRPO	Full GC-GRPO	**GAIA**	**BrowseComp**	**BrowseComp-ZH**	**SEAL-0**	**XBench-DS**
×	×	×	×	70.87	29.54	32.18	31.53	66.00
×	×	×	✓	77.67	31.51	32.87	36.03	71.00
✓	×	×	✓	80.58	48.50	44.64	45.95	80.00
✓	✓	×	✓	**85.44**	**48.66**	**47.75**	**47.75**	**82.00**
✓	✓	✓	×	82.52	47.95	46.02	45.95	81.00

5 Related Work

Training experience and supervision for search agents. Search-R1 and ReSearch use RL for search (Jin et al., 2025; Chen et al., 2025). WebExplorer and WebShaper synthesize tasks (Liu et al., 2025a; Tao et al., 2026). SearchGym simulates a search environment (Zhang et al., 2026b). C-GRPO and ABSeeker add citation-aware or answer-backtracked supervision (Zhang et al., 2026a; Lu et al., 2026). Our framework routes evidence-grounded questions by the current agent's success and trains with final-answer rewards, without additional process-reward annotations.

Adaptive use of training questions. Adaptive curricula change which data or objective a policy uses as it improves. EvolveSearch uses selected RL trajectories for SFT before later RL rounds (Zhang et al., 2025); query recycling trains on mixed-reward groups and retains zero-variance queries for resampling (Coelho et al., 2026). Goldilocks RL predicts question difficulty for the current student, whereas Dr. Zero adapts a question proposer using solver feedback (Mahrooghi et al., 2026; Yue et al., 2026). We keep the generator fixed and use separate success probes to route questions to SFT, RL, or reassessment. RL then samples fresh groups after SFT and retains them regardless of their realized reward variation. This protocol assigns a question's learning role before the RL rollout, rather than selecting that rollout by its observed rewards.

Group-relative optimization and gradient control variates. GRPO obtains advantages from grouped rewards without a learned critic (Shao et al., 2024). GSPO changes policy-ratio clipping, Dr. GRPO changes normalization, DaGRPO masks low-distinctiveness sample pairs, and Discounted Beta–Bernoulli estimation uses reward history to estimate success probabilities (Zheng et al., 2025; Liu et al., 2025b; Xie et al., 2025; Kim et al., 2026). Policy-gradient variance reduction uses zero-mean likelihood scores, optimal baselines, and learned control variates (Greensmith et al., 2004; Zhong et al., 2021); RLOO uses other responses' rewards as a leave-one-out baseline (Ahmadian et al., 2024). Unlike PCGrad's multi-task gradient projection and HamiPose's source-target calibration (Yu et al., 2020; Li et al., 2026b), GC-GRPO retains GRPO's group-relative advantages and constructs zero-mean reference gradients from token prefixes and peer rewards. Historical coefficients fix the correction before sampling, preserving its conditional mean; their fit determines error reduction.

6 Conclusion

We develop an adaptive training framework for search agents that pairs question routing with GC-GRPO gradient correction. Routing assigns questions to SFT, RL, or reassessment according to the agent's current success, while GC-GRPO uses zero-mean reference gradients to reduce sampling noise without changing the expected GRPO gradient at the sampling policy. Apt, the resulting 4B agent, improves over its pretrained base model on all five benchmarks. Controlled comparisons show benefits from both components, and a separate gradient diagnostic finds lower sampling error with GC-GRPO. These results indicate that matching questions to suitable learning objectives and improving the precision of reward-driven updates are complementary ways to train search agents. The next challenge is to retain these gains as agents scale and search environments change.

AI USE STATEMENT

We used large language models to assist with writing and revising experimental code and with language editing and polishing of the manuscript. They also helped organize some experimental results and identify grammatical errors in the text.

ETHICS STATEMENT

This work involves neither human participants nor animal experiments, and it does not target private personal information or security-sensitive data for training, evaluation, or release.

REPRODUCIBILITY STATEMENT

Sections 2 and 3 describe question construction, training-data routing, and GC-GRPO; Appendices A and B provide the derivation and implementation details of the gradient correction. Section 4.1 and Appendices C and D document model and training settings, benchmark versions, tool interfaces, and grading procedures. Sections 4.3–4.5 present the controlled comparisons and ablations. We will publicly release all code developed for this work, including the adaptive training framework and GC-GRPO implementation.

REFERENCES

Arash Ahmadian, Chris Cremer, Matthias Gallé, Marzieh Fadaee, Julia Kreutzer, Olivier Pietquin, Ahmet Üstün, and Sara Hooker. Back to basics: Revisiting REINFORCE-style optimization for learning from human feedback in LLMs. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 12248–12267, 2024. doi: 10.18653/v1/2024.acl-long.662. URL https://aclanthology.org/2024.acl-long.662/.

Lei Bai, Zongsheng Cao, Yang Chen, Zhiyao Cui, Shangheng Du, Yue Fan, Shiyang Feng, Zijie Guo, Haonan He, Liang He, Xiaohan He, Shuyue Hu, Yusong Hu, Songtao Huang, Yichen Jiang, Hao Li, Xin Li, Dahua Lin, Weihao Lin, Fenghua Ling, Dongrui Liu, Zhuo Liu, Wenjie Lou, Runmin Ma, Chunjiang Mu, Haoyang Peng, Tianshuo Peng, Jinxin Shi, Luohe Shi, Boyuan Sun, Zelin Tan, Shengji Tang, Yan Teng, Qianyi Wang, Xiaosong Wang, Yiming Wu, Yi Xie, Xiangchao Yan, Jingqi Ye, Peng Ye, Fangchen Yu, Jiakang Yuan, Bihao Zhan, Bo Zhang, Chen Zhang, Shufei Zhang, Shuaiyu Zhang, Wenlong Zhang, Yiqun Zhang, Junpeng Zhao, Zhijie Zhong, Bowen Zhou, and Yuhao Zhou. Scaling the horizon, not the parameters: Reaching trillion-parameter performance with a 35B agent, 2026. URL https://arxiv.org/abs/2606.30616.

Haotian Chen, Xin Cong, Shengda Fan, Yuyang Fu, Ziqin Gong, Yaxi Lu, Yishan Li, Boye Niu, Chengjun Pan, Zijun Song, Huadong Wang, Yesai Wu, Yueying Wu, Zihao Xie, Yukun Yan, Zhong Zhang, Yankai Lin, Zhiyuan Liu, and Maosong Sun. AgentCPM-Explore: Realizing long-horizon deep exploration for edge-scale agents, 2026. URL https://arxiv.org/abs/2602.06485.

Mingyang Chen, Linzhuang Sun, Tianpeng Li, Haoze Sun, Yijie Zhou, Chenzheng Zhu, Haofen Wang, Jeff Z. Pan, Wen Zhang, Huajun Chen, Fan Yang, Zenan Zhou, and Weipeng Chen. ReSearch: Learning to reason with search for LLMs via reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 38, 2025. doi: 10.52202/085713-2858.

João Coelho, João Magalhães, Bruno Martins, and Chenyan Xiong. Effective reinforcement learning for agentic search by recycling zero-variance queries during training. *arXiv preprint arXiv:2606.10709*, 2026. URL https://arxiv.org/abs/2606.10709.

DeepSeek-AI. DeepSeek Harness. Official software repository, 2026. URL https://github.com/deepseek-ai/deepseek-harness. Release dsh-v0.1.1-rc.2.

Evan Greensmith, P. Bartlett, and Jonathan Baxter. Variance reduction techniques for gradient estimates in reinforcement learning. *Journal of Machine Learning Research*, 5:1471–1530, 2004. URL https://www.jmlr.org/papers/volume5/greensmith04a/greensmith04a.pdf.

InternScience Team. Agents-A1-4B: Model card. Official model release, 2026. URL https://huggingface.co/InternScience/Agents-A1-4B. Long-horizon search comparison, including 4B and 35B-A3B systems; accessed September 24, 2026.

Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan Ö. Arık, Dong Wang, Hamed Zamani, and Jiawei Han. Search-R1: Training LLMs to reason and leverage search engines with reinforcement learning. In *Conference on Language Modeling*, 2025. URL https://openreview.net/forum?id=Rwhi91ideu.

Haechan Kim, Soohyun Ryu, Gyouk Chu, Doohyuk Jang, and Eunho Yang. Discounted beta–bernoulli reward estimation for sample-efficient reinforcement learning with verifiable rewards. In *International Conference on Machine Learning*, 2026. URL https://arxiv.org/abs/2603.18444v2.

Baixuan Li, Jialong Wu, Wenbiao Yin, Kuan Li, Zhongwang Zhang, Huifeng Yin, Zhengwei Tao, Liwen Zhang, Pengjun Xie, Jingren Zhou, Yong Jiang, Wentao Zhang, and Zhiqiang Gao. Nested browser-use learning for agentic information seeking. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 22911–22923, San Diego, California, United States, jul 2026a. Association for Computational Linguistics. doi: 10.18653/v1/2026.acl-long.1049. URL https://aclanthology.org/2026.acl-long.1049/.

Jiawen Li, Fei Jiang, Dandan Zhu, and Aimin Zhou. HamiPose: Hamiltonian optimization for unsupervised domain adaptive pose estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 13856–13865, 2026b. URL https://openaccess.thecvf.com/content/CVPR2026/papers/Li_HamiPose_Hamiltonian_Optimization_for_Unsupervised_Domain_Adaptive_Pose_Estimation_CVPR_2026_paper.pdf.

Kuan Li, Zhongwang Zhang, Huifeng Yin, Rui Ye, Yida Zhao, Liwen Zhang, Litu Ou, Dingchu Zhang, Xixi Wu, Xinmiao Yu, Jialong Wu, Xinyu Wang, Zile Qiao, Zhen Zhang, Yong Jiang, Pengjun Xie, Fei Huang, Zhi-Qin John Xu, Shuai Wang, Minhao Cheng, and Jingren Zhou. WebSailor-V2: Bridging the chasm to proprietary agents via synthetic data and scalable reinforcement learning. In *The Fourteenth International Conference on Learning Representations*, 2026c. URL https://openreview.net/forum?id=HuP16O5SJf.

Zhuofeng Li, Dongfu Jiang, Xueguang Ma, Haoxiang Zhang, Ping Nie, Yuyu Zhang, Kai Zou, Jianwen Xie, Yu Zhang, and Wenhu Chen. OpenResearcher: A fully open pipeline for long-horizon deep research trajectory synthesis. *arXiv preprint arXiv:2603.20278*, 2026d. URL https://arxiv.org/abs/2603.20278.

Junteng Liu, Yunji Li, Chi Zhang, Jingyang Li, Aili Chen, Ke Ji, Weiyu Cheng, Zijia Wu, Chengyu Du, Qidi Xu, Jiayuan Song, Zhengmao Zhu, Wenhu Chen, Pengyu Zhao, and Junxian He. WebExplorer: Explore and evolve for training long-horizon web agents. *arXiv preprint arXiv:2509.06501*, 2025a. URL https://arxiv.org/abs/2509.06501.

Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. Understanding R1-Zero-like training: A critical perspective. In *Second Conference on Language Modeling*, 2025b. URL https://openreview.net/forum?id=5PAF7PAY2Y.

Yijun Lu, Rui Ye, Jiajun Wang, Yuwen Du, Tian Jin, Songhua Liu, and Siheng Chen. ABSeeker: Training long-horizon search agents via answer-backtracked credit assignment, 2026. URL https://arxiv.org/abs/2608.05102.

Ilia Mahrooghi, Aryo Lotfi, and Emmanuel Abbe. Goldilocks RL: Tuning task difficulty to escape sparse rewards for reasoning. *arXiv preprint arXiv:2602.14868*, 2026. URL https://arxiv.org/abs/2602.14868v2.

Grégoire Mialon, Clémentine Fourrier, Craig Swift, Thomas Wolf, Yann LeCun, and Thomas Scialom. GAIA: A benchmark for general AI assistants. In *International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=fibxvahvs3.

MiroMind Team. MiroThinker-1.7 & H1: Towards heavy-duty research agents via verification. *arXiv preprint arXiv:2603.15726*, 2026. URL https://arxiv.org/abs/2603.15726.

Thinh Pham, Nguyen Nguyen, Pratibha Zunjare, Weiyuan Chen, Yu-Min Tseng, and Tu Vu. SealQA: Raising the bar for reasoning in search-augmented language models. In *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/a0e0f9c39b7f6d07a266a3326216ec40-Abstract-Conference.html.

Bince Qu, Wanli Li, Bo Pan, Jianyu Zhang, Zheng Liu, Pan Zhang, Wei Chen, and Bo Zhang. LiteResearcher: A scalable agentic RL training framework for deep research agent. In *Conference on Language Modeling*, 2026. URL https://arxiv.org/abs/2604.17931v5. Accepted to COLM 2026.

Qwen Team. Qwen3.5-35B-A3B model card. Official Hugging Face model card, 2026. URL https://huggingface.co/Qwen/Qwen3.5-35B-A3B. Comparison table includes Qwen3.5-27B and Qwen3.5-35B-A3B; accessed 2026-09-26.

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

Zhengwei Tao, Jialong Wu, Wenbiao Yin, Pu Wu, Junkai Zhang, Baixuan Li, Haiyang Shen, Kuan Li, Liwen Zhang, Xinyu Wang, Wentao Zhang, Yong Jiang, Pengjun Xie, Fei Huang, and Jingren Zhou. WebShaper: Agentically data synthesizing via information-seeking formalization. In *International Conference on Learning Representations*, 2026. URL https://iclr.cc/virtual/2026/poster/10008023.

Tongyi DeepResearch Team. Tongyi deepresearch technical report. *arXiv preprint arXiv:2510.24701*, 2025. URL https://arxiv.org/abs/2510.24701.

Venus Team, Sunhao Dai, Yong Deng, Jinzhen Lin, Yusheng Song, Guoqing Wang, Xiaofeng Wu, Yuqi Zhou, Shuo Yang, Zhenzhe Ying, Zhanwei Zhang, Changhua Meng, and Weiqiang Wang. DR-Venus: Towards frontier edge-scale deep research agents with only 10K open data, 2026. URL https://arxiv.org/abs/2604.19859.

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. URL https://arxiv.org/abs/2504.12516.

Ronald J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8:229–256, 1992. doi: 10.1007/BF00992696. URL https://doi.org/10.1007/BF00992696.

xbench Team. Eval card xbench-DeepSearch. Official benchmark evaluation card, 2025. URL https://xbench.org/files/Eval%20Card%20xbench-DeepSearch.pdf. Accessed 2026-09-23.

Jian Xie, Tianhe Lin, Zilu Wang, Yuting Ning, Yuekun Yao, Tianci Xue, Zhehao Zhang, Zhongyang Li, Kai Zhang, Yufan Wu, Shijie Chen, Boyu Gou, Mingzhe Han, Yifei Wang, Vint Lee, Xinpeng Wei, Xiangjun Wang, Yu Su, and Huan Sun. Quest: Training frontier deep research agents with fully synthetic tasks. *arXiv preprint arXiv:2605.24218*, 2026. URL https://arxiv.org/abs/2605.24218.

Xuan Xie, Xuan Wang, Wenjie Wang, Shuai Chen, and Wei Lin. DaGRPO: Rectifying gradient conflict in reasoning via distinctiveness-aware group relative policy optimization. *arXiv preprint arXiv:2512.06337*, 2025. URL https://arxiv.org/abs/2512.06337.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations*, 2023. URL https://arxiv.org/abs/2210.03629.

Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. Gradient surgery for multi-task learning. In *Advances in Neural Information Processing Systems*, 2020. URL https://papers.neurips.cc/paper/2020/hash/3fe78a8acf5fda99de95303940a2420c-Abstract.html.

Zhenrui Yue, Kartikeya Upasani, Xianjun Yang, Suyu Ge, Shaoliang Nie, Yuning Mao, Zhe Liu, and Dong Wang. Dr. Zero: Self-evolving search agents without training data. *arXiv preprint arXiv:2601.07055*, 2026. URL https://arxiv.org/abs/2601.07055.

Ding-Chu Zhang, Yida Zhao, Jialong Wu, Liwen Zhang, Baixuan Li, Wenbiao Yin, Yong Jiang, Yu-Feng Li, Kewei Tu, Pengjun Xie, and Fei Huang. EvolveSearch: An iterative self-evolving search agent. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 13123–13136. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.emnlp-main.663. URL https://aclanthology.org/2025.emnlp-main.663/.

Jiajie Zhang, Xin Lv, Ling Feng, Lei Hou, and Juanzi Li. Chaining the evidence: Robust reinforcement learning for deep search agents with citation-aware rubric rewards. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 20744–20765, San Diego, California, United States, jul 2026a. Association for Computational Linguistics. doi: 10.18653/v1/2026.acl-long.950. URL https://aclanthology.org/2026.acl-long.950/.

Xichen Zhang, Ziyi He, Yinghao Zhu, Sitong Wu, Shaozuo Yu, Meng Chu, Wenhu Zhang, Haoru Tan, and Jiaya Jia. SearchGym: Bootstrapping real-world search agents via cost-effective and high-fidelity environment simulation. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 18633–18665. Association for Computational Linguistics, 2026b. doi: 10.18653/v1/2026.acl-long.848. URL https://aclanthology.org/2026.acl-long.848/.

Chujie Zheng, Shixuan Liu, Mingze Li, Xiong-Hui Chen, Bowen Yu, Chang Gao, Kai Dang, Yuqiong Liu, Rui Men, An Yang, Jingren Zhou, and Junyang Lin. Group sequence policy optimization. *arXiv preprint arXiv:2507.18071*, 2025. URL https://arxiv.org/abs/2507.18071.

Yuanyi Zhong, Yuan Zhou, and Jian Peng. Coordinate-wise control variates for deep policy gradients. *arXiv preprint arXiv:2107.04987*, 2021. URL https://arxiv.org/abs/2107.04987.

Hongyi Zhou, Kai Ye, Erhan Xu, Jin Zhu, Ying Yang, Shijin Gong, and Chengchun Shi. Demystifying group relative policy optimization: Its policy gradient is a U-statistic. *arXiv preprint arXiv:2603.01162*, 2026. URL https://arxiv.org/abs/2603.01162.

Peilin Zhou, Bruce Leon, Xiang Ying, Can Zhang, Yifan Shao, Qichen Ye, Dading Chong, Zhiling Jin, Chenxuan Xie, Meng Cao, Yuxin Gu, Sixin Hong, Jing Ren, Jian Chen, Chao Liu, and Yining Hua. BrowseComp-ZH: Benchmarking web browsing ability of large language models in chinese. *arXiv preprint arXiv:2504.19314*, 2025. URL https://arxiv.org/abs/2504.19314.

Jialiang Zhu, Gongrui Zhang, Xiaolong Ma, Lin Xu, Miaosen Zhang, Ruiqi Yang, Song Wang, Kai Qiu, Zhirong Wu, Qi Dai, Ruichun Ma, Bei Liu, Yifan Yang, Chong Luo, Zhengyuan Yang, Linjie Li, Lijuan Wang, Weizhu Chen, Xin Geng, and Baining Guo. RE-TRAC: REcursive TRAjectory Compression for deep search agents. In *International Conference on Machine Learning*, 2026. URL https://arxiv.org/abs/2602.02486.

A Score Identity and Conditional Error

Conditioning and reference weights. Fix the current parameters, question batch, and information available before generation, denoted jointly by $\mathcal{F}$. The coefficient $\boldsymbol{c}$ is measurable with respect to $\mathcal{F}$. For trajectory i, let τ_{-i} denote all other trajectories in its question group and assume trajectories have independent generation and tool-response randomness conditional on their questions. The peer reward $\bar{R}_{-i}$ depends only on τ_{-i}. Conditional on $\mathcal{F}$, τ_{-i}, and the prefix h_{it}, the weight $w_{it,j}$ is therefore known before token a_{it} is drawn. This remains true for the termination probability and entropy features: they depend on the current policy and prefix, not on the sampled token. Their numerical values, peer rewards, and the coefficient are detached in the auxiliary loss.

Let m_{it} indicate that opportunity t uses stochastic sampling from the policy. It equals zero for deterministic finalization and, if trajectories are padded to the fixed budget, for inactive positions. The sampling mode is fixed before drawing the current token, so a sampled termination token is included. Assume a finite generation budget, integrable scores with finite second moments, parameter-independent support, and that differentiation can pass through the probability sum. At an active opportunity,

$$\begin{aligned}&\mathbb{E}[m_{it}w_{it,j}\nabla_\theta \log \pi_\theta(a_{it} \mid h_{it}) \mid \mathcal{F}, \tau_{-i}, h_{it}] \\ &\quad = m_{it}w_{it,j}\sum_a \pi_\theta(a \mid h_{it})\nabla_\theta \log \pi_\theta(a \mid h_{it}) = 0.\end{aligned} \tag{11}$$

The indicator m_{it} is determined by the prefix, so stopping after an observed token is allowed. Summing over the bounded opportunities and trajectories, then applying iterated expectation, gives $\mathbb{E}[Z_j \mid \mathcal{F}] = 0$ for every reference. Hence

$$\mathbb{E}[X - \mathcal{Z}\boldsymbol{c} \mid \mathcal{F}] = \mathbb{E}[X \mid \mathcal{F}] = \mu. \tag{12}$$

References need not be independent: peer weights couple different trajectory terms, and their correlations enter the joint Gram matrix. The exclusion of the current trajectory's reward is essential. Substituting its group-normalized advantage into the reference weight would violate the conditioning argument. The argument also requires that other trajectories do not change the current trajectory's environment through shared mutable state.

Sampling and objective boundaries. We use the temperature-adjusted sampling distribution at stochastic positions. In the implementation examined here, temperature is 0.8 with no top-p or top-k truncation. Deterministic finalization remains in the baseline loss but has zero sampling score. Tool observations are context, not generated tokens. The position feature is normalized by the fixed generation budget; division by the realized trajectory length would generally invalidate Eq. 11. Likewise, post-generation rejection of unsuccessful trajectories is not covered by this identity. The training construction uses one update per newly generated batch at its sampling policy; an uncorrected score evaluated after further updates on old trajectories has no such guarantee.

The expectation preserved here is that of the implemented GRPO gradient. The group estimator has also been analyzed as a U-statistic (Zhou et al., 2026). With group-standardized rewards and its sampled KL term, this gradient need not be an unbiased gradient of expected task reward. The identity applies before gradient-norm clipping and Adam; it does not assert equal expected parameter displacement after those nonlinear operations.

Joint error decomposition. All expectations below condition on $\mathcal{F}$. With $\mathcal{Z} = [Z_1, \ldots, Z_K]$, define

$$\Gamma = \mathbb{E}[\mathcal{Z}^\top \mathcal{Z}], \qquad \boldsymbol{b} = \mathbb{E}[\mathcal{Z}^\top (X - \mu)] = \mathbb{E}[\mathcal{Z}^\top X], \qquad V_X = \mathbb{E}\|X - \mu\|^2. \tag{13}$$

Expanding the squared norm gives $V_X - 2\boldsymbol{c}^\top \boldsymbol{b} + \boldsymbol{c}^\top \Gamma \boldsymbol{c}$. If v belongs to the null space of Γ, then $\mathcal{Z}v = 0$ almost surely, so $v^\top \boldsymbol{b} = 0$. Thus $\boldsymbol{b}$ belongs to the range of Γ, and completing the square with $\boldsymbol{c}^* = \Gamma^\dagger \boldsymbol{b}$ gives

$$\mathbb{E}\|X - \mathcal{Z}\boldsymbol{c} - \mu\|^2 = V_X - \boldsymbol{b}^\top \Gamma^\dagger \boldsymbol{b} + (\boldsymbol{c} - \boldsymbol{c}^*)^\top \Gamma (\boldsymbol{c} - \boldsymbol{c}^*). \tag{14}$$

Here $\Gamma^\dagger$ is the Moore–Penrose pseudoinverse. The optimal correction removes $\boldsymbol{b}^\top \Gamma^\dagger \boldsymbol{b}$; approximate coefficients incur the final, nonnegative error term. Adding references cannot worsen the population optimum because zero coefficients recover the smaller family. Finite-data fitting, regularization, and policy drift can nevertheless increase the realized error. Mean preservation holds for any finite coefficient fixed before sampling, whereas error reduction requires a sufficiently accurate coefficient.

B Coefficient Estimation and Implementation

Feature family and baseline interpretation. Partition the coefficient vector into two five-dimensional blocks, $\boldsymbol{c}^{(0)}$ and $\boldsymbol{c}^{(1)}$. The scalar baseline implemented by the references is

$$\boldsymbol{w}_{it}^{\top}\boldsymbol{c} = \phi_{it}^{\top}\boldsymbol{c}^{(0)} + (\bar{R}_{-i} - \tfrac{1}{2})\phi_{it}^{\top}\boldsymbol{c}^{(1)}.$$

Thus the second block allows the coefficients of the prefix basis to vary linearly with peer success. The constant feature includes the simplest uniform baseline, while the other features expand the family using information available before the current token. For a termination-token set $\mathcal{E}(h_{it})$ specified by the generation protocol, the distribution features are

$$p_{\mathrm{end},it} = \sum_{a\in\mathcal{E}(h_{it})} \pi_\theta(a \mid h_{it}),$$
$$H_{it} = -\sum_{a} \pi_\theta(a \mid h_{it}) \log \pi_\theta(a \mid h_{it}).$$

In the controlled gradient diagnostic, $B_{\mathrm{gen}} = 98{,}304$ generated tokens across the trajectory, distinct from the 8,192-token response limit for an individual model call. The ending-token set is $\mathcal{E}(h_{it}) = \{\texttt{<|im_end|>}\}$, whose token ID is 151645 in the diagnostic model. This token ends an assistant message or model call, which may be followed by another search turn. The literal `<answer>` and `</answer>` tags delimit answer content but do not define the ending-token event.

Termination probability and entropy describe the sampling distribution, not answer correctness. The score identity permits any detached prefix features satisfying its moment assumptions, so it does not uniquely select these five features or establish their optimality. They define a compact approximation family. Its error reduction depends on correlation with the original gradient and the accuracy of coefficient estimation, as quantified by Eq. 6.

A token regression for the coefficient. Directly forming the full-gradient regression moments requires separate reference gradients. Instead, we estimate the shared coefficient with a token regression whose statistics can be computed without an auxiliary backward pass. To connect this regression to Eq. 3, write $\ell_{it} = \log \pi_\theta(a_{it} \mid h_{it})$ and $\ell_{it}^{\mathrm{ref}} = \log \pi_{\mathrm{ref}}(a_{it} \mid h_{it})$. The sampled KL term in Eq. 21 contributes

$$-\nabla_\theta \mathcal{L}_{\mathrm{KL}} = \frac{1}{N}\sum_{i=1}^{N}\sum_{t=1}^{L_i} \frac{\eta_{\mathrm{KL}}}{L_i} \left\{\exp(\ell_{it}^{\mathrm{ref}} - \ell_{it}) - 1\right\} \nabla_\theta \log \pi_\theta(a_{it} \mid h_{it}).$$

At the sampling policy, where the importance ratio equals one and clipping is locally inactive, combining this term with the advantage contribution gives

$$\begin{aligned} X &= \frac{1}{N}\sum_{i=1}^{N}\sum_{t=1}^{L_i} \alpha_{it} \nabla_\theta \log \pi_\theta(a_{it} \mid h_{it}), \\ \alpha_{it} &= \frac{A_i + \eta_{\mathrm{KL}}\{\exp(\ell_{it}^{\mathrm{ref}} - \ell_{it}) - 1\}}{L_i}, \qquad \eta_{\mathrm{KL}} = 0.001. \end{aligned} \tag{15}$$

Here L_i is the original loss's generated-token count. The factor $1/N$ remains outside the token sum, and the KL contribution above uses the gradient-ascent sign convention. Thus Eq. 15, equivalently Eq. 7, collects the original terms of Eq. 3 without changing its token set or objective.

For an affine output layer with last hidden representation r_{it}, bias, and temperature T, its sampling-score squared norm is

$$\nu_{it} = T^{-2}(1 + \|r_{it}\|^2)\|e_{a_{it}} - \pi_\theta(\cdot \mid h_{it})\|^2, \tag{16}$$

where $e_{a_{it}}$ is the one-hot token vector. The common factor T^{-2} can be omitted from both regression moments without changing the normalized-ridge solution below. Using the ten-dimensional weights $\boldsymbol{w}_{it}$ from Section 3, each batch supplies

$$\widetilde{\Gamma}_k = \frac{1}{N}\sum_{i,t} m_{it}\nu_{it}\boldsymbol{w}_{it}\boldsymbol{w}_{it}^{\top}, \qquad \widetilde{\boldsymbol{b}}_k = \frac{1}{N}\sum_{i,t} m_{it}\nu_{it}\alpha_{it}\boldsymbol{w}_{it}. \tag{17}$$

These are weighted least-squares statistics for predicting the token coefficient α_{it}. They approximate the full-gradient regression by retaining output-layer contributions within each token. They omit hidden-layer derivatives, inner products across tokens and trajectories, and the interaction between deterministic-finalization gradients in X and stochastic scores in $\mathcal{Z}$. Accordingly, $\widetilde{\Gamma}_k$ and $\widetilde{\boldsymbol{b}}_k$ are not unbiased estimates of the full-gradient moments, and their solution is not claimed to be $\boldsymbol{c}^*$.

Update order and running statistics. At the start of step k, we fix $\boldsymbol{c}_k$ from previously committed statistics. The policy then generates trajectories, receives rewards, computes GRPO advantages, and forms the corrected loss. Regression observations are computed from this same batch with gradients disabled. Only after the parameter update succeeds are they committed to exponential moving averages:

$$M_k^{\Gamma} = \beta M_{k-1}^{\Gamma} + (1-\beta)\widetilde{\Gamma}_k, \qquad M_k^{b} = \beta M_{k-1}^{b} + (1-\beta)\widetilde{\boldsymbol{b}}_k, \qquad \beta = e^{-1/16}. \tag{18}$$

Starting from zero, divide the moments by $1-\beta^k$ after k successful updates to obtain $\widehat{\Gamma}_k$ and $\widehat{\boldsymbol{b}}_k$. These symbols refer to smoothed proxy moments. Let $D_k = \operatorname{diag}(\max\{(\widehat{\Gamma}_k)_{jj}, 10^{-30}\})$, with the maximum applied separately to each diagonal entry. The next coefficient is

$$\boldsymbol{c}_{k+1} = D_k^{-1/2}\Big(D_k^{-1/2}\widehat{\Gamma}_k D_k^{-1/2} + 0.01I\Big)^{-1} D_k^{-1/2}\widehat{\boldsymbol{b}}_k. \tag{19}$$

During warmup, the coefficients remain zero while the controller collects moments. The same ten coefficients then apply across parameters, tokens, trajectories, and questions. Only the ten cross moments, 55 distinct symmetric Gram entries, coefficient vector, and counters persist. No past trajectory-specific records or full gradient vectors are retained. Computing statistics from the current batch is compatible with the mean identity because they affect only subsequent coefficients.

Fused differentiation and cost. Define $S_j = N^{-1}\sum_{i,t} m_{it}\,\mathrm{sg}(w_{it,j})\ell_{it}$, where sg denotes stop-gradient. Then

$$\mathcal{L}_{\mathrm{corr}} = \mathcal{L}_{\mathrm{GRPO}} + \sum_{j=1}^{K} \mathrm{sg}(c_{k,j})S_j, \qquad -\nabla_\theta \mathcal{L}_{\mathrm{corr}} = X - \mathcal{Z}\boldsymbol{c}_k = Y. \tag{20}$$

The reference terms are combined as scalars before differentiation, so training needs one backward pass and no reference-gradient bank. A token's added scalar weight is $\boldsymbol{w}_{it}^{\top}\boldsymbol{c}_k$; it is computed from ten shared coefficients rather than fitted separately for that token. The original clipped GRPO and KL terms remain intact. Inserting a modified advantage inside the clipped surrogate would not generally implement this additive loss.

The proxy introduces feature computation, vocabulary reductions, and a 10×10 linear solve. A forward-reuse implementation shares policy and reference evaluations with the baseline loss.

Baseline loss and implementation scope. For loss-side token likelihoods at temperature 0.8, let $\rho_{it} = \exp(\ell_{it} - \mathrm{sg}(\ell_{it}^{\mathrm{old}}))$ and $d_{it} = \ell_{it} - \ell_{it}^{\mathrm{ref}}$. The baseline is

$$\mathcal{L}_{\mathrm{GRPO}} = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{L_i}\sum_{t=1}^{L_i}\left[-\min\{\rho_{it}A_i, \mathrm{clip}(\rho_{it}, .8, 1.2)A_i\} + .001(e^{-d_{it}} - 1 + d_{it})\right]. \tag{21}$$

The reducer pools generated tokens across all model-call segments of a trajectory before taking its mean, then averages trajectories. Advantages use the sample standard deviation of the question group's rewards, with 10^{-6} added to the denominator. The reference is frozen at stage initialization. Entropy regularization and dual clipping are disabled. The denominator of ρ_{it} is a detached pre-update likelihood. Deterministic finalization remains in this surrogate with the loss-side temperature, but its auxiliary score is zero. Weight decay and gradient clipping, if applied, work outside this loss.

C Evaluation Protocols

This section documents benchmark versions, inference settings, and tool interfaces.

Table 5: **Benchmark scope.** Evaluation uses binary answer correctness.

Benchmark	Questions	Evaluated split / scope
BrowseComp	1,266	Full set
BrowseComp-ZH	289	Full Chinese set
GAIA	103	2023 text-only validation
SEAL-0	111	SEAL-0 subset of SealQA
XBench-DS	100	Original release

C.1 Datasets

Table 5 identifies the evaluation sets used for the five benchmark comparisons.

BrowseComp (Wei et al., 2025) tests persistent web browsing to find hard-to-locate facts with short, verifiable answers. BrowseComp-ZH (Zhou et al., 2025) tests Chinese-web browsing with multi-hop questions across 11 domains. GAIA (Mialon et al., 2024) poses real-world questions that require reasoning, web browsing, tool use, and, in the full benchmark, multimodal understanding. We use its 103 text-only validation questions, excluding those that require attached files. SEAL-0, the main subset of SealQA (Pham et al., 2026), tests factual reasoning when web search returns conflicting, noisy, or unhelpful evidence. XBench-DS (xbench Team, 2025) evaluates deep search through questions that require planning, retrieval, and reasoning across web sources. We use its original 100-question release.

C.2 Inference and Grading

All five benchmarks use the trained 4B policy with live-web retrieval. The agent can search, read pages, and run Python before submitting a final answer. Serper provides search results, while Jina fetches pages. A separate Qwen3.8-27B service extracts evidence relevant to the agent's reading goal from fetched pages. After the trajectory ends, a Qwen3.8-27B judge compares the final answer with the question and reference answer. It accepts semantically equivalent, complete answers and rejects contradictions or missing required information. Incomplete trajectories and empty final answers are scored incorrect.

C.3 Harness Composition and Tool Interfaces

We use DeepSeek Harness v0.1.1-rc.2 through its Python SDK, JSON-RPC runtime, and plugins. A stdio-based MCP server exposes the task tools. Separate plugins restrict native tools, stop generation after a successful QA submission, and finalize search answers. Each trajectory records the tool schemas visible to the model and a session trace.

Table 6: **Tools available at each stage.** Training uses the local corpus; benchmark evaluation uses live-web retrieval.

Stage	Available tools	Retrieval backend
QA generation	`search`, `read_page`, `submit_qa`, `todo_write`	Local corpus
Tool-free probe	None	None
Search probe / RL	`search`, `read_page`, `PythonInterpreter`	Local corpus
Benchmark evaluation	`search`, `read_page`, `PythonInterpreter`	Serper search; Jina Reader

Only the generator receives `submit_qa` and the native `todo_write` tool. Shell, filesystem, and subagent tools are unavailable at every stage. The generator's `submit_qa` call contains the question, reference answer, optional aliases, reasoning steps, evidence records, and language. Evidence records contain a search query, URL, quotation, and the fact supported. Subsequent validation checks that URLs and quotations occur in the same trajectory's observed retrieval results. The generator has no Python execution tool. Search agents receive the question rather than the reference answer and may use Python for calculations or processing retrieved evidence.

Tool input schemas. The following JSON Schema objects list the model-visible input fields and validation constraints; field descriptions are omitted for space. The displayed `search` name corresponds to `mcp__custom__google_search`. The other custom tools carry the same MCP prefix, while `todo_write` is a native Harness tool.

submit_qa

```
{
  "type": "object",
  "properties": {
    "question": {"type": "string", "minLength": 20, "maxLength": 2000},
    "answer": {"type": "string", "minLength": 1, "maxLength": 100},
    "answer_aliases": {
      "type": "array", "maxItems": 8,
      "items": {"type": "string", "minLength": 1, "maxLength": 100}
    },
    "language": {"type": "string", "minLength": 2, "maxLength": 16},
    "reasoning_steps": {
      "type": "array", "minItems": 2, "maxItems": 12,
      "items": {"type": "string", "minLength": 4, "maxLength": 1000}
    },
    "evidence": {
      "type": "array", "minItems": 2, "maxItems": 16,
      "items": {
        "type": "object",
        "properties": {
          "query": {"type": "string", "minLength": 1, "maxLength": 2048},
          "url": {"type": "string", "minLength": 1, "maxLength": 8192},
          "quote": {"type": "string", "minLength": 20, "maxLength": 4000},
          "supports": {"type": "string", "minLength": 4, "maxLength": 1000}
        },
        "required": ["query", "url", "quote", "supports"],
        "additionalProperties": false
      }
    }
  },
  "required": ["question", "answer", "reasoning_steps", "evidence"],
  "additionalProperties": false
}
```

search

```
{
  "type": "object",
  "properties": {
    "query": {
      "type": "string",
      "minLength": 1,
      "maxLength": 2048
    }
  },
  "required": ["query"],
  "additionalProperties": false
}
```

read_page

```
{
  "type": "object",
  "properties": {
    "url": {
      "type": "string",
      "minLength": 1,
      "maxLength": 8192
    },
    "goal": {
      "type": "string",
      "minLength": 1,
      "maxLength": 4096
    }
  },
  "required": ["url", "goal"],
  "additionalProperties": false
}
```

PythonInterpreter

```
{
  "type": "object",
  "properties": {
    "code": {
      "type": "string",
      "minLength": 1,
      "maxLength": 100000
    }
  },
  "required": ["code"],
  "additionalProperties": false
}
```

todo_write

```
{
  "type": "object",
  "properties": {
    "todos": {
      "type": "array",
      "items": {
        "type": "object",
        "properties": {
          "content": {"type": "string"},
          "status": {
            "type": "string",
            "enum": [
              "pending", "in_progress",
              "completed"
            ]
          }
        },
        "required": ["content", "status"],
        "additionalProperties": false
      }
    }
  },
  "required": ["todos"]
}
```

Local corpus and retrieval. The local search engine serves a fixed corpus combining the January 2026 English Wikipedia dump and OpenResearcher-Corpus (Li et al., 2026d).[2] The LanceDB index contains 21,484,094 page records and 41,615,719 text chunks. The preprocessing pipeline removes markup and short documents, then splits text into chunks with a target length of 512 tokens and 64-token overlap, with sentence-boundary adjustment and short-tail merging. Chunks are encoded with Qwen3-Embedding-4B into 2,560-dimensional vectors and indexed with IVF-PQ using the dot-product metric; a separate page table retains titles and full text.

QA generation, probing, and RL use vector retrieval. Retrieved chunks are grouped by their parent page, and pages are ranked by their best-matching chunk (`vector/min`). The agent requests up to 50 page results per query, each with a title, URL, and snippet from the matched chunk. Returned URLs retain an internal page identifier so that `read_page` retrieves the corresponding stored text. The server defaults are 300 candidate chunks, 100 probed IVF partitions, a refinement factor of 2, and a 500-character snippet limit.

The search agent is served with a 131,072-token context window during question probing. Search trajectories are limited to 150 steps and 102,400 tokens; SFT and RL also use a 102,400-token sequence limit. The RL server is configured for 102,912 tokens, including a 512-token margin. Benchmark execution uses a 262,144-token server context and trajectory limit, allows up to 300 steps, and reserves the final 8,192 tokens for answering. The inference adapter controls benchmark sampling at temperature 0.85, top-$p = 0.95$, and top-$k = 20$. At finalization, the harness restricts search tools and requests a final answer. Automatic context compaction is disabled.

D Training Settings

The production run is configured for up to 50 rounds. Each round uses five SFT epochs followed by one GRPO epoch. The frozen DeepSeek-v4-flash generator proposes 128 new questions at temperature 0.8, using vector retrieval and up to 150 generation turns. Evidence validation checks traceable source visits and matching quotations. Tool-free screening uses one response at temperature zero. For questions not solved without tools, $G = 8$ search trajectories are sampled at temperature 0.8 with top-$p = 1$. A complete group is required before assigning the learning role from its success count, and search-trajectory success requires at least one successful page retrieval.

SFT uses Adam, a cosine learning rate from 5×10^{-7} to 10^{-7}, no warmup, weight decay 0.1, and gradient clipping at 1.0. Its global batch size is 64, microbatch size is 1, and worker sequence limit is 102,400. An admission filter excludes trajectories above 60,000 estimated tokens rather than

[2]Corpus sources: `https://dumps.wikimedia.org/enwiki/20260101/`; `https://huggingface.co/datasets/OpenResearcher/OpenResearcher-Corpus`.

truncating them. Packing is disabled. The loss covers generated content, excludes tool observations, and follows the configured empty-thinking mask.

RL uses eight questions with eight trajectories each per update, a 102,400-token context limit, an 8,192-token response limit, and temperature 0.8. The global batch contains 64 trajectories; context-split segments are not additional trajectories. GRPO uses reward-standard-deviation normalization, clipping thresholds 0.2/0.2, a k_3 KL coefficient of 0.001, and no entropy bonus. Adam uses learning rate 5×10^{-7}, $(\beta_1, \beta_2) = (0.9, 0.98)$, weight decay 0.1, and gradient clipping at 1.0. The configured actor occupies eight H200 GPUs with context parallelism 8. Each new batch produces one optimizer update.

E Extended Comparison

Table 7: Extended comparison of Apt with general-purpose models and selected search agents. BC denotes BrowseComp, BC-ZH denotes BrowseComp-ZH, and XBench denotes XBench-DS. Scores are percentages. Rows citing AgentCPM-Explore reproduce values from its Table 1. Gray rows show general-purpose models; the best and second-best scores are highlighted only among the listed search agents as **best** and second-best.

Model / system	Size	GAIA	BC	BC-ZH	SEAL-0	XBench
General-purpose models (with tools)						
Qwen3.5 (initial)	4B	70.87	29.56	32.18	31.53	66.00
Qwen3.5 (Qwen Team, 2026)	27B	–	61.0	62.1	47.2	–
Qwen3.5 (Qwen Team, 2026)	35B-A3B	59.8	61.0	69.5	41.4	77.0
Nex-N2-mini (InternScience Team, 2026)	35B-A3B	82.5	74.1	–	49.6	82.0
Agents-A1 (Bai et al., 2026)	35B	96.0	75.5	–	56.4	86.0
MiniMax-M2 (Chen et al., 2026)	230B-A10B	75.7	44.0	48.5	–	72.0
GLM-4.6 (Chen et al., 2026)	355B-A32B	71.9	45.1	49.5	–	70.0
Search agents						
AgentCPM-Explore (Chen et al., 2026)	4B	63.9	24.1	29.1	40.5	70.0
LiteResearcher (Qu et al., 2026)	4B	71.3	–	–	41.8	78.0
QUEST-SFT (Xie et al., 2026)	4B	77.7	40.0	–	–	–
C-GRPO (Zhang et al., 2026a)	4B	50.2	17.5	24.7	–	54.0
DR-Venus-RL (Venus Team et al., 2026)	4B	64.4	29.1	37.7	–	74.7
ABSeeker (Lu et al., 2026)	4B	81.6	**55.3**	**52.9**	–	77.0
RE-TRAC (Zhu et al., 2026)	4B	70.4	30.0	36.1	–	76.6
NestBrowse (Li et al., 2026a)	4B	68.9	22.4	28.4	–	74.0
WebExplorer (Liu et al., 2025a)	8B	50.0	15.7	32.0	–	53.7
MiroThinker (Chen et al., 2026)	8B	66.4	31.1	40.2	40.4	60.6
C-GRPO (Zhang et al., 2026a)	30B-A3B	56.3	24.8	33.3	–	57.7
WebSailor-V2 (Li et al., 2026c)	30B-A3B	74.1	35.3	44.1	–	73.7
IterResearch (Chen et al., 2026)	30B-A3B	72.8	37.3	45.2	39.6	–
WebLeaper (Chen et al., 2026)	30B-A3B	73.2	38.8	–	**48.6**	72.0
Tongyi DeepResearch (Tongyi DeepResearch Team, 2025)	30.5B-A3.3B	70.9	43.4	46.7	–	75.0
ASearcher-QwQ v2 (Chen et al., 2026)	32B	58.7	–	–	–	51.1
WebDancer-QwQ (Chen et al., 2026)	32B	51.5	3.8	18.0	–	38.3
Apt (ours)	4B	**85.44**	48.66	47.75	47.75	**82.00**

Table 7 extends the search-agent comparison with general-purpose models. External scores come from the cited papers, model cards, or comparison tables and may use different evaluation protocols.