TRACE: TURN-LEVEL REWARD ASSIGNMENT VIA CREDIT ESTIMATION FOR LONG-HORIZON AGENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Multi-turn agents solve complex tasks through extended sequences of tool interactions before producing a final answer, making credit assignment a fundamental challenge during post-training. Outcome rewards provide reliable supervision for short-horizon reasoning, but become sparse and high-variance as trajectories grow to tens or hundreds of tool calls. They can also be misleading: a failed rollout may contain many useful actions that move the agent closer to the goal, yet outcome-only training assigns them the same negative advantage as the eventual mistake. We propose TRACE (Turn-level Reward Assignment via Credit Estimation), a dense credit-assignment method for agentic reinforcement learning. TRACE represents rollouts as state transitions at tool-call boundaries, obtains gold-answer log-probabilities from a frozen reference model, transforms them into log-ratio state values, and derives per-action rewards as Temporal-Difference changes in those values. This requires no additional critic or process-label training, and its one-step log-ratio TD component telescopes across redundant tool calls. On long-horizon complex search, TRACE substantially improves base-model tool-use ability using pure RL, without a cold-start supervised fine-tuning stage, an agentic mid-training stage, or training on live-web data. On the closed-web BrowseComp-Plus benchmark, it raises Qwen3-4B from 7.2 to 35.6 and Qwen3-30B-A3B from 8.4 to 42.6. The learned search behavior also transfers to open-web benchmarks, and the learning curves show earlier improvement and faster convergence during RL training.

1 INTRODUCTION

Large language model (LLM) agents increasingly solve complex tasks by reasoning and acting through many interactions with external environments, including web navigation, software engineering, and general computer use (Yao et al., 2023; Schick et al., 2023; Nakano et al., 2021; Yao et al., 2022; Deng et al., 2023; Zhou et al., 2024; Jimenez et al., 2024; Xie et al., 2024). Reinforcement learning with verifiable rewards (RLVR) has been effective for single-turn reasoning tasks such as mathematics and coding, where a deterministic checker can evaluate the final answer and provide a clean outcome reward (Shao et al., 2024; Guo et al., 2025). Applying the same idea to agents is appealing but insufficient: an agentic rollout may contain dozens of search, open, find, code, or other tool-use decisions before the final answer is verified. A single terminal reward says whether the trajectory succeeded, but it does not reveal which intermediate actions gathered necessary evidence, which were redundant, or which derailed the agent.

Dense turn-level rewards are therefore desirable, but obtaining them is challenging because partial progress in agent trajectories is not directly observed. As illustrated in Figure 1, a failed rollout can begin with useful search and open actions that make the answer easier to infer, while a successful rollout can contain redundant searches or accidental openings that do not contribute to the final answer. Outcome-only training assigns the same trajectory-level advantage to all of these turns, under-crediting productive exploration, over-crediting irrelevant actions, and increasing gradient variance as horizons grow (Sutton & Barto, 2018; Arjona-Medina et al., 2019; Ye et al., 2025). Prior work on process supervision offers finer feedback, but typically requires step-level labels, a strong LLM judge to score intermediate behavior (Zheng et al., 2023), or a trained process reward model whose scores may drift away from final-answer correctness (Uesato et al., 2022; Lightman et al., 2023; Wang et al., 2024a; Setlur et al., 2024; Yuan et al., 2024). This raises the central question of

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.

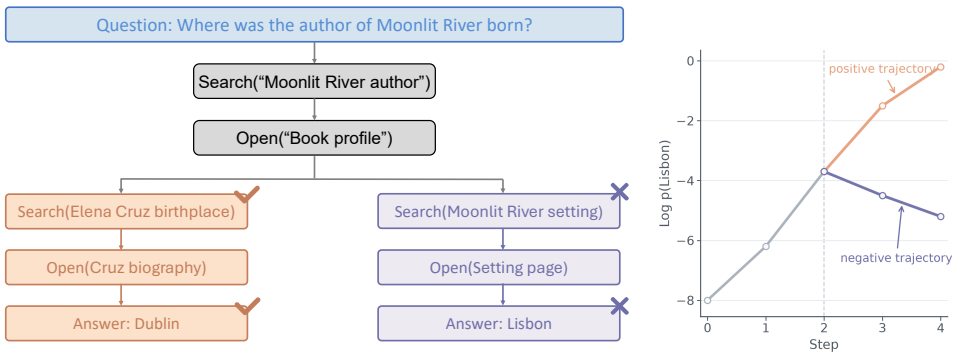

Figure 1: **Credit assignment at tool-call boundaries in a search trajectory.** The figure illustrates a long-horizon search rollout decomposed into tool-call turns. Early search and open actions can add task-relevant evidence to the transcript even when a later branch leads to an incorrect final answer. Outcome-reward training attaches one trajectory-level advantage to all actions in the rollout, whereas TRACE computes prefix values at tool boundaries and assigns turn credit from adjacent value changes.

this work: *how can we identify the tool calls that actually move a long-horizon agent toward the answer, and turn that signal into dense rewards without relying on step labels, a strong judge, or a trained process reward model?*

We propose TRACE (**T**urn-level **R**eward **A**ssignment via **C**redit **E**stimation), a critic-free credit-assignment framework that keeps the final verifier as the anchor while adding dense credit at tool-call boundaries. The key idea is to use a frozen reference model not as a judge, but as a stable probe of whether each trajectory prefix makes the gold answer more predictable. TRACE represents a rollout as state transitions at tool-call boundaries, scores each prefix by the reference model's gold-answer log-probability, and converts the score into a log-ratio state value measuring progress toward the answer. It then assigns turn rewards using temporal-difference (TD) changes between adjacent values: a tool call receives positive credit when its returned observation increases answer predictability, near-zero credit when it adds no useful evidence, and negative credit when it moves the trajectory away from the answer. Because the one-step TD credits telescope, redundant intermediate turns cannot inflate this component, while the cumulative credit remains aligned with the final reference-model state. The resulting rewards combine this dense TD signal with the standard outcome-level advantage (Sutton, 1988), preserving verifiable success as the final training target while distinguishing which turns deserve credit.

We evaluate TRACE on long-horizon complex search, a setting where agents must learn to interact with an environment, explore through multiple tool calls, and refine later actions from earlier observations. Because common multi-hop benchmarks can be solved by strong agents in only a few turns (Yang et al., 2018), we train on deeper synthetic search questions and evaluate on both closed-web and open-web deep-research benchmarks. Our recipe is pure RL: it uses outcome-level and turn-level rewards directly, without a cold-start supervised fine-tuning stage, an agentic mid-training stage, live-web training data, a strong judging model, or a trained process reward model. Despite this minimal supervision, TRACE substantially improves base-model tool use in both closed-web and open-web settings. On closed-web BrowseComp-Plus (Chen et al., 2025c), it raises Qwen3-4B from 7.2 to 35.6 and Qwen3-30B-A3B from 8.4 to 42.6. The learned behavior also transfers to open-web retrieval, with the 30B-A3B agent reaching 12.9 on BrowseComp, 52.0 on GAIA, and 45.0 on xbench-DeepSearch. In addition to final performance, the training curves show that TRACE starts improving earlier and converges faster, indicating that turn-level credit makes long-horizon tool use easier to learn from pure RL. These results show that verifier-anchored turn-level credit can teach base models to explore, interact with the environment, and refine tool use for complex long-horizon tasks, while remaining independent of expensive process supervision or strong judge-based feedback.

2 PRELIMINARIES

2.1 AGENTIC REINFORCEMENT LEARNING

In agentic reinforcement learning, a policy LLM π_θ solves a prompt $x \sim \mathcal{D}$ by interleaving assistant tokens, tool calls from an available tool set $\mathcal{T}$, and tool observations before producing a final answer. A common KL-regularized training objective is

$$\max_{\pi_\theta} \mathbb{E}_{x\sim\mathcal{D},\,\tau\sim\pi_\theta(\cdot|x;\mathcal{T})}\left[r_\phi(x,\tau)\right] - \beta_{\mathrm{KL}}\,\mathbb{D}_{\mathrm{KL}}[\pi_\theta(\tau \mid x;\mathcal{T}) \,\|\, \pi_{\mathrm{ref}}(\tau \mid x;\mathcal{T})]\,, \tag{1}$$

where τ denotes the complete rollout, r_ϕ is the reward function, π_{ref} is a reference policy, and β_{KL} controls the strength of the KL penalty. In reinforcement learning with verifiable rewards, r_ϕ is usually an outcome reward that is observed only after the rollout is complete, for example by checking whether the final answer satisfies an automatically verifiable criterion (Sutton & Barto, 2018; Shao et al., 2024). Unlike single-turn reasoning, agentic rollouts induce a distribution over both the intermediate interaction trajectory and the final answer (Yao et al., 2023; Schick et al., 2023; Nakano et al., 2021; Yao et al., 2022). We write the interaction part as alternating policy actions and environment observations. Let $\mathcal{R} = ((a_1, o_1), \ldots, (a_{T_\mathcal{R}}, o_{T_\mathcal{R}}))$ and $H_k = (x, a_1, o_1, \ldots, a_{k-1}, o_{k-1})$. Then

$$P_\theta(\mathcal{R}, y \mid x;\mathcal{T}) = \underbrace{\prod_{k=1}^{T_\mathcal{R}} \pi_\theta(a_k \mid H_k;\mathcal{T})\, P_{\mathrm{env}}(o_k \mid H_k, a_k;\mathcal{T})}_{\text{Tool interaction}} \cdot \underbrace{\prod_{t=1}^{T_y} \pi_\theta(y_t \mid y_{<t}, H_{T_\mathcal{R}+1};\mathcal{T})}_{\text{Answer generation}}, \tag{2}$$

where $\mathcal{R}$ is the multi-turn trajectory, including tool-call feedback, and y is the final answer. Gradients are taken through assistant actions and answer tokens, while tool observations are environment outputs rather than policy-generated tokens. This decomposition highlights the source of the credit-assignment problem: the terminal reward supervises a long sequence of heterogeneous policy decisions, but does not identify which turns made the answer more or less likely.

For credit assignment, it is useful to view the interaction trajectory at tool-call boundaries. Given a rollout with $T_\mathcal{R}$ tool interactions, let S_k denote the prefix containing the prompt and the first k action–observation pairs, optionally followed by the final-answer opener used to score answer tokens. The transition from S_k to S_{k+1} is induced by the next policy action and environment observation (a_{k+1}, o_{k+1}). These prefix transitions are the natural units for assigning credit to tool use, because they isolate how each interaction changes the information available to the agent.

Typical RL training builds on outcome-based agentic RL algorithms such as Group Relative Policy Optimization (GRPO) (Guo et al., 2025), which compare multiple rollouts from the same prompt and avoid a learned critic for the outcome-reward stage (Shao et al., 2024). However, the resulting group-relative advantage is still attached to the entire rollout. It can rank a successful trajectory above failed alternatives, but it cannot distinguish useful tool calls from redundant or harmful ones within the same trajectory. This limitation motivates a reward construction that preserves the reliability of outcome verification while exposing denser signal at the prefix-transition level. TRACE obtains this signal from temporal-difference (TD) changes in a prefix value (Sutton, 1988); Appendix A.1 reviews TD credit assignment.

3 TURN-LEVEL REWARD ASSIGNMENT VIA CREDIT ESTIMATION

We introduce TRACE, a critic-free credit-assignment method for long-horizon agent post-training. Instead of learning a critic over sparse terminal rewards, TRACE measures whether each tool interaction makes the gold answer more predictable under a frozen reference model. It splits rollouts at tool-call boundaries, maps prefix answer log-probabilities to log-ratio state values, uses TD changes in those values as turn-level rewards, and mixes the resulting turn-level advantage with the standard GRPO outcome advantage. The outcome reward remains the final arbiter of task success. Figure 2 illustrates the framework, and Algorithm 1 (Appendix A.2) summarizes the procedure.

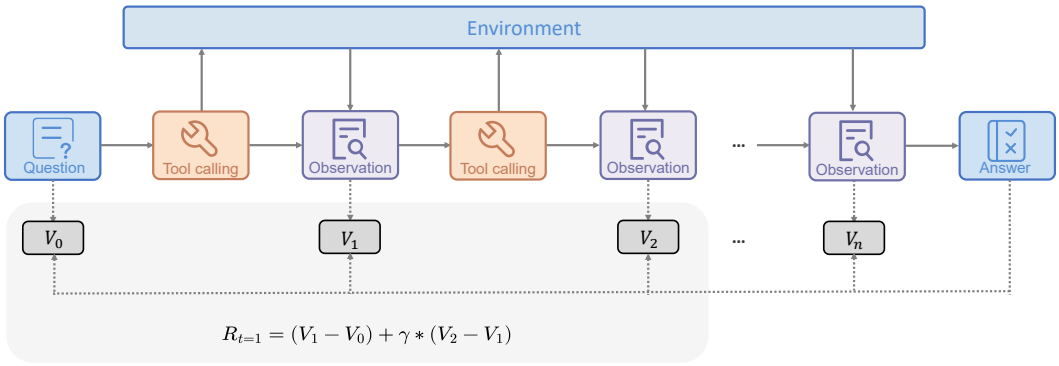

Figure 2: **TRACE reward construction at tool-call boundaries.** The figure shows a long-horizon rollout segmented by tool calls and observations. Each prefix state is scored by a frozen reference model, transformed into a log-ratio value, and compared with adjacent prefix values to obtain turn-level TD credit. The turn-level advantage is then combined with the trajectory-level outcome advantage for policy optimization.

3.1 Tool-Boundary States

Following the prefix-transition view in Section 2.1, we instantiate the notation used by TRACE. For a sampled trajectory on question x with gold answer $y^\star$, write

$$\tau = \big(x,\ a_1,\ o_1,\ a_2,\ o_2,\ \ldots,\ a_T,\ o_T,\ \hat{y}\big), \tag{3}$$

where a_k is the k-th assistant action, o_k is the corresponding tool observation, $\hat{y}$ is the final answer span, and T is the number of tool interactions. The terminal verifier returns $R = R(\hat{y}, y^\star)$. TRACE constructs one credit target for each transition $S_k \rightarrow S_{k+1}$ induced by (a_{k+1}, o_{k+1}), while the policy-gradient loss is applied only to policy-generated assistant actions and answer tokens; tool observations remain environment outputs and are masked from the loss. The remaining question is how to score whether a prefix transition improves the trajectory's ability to reach the gold answer.

3.2 State Value via Log-Ratio Answer Probability

Reference answer score. Long-horizon agentic RL makes critic learning brittle: the trajectory reward is sparse, observations shift the state after each tool action, and failed trajectories can still contain useful action–observation transitions. TRACE therefore replaces a learned critic with a *frozen* reference model π_{ref} (a copy of the policy initialization), used only to estimate how ready the single-trajectory prefix is to produce the gold answer. This is a natural proxy for progress in RLVR settings: the gold answer is available during training, and a transcript that has collected relevant evidence should make that answer easier for a fixed language model to conditionally predict. Freezing the model keeps this probe stable across policy updates and prevents it from becoming another learned reward model. For state S_k, the average gold-answer log-probability is

$$\bar{\ell}_k = \frac{1}{|y^\star|} \sum_{t=1}^{|y^\star|} \log \pi_{\text{ref}}(y_t^\star \mid S_k,\ y_{<t}^\star) \ \leq\ 0. \tag{4}$$

Larger (less negative) $\bar{\ell}_k$ means that the current transcript already contains more evidence for generating $y^\star$. All $\bar{\ell}_k$ are obtained in a single batched forward pass and are never optimized; final-answer correctness is still determined by the outcome verifier.

Log-ratio state value. The score $\bar{\ell}_k$ is a prefix-readiness measure, but its raw scale is not directly suitable for credit assignment. Raw changes in $\bar{\ell}_k$ can be misleading: the same absolute gain is minor when the trajectory is far from the answer but important when it nearly resolves the answer. For instance, shrinking the remaining gap from 0.2 to 0.1 removes half the uncertainty, whereas $5.1 \rightarrow 5.0$ barely changes it despite an identical raw gain. We therefore model the state value as

relative closure of the initial answer-likelihood gap. Define $d_k = -\bar{\ell}_k + \epsilon$ with offset $\epsilon > 0$, and set

$$V(S_k) \;=\; \log \frac{d_0}{d_k} \;=\; \log \frac{-\bar{\ell}_0 + \epsilon}{-\bar{\ell}_k + \epsilon}. \tag{5}$$

Thus $V(S_0) = 0$, and a larger $V(S_k)$ means that the action–observation history has closed a larger *fraction* of the initial gap. The later TD difference $V(S_{k+1}) - V(S_k)$ therefore measures the value added by the transition induced by action a_{k+1} and observation o_{k+1}.

3.3 Turn-Level Reward via Temporal-Difference Credit

One-step TD action credit. With no intermediate environment reward and discount 1, the credit assigned to the transition $S_k \xrightarrow{a_{k+1},\, o_{k+1}} S_{k+1}$ is the one-step TD change in value,

$$\delta_k \;=\; V(S_{k+1}) - V(S_k) \;=\; \log \frac{d_k}{d_{k+1}}, \qquad k = 0, \ldots, T-1. \tag{6}$$

This is positive when the action and returned observation make the gold answer more likely, equals $\log 2$ when the transition halves the remaining gap, is 0 when the gap is unchanged, and is negative when the transition moves away from the answer. Because the credits telescope,

$$\sum_{k=0}^{T-1} \delta_k \;=\; V(S_T) - V(S_0) \;=\; \log \frac{-\bar{\ell}_0 + \epsilon}{-\bar{\ell}_T + \epsilon},$$

so total credit depends only on the endpoints: redundant intermediate transitions cannot inflate it, and the agent is not rewarded for padding a trajectory. Appendix A.5 compares this log-ratio credit with raw and linearized alternatives.

K-step turn credit. The one-step TD credit above can miss delayed tool effects: for example, a `browser.search` may only surface candidate links, while the answer likelihood jumps after a later `browser.open` exposes decisive evidence. We therefore use a truncated K-step TD backup with $K \geq 1$: each turn receives discounted future TD progress from the current transition and the next few transitions in the same trajectory. For rollout g and turn k, let

$$h_{g,k} = \min(k + K - 1,\, T_g - 1) \tag{7}$$

be the last transition included in the backup window. The local progress credit assigned to the current turn is

$$c_{g,k}^{(K)} = \frac{1}{Z_{g,k}} \sum_{u=k}^{h_{g,k}} \gamma_{\mathrm{td}}^{u-k} \delta_{g,u}, \qquad Z_{g,k} = \sum_{u=k}^{h_{g,k}} \gamma_{\mathrm{td}}^{u-k}, \tag{8}$$

where γ_{td} discounts delayed evidence and $Z_{g,k}$ is the K-step normalization factor used in our reported runs. We use $K = 0$ only as a shorthand for disabling the dense TD backup entirely, not as an input to Eq. 8. When the backup window reaches the end of the trajectory, we additionally anchor the last turns to the verifiable outcome signal:

$$r_{g,k}^{\mathrm{turn}} = c_{g,k}^{(K)} + \mathbf{1}[h_{g,k} = T_g - 1]\, \lambda_{\mathrm{term}}\, \gamma_{\mathrm{td}}^{T_g - k}\, A_g^{\mathrm{out}}. \tag{9}$$

Here A_g^{out} is the standard GRPO group-relative advantage computed from the terminal reward, and λ_{term} controls the strength of the terminal anchor. The exponent $T_g - k$ treats final-answer generation as one additional transition after the last tool observation, so the last tool turn is discounted by one step before receiving terminal-outcome fill. This form makes the two sources of credit explicit: $c_{g,k}^{(K)}$ captures local answer-likelihood progress, while the terminal fill only affects turns whose look-ahead window reaches the final outcome. The endpoint telescoping property above applies exactly to the one-step component δ_k; the K-step backup and terminal fill intentionally trade this exact endpoint-only form for delayed credit propagation and outcome anchoring.

Joint optimization with outcome reward. We do not group-normalize the turn values. Instead, the TD-derived turn credit is used directly as a trajectory-local auxiliary signal and is jointly optimized with the GRPO outcome reward. For the outcome component, we use the standard group-relative normalization

$$A_g^{\text{out}} = \begin{cases} (R_g - \bar{R})/\sigma_R, & \sigma_R > 0, \\ 0, & \sigma_R = 0, \end{cases} \quad \bar{R} = G^{-1} \sum_{h=1}^{G} R_h, \quad \sigma_R = \sqrt{G^{-1} \sum_{h=1}^{G} (R_h - \bar{R})^2}.$$

For each token generated as part of tool-interaction action $a_{g,k+1}$, we set $\text{turn}(t) = k$; final-answer tokens use the outcome component with the answer-tail weighting described in Appendix A.4. The mixed per-token advantage for tool-interaction tokens is

$$\hat{A}_{g,t} = \alpha_{\text{out}} A_g^{\text{out}} + \alpha_{\text{turn}} r^{\text{turn}}_{g,\,\text{turn}(t)}, \tag{10}$$

where $\alpha_{\text{out}}, \alpha_{\text{turn}} \geq 0$ control the relative strength of terminal correctness and turn-level credit. Let $\mathcal{I}_g$ be the assistant-token indices used in the policy-gradient loss for rollout g, and let $\rho_{g,t}(\theta) = \pi_\theta(a_{g,t} \mid s_{g,t})/\pi_{\text{old}}(a_{g,t} \mid s_{g,t})$, where π_{old} is the frozen behavior-policy snapshot that generated the current rollout batch. We optimize the clipped GRPO objective

$$\mathcal{J}_{\text{TRACE}}(\theta) = \frac{1}{G} \sum_{g=1}^{G} \frac{1}{|\mathcal{I}_g|} \sum_{t \in \mathcal{I}_g} \min\Big(\rho_{g,t}(\theta) \hat{A}_{g,t}, \text{clip}(\rho_{g,t}(\theta), 1 - c_-, 1 + c_+)\, \hat{A}_{g,t} \Big), \tag{11}$$

which keeps the standard GRPO group-relative outcome signal while adding dense turn-level credit without normalizing the turn values across the prompt group.

4 Experiments

4.1 Experimental Setup

Training datasets. We train on synthetic multi-document search tasks built over the offline corpus released by OpenResearcher (Li et al., 2026). Unlike standard multi-hop QA benchmarks, which can often be solved with one search and one document open, these tasks require chained retrieval over multiple irreplaceable evidence documents (Appendix A.9).

Agentic harness. The policy acts in a ReAct-style harness (Yao et al., 2023) with three browser actions, `browser.search`, `browser.open`, and `browser.find`, over the OpenResearcher offline corpus served through a FAISS index built with Qwen3-Embedding-8B (Zhang et al., 2025c). The outcome reward checks normalized exact-match correctness of the final `<answer>` span (Appendix A.8).

Models. We use Qwen3-4B-Thinking-2507 as the primary controlled setting and Qwen3-30B-A3B-Thinking-2507 to test transfer to a larger backbone (Yang et al., 2025a). All controlled RL runs start directly from the base search policy without a cold-start supervised fine-tuning stage, a mid-training stage, or training on live-web data.

Training protocol. We optimize each student with Adam at a constant learning rate of 10^{-6}, a global batch size of 128, 8 rollouts per prompt, and up to 60 tool turns. TRACE uses the log-ratio credit in Eq. 6 with $K = 3$, $\gamma_{\text{td}} = 0.8$, $\alpha_{\text{out}} = 1.0$, and $\alpha_{\text{turn}} = 0.2$; Appendix A.4 gives the full protocol and hyperparameters.

Baselines. We compare with external deep-research agents, ASearcher-QwQ-32B (Gao et al., 2025), WebDancer-32B (Wu et al., 2025b), CutBill-30B-A3B (Wu et al., 2025a), and TongyiDS-30B-A3B (Tongyi DeepResearch Team et al., 2025), whose reported numbers are non-controlled reference points. The controlled baselines start from the same Qwen3 backbones: Base denotes the untrained search policy, and we train GRPO (Guo et al., 2025), GSPO (Zheng et al., 2025a), and GiGRPO (Feng et al., 2025) with the same browser action space, rollout protocol, training data, terminal reward, and evaluation interface as TRACE.

Model	BrowseComp-Plus	BrowseComp	GAIA	xbench-DeepSearch	Avg
Deep-research agents					
ASearcher-QwQ-32B	28.9	5.2	52.8	42.0	32.2
WebDancer-32B	32.4	3.8	51.5	39.0	31.7
CutBill-30B-A3B	35.1	13.4	46.6	43.0	34.5
TongyiDS-30B-A3B	44.4	43.4	70.9	75.0	58.4
Qwen3-4B-Thinking-2507					
Base	7.2	3.3	24.2	19.0	13.4
GRPO	30.0	5.1	38.8	44.0	29.5
GSPO	29.7	5.4	36.7	41.0	28.2
GiGRPO	27.7	4.4	37.9	36.0	26.5
TRACE	35.6	6.7	44.6	49.0	34.0
Qwen3-30B-A3B-Thinking-2507					
Base	8.4	4.4	34.1	20.0	16.7
GRPO	36.4	10.8	45.6	37.0	32.5
GSPO	39.7	11.8	46.6	35.0	33.3
GiGRPO	33.0	10.1	44.7	31.0	29.7
TRACE	42.6	12.9	52.0	45.0	38.1

Table 1: **Performance comparison on long-horizon deep-research benchmarks.** BrowseComp-Plus is evaluated in the closed-web setting, while BrowseComp, GAIA, and xbench-DeepSearch are evaluated with open-web retrieval. Avg reports the unweighted average over all four benchmarks. The top block lists external deep-research agents for reference, while the Qwen3 blocks compare controlled RL variants initialized from the same backbone and trained with the same browser interface, data, terminal reward, and evaluation protocol. Highlighted rows denote TRACE.

Evaluation. The closed-web benchmark is BrowseComp-Plus (Chen et al., 2025c), served by an offline Qwen3-Embedding-8B FAISS index over its released corpus. Open-web transfer is measured on BrowseComp (Wei et al., 2025a), GAIA (Mialon et al., 2024), and the Chinese QA benchmark xbench-DeepSearch (Chen et al., 2025a), with retrieval served by the Serper API (Serper.dev, 2026). Controlled ablations are single training runs, so small differences should be read as directional. Appendix A.3 gives the full experimental setup.

4.2 Main Results

TRACE remarkably improves base-model tool use and transfers beyond the training corpus. Table 1 shows that TRACE substantially improves base-model tool use while retaining generalization beyond the training environment. The most direct evidence comes from the closed-web BrowseComp-Plus setting, where TRACE raises Qwen3-4B from 7.2 to 35.6 and Qwen3-30B-A3B from 8.4 to 42.6 using only RL with outcome-level and turn-level rewards. This is a strong gain for a minimal recipe: the models are not first adapted with cold-start SFT, agentic mid-training, or live-web data, so the improvement suggests that the base backbones already contain useful reasoning and tool-use primitives, but need a denser credit signal to organize them into effective long-horizon behavior. The external-agent comparisons give the result additional context. Although those systems may use different data and training pipelines, the 4B TRACE agent already surpasses several larger deep-research agents on BrowseComp-Plus, while the 30B-A3B agent approaches the strong TongyiDS-30B-A3B result. The more important controlled test is whether this behavior transfers beyond the closed training corpus: despite being trained in a closed synthetic search environment, TRACE improves both backbones on open-web benchmarks served by an external search API; with Qwen3-30B-A3B, it reaches 12.9 on BrowseComp, 52.0 on GAIA, and 45.0 on the Chinese QA benchmark xbench-DeepSearch. This pattern suggests that TRACE is not merely fitting the closed retrieval corpus within our controlled setup; it teaches a more general interaction strategy for searching, reading observations, and refining later actions across different corpora and languages, though it does not close the gap to the strongest external deep-research system in Table 1.

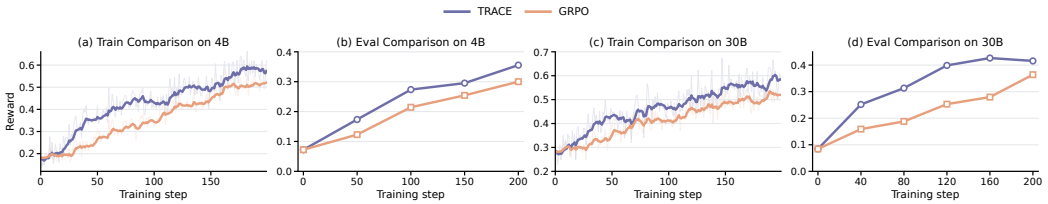

Figure 3: **Learning dynamics on BrowseComp-Plus.** (a) Training reward on Qwen3-4B. (b) Evaluation accuracy on Qwen3-4B. (c) Training reward on Qwen3-30B-A3B. (d) Evaluation accuracy on Qwen3-30B-A3B.

TRACE outperforms other RL methods across model scales. The controlled RL comparison clarifies why the gains come from credit assignment rather than from a change in model, data, or environment. All RL variants use the same backbone, browser interface, training data, terminal reward, and evaluation protocol; what differs is how the policy-gradient signal is constructed. Against outcome-only GRPO, TRACE asks the cleanest question: if the final correctness reward is unchanged, does adding turn-level TD credit help? The answer is yes on both scales, with the 4B four-benchmark average rising to 34.0 from GRPO's 29.5 and the 30B-A3B average rising to 38.1 from 32.5. This indicates that terminal rewards alone do not expose enough information about which parts of a long trajectory should be reinforced. GSPO and GiGRPO address a different aspect of RL optimization by changing sequence- or group-level estimation, but they still operate primarily at the trajectory level: they can compare rollouts, yet they do not assign credit to the intermediate interaction that made later success more likely. This distinction explains why TRACE is especially beneficial on BrowseComp-Plus and xbench-DeepSearch, where success depends on a chain of dependent environment interactions rather than a single decisive final answer. By keeping outcome correctness as the global anchor and adding temporally aligned progress credit within the trajectory, TRACE supplies the kind of supervision that long-horizon agent training is missing.

4.3 Learning Dynamics of TRACE

TRACE makes the model converge faster and to a better policy. Figure 3 shows that the TRACE curves start rising earlier, climb more steeply in the early stage, and stabilize at a higher plateau than the outcome-reward baselines. The same ordering appears in held-out BrowseComp-Plus accuracy with Qwen3-30B-A3B, where the 160-step TRACE checkpoint already exceeds the 200-step outcome-reward baseline. On Qwen3-4B, TRACE also expands interaction length earlier and faster than GRPO during early training; Appendix A.6 discusses both trends.

4.4 Ablation Study

Log-ratio credit assignment is more effective in this run. We ablate the turn-level credit format on BrowseComp-Plus while keeping the same Qwen3-4B training setup. Starting from outcome-only GRPO (30.0), adding dense transition rewards based on the raw log-probability delta improves the score to 32.4, and normalizing by the remaining answer-likelihood gap further improves it to 34.6. The proposed log-ratio TD credit achieves the best score in this run (35.5), suggesting that relative gap closure provides a more effective credit signal than absolute likelihood changes. This supports the design in Eq. 6: the log-ratio form better normalizes credit across states with different confidence levels and preserves a telescoping structure that discourages redundant trajectory extension. Appendix A.7 further ablates the turn-level reward coefficient, the TD look-ahead horizon K, and the reference checkpoint used for answer-readiness scoring.

Method	BC-Plus
GRPO	30.0
+ raw delta	32.4
+ remaining gap	34.6
+ log-ratio	35.5

Table 2: Credit-assignment ablation on Qwen3-4B.

5 Related Work

Post-training of LLMs for reasoning and agentic tasks. Reinforcement learning with verifiable rewards improves mathematical and symbolic reasoning by optimizing a trajectory-level outcome signal with clipped policy-gradient updates (Shao et al., 2024; Guo et al., 2025; Schulman et al., 2017), but this signal becomes less informative as decisions grow into multi-hop or browsing trajectories (Yang et al., 2018; Wei et al., 2025a; Chen et al., 2025c). Agentic post-training extends this regime to tool use, web navigation, software engineering, and computer control (Nakano et al., 2021; Yao et al., 2023; Schick et al., 2023; Qin et al., 2023; Zeng et al., 2023; Yao et al., 2022; Deng et al., 2023; Zhou et al., 2024; Jimenez et al., 2024; Xie et al., 2024). Search and deep-research agents apply outcome-based RL, synthetic trajectory construction, and tool-use training to search-augmented reasoning and deep browsing (Li et al., 2025b; Chen et al., 2025b; Jin et al., 2025b; Song et al., 2025; Sun et al., 2025; Mu et al., 2025; Li et al., 2025c; Wu et al., 2025b; Li et al., 2025a; Zheng et al., 2025b; Li et al., 2026; Wang et al., 2026b; Gao et al., 2025). Rather than designing a new agent interface or adding a stronger teacher, we study how to make the training signal more informative inside a multi-turn trajectory.

Credit assignment for long-horizon language agents. Temporal-difference learning, potential-based reward shaping, and return decomposition address delayed rewards by crediting the transitions that change the value of the current state (Sutton, 1988; Ng et al., 1999; Arjona-Medina et al., 2019). In language-model reasoning, process supervision and process reward models provide step-level feedback (Uesato et al., 2022; Lightman et al., 2023; Li & Li, 2025), while recent methods derive step credit from changes in the probability of reaching a correct answer, Monte Carlo rollouts, or policy–reference log-likelihood ratios (Setlur et al., 2024; Wang et al., 2024a; Yuan et al., 2024). For multi-turn agents, recent work studies stepwise rewards, hindsight credit, memory-aware credit assignment, search-agent RL, and training infrastructure for long-horizon tool use (Feng et al., 2025; Wang et al., 2025a; Dong et al., 2025b; Wang et al., 2025d; Zhao et al., 2025a; Wang et al., 2026a; Tan et al., 2026; Yan et al., 2026; Qu et al., 2025; Dong et al., 2025a; Jin et al., 2025a; Xi et al., 2025b; Chai et al., 2025; Zhao et al., 2025b; Zhang et al., 2025a). TRACE shares the goal of assigning credit below the trajectory level, but avoids training an additional critic or progress estimator, does not require Monte Carlo continuations, and does not depend on revisiting identical states. Appendix C provides an extended discussion.

6 Conclusion

We presented TRACE, a critic-free method for assigning turn-level credit in long-horizon agentic RL. At tool-call boundaries, a frozen reference model maps gold-answer log-probabilities to log-ratio state values, whose TD changes provide local rewards. Combining this signal with the GRPO outcome advantage preserves final-answer correctness as the training anchor without step labels, a strong LLM judge, or a trained critic or process reward model. TRACE outperforms outcome-only GRPO on all four benchmarks at both model scales, raising the four-benchmark average from 29.5 to 34.0 for Qwen3-4B and from 32.5 to 38.1 for Qwen3-30B-A3B. On BrowseComp-Plus, it improves the base policies from 7.2 to 35.6 and from 8.4 to 42.6, with gains transferring to open-web evaluation. These results establish frozen-reference turn credit as an effective signal for long-horizon search with compact, verifiable answers; long, structured, or open-ended outputs will require alternative state-value estimators. Appendix B discusses this limitation.

AI use statement

In this work, we used generative AI tools to generate synthetic data sets: the multi-document training questions were produced by an LLM-based generation pipeline with rejection sampling, and a separate LLM call independently re-derived each answer before an example was retained (Appendix A.9). We have not used generative AI tools to develop theoretical models or conceptual frameworks, formulate or prove mathematical claims, propose or refine hypotheses, or interpret results, and translation is not applicable to this work. Additionally, we used generative AI tools to draft and edit parts of the paper, create and edit software code, create and edit figures, search for and summarize related literature, suggest experimental parameters, format references, and suggest the title and structure of the paper. We have reviewed all AI-assisted work: every synthetic ques-

tion passed programmatic checks and independent LLM answer verification (Appendix A.9.3), the authors reviewed and tested AI-assisted code before using it in experiments, suggested hyperparameters were adopted only after evaluation in our own training runs, and all references were checked against the original publications. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REFERENCES

Jose A. Arjona-Medina, Michael Gillhofer, Michael Widrich, Thomas Unterthiner, Johannes Brandstetter, and Sepp Hochreiter. RUDDER: Return decomposition for delayed rewards. In *Advances in Neural Information Processing Systems*, 2019.

Ramakrishna Bairi, Atharv Sonwane, Aditya Kanade, D. C. Vageesh, Arun Iyer, Suresh Parthasarathy, Sriram Rajamani, B. Ashok, and Shashank Shet. CodePlan: Repository-level coding using LLMs and planning. *Proceedings of the ACM on Software Engineering*, 1(FSE): 675–698, 2024. doi: 10.1145/3643757.

Islem Bouzenia, Premkumar Devanbu, and Michael Pradel. RepairAgent: An autonomous, LLM-based agent for program repair. In *Proceedings of the 47th IEEE/ACM International Conference on Software Engineering*, pp. 2188–2200, 2025. doi: 10.1109/ICSE55347.2025.00157.

Jiajun Chai, Guojun Yin, Zekun Xu, Chuhuai Yue, Yi Jia, Siyu Xia, Xiaohan Wang, Jiwen Jiang, Xiaoguang Li, Chengqi Dong, Hang He, and Wei Lin. RLFactory: A plug-and-play reinforcement learning post-training framework for LLM multi-turn tool-use. *arXiv preprint arXiv:2509.06980*, 2025.

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.

Mingyang Chen, Tianpeng Li, Haoze Sun, Yijie Zhou, Chenzheng Zhu, Fan Yang, Zenan Zhou, Weipeng Chen, Haofen Wang, Jeff Z. Pan, Wen Zhang, and Huajun Chen. ReSearch: Learning to reason with search for LLMs via reinforcement learning. *arXiv preprint arXiv:2503.19470*, 2025b.

Zijian Chen, Xueguang Ma, Shengyao Zhuang, Ping Nie, Kai Zou, et al. BrowseComp-Plus: A more fair and transparent evaluation benchmark of deep-research agent. *arXiv preprint arXiv:2508.06600*, 2025c.

Xiang Deng, Yu Gu, Boyuan Zheng, Shijie Chen, Samuel Stevens, Boshi Wang, Huan Sun, and Yu Su. Mind2Web: Towards a generalist agent for the web. In *Advances in Neural Information Processing Systems*, 2023.

Guanting Dong, Yifei Chen, Xiaoxi Li, Jiajie Jin, Hongjin Qian, Yutao Zhu, Hangyu Mao, Guorui Zhou, Zhicheng Dou, and Ji-Rong Wen. Tool-Star: Empowering LLM-brained multi-tool reasoner via reinforcement learning. *arXiv preprint arXiv:2505.16410*, 2025a.

Guanting Dong, Hangyu Mao, Kai Ma, Licheng Bao, Yifei Chen, et al. Agentic reinforced policy optimization. *arXiv preprint arXiv:2507.19849*, 2025b.

Alexandre Drouin, Maxime Gasse, Massimo Caccia, Issam H. Laradji, Manuel Del Verme, Tom Marty, Léo Boisvert, Megh Thakkar, Quentin Cappart, David Vazquez, Nicolas Chapados, and Alexandre Lacoste. WorkArena: How capable are web agents at solving common knowledge work tasks? *arXiv preprint arXiv:2403.07718*, 2024.

Lang Feng, Zhenghai Xue, Tingcong Liu, and Bo An. Group-in-group policy optimization for LLM agent training. *arXiv preprint arXiv:2505.10978*, 2025.

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.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. DeepSeek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning. *Nature*, 645:633–638, 2025. doi: 10.1038/s41586-025-09422-z.

Sirui Hong, Mingchen Zhuge, Jiaqi Chen, Xiawu Zheng, Yuheng Cheng, Ceyao Zhang, Jinlin Wang, Zili Wang, Steven Ka Shing Yau, Zijuan Lin, Liyang Zhou, Chenyu Ran, Lingfeng Xiao, Chenglin Wu, and Jürgen Schmidhuber. MetaGPT: Meta programming for a multi-agent collaborative framework. In *International Conference on Learning Representations*, 2024.

Naman Jain, Jaskirat Singh, Manish Shetty, Liang Zheng, Koushik Sen, and Ion Stoica. R2E-Gym: Procedural environments and hybrid verifiers for scaling open-weights SWE agents. *arXiv preprint arXiv:2504.07164*, 2025.

Carlos E. Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik Narasimhan. SWE-bench: Can language models resolve real-world GitHub issues? In *International Conference on Learning Representations*, 2024.

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

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*, 2025b.

Jing Yu Koh, Robert Lo, Lawrence Jang, Vikram Duvvur, Ming Chong Lim, Po-Yu Huang, Graham Neubig, Shuyan Zhou, Ruslan Salakhutdinov, and Daniel Fried. VisualWebArena: Evaluating multimodal agents on realistic visual web tasks. *arXiv preprint arXiv:2401.13649*, 2024.

Kuan Li, Zhongwang Zhang, Huifeng Yin, Liwen Zhang, Litu Ou, Jialong Wu, Wenbiao Yin, Baixuan Li, Zhengwei Tao, Xinyu Wang, et al. WebSailor: Navigating super-human reasoning for web agent. *arXiv preprint arXiv:2507.02592*, 2025a.

Minghao Li, Yingxiu Zhao, Bowen Yu, Feifan Song, Hangyu Li, Haiyang Yu, Zhoujun Li, Fei Huang, and Yongbin Li. API-Bank: A comprehensive benchmark for tool-augmented LLMs. *arXiv preprint arXiv:2304.08244*, 2023.

Wendi Li and Yixuan Li. Process reward model with q-value rankings. In *Proceedings of the International Conference on Learning Representations*, 2025.

Xiaoxi Li, Guanting Dong, Jiajie Jin, Yuyao Zhang, Yujia Zhou, Yutao Zhu, Peitian Zhang, and Zhicheng Dou. Search-o1: Agentic search-enhanced large reasoning models. *arXiv preprint arXiv:2501.05366*, 2025b.

Xiaoxi Li, Jiajie Jin, Guanting Dong, Hongjin Qian, Yongkang Wu, Ji-Rong Wen, Yutao Zhu, and Zhicheng Dou. WebThinker: Empowering large reasoning models with deep research capability. *arXiv preprint arXiv:2504.21776*, 2025c.

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*, 2026.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. *arXiv preprint arXiv:2305.20050*, 2023.

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.

Jinyi Mu, Yifei Lu, and Michelle Smith. Llm-assisted incrementality (uplift) modeling for email advertising: From feature interactions to interpretable audience–creative–channel policies. *Journal of Advanced Computing Systems*, 3(1):31–48, 2023.

Jinyi Mu, Tong Ye, and Priya Patel. Offline counterfactual evaluation for advertising and recommendation slot policies: A reproducible study on the open bandit dataset (small). *Journal of Technology Informatics and Engineering*, 4(3):521–543, 2025.

Reiichiro Nakano, Jacob Hilton, Suchir Balaji, Jeff Wu, Long Ouyang, Christina Kim, Christopher Hesse, Shantanu Jain, Vineet Kosaraju, William Saunders, et al. WebGPT: Browser-assisted question-answering with human feedback. *arXiv preprint arXiv:2112.09332*, 2021.

Andrew Y. Ng, Daishi Harada, and Stuart Russell. Policy invariance under reward transformations: Theory and application to reward shaping. In *Proceedings of the Sixteenth International Conference on Machine Learning*, pp. 278–287, 1999.

Jiayi Pan, Xingyao Wang, Graham Neubig, Navdeep Jaitly, Heng Ji, Alane Suhr, and Yizhe Zhang. Training software engineering agents and verifiers with SWE-Gym. In *International Conference on Machine Learning*, 2025.

Yichen Pan, Dehan Kong, Sida Zhou, Cheng Cui, Yifei Leng, Bing Jiang, Hangyu Liu, Yanyi Shang, Shuyan Zhou, Tongshuang Wu, and Zhengyang Wu. WebCanvas: Benchmarking web agents in online environments. *arXiv preprint arXiv:2406.12373*, 2024.

Chen Qian, Wei Liu, Hongzhang Liu, Nuo Chen, Yufan Dang, Jiahao Li, Cheng Yang, Weize Chen, Yusheng Su, Xin Cong, Juyuan Xu, Dahai Li, Zhiyuan Liu, and Maosong Sun. ChatDev: Communicative agents for software development. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics*, pp. 15174–15186, 2024. doi: 10.18653/v1/2024.acl-long.810.

Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, et al. ToolLLM: Facilitating large language models to master 16000+ real-world apis. *arXiv preprint arXiv:2307.16789*, 2023.

Yun Qu, Yuhang Jiang, Boyuan Wang, Yixiu Mao, Cheems Wang, Chang Liu, and Xiangyang Ji. Latent reward: LLM-empowered credit assignment in episodic reinforcement learning. *arXiv preprint arXiv:2412.11120*, 2025.

Christopher Rawles, Sarah Clinckemaillie, Yifan Chang, Jonathan Waltz, Gabrielle Lau, Marybeth Fair, Alice Li, William Bishop, Wei Li, Folawiyo Campbell-Ajala, Daniel Toyama, Robert Berry, Divya Tyamagundlu, Timothy Lillicrap, and Oriana Riva. AndroidWorld: A dynamic benchmarking environment for autonomous agents. *arXiv preprint arXiv:2405.14573*, 2025.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. In *Advances in Neural Information Processing Systems*, 2023.

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

Serper.dev. Serper: Google search API. https://serper.dev, 2026.

Amrith Setlur, Chirag Nagpal, Adam Fisch, Xinyang Geng, Jacob Eisenstein, Rishabh Agarwal, Alekh Agarwal, Jonathan Berant, and Aviral Kumar. Rewarding progress: Scaling automated process verifiers for LLM reasoning. *arXiv preprint arXiv:2410.08146*, 2024.

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

Noah Shinn, Federico Cassano, Edward Berman, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. *arXiv preprint arXiv:2303.11366*, 2023.

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.

Hao Sun, Zile Qiao, Jiayan Guo, Xuanbo Fan, Yingyan Hou, Yong Jiang, Pengjun Xie, Fei Huang, and Yan Zhang. ZeroSearch: Incentivize the search capability of LLMs without searching. *arXiv preprint arXiv:2505.04588*, 2025.

Richard S. Sutton. Learning to predict by the methods of temporal differences. *Machine Learning*, 3:9–44, 1988.

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 2 edition, 2018.

Hui-Ze Tan, Xiao-Wen Yang, Hao Chen, Jie-Jing Shao, Yi Wen, Yuteng Shen, Weihong Luo, Xiku Du, Lan-Zhe Guo, and Yu-Feng Li. Hindsight credit assignment for long-horizon LLM agents. *arXiv preprint arXiv:2603.08754*, 2026.

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.

Jonathan Uesato, Nate Kushman, Ramana Kumar, Francis Song, Noah Siegel, Lisa Wang, Antonia Creswell, Geoffrey Irving, and Irina Higgins. Solving math word problems with process- and outcome-based feedback. *arXiv preprint arXiv:2211.14275*, 2022.

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 *International Conference on Learning Representations*, 2026a.

Hanlin Wang, Chak Tou Leong, Jiashuo Wang, Jian Wang, and Wenjie Li. SPA-RL: Reinforcing LLM agents via stepwise progress attribution. *arXiv preprint arXiv:2505.20732*, 2025a.

Haoran Wang, Zhenyu Hou, Yao Wei, Jie Tang, and Yuxiao Dong. SWE-Dev: Building software engineering agents with training and inference scaling. In *Findings of the Association for Computational Linguistics: ACL 2025*, 2025b.

Peiyi Wang, Lei Li, Zhihong Shao, R. X. Xu, Damai Dai, Yifei Li, Deli Chen, Y. Wu, and Zhifang Sui. Math-Shepherd: Verify and reinforce LLMs step-by-step without human annotations. *arXiv preprint arXiv:2312.08935*, 2024a.

Xingyao Wang, Yangyi Chen, Lifan Yuan, Yizhe Zhang, Yunzhu Li, Hao Peng, and Heng Ji. Executable code actions elicit better LLM agents. In *International Conference on Machine Learning*, 2024b.

Xingyao Wang, Boxuan Li, Yufan Song, Frank F. Xu, Xiangru Tang, Mingchen Zhuge, Jiayi Pan, Yueqi Song, Bowen Li, Jaskirat Singh, et al. OpenHands: An open platform for AI software developers as generalist agents. In *International Conference on Learning Representations*, 2025c.

Yifan Wang, Jinyi Mu, Mayank Jobanputra, Yu Wang, Ji-Ung Lee, Soyoung Oh, Isabel Valera, and Vera Demberg. Sparse mixture-of-experts reward models learn interpretable and specialized experts for personalized preference modeling. *arXiv preprint arXiv:2606.04284*, 2026b.

Ziliang Wang, Xuhui Zheng, Kang An, Cijun Ouyang, Jialu Cai, Yuhang Wang, and Yichao Wu. StepSearch: Igniting LLMs search ability via step-wise proximal policy optimization. *arXiv preprint arXiv:2505.15107*, 2025d.

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

Yuxiang Wei, Olivier Duchenne, Jade Copet, Quentin Carbonneaux, Lingming Zhang, Daniel Fried, Gabriel Synnaeve, Rishabh Singh, and Sida I. Wang. SWE-RL: Advancing LLM reasoning via reinforcement learning on open software evolution. In *Advances in Neural Information Processing Systems*, 2025b.

Jiahao Wu, Zhongwen Xu, Qiang Fu, and Wei Yang. Cut the bill, keep the turns: Affordable multi-turn search RL. project page, December 2025a.

Jialong Wu, Baixuan Li, Runnan Fang, Wenbiao Yin, Liwen Zhang, Zhengwei Tao, Dingchu Zhang, Zekun Xi, Gang Fu, Yong Jiang, Pengjun Xie, Fei Huang, and Jingren Zhou. WebDancer: Towards autonomous information seeking agency. *arXiv preprint arXiv:2505.22648*, 2025b.

Yunjia Xi, Jianghao Lin, Yongzhao Xiao, Zheli Zhou, Rong Shan, Te Gao, Jiachen Zhu, Weiwen Liu, Yong Yu, and Weinan Zhang. A survey of LLM-based deep search agents: Paradigm, optimization, evaluation, and challenges. *arXiv preprint arXiv:2508.05668*, 2025a.

Zhiheng Xi, Jixuan Huang, Chenyang Liao, Baodai Huang, Honglin Guo, Jiaqi Liu, Rui Zheng, Junjie Ye, Jiazheng Zhang, Wenxiang Chen, et al. AgentGym-RL: Training LLM agents for long-horizon decision making through multi-turn reinforcement learning. *arXiv preprint arXiv:2509.08755*, 2025b.

Chunqiu Steven Xia, Yinlin Deng, Soren Dunn, and Lingming Zhang. Agentless: Demystifying LLM-based software engineering agents. *arXiv preprint arXiv:2407.01489*, 2024.

Tianbao Xie, Danyang Zhang, Jixuan Chen, Xiaochuan Li, Siheng Zhao, Ruisheng Cao, Toh Jing Hua, Zhoujun Cheng, et al. OSWorld: Benchmarking multimodal agents for open-ended tasks in real computer environments. *arXiv preprint arXiv:2404.07972*, 2024.

Sikuan Yan, Ahmed Bahloul, Ercong Nie, Susanna Schwarzmann, Riccardo Trivisonno, Volker Tresp, and Yunpu Ma. Memory-R2: Fair credit assignment for long-horizon memory-augmented LLM agents. *arXiv preprint arXiv:2605.21768*, 2026.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025a.

John Yang, Akshara Prabhakar, Karthik Narasimhan, and Shunyu Yao. InterCode: Standardizing and benchmarking interactive coding with execution feedback. In *Advances in Neural Information Processing Systems*, volume 36, pp. 23826–23854, 2023.

John Yang, Carlos E. Jimenez, Alexander Wettig, Kilian Lieret, Shunyu Yao, Karthik Narasimhan, and Ofir Press. SWE-agent: Agent-computer interfaces enable automated software engineering. *arXiv preprint arXiv:2405.15793*, 2024.

John Yang, Kilian Lieret, Carlos E. Jimenez, Alexander Wettig, Kabir Khandpur, Yanzhe Zhang, Binyuan Hui, Ofir Press, Ludwig Schmidt, and Diyi Yang. SWE-smith: Scaling data for software engineering agents. In *Advances in Neural Information Processing Systems*, 2025b.

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

Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. WebShop: Towards scalable real-world web interaction with grounded language agents. In *Advances in Neural Information Processing Systems*, 2022.

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.

Chenlu Ye, Zhou Yu, Ziji Zhang, Hao Chen, Narayanan Sadagopan, Jing Huang, Tong Zhang, and Anurag Beniwal. Beyond correctness: Harmonizing process and outcome rewards through RL training. *arXiv preprint arXiv:2509.03403*, 2025.

Lifan Yuan, Wendi Li, Huayu Chen, Ganqu Cui, Ning Ding, Kaiyan Zhang, Bowen Zhou, Zhiyuan Liu, and Hao Peng. Free process rewards without process labels. *arXiv preprint arXiv:2412.01981*, 2024.

Aohan Zeng, Mingdao Liu, Rui Lu, Bowen Wang, Xiao Liu, Yuxiao Dong, and Jie Tang. AgentTuning: Enabling generalized agent abilities for LLMs. *arXiv preprint arXiv:2310.12823*, 2023.

Chi Zhang, Zhao Yang, Jiaxuan Liu, Yucheng Han, Xin Chen, Zebiao Huang, Bin Fu, and Gang Yu. AppAgent: Multimodal agents as smartphone users. *arXiv preprint arXiv:2312.13771*, 2023.

Hanchen Zhang, Xiao Liu, Bowen Lv, Xueqiao Sun, Bohao Jing, Iat Long Iong, Zhenyu Hou, Zehan Qi, Hanyu Lai, Yifan Xu, et al. AgentRL: Scaling agentic reinforcement learning with a multi-turn, multi-task framework. *arXiv preprint arXiv:2510.04206*, 2025a.

Kechi Zhang, Jia Li, Ge Li, Xianjie Shi, and Zhi Jin. CodeAgent: Enhancing code generation with tool-integrated agent systems for real-world repo-level coding challenges. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics*, pp. 13643–13658, 2024a. doi: 10.18653/v1/2024.acl-long.737.

Weizhi Zhang, Yangning Li, Yuanchen Bei, Junyu Luo, Guancheng Wan, Liangwei Yang, Chenxuan Xie, Yuyao Yang, Wei-Chieh Huang, Chunyu Miao, et al. From web search towards agentic deep research: Incentivizing search with reasoning agents. *arXiv preprint arXiv:2506.18959*, 2025b.

Yanzhao Zhang, Mingxin Li, Dingkun Long, Xin Zhang, Huan Lin, Baosong Yang, Pengjun Xie, An Yang, Dayiheng Liu, Junyang Lin, Fei Huang, and Jingren Zhou. Qwen3 embedding: Advancing text embedding and reranking through foundation models. *arXiv preprint arXiv:2506.05176*, 2025c.

Yuntong Zhang, Haifeng Ruan, Zhiyu Fan, and Abhik Roychoudhury. AutoCodeRover: Autonomous program improvement. In *Proceedings of the ACM International Symposium on Software Testing and Analysis*, pp. 1592–1604, 2024b. doi: 10.1145/3650212.3680384.

Qingfei Zhao, Ruobing Wang, Dingling Xu, Daren Zha, and Limin Liu. R-Search: Empowering LLM reasoning with search via multi-reward reinforcement learning. *arXiv preprint arXiv:2506.04185*, 2025a.

Weikang Zhao, Xili Wang, Chengdi Ma, Lingbin Kong, Zhaohua Yang, Mingxiang Tuo, Xiaowei Shi, Yitao Zhai, and Xunliang Cai. MUA-RL: Multi-turn user-interacting agent reinforcement learning for agentic tool use. *arXiv preprint arXiv:2508.18669*, 2025b.

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*, 2025a.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging LLM-as-a-judge with MT-Bench and chatbot arena. In *Advances in Neural Information Processing Systems*, 2023.

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

Shuyan Zhou, Frank F. Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, Uri Alon, and Graham Neubig. WebArena: A realistic web environment for building autonomous agents. In *International Conference on Learning Representations*, 2024.

A Appendix

A.1 Temporal-Difference Credit Assignment

Temporal-difference (TD) learning estimates progress by comparing the value of two adjacent states (Sutton, 1988). Given a value function $V(s)$, an immediate reward r_t, and a discount γ, the one-step TD error is

$$\delta_t = r_t + \gamma V(s_{t+1}) - V(s_t). \tag{12}$$

The TD error is positive when the transition leads to a state whose realized or estimated future return is higher than expected, and negative when it leads to a worse state. In sparse-reward agentic tasks, intermediate environment rewards are usually unavailable. If we set $r_t = 0$ before termination and use $\gamma = 1$, the TD form reduces to the value difference $V(s_{t+1}) - V(s_t)$, which naturally measures the incremental progress made by one transition. The difficulty is that long-horizon agent tasks usually lack a reliable dense value function over partially completed tool trajectories. Section 3 addresses this missing ingredient: how to obtain a stable prefix-level progress signal without training a critic or collecting step-level process labels.

A.2 TRACE Algorithm

Algorithm 1 summarizes TRACE reward construction and the policy update for one prompt group (Section 3).

A.3 Detailed Experimental Setup

This subsection gives the complete model, baseline, and evaluation setup summarized in Section 4.

Models. To evaluate whether the same credit-assignment signal works across model scale, we instantiate the search policy with two Qwen3 thinking backbones: Qwen3-4B-Thinking-2507 and Qwen3-30B-A3B-Thinking-2507 (Yang et al., 2025a). The 4B model is our primary controlled setting for comparing optimization objectives and ablations, while the 30B-A3B model tests whether TRACE transfers to a larger agentic backbone. Unless otherwise stated, both models use the same browser action space, rollout protocol, terminal reward, training data, and evaluation interface. Importantly, all controlled RL runs start directly from the base search policy without a cold-start supervised fine-tuning stage, a mid-training stage, or training on live-web data.

Baselines. We compare TRACE with both external deep-research agents and controlled RL baselines. The external agents include ASearcher-QwQ-32B (Gao et al., 2025), WebDancer-32B (Wu et al., 2025b), CutBill-30B-A3B (Wu et al., 2025a), and TongyiDS-30B-A3B (Tongyi DeepResearch Team et al., 2025); these systems provide strong reference points for long-horizon search, although they may use different data, model backbones, training pipelines, and benchmark harnesses, so we treat their reported numbers as non-controlled reference points. The controlled baselines are initialized from the same Qwen3 backbones as TRACE. Base denotes the untrained search policy. GRPO (Guo et al., 2025) uses only the terminal outcome reward, GSPO (Zheng et al., 2025a) replaces token-level importance ratios with sequence-level ratios, and GiGRPO (Feng et al., 2025) applies group-in-group policy optimization for agent training. For all controlled baselines, we keep the browser action space, rollout protocol, training data, terminal reward, and evaluation interface fixed, so performance differences primarily reflect the optimization objective and credit-assignment signal.

Evaluation. We evaluate deep-research ability in both closed-web and open-web settings. The closed-web benchmark is BrowseComp-Plus (Chen et al., 2025c), for which we build an offline search engine from the officially released corpus using a Qwen3-Embedding-8B (Zhang et al., 2025c) FAISS index. This setting measures long-horizon reasoning under a reproducible retrieval environment that matches the training interface. To test transfer beyond the closed training corpus, we also evaluate on three open-web benchmarks: BrowseComp (Wei et al., 2025a), GAIA (Mialon et al., 2024), and xbench-DeepSearch (Chen et al., 2025a), where retrieval is served by the Serper API (Serper.dev, 2026). Among them, xbench-DeepSearch is a Chinese QA benchmark, allowing us to test whether a training recipe learned in the closed search setting transfers across languages

Algorithm 1 TRACE reward construction and policy update for one prompt group.

Require: prompt x, gold answer $y^\star$, current policy π_θ, behavior snapshot π_{old}, frozen reference π_{ref}, group size G, horizon K, discount γ_{td}, offset ϵ, terminal scale λ_{term}, weights $\alpha_{\text{out}}, \alpha_{\text{turn}}$, clip bounds c_-, c_+

Sample G trajectories $\{\tau_g\}_{g=1}^{G} \sim \pi_{\text{old}}(\cdot \mid x)$, where $\tau_g = (x, a_{g,1}, o_{g,1}, \ldots, a_{g,T_g}, o_{g,T_g}, \hat{y}_g)$.

Compute terminal rewards $R_g = R(\hat{y}_g, y^\star)$ and outcome advantages $A_g^{\text{out}} = \text{GroupNorm}(R_g; \{R_h\}_{h=1}^{G})$, returning 0 for all g if the group standard deviation is 0.

for $g = 1, \ldots, G$ **do**

  Form answer-ready prefixes $S_{g,0}, \ldots, S_{g,T_g}$ from τ_g.

  for $k = 0, \ldots, T_g$ **do**

    $$\bar{\ell}_{g,k} = \frac{1}{|y^\star|} \sum_{t=1}^{|y^\star|} \log \pi_{\text{ref}}(y_t^\star \mid S_{g,k}, y_{<t}^\star).$$

    $$d_{g,k} = -\bar{\ell}_{g,k} + \epsilon, \qquad V_{g,k} = V(S_{g,k}) = \log \frac{d_{g,0}}{d_{g,k}}.$$

  end for

  for $k = 0, \ldots, T_g - 1$ **do**

    $$\delta_{g,k} = V_{g,k+1} - V_{g,k} = \log \frac{d_{g,k}}{d_{g,k+1}}.$$

    if $K = 0$ **then**

      $c_{g,k}^{(K)} = 0$ and $h_{g,k} = k$.

    else

      $h_{g,k} = \min(k + K - 1, T_g - 1)$.

      $$Z_{g,k} = \sum_{u=k}^{h_{g,k}} \gamma_{\text{td}}^{u-k}, \qquad c_{g,k}^{(K)} = Z_{g,k}^{-1} \sum_{u=k}^{h_{g,k}} \gamma_{\text{td}}^{u-k} \delta_{g,u}.$$

    end if

    $$r_{g,k}^{\text{turn}} = c_{g,k}^{(K)} + \mathbf{1}[h_{g,k} = T_g - 1] \lambda_{\text{term}} \gamma_{\text{td}}^{T_g - k} A_g^{\text{out}}.$$

  end for

end for

Let $\mathcal{I}_g$ be the assistant-token indices of trajectory g.

For each g and $t \in \mathcal{I}_g$, set $\hat{A}_{g,t} = \alpha_{\text{out}} A_g^{\text{out}} + \alpha_{\text{turn}} r_{g,\text{turn}(t)}^{\text{turn}}$.

$$\rho_{g,t}(\theta) = \frac{\pi_\theta(a_{g,t} \mid s_{g,t})}{\pi_{\text{old}}(a_{g,t} \mid s_{g,t})}.$$

Update π_θ by maximizing $\mathcal{J}_{\text{TRACE}}(\theta)$:

$$\frac{1}{G} \sum_{g=1}^{G} \frac{1}{|\mathcal{I}_g|} \sum_{t \in \mathcal{I}_g} \min\Big(\rho_{g,t}(\theta) \hat{A}_{g,t}, \text{clip}(\rho_{g,t}(\theta), 1 - c_-, 1 + c_+) \hat{A}_{g,t}\Big).$$

as well as across retrieval environments. Across all evaluations, rollouts stop when the model emits a final `<answer>` span, reaches the tool-turn limit, or hits a context or generation limit. Unless otherwise stated, controlled ablations are single training runs, so small differences should be read as directional rather than as variance-adjusted conclusions.

A.4 Training Hyperparameters

Training protocol. We optimize each student with Adam at a constant learning rate of 10^{-6}, using a global batch size of 128 and 8 rollouts per prompt, with up to 60 tool turns per trajectory. TRACE computes turn credit with the log-ratio formulation in Eq. 6, using training gap offset $\epsilon_{\text{train}} = 10^{-1}$. The look-ahead advantage in Eq. 9 uses horizon $K = 3$ and discount $\gamma_{\text{td}} = 0.8$, with terminal-outcome fill enabled at scale 2.0. The final per-token advantage in Eq. 10 mixes outcome and turn signals with $\alpha_{\text{out}} = 1.0$ and $\alpha_{\text{turn}} = 0.2$; the token-level distillation term is disabled in this work. Tokens in the final answer span are weighted by the group-mean rule at scale 0.05. Tables 3 and 4 list the full script-level training, rollout, and turn-credit hyperparameters.

Tables 3 and 4 summarize the script-level hyperparameters used for the Qwen3-30B-A3B-Thinking-2507 Search-R1 training run. The shared launch script keeps turn-level scoring disabled by default for a single-node colocated layout unless a remote reference-model scoring endpoint is provided; TRACE runs enable this path and use the settings in Table 4.

Category	Hyperparameter	Value
Rollout	Rollout engine	SGLang
	GPUs per rollout engine	1
	Running requests	64
	Rollout batch size	64
	Samples per prompt	8
	Sampling temperature	1.0
	Sampling top-p	1.0
	Max trajectory length	48,000 tokens
	Max response tokens	4096 tokens
	Max tool turns	60
	Max observation tokens	2048 tokens
	Rollout timeout	240s
Evaluation	Eval samples per prompt	1
	Eval max response length	8192 tokens
	Eval max context length	72,000 tokens
	Eval max tool turns	80 tool turns
Optimization	Global batch size	128
	Optimizer	Adam
	Learning rate schedule	Constant
	Learning rate	1×10^{-6}
	Weight decay	0.01
	Adam β_1	0.9
	Adam β_2	0.98
	Optimizer offload	CPU offload with overlapped D2H/H2D transfer
	KL loss	0
	Entropy coefficient	0.00
	Clip ratio lower bound	0.20
	Clip ratio upper bound	0.28
	Loss accounting	Per-token loss
	TIS	Enabled

Table 3: Core training, rollout, and optimization hyperparameters for the Qwen3-30B-A3B-Thinking-2507 Search-R1 training run.

Category	Hyperparameter	Value
Outcome reward	Outcome reward coefficient	1.0
	Format score	0.1
	Zero-advantage handling	Return zero when the group advantage is zero
Turn credit	Turn-reward scoring	Enabled for TRACE runs with a remote reference-model scoring endpoint
	Turn-reward weight	0.2
	Training gap offset ϵ_{train}	0.1
	Ground-truth field	`ground_truth`
	Advantage mode	`turn_advantages`
	Remaining-space transform	Enabled
	Transform mode	`log_ratio`
	Legacy transform stabilizer	0.001
	TD look-ahead horizon	$K = 3$
	TD discount	$\gamma_{\text{td}} = 0.8$
	K-step normalization	Enabled
	Terminal-outcome fill	Enabled
	Terminal-outcome scale	2.0
	Advantage normalization	`none`
	Post-group normalization	False
	Advantage clipping	0.0

Table 4: TRACE-specific reward and turn-credit hyperparameters from the Search-R1 launch script. The training gap offset is the ϵ used in $d_k = -\bar{\ell}_k + \epsilon$; the legacy transform stabilizer is a separate script parameter and is not used as the log-ratio gap offset in the reported TRACE runs.

A.5 Comparison with Raw Log-Probability Differences

This section expands on the choice of the log-ratio state value in Eq. 5 and the resulting TD action reward in Eq. 6. For a fixed rollout, write $\bar{\ell}_k$ for the raw average gold-answer log-probability and $d_k = -\bar{\ell}_k + \epsilon$ for the remaining answer-likelihood gap. The raw log-probability difference $\Delta_k^{\ell} = \bar{\ell}_{k+1} - \bar{\ell}_k$ telescopes to $\bar{\ell}_T - \bar{\ell}_0$ and therefore preserves the ordering of two rollouts from the same prompt when they share the same initial state. However, raw differences measure absolute improvement in log-probability, not the fraction of the remaining gap that a turn closes. The linear remaining-gap score $\widetilde{\Delta}_k^{\text{lin}} = (\bar{\ell}_{k+1} - \bar{\ell}_k)/d_k$ and the TD reward from the log-ratio value, $C_k = V_{k+1} - V_k = \log(d_k/d_{k+1})$, both correct this scale issue by rewarding relative gap reduction, but only the log-ratio value also retains an endpoint-only telescoping form.

Property	Raw delta Δ_k^{ℓ}	Linear remaining gap $\widetilde{\Delta}_k^{\text{lin}}$	Log-ratio TD reward C_k
Definition	$\bar{\ell}_{k+1} - \bar{\ell}_k$	$(\bar{\ell}_{k+1} - \bar{\ell}_k)/d_k$	$V_{k+1} - V_k = \log(d_k/d_{k+1})$
Telescoping	$\bar{\ell}_T - \bar{\ell}_0$	No endpoint-only form	$V_T - V_0 = \log(d_0/d_T)$
Order preservation	Yes, for shared $\bar{\ell}_0$	Not guaranteed	Yes, for shared $\bar{\ell}_0$
Per-step unit	Absolute improvement	Relative gap reduction	Relative gap reduction
Large remaining gap	Uniform scaling	Downweights early large-gap steps	Downweights early large-gap steps
Small remaining gap	Uniform scaling	Strongly amplifies near-terminal steps	Moderately amplifies near-terminal steps
Numerical behavior	Stable	Can spike when d_k is small	Finite with offset and logarithmic growth

Table 5: Comparison of raw log-probability differences, a linear remaining-gap normalization, and the TD reward from the log-ratio value used by TRACE. The log-ratio value is the only relative-gap formulation whose cumulative action reward depends only on the initial and final states.

An offline diagnostic on a held-out BrowseComp-Plus evaluation trace further supports this choice. We compared the cumulative score from each formulation against the final reference-model log-probability and the binary outcome reward over 830 rollouts containing 3742 tool turns, using a separate diagnostic offset $\epsilon_{\text{diag}} = 10^{-3}$ for the two remaining-gap normalizations. The log-ratio

score has the strongest correlation with both endpoint quality and final correctness, while also giving the best pairwise ranking accuracy among rollouts with similar initial raw log-probabilities.

Metric	Raw	Linear	Log-ratio
Correlation with final $\bar{\ell}_T$	0.425	0.721	**0.751**
Correlation with positive outcome reward	0.603	0.680	**0.713**
Pairwise ranking accuracy, $\lvert\Delta\bar{\ell}_0\rvert \leq 0.2$	97.34%	93.13%	**98.24%**

Table 6: Offline comparison of transition-credit formulations on a held-out evaluation trace. Scores are computed by summing turn-level credits within each rollout before comparing to the final reference-model score or outcome reward.

The difference is visible at the turn level. Consider two transitions with nearly identical raw gains: one moves from $\bar{\ell} = -5.1187$ to $\bar{\ell} = -1.5712$ ($\Delta^{\ell} = 3.5475$), while another moves from $\bar{\ell} = -10.6570$ to $\bar{\ell} = -7.1061$ ($\Delta^{\ell} = 3.5509$). Raw delta treats them as essentially equal. In contrast, the corresponding log-ratio TD rewards are 1.1806 and 0.4052, respectively, because the first transition closes a much larger fraction of its remaining gap. This behavior is desirable for long-horizon search: decisive evidence localization near the answer should receive more credit than an equally sized absolute improvement that still leaves the rollout far from predicting the gold answer.

A.6 Learning Dynamics

TRACE makes the model converge faster and to a better policy. Figure 3 compares the learning dynamics of TRACE and outcome-reward baselines across four panels. The first visible trend is that all methods improve as training proceeds, but the TRACE curves start rising earlier. The second trend is that the slope of TRACE is steeper in the early stage, indicating faster reward acquisition rather than only a late-stage separation. The third trend is that the gap does not disappear after the initial rise: TRACE stabilizes at a higher plateau, suggesting that the faster optimization also leads to a better learned policy. Finally, the same ordering appears in held-out accuracy on BrowseComp-Plus with Qwen3-30B-A3B, where the 160-step TRACE checkpoint already exceeds the 200-step outcome-reward baseline. These observations suggest that the reward-curve improvement reflects earlier acquisition of transferable long-horizon search behavior. A likely reason is that outcome-only training assigns one terminal signal to the whole trajectory, so early failed rollouts can update useful intermediate progress together with the final mistake. TRACE reduces this temporal mismatch by assigning turn-level TD credit when an interaction makes the gold answer more predictable under the reference model, while still keeping the outcome reward in the objective to enforce final correctness.

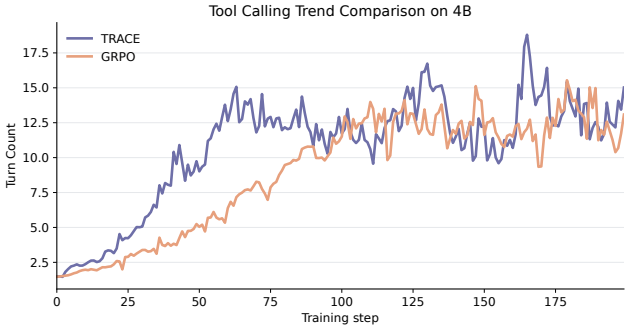

Figure 4: Trajectory-scale dynamics on BrowseComp-Plus with Qwen3-4B. The TRACE curve increases earlier and faster than GRPO, indicating that the two training objectives expand interaction length at different rates during the early training stage.

TRACE scales up interaction length faster in the beginning stage. Figure 4 shows a clear difference in trajectory scale between TRACE and GRPO on the Qwen3-4B backbone: the TRACE curve starts to rise earlier and grows faster, while the GRPO curve increases more slowly. This suggests that the objectives differ in how they encourage longer environment interaction during

early training. With outcome-only GRPO, longer early trajectories often still fail at the final answer, so the whole rollout receives low or negative relative advantage and the objective cannot separate unhelpful extra interaction from intermediate progress. TRACE provides a more local signal: when an interaction and its observation increase answer readiness, the turn-level TD credit can reward that progress before the final answer is correct. This gives a smoother incentive for scaling interaction length, while the outcome reward keeps optimization tied to final correctness.

A.7 Additional Ablations

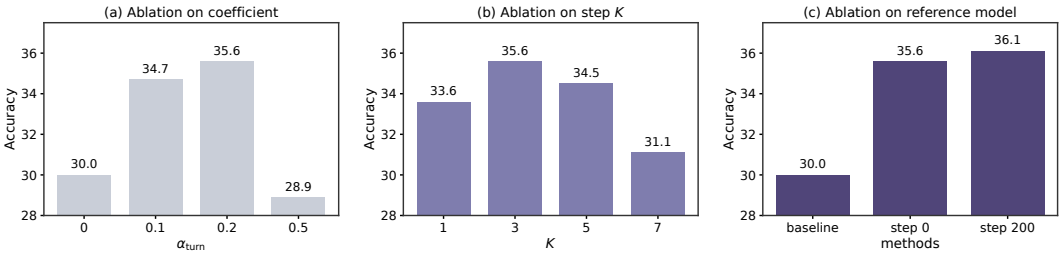

Figure 5: **Ablations on BrowseComp-Plus with Qwen3-4B.** (a) Turn-level reward coefficient. (b) TD look-ahead horizon K. (c) Reference checkpoint for answer-readiness scoring.

Ablation on turn-level reward coefficient. Figure 5(a) shows that the weight assigned to the dense turn-level signal must be chosen moderately. Setting $\alpha_{\text{turn}} = 0$ removes the turn-level signal and obtains only 30.0, matching outcome-only GRPO. Performance rises to 34.7 at $\alpha_{\text{turn}} = 0.1$ and peaks at 35.6 at the reported setting $\alpha_{\text{turn}} = 0.2$, but drops to 28.9 at $\alpha_{\text{turn}} = 0.5$. A small coefficient underuses the extra supervision from prefix-level progress, while an overly large coefficient can make local reference-model readiness dominate the final correctness signal. This pattern supports the use of turn-level reward as an auxiliary credit signal rather than a replacement for the outcome reward.

Ablation on TD look-ahead horizon K. Figure 5(b) varies the TD look-ahead horizon K, which controls how far delayed transition-level progress is propagated. One-step credit ($K = 1$) reaches 33.6, and the reported setting $K = 3$ reaches the best accuracy of 35.6, which indicates that delayed evidence from later `open` and `find` operations should be assigned back to the preceding retrieval decisions. However, longer horizons reduce accuracy to 34.5 at $K = 5$ and 31.1 at $K = 7$, suggesting that overemphasizing the propagated signal introduces noise from loosely related later turns and weakens the temporal locality of the reward.

Ablation on reference model for probability estimation. Finally, Figure 5(c) studies which reference checkpoint should be used to compute the gold-answer prefix probabilities. The baseline without reference-model progress scoring reaches 30.0, while using the initialization checkpoint at step 0 raises accuracy to 35.6. Replacing it with a slightly updated checkpoint at step 200 obtains a similar score of 36.1 in this run. The small gap between the two reference checkpoints suggests that TRACE does not rely on a specially strong or heavily tuned reference model. Instead, the reference model mainly serves as a stable value-estimation anchor: it provides prefix-level answer-readiness scores from which relative TD progress can be computed. In practice, the base checkpoint is already sufficient for this role, which makes the method easier to apply and reduces dependence on selecting an advanced teacher or a carefully adapted scoring model.

A.8 Search-Agent Task and Interface

Agentic harness. The policy is trained in a ReAct-style reasoning-and-acting harness (Yao et al., 2023), where each assistant turn contains private reasoning and either one browser action or a final answer. The closed-corpus harness serves documents from the OpenResearcher offline corpus through a FAISS retrieval index built with Qwen3-Embedding-8B (Zhang et al., 2025c). The browser interface exposes three actions, `browser.search`, `browser.open`, and `browser.find`; the final response must be emitted inside `<answer>` tags. The outcome reward

checks normalized exact-match answer correctness, with a small format component for well-formed answer blocks. The rest of this subsection gives the full tool schema, rollout stopping rules, and masking convention.

We instantiate TRACE on a deliberately simple closed-corpus search task. Given a question x, the agent follows an interleaved reasoning-and-acting pattern (Yao et al., 2023): each assistant turn contains private reasoning and either one browser tool call or a final answer. If the turn contains a valid tool call, the environment executes it against the retrieval corpus and appends the resulting observation before the next assistant turn. Keeping this interface small helps isolate the effect of the training signal from improvements due to a more specialized planner or tool stack.

Rollouts stop when the model emits a final `<answer>` span, reaches the tool-turn limit, or hits a context or generation limit. Training loss is applied only to assistant-generated tokens, while tool observations are masked. The outcome reward checks whether the normalized final answer exactly matches the gold answer, with a small format component for well-formed answers. We also record lightweight trajectory diagnostics, including tool-call counts, repeated queries, reopened documents, repeated in-document searches, navigation errors, and answer-format validity.

The environment exposes three browsing tools over the retrieval corpus. Table 7 summarizes the tool schema used in all experiments.

Tool	Argument	Type/default	Description
`browser.search`	`query`	string, required	Search query string.
	`topn`	integer, default 10	Number of top results to return.
`browser.open`	`id`	integer, default -1	Result index from the most recent search, using zero-based indexing; -1 rereads the current page.
	`loc`	integer, default -1	Starting line number; -1 uses the beginning of the page or a relevant passage.
	`num_lines`	integer, default -1	Number of lines to display; -1 uses the environment default.
`browser.find`	`pattern`	string, required	Exact text pattern to search for in the currently open document.
	`cursor`	integer, default -1	Page cursor to search in; -1 searches the current page.

Table 7: Browser tool schema exposed to the search agent. Search returns titles, URLs, and summaries for the top retrieved documents; open displays document content from the closed corpus; find returns matching line numbers in the currently open document.

The assistant's reasoning, tool call, and final-answer formatting tokens are part of the assistant turn for training purposes, while tool observations are environment outputs and remain masked from the policy-gradient loss.

A.9 Synthetic Training Data

Training datasets. A central question in this work is whether dense turn-level credit can improve long-horizon tool use when the only verifiable supervision is final-answer correctness. To examine this question in a controlled setting, we train agents on synthetic multi-document search tasks built over the offline corpus released by OpenResearcher (Li et al., 2026). Standard multi-hop QA benchmarks are often too short for this purpose: many examples can be solved with one search and one document open, leaving little room to study delayed credit from later evidence gathering. Our training data instead requires chained retrieval over multiple irreplaceable evidence documents, so early queries, document-selection decisions, and in-document localization steps can all affect the final answer. The remainder of this subsection describes the data generation, filtering, and verification pipeline.

Training a search agent with verifiable reinforcement learning requires questions whose rewards remain informative beyond the single-hop retrieval regime. If the training distribution is dominated by direct lookup questions, the agent quickly saturates: once it can issue one or two effective searches, additional training provides little pressure to learn deeper information gathering or cross-document

synthesis. We therefore construct a synthetic training set of hard multi-document identification questions designed to require chained search, reasoning over at least two irreplaceable source documents, and resistance to trivial answer lookup.

A.9.1 GENERATION PIPELINE

The pipeline starts from a document corpus, stored as parquet files or a Hugging Face dataset, together with a related-document index in JSONL format. For each anchor document, we retrieve the top-K_{rel} related documents whose similarity score exceeds a minimum threshold and use this set as the source context for question generation. This related-document context increases the probability that a valid question can connect facts across documents while still requiring the model to identify which documents are necessary.

For every anchor document, we run a best-of-B generation procedure with rejection sampling. Each independent candidate is generated by an LLM prompted to produce a hard question, its answer, and an explicit reasoning chain over the provided sources. If a candidate fails verification, the rejection reason is fed back to the generator for up to a fixed number of regeneration attempts. Candidates that pass the initial checks are then sent to an independent answer verification stage: a separate LLM call receives only the question and source documents, not the proposed answer, and must derive the answer from scratch. We discard the example unless the independently derived answer matches the proposed answer under fuzzy matching. Among the remaining candidates, we keep the highest-scoring example and write it to the training JSONL.

A.9.2 QUESTION TEMPLATES

The generator is constrained to produce questions from a small set of templates that naturally require cross-document reasoning. Table 8 summarizes the templates used in our data synthesis pipeline.

Type	Description
`BRIDGE_ENTITY`	Identify a masked entity, then retrieve one of its attributes from another document.
`INTERSECTION`	Find the unique entity satisfying independent conditions from different documents.
`COUNTING_FILTERED`	Count members of a set defined by cross-document criteria.
`COMPARATIVE`	Compare obfuscated entities using facts distributed across documents.
`REVERSE_LOOKUP`	Given an obfuscated output, identify its creator or source through another document.

Table 8: Question templates used to synthesize hard multi-document QA examples.

To prevent direct keyword lookup, each question applies at least one obfuscation operation. Time expressions may be blurred into relative or approximate descriptions; entity names may be masked; names may be replaced by role descriptions; and quantitative constraints may be converted into qualitative ones. These transformations force the agent to first resolve what should be searched for and only then retrieve the evidence needed to answer.

A.9.3 VERIFICATION AND SELECTION

Quality control combines programmatic checks with LLM-based verification. The programmatic check requires the reasoning chain to cite at least two distinct source URLs. The LLM verifier then evaluates whether the question is actually hard, grounded, and multi-document. It rejects candidates whose answer appears directly in the question, whose constraints can be satisfied from a single source, whose answer is retrievable with a trivial query, or whose reasoning chain lacks a genuine synthesis step. It also filters unsupported or hallucinated answers and examples in which one of the cited documents is redundant.

The final selection score favors candidates that pass both quality and answer verification, succeed in fewer regeneration attempts, use longer reasoning chains, cite more distinct documents, and assign primary roles to all cited documents. It penalizes questions that leak the answer or produce overly long answers. This scoring rule selects examples that are not merely answerable, but useful for

training long-horizon search behavior: the agent must issue multiple queries, maintain intermediate entities, and combine evidence from multiple documents before producing the final answer.

A.10 Qualitative Analysis of the Turn Credit

Outcome-only GRPO assigns one trajectory-level advantage to every action in a rollout, obscuring whether an individual turn retrieves decisive evidence, adds redundant context, or discards an answer already supported by the transcript. We examine five successful and five failed trajectories from rollout batches 0140–0149. In each successful case, we highlight the tool call that closes most of the answer-likelihood gap; in each failed case, the trajectory first reaches an answer-secured prefix and then loses it after one diagnostic tool call. Each example reports the complete question and answer, the one-step credit sequence, and the most diagnostic tool call together with its credit.

For a trajectory with T tool calls, `search_r1_turn_values` contains $T + 2$ frozen-reference values: one before tool use, one after each tool observation, and one after answer generation. Larger V indicates greater closure of the gold-answer likelihood gap; we use $V \geq -0.3$ as a diagnostic definition of an *answer-secured* prefix. Following Eq. 6, $\delta_k = V(S_{k+1}) - V(S_k)$ is assigned to zero-based tool turn k, with the final difference assigned to answer generation. We mark positive progress in green, harmful transitions in red, and near-zero or mixed credit in gray. Tool calls use the notation `browser.tool(argument=value)`; search-relative `open` IDs are accompanied by the returned page name.

A.10.1 Successful Trajectories: Localizing the Turn That Wins

Positive turn example P1: A decisive search resolves the distractor

Question. Starting from the job ad for a major Canadian flag carrier where line-maintenance applicants are asked to speak English plus German, trace to that carrier's separate maintenance leadership posting that gives an edge to bilingual candidates, then compare this carrier against its express-partner company that says it was recognized as a top small/medium employer around the mid-2010s. Which company is associated with the mid-2010s employer recognition (rather than the bilingual-preference maintenance posting)?

Answer: *Air Georgian.* **Final answer:** ✓ **Correct**

Turn-credit distribution.

$$\delta = [+6.39, +0.59, +0.10, +0.05].$$

Critical turn.

Turn 0. `browser.search(query=question)`
Turn credit: $\delta_0 = +6.39$. The salient entity in the question is Air Canada, but the result set surfaces the Air Georgian careers page and thereby introduces the non-obvious express-partner answer. The subsequent page open secures the answer, while the literal `find(pattern="2015")` adds only $+0.10$ after it is already secured.

Positive turn example P2: Only the answer-bearing document earns credit

Question. In the rail-safety material from around the early 2010s, a conference is said to be opened by a senior infrastructure executive (not named here). Follow that trail to the police-led plan later described as helping prevent rail suicides, then move to a later partnership update and identify the person quoted as managing the suicide-prevention programme for the rail industry. Who is that person?

Answer: *Ian Stevens.* **Final answer:** ✓ **Correct**

Turn-credit distribution.

$$\delta = [+1.63, -1.03, -1.04, +6.05, +0.03].$$

Critical turn.

Turn 3. `browser.open(id=2)` (Samaritans–Network Rail partnership update)
Turn credit: $\delta_3 = +6.05$. After two plausible but non-answer-bearing opens, this page names Ian Stevens and supplies the missing discriminative fact.

Positive turn example P3: A decisive open makes confirmation redundant

Question. Identify the venue where the Nepali boxer was defeated by the Filipino Olympian in a unanimous decision, if that Filipino was the same athlete once knocked out in a world-title rematch by the Thai ex-IBF flyweight champion who made an impressive Rio start around the mid-2010s.

Answer: *Jakarta Expo Boxing Hall*. **Final answer:** ✓ **Correct**

Turn-credit distribution.

$$\delta = [+0.88, +5.86, +0.00, -0.00].$$

Critical turn.

Turn 1. `browser.open(id=1)` (Rappler match report)
Turn credit: $\delta_1 = +5.86$. The report supplies "at the Jakarta Expo Boxing Hall" and immediately secures the venue. The subsequent literal `find` receives $+0.00$, illustrating that repeated confirmation cannot inflate the telescoping signal.

Positive turn example P4: The search-result snippet secures the answer

Question. Starting from the maker described as doing Friday evening Arizona "drops" and claiming a no-open-edge signature stitch, follow that trail to the similarly sized product from the black-label line for everyday-carry users, then compare those with the summer-themed, hand-dyed Atlanta shop item that comes with a holographic extra. Which brand is the one that explicitly says no other maker uses its edge/lanyard construction method?

Answer: *Godspeed Tactical*. **Final answer:** ✓ **Correct**

Turn-credit distribution.

$$\delta = [+5.48, +0.06, +0.00].$$

Critical turn.

Turn 0. `browser.search(query=question)`
Turn credit: $\delta_0 = +5.48$. The result snippet exposes the Godspeed Tactical identity and exclusivity claim before any page is opened. The decisive credit therefore belongs to the search-result observation rather than the subsequent page read.

Positive turn example P5: A reformulated search rescues a stalled rollout

Question. In a software podcast episode from around the early 2020s featuring two engineering guests, an ad segment promotes an open-source time-series database and references that brand's event series. For the London edition that was moved online during the pandemic period, what was the earlier company name (from around the early 2010s) that immediately preceded the host company's later name change associated with that database?

Answer: *Errplane*. **Final answer:** ✓ **Correct**

Turn-credit distribution.

$$\delta = [+1.18, -0.12, -0.19, -0.14, + 0.43, +5.44, +0.02, +0.00].$$

Critical turn.

Turn 5. `browser.search(query="influxdata company name change from early 2010s to later name")`
Turn credit: $\delta_5 = +5.44$. After four stalled probes, this reformulated query returns "Errplane became InfluxData in 2015," localizing nearly all progress in the eight-turn trajectory to this rescue query.

A.10.2 Negative Trajectories: Localizing the Tool Call That Loses

Every failure below reaches an answer-secured prefix and then undergoes a single large negative transition. The examples differ in what triggers the collapse: a query-echo self-match, an over-interpreted no-match, a match in the wrong context, an over-literal type comparison, or a phrase copied from the question rather than read from the page.

Negative turn example N1: A query-echo self-match discards a secured answer

Question. In a post from around the mid-2010s by a youth organizer planning an end-of-season teen church overnight, the writer jokes about a very high-voltage fence and doubts a loft drop trust exercise. Use that to find another reflection from roughly the same era where students mention trusting the rope-holder during a heights course, then compare the fear-focused activity named there with the similarly fear-themed activity listed in a different school day-log that also mentions climbing, archery, a blindfold obstacle course, and a ladder challenge. Which fear-themed activity title comes later in alphabetical order?

Gold answer: *Leap of Faith.* **Model answer:** *loft drop trust exercise.*
Final answer: × **Incorrect**

Turn-credit distribution.

$$\delta = [+1.47, +0.13, +1.49, -0.00, -4.83, -0.78].$$

Critical turn.

Turn 4. `browser.find(pattern="loft drop trust exercise", cursor=0)`
Turn credit: $\delta_4 = -4.83$. After the gold answer is secured, this call matches the model's earlier query echo rather than a page-supported title. Accepting its own paraphrase as a second candidate discards the answer and produces the sharp negative credit.

Negative turn example N2: A no-match becomes a false factual negative

Question. Starting from the article about a TV doctor's health-content partnership launched on a global blood-disorder awareness observance in the late 2010s, use that observance date to find another program episode from around the mid-2010s featuring three women in the same disease space. Then use the early-2010s Pittsburgh interview with the longtime local nonprofit executive (described by role, not name) to filter that trio by shared family name. How many women in the trio match that surname?

Gold answer: 2. **Model answer:** 0. **Final answer:** × **Incorrect**

Turn-credit distribution.

$$\delta = [+2.31, +0.10, -1.32, +0.09, + 0.35, +0.98, -4.05, -1.09].$$

Critical turn.

Turn 6. `browser.find(pattern="Bush")` (returns "No matches")
Turn credit: $\delta_6 = -4.05$. Although the preceding prefix is answer-secured, the model interprets absence from an incomplete episode page as evidence that none of the women is a Bush. The negative credit localizes this invalid factual-negative inference.

Negative turn example N3: A successful find locks onto distractor context

Question. Starting from a celebrity-photo page from around the mid-2010s that praises two aristocratic sisters for attending separate events in England, follow to the article comparing the elder future queen consort's conservative red charity gown with her younger sibling's more revealing red look earlier that month. Using those two sisters as the candidate set, then check a later royal-fashion tag page from the early 2020s and count how many of those candidates are explicitly mentioned in the sentence about children serving as page attendants and bridesmaid at the younger sister's wedding.

Gold answer: 2. **Model answer:** 1. **Final answer:** × **Incorrect**

Turn-credit distribution.

$$\delta = [+2.19, -0.35, +0.10, -1.03, +0.96, + 0.50, -3.63, +0.19, -1.22].$$

Critical turn.

Turn 6. `browser.find(pattern="Pippa Middleton")`
Turn credit: $\delta_6 = -3.63$. The call succeeds, but matches "as she did for her aunt Pippa Middleton last year," a sentence about the wrong wedding. From a just-secured prefix, the model anchors on this superficially relevant single-name match and later answers 1. Tool success is therefore not equivalent to information progress.

Negative turn example N4: Exact-string matching replaces a type comparison

Question. Using a post from around the early 2010s by the person who says he ordered interchangeable raw-brass ring parts as your starting clue, go to a different instrument-forum thread that references a repair specialist by first name and mentions that writer considering an instrument with many strings. From that thread, take the distinct instrument types named there as the candidate set. Then check a separate dealer page listing "other instrument offerings" from the mid-2000s era and count how many of those candidate types are present there.

Gold answer: 1. **Model answer:** 0. **Final answer:** × **Incorrect**

Turn-credit distribution.

$$\delta = [+1.42, -0.10, +0.82, +0.08, -3.41, -0.97].$$

Critical turn.

Turn 4. `browser.find(pattern="autoharp")` (returns "No matches")
Turn credit: $\delta_4 = -3.41$. The dealer page has already secured the answer, but the task requires comparing the candidates by instrument type. The model instead probes one literal string and treats its absence as proof that the intersection is empty, discarding the page-level evidence.

Negative turn example N5: A prompt phrase is mistaken for page text

Question. Using the workshop notice from around the early 2010s that mentions a big northern English city and a hotel-branded venue, trace forward to a later full-day procurement training page to recover a per-attendee cost, then apply that amount as a strict threshold to the industry-only Canadian pricing workshops held a few years afterward. How many city sessions in that Canadian notice have fees greater than that threshold?

Gold answer: 2. **Model answer:** 0. **Final answer:** × **Incorrect**

Turn-credit distribution.

$$\delta = [+2.00, +0.43, +0.06, -2.46, + 0.40, +1.98, -3.48, -1.03].$$

Critical turn.

Turn 6. `browser.find(pattern="fees greater than")` (returns "No matches")
Turn credit: $\delta_6 = -3.48$. The Canadian workshop page has already supplied the city-level fees. The model nevertheless searches for a comparison phrase copied from the question rather than comparing the listed numbers, then over-interprets the no-match and answers 0.

Across the positive cases, nearly all credit lands on the `search` or `open` that first supplies the missing discriminative fact; redundant probes and post-hoc `find` calls remain near zero. Across the negative cases, all five trajectories first reach the answer-secured region, after which one misleading `find` receives the dominant penalty. The calls cover both empty matches that are over-interpreted and successful matches in the wrong context, showing that tool completion alone does not imply progress. Outcome-only GRPO cannot express any of these within-trajectory distinctions because it assigns every turn the same trajectory-level advantage.

B LIMITATIONS

Our empirical validation focuses on long-horizon search agents whose final answers are short and can be compared against a known ground truth. This setting matches the core design of TRACE: the frozen reference model estimates the value of a trajectory prefix by measuring how much more predictable the gold answer becomes after each tool interaction. For tasks with compact answers, this provides a simple and stable proxy for progress. However, the same value estimation strategy may be less direct for agentic tasks whose correct outputs are long, structured, or open-ended. For example, a code agent may need to produce a multi-file patch rather than a short answer string, and an open-ended assistant may need to satisfy underspecified user preferences rather than match a single gold response. In such settings, it is unclear whether gold-output log-probability under a frozen reference model is still a reliable state-value proxy.

This limitation does not affect the main claim that turn-level credit can reduce the sparsity of outcome-only agentic RL, but it does bound the current scope of the method. Extending TRACE

to richer agent tasks may require alternative state-value targets, such as execution-based progress signals for coding, structured task specifications, or decomposed verifiable subgoals. We leave the design of reliable credit estimators for long-output and open-ended agents to future work.

C Extended Related Work

Post-training of LLMs for reasoning and agentic tasks. Post-training has become a central recipe for turning pretrained language models into capable problem solvers. For reasoning tasks, reinforcement learning with verifiable rewards has shown that outcome supervision can substantially improve mathematical and symbolic problem solving when correctness can be checked automatically (Shao et al., 2024; Guo et al., 2025). These methods typically optimize a trajectory-level signal with clipped policy-gradient updates (Schulman et al., 2017): a complete solution receives positive or negative feedback according to whether the final answer is correct. This simple regime is effective for short-horizon reasoning, but becomes less informative as decisions grow into multi-hop or browsing trajectories (Yang et al., 2018; Wei et al., 2025a; Chen et al., 2025c). LLM post-training has also been extended from single-response reasoning to agentic settings in which models interact with tools, environments, users, or external feedback over multiple turns. Early systems such as WebGPT and ReAct introduced the interleaving of language-model reasoning, actions, and observations (Nakano et al., 2021; Yao et al., 2023). Subsequent work improves tool use through self-supervised tool-call prediction, instruction tuning on tool-use traces, and agent-specific tuning corpora (Schick et al., 2023; Qin et al., 2023; Zeng et al., 2023), while related approaches use feedback across trials to revise plans, memory, or behavior (Shinn et al., 2023). This agentic post-training landscape now covers a wide range of tasks beyond search. Representative settings include shopping and web navigation (Yao et al., 2022; Deng et al., 2023; Zhou et al., 2024; Koh et al., 2024; Drouin et al., 2024; Pan et al., 2024), API and tool use (Schick et al., 2023; Qin et al., 2023; Li et al., 2023), software engineering agents and related coding systems, real-computer and mobile-device control (Xie et al., 2024; Rawles et al., 2025; Zhang et al., 2023), and multi-turn user-interactive tool use (Zhao et al., 2025b; Chai et al., 2025; Xi et al., 2025b; Zhang et al., 2025a). Within software engineering, representative work spans role-based multi-agent development and interactive code execution (Hong et al., 2024; Qian et al., 2024; Yang et al., 2023; Wang et al., 2024b); repository-level evaluation, planning, and repair (Jimenez et al., 2024; Yang et al., 2024; Zhang et al., 2024a; Bairi et al., 2024; Zhang et al., 2024b; Bouzenia et al., 2025; Wang et al., 2025c; Xia et al., 2024); and executable training environments, scalable task generation, and RL or inference-scaling recipes (Pan et al., 2025; Yang et al., 2025b; Wei et al., 2025b; Jain et al., 2025; Wang et al., 2025b; Mu et al., 2023). Search and deep-research agents form one important branch of this broader agentic post-training trend, with recent work applying outcome-based RL, synthetic trajectory construction, and tool-use training to search-augmented reasoning and deep browsing (Li et al., 2025b; Chen et al., 2025b; Jin et al., 2025b; Song et al., 2025; Sun et al., 2025; Mu et al., 2025; Li et al., 2025c; Wu et al., 2025b; Li et al., 2025a; Zheng et al., 2025b; Li et al., 2026; Wang et al., 2026b; Gao et al., 2025). Recent surveys and position papers summarize these developments under deep search, tool-use agents, and agentic deep research (Xi et al., 2025a; Zhang et al., 2025b). Our work is in this general post-training regime: rather than designing a new agent interface or adding a stronger teacher, we study how to make the training signal more informative inside a multi-turn trajectory.

Credit assignment for long-horizon language agents. Credit assignment is a long-standing challenge in reinforcement learning when the reward is delayed until the end of an extended trajectory (Sutton & Barto, 2018). Temporal-difference learning addresses this problem by estimating how each transition changes the value of the current state (Sutton, 1988). Potential-based reward shaping similarly uses differences in a state potential to provide denser feedback while preserving the underlying task objective under standard assumptions (Ng et al., 1999), and return-decomposition methods such as RUDDER redistribute delayed rewards to the decisions that make the final outcome predictable (Arjona-Medina et al., 2019). In language-model reasoning, process supervision and process reward models provide fine-grained feedback by labeling or modeling intermediate reasoning steps (Uesato et al., 2022; Lightman et al., 2023; Li & Li, 2025). These approaches motivate dense supervision, but they often require a learned critic, human or synthetic step labels, process-reward annotation, Monte Carlo continuations, or task-specific reward features. Recent work has begun to adapt fine-grained credit assignment to language-model policies. For single-response reasoning, Setlur et al. (2024) formalize a process reward as the change in the probability of eventually reaching

a correct answer, Wang et al. (2024a) estimate per-step values from Monte Carlo rollouts, and Yuan et al. (2024) derive an implicit process reward from the log-likelihood ratio between a policy and a fixed reference model. For multi-turn agents, recent work studies stepwise rewards, hindsight credit, memory-aware credit assignment, search-agent RL, and training infrastructure for long-horizon tool use (Feng et al., 2025; Wang et al., 2025a; Dong et al., 2025b; Wang et al., 2025d; Zhao et al., 2025a; Wang et al., 2026a; Tan et al., 2026; Yan et al., 2026; Qu et al., 2025; Dong et al., 2025a; Jin et al., 2025a; Xi et al., 2025b; Chai et al., 2025; Zhao et al., 2025b; Zhang et al., 2025a). TRACE shares the goal of assigning credit below the trajectory level, but avoids training an additional critic or progress estimator, does not require Monte Carlo continuations, and does not depend on revisiting identical states. At a high level, TRACE complements outcome-based post-training with turn-level feedback, allowing long-horizon agents to learn from useful intermediate progress rather than only from final success or failure.