AHEAD: ADAPTIVE HINDSIGHT WITH ENVIRONMENT-AUGMENTED DISTILLATION FOR AGENTIC RL

Anonymous authors
Paper under double-blind review

ABSTRACT

Training multi-turn LLM agents with reinforcement learning typically relies on trajectory-level rewards, which assign a uniform advantage to every step and cannot identify which decisions led to success or failure. Self-distillation methods can provide finer-grained supervision by augmenting RL with privileged information. However, existing approaches usually apply the same privileged information uniformly to every step, ignoring an asymmetry between steps: routine steps need little additional guidance, while critical error steps need explicit corrective direction. We propose **AHEAD**, a step-aware self-distillation method that matches different supervision sources to different step types. The teacher receives environment feedback on all steps as a grounded dense signal, and additionally receives LLM-generated corrective hints on error steps to supply the direction that environment feedback lacks. The method modifies only the GRPO advantage. On ALFWorld, WebShop, and Search-based QA at three model scales, **AHEAD** improves over GRPO ($+17.2$ points on ALFWorld and $+11.0$ on WebShop success rate at 7B) and over self-distillation baselines on ALFWorld at 7B and 1.7B. It also reaches a given success rate in fewer training steps and solves tasks in fewer interaction steps than GRPO.

1 INTRODUCTION

Reinforcement learning has become a central approach for post-training multi-turn LLM agents (Shao et al., 2024; Dong et al., 2025; Feng et al., 2025). In contrast to single-turn reasoning, a multi-turn agent interacts with its environment over an extended horizon, where each action changes future observations and shapes subsequent decisions. Methods such as GRPO (Shao et al., 2024) train policies from trajectory-level environment rewards without requiring a critic, but assign a uniform advantage to every token in the trajectory. However, not all steps contribute equally to the outcome. Most are routine: *the agent acts reasonably and the environment progresses normally*. A small number of steps, however, largely determine the outcome. A wrong object selection, an invalid action, or a misguided search query can compromise the entire episode. Assigning the same gradient signal to both types gives the policy no way to single out the decisions that determined the outcome.

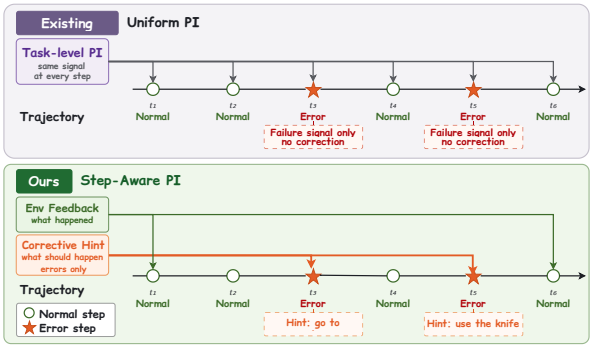


Figure 1: Existing self-distillation methods give the teacher the same task-level privileged information (PI), such as a skill, at every step (top), so error steps such as t_3 and t_5 get no step-specific correction. **AHEAD** (bottom) gives the teacher environment feedback at every step and adds an LLM-generated corrective hint at error steps. The feedback (e.g., "Nothing happens") only signals failure; the hint states what the agent should have done.

On-policy self-distillation offers denser supervision by augmenting a teacher branch with privileged information (PI) and comparing its token-level predictions against the unaugmented student (Zhao et al., 2026; Yang et al., 2026; Lu et al., 2026b), producing a log-probability gap that serves as a

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.

token-level credit-assignment signal. The informativeness of this signal, however, depends on what PI the teacher receives. Existing methods use task-level PI, such as retrieved skills (Lu et al., 2026b; Wang et al., 2026a) or reference answers (Zhao et al., 2026), that provides the same guidance to every step in the trajectory. This ignores an asymmetry: routine steps need only confirmation that the action was appropriate, while error steps need specific corrective direction telling the agent what it should have done instead. Task-level PI, which summarizes general workflows rather than diagnosing individual decisions, cannot provide such step-specific corrective information.

Multi-turn environments already provide a step-specific signal, the environment's feedback after each action, which existing self-distillation methods do not use as privileged information. After each action, the environment returns an observation that is unavailable to the student when generating its response, making it a natural source of PI. For routine steps, this feedback is sufficient: signals such as "You arrived at shelf 2" confirm the action's outcome, and the teacher's assessment shifts only mildly. For error steps, environment feedback is informative but insufficient: signals such as "Nothing happens" indicate failure but reveal neither the cause nor the corrective action. LLM-generated corrective hints (e.g., "Go to diningtable 1 to find the target object") state what the agent should have done at that step. Together, the two sources give weak confirmatory signals on routine steps and strong corrective signals on error steps.

Based on this observation, we propose **AHEAD** (**A**daptive **H**indsight with **E**nvironment-**A**ugmented **D**istillation) for multi-turn agentic RL. **AHEAD** keeps the advantage reweighting of RLSD (Yang et al., 2026) and changes two design choices: what privileged information the teacher sees at each step, and which trajectories the teacher signal is applied to. RLSD gives the teacher the same task-level PI at every step and reweights both successful and failed trajectories. **AHEAD** instead injects environment feedback at every step, adds an LLM-generated corrective hint only at error steps identified by an LLM analyzer, and applies the resulting signal only to failed trajectories; successful trajectories, whose GRPO advantage already points in the right direction, keep the vanilla advantage. At inference time, no PI, LLM calls, or environment feedback injection are needed.

We validate **AHEAD** across the Qwen2.5 (Yang et al., 2024) and Qwen3 (Yang et al., 2025) model families on three benchmarks for LLM-based agents: ALFWorld (Shridhar et al., 2020), WebShop (Yao et al., 2022), and Search-based QA (Jin et al., 2025). **AHEAD** improves over GRPO ($+17.2$ points on ALFWorld and $+11.0$ on WebShop success rate at 7B) and outperforms self-distillation baselines such as SDAR (Lu et al., 2026b), Skill-SD (Wang et al., 2026a), and RLSD (Yang et al., 2026) on ALFWorld at 7B and 1.7B and on WebShop success rate at 3B and 1.7B, while remaining comparable elsewhere.

Our contributions are:

- Motivated by the asymmetry between routine and error steps, we propose **AHEAD**, which keeps the advantage reweighting of RLSD and changes what the teacher sees (environment feedback at every step, a corrective hint at error steps) and where the signal applies (failed trajectories only).
- We validate **AHEAD** on three agentic benchmarks across three model scales, improving over GRPO at every scale and over self-distillation baselines on ALFWorld at 7B and 1.7B and on WebShop success rate at 3B and 1.7B, with comparable results elsewhere.
- We show that the gain comes from what the teacher sees rather than from a stronger LLM: baselines trained with skills from the same LLM still trail **AHEAD**, and the gain holds across analyzers, including an open-weight one.

2 METHOD

We present **AHEAD**, which builds step-aware privileged information for multi-turn agent self-distillation and uses the resulting token-level signal to reweight the GRPO advantage. Figure 2 illustrates the overall pipeline.

2.1 PRELIMINARIES

Problem Setting. We consider a multi-turn setting, in which an agent interacts with an environment over a finite horizon. At step t, the agent receives an observation o_t and maintains an interaction

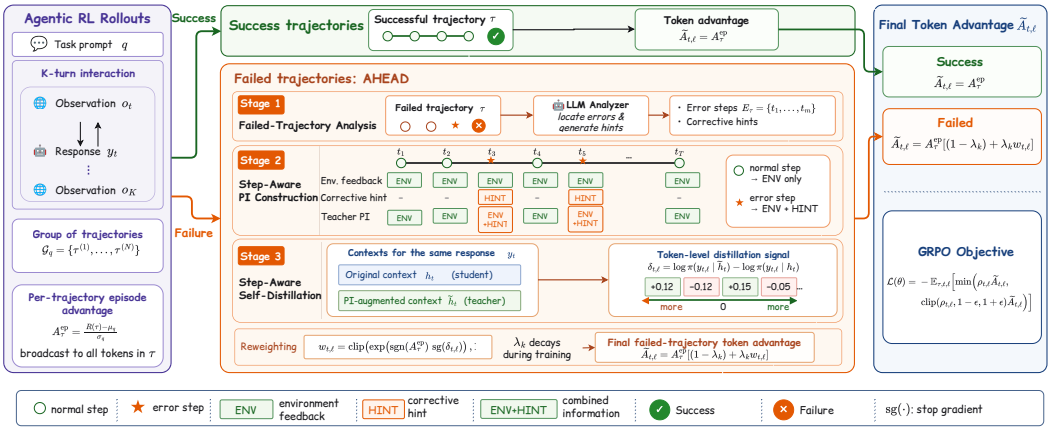


Figure 2: Overview of AHEAD . **Stage 1** identifies error steps in failed trajectories via an LLM analyzer. **Stage 2** constructs step-aware privileged information: environment feedback for routine steps, environment feedback combined with an LLM corrective hint for error steps. **Stage 3** computes a token-level self-distillation signal $\delta_{t,\ell}$ by comparing the policy's log-probabilities under original and PI-augmented contexts, and converts it into a bounded reweight of the GRPO advantage. Successful trajectories bypass the PI pipeline entirely and retain the vanilla GRPO advantage.

history $h_t = (o_0, y_0, o_1, y_1, \ldots, o_t)$, where y_i denotes the response generated at step i. The policy π_θ generates the next response as $y_t \sim \pi_\theta(\cdot \mid h_t)$. A completed trajectory is $\tau = \{(o_t, y_t)\}_{t=0}^{T-1}$, with terminal outcome reward $R(\tau)$.

GRPO. For each task prompt q, GRPO samples N trajectories $\mathcal{G}_q = \{\tau^{(1)}, \ldots, \tau^{(N)}\}$ and computes a group-relative advantage:

$$A^{\text{ep}} = \frac{R(\tau) - \mu_q}{\sigma_q}, \tag{1}$$

where μ_q and σ_q are the group mean and standard deviation. This scalar is broadcast to every token in the trajectory. The policy is optimized with the clipped surrogate:

$$\mathcal{L}_{\text{GRPO}}(\theta) = -\mathbb{E}_{\tau,t,\ell}[\min(\rho_{t,\ell}\, A^{\text{ep}},\ \hat{\rho}_{t,\ell}\, A^{\text{ep}})]\,, \tag{2}$$

where $\rho_{t,\ell} = \pi_\theta(y_{t,\ell} \mid h_t, y_{t,<\ell}) \,/\, \pi_{\theta_{\text{old}}}(y_{t,\ell} \mid h_t, y_{t,<\ell})$ is the token-level importance ratio and $\hat{\rho}_{t,\ell} = \text{clip}(\rho_{t,\ell}, 1-\epsilon, 1+\epsilon)$ is its clipped counterpart. Since A^{ep} is identical for every token, GRPO provides no mechanism to distinguish which steps or tokens were responsible for the outcome.

On-Policy Self-Distillation. On-policy self-distillation (Zhao et al., 2026; Yang et al., 2026) uses the same model as both teacher and student: the student scores each token under the original context h_t, while the teacher receives privileged information (PI) unavailable at decision time. The log-probability gap

$$\delta_{t,\ell} = \log \pi_{\theta_{\text{old}}}(y_{t,\ell} \mid \tilde{h}_t, y_{t,<\ell}) - \log \pi_{\theta_{\text{old}}}(y_{t,\ell} \mid h_t, y_{t,<\ell}) \tag{3}$$

measures how the PI revises the model's assessment of each sampled token. If $\delta_{t,\ell} > 0$, the PI-augmented teacher assigns higher probability to the token than the student, endorsing the student's choice; if $\delta_{t,\ell} < 0$, the teacher disfavors it, suggesting the token should be suppressed. We therefore focus on what PI to inject into $\tilde{h}_t$.

2.2 STEP-AWARE PRIVILEGED INFORMATION

AHEAD constructs different PI for routine and error steps. We first describe how error steps are identified, then how the two PI sources are combined into a step-aware teacher context.

Error Step Identification (Stage 1). After a trajectory completes with a failure outcome, the full trajectory record (observations, actions, environment feedback, and terminal outcome) is passed to an LLM-based analyzer, together with a successful rollout of the same task from its GRPO group. The analyzer identifies steps whose actions were critical errors (e.g., picking up the wrong object, navigating to an irrelevant location). We denote the set of identified error steps as $\mathcal{E}_\tau$.

Environment Feedback as PI. After the agent generates an action at step t, the environment returns an observation o_{t+1}. This feedback is not available to the student when generating y_t, making it a natural source of PI. For routine steps, it confirms that the action was appropriate (e.g., "You pick up the mug from shelf 2"). For error steps, it signals that something went wrong (e.g., "Nothing happens"). Environment feedback is grounded and local, reflecting the actual consequence of the agent's specific action. However, it only describes *what happened*, not *what should have happened*.

LLM Corrective Hints as PI. For each error step $t \in \mathcal{E}_\tau$, the LLM analyzer generates a corrective hint describing what the agent should have done instead. For example, if the agent attempted to pick up an object from a wrong location, the hint might state: "You should first go to diningtable 1 to find the target object." While environment feedback can only signal that an action failed, the corrective hint provides the missing direction: what the alternative should be.

PI Construction (Stage 2). We construct the PI-augmented context $\tilde{h}_t$ by injecting the appropriate PI:

$$\tilde{h}_t = \begin{cases} H(h_t,\, \Phi_t^{\mathrm{env}},\, \Phi_t^{\mathrm{llm}}) & t \in \mathcal{E}_\tau \\ H(h_t,\, \Phi_t^{\mathrm{env}}) & t \notin \mathcal{E}_\tau \end{cases} \tag{4}$$

where $H(\cdot)$ appends PI to the history, Φ_t^{env} is the environment feedback, and Φ_t^{llm} is the corrective hint. Error steps receive both sources simultaneously: the feedback identifies the failure, and the hint supplies the correction. Because the teacher sees richer PI on error steps, the resulting $|\delta_{t,\ell}|$ is larger than on routine steps (Appendix F.3), producing stronger token-level signals on error steps. The LLM analyzer runs only on failed trajectories whose group contains a success; environment feedback costs nothing extra and covers all steps.

2.3 Selective Trajectory Filtering

Not all trajectories benefit equally from PI-based supervision. Failed trajectories carry a uniform negative advantage but lack information about which steps were responsible. The self-distillation signal δ can supply this missing step-level information. Successful trajectories already carry positive advantages that provide the correct gradient direction; injecting PI risks introducing noise when the feedback is not aligned with the tokens that led to success.

AHEAD therefore applies PI-based reweighting only to failed trajectories. Let $\mathcal{M}_\tau$ be the steps that are reweighted: all steps of τ if it failed, since each of them has environment feedback as PI, and none if it succeeded. For steps outside $\mathcal{M}_\tau$, $\tilde{A}_{t,\ell} = A^{\mathrm{ep}}$.

2.4 Advantage Reweighting (Stage 3)

Following RLSD (Yang et al., 2026), we convert the token-level self-distillation signal into a multiplicative weight on the GRPO advantage, rather than using it as a separate distillation loss. This ensures that the environment reward determines whether the policy is reinforced or penalized, while the distillation signal only adjusts how strongly each token is updated.

We exponentiate the gap, gated by the sign of the episode advantage:

$$w_{t,\ell} = \mathrm{clip}\big(\exp\big(\mathrm{sgn}(A^{\mathrm{ep}})\,\delta_{t,\ell}\big),\ 1{-}\varepsilon_w,\ 1{+}\varepsilon_w\big) \tag{5}$$

$$\tilde{A}_{t,\ell} = A^{\mathrm{ep}} \cdot \big[(1-\lambda_k) + \lambda_k \cdot w_{t,\ell}\big] \tag{6}$$

where $\varepsilon_w = 0.2$. The weight $w_{t,\ell}$ carries no gradient, because $\delta_{t,\ell}$ is computed under $\pi_{\theta_{\mathrm{old}}}$. Since **AHEAD** applies reweighting only to failed trajectories (Sec. 2.3), where $A^{\mathrm{ep}} < 0$, tokens disfavored by the teacher ($\delta < 0$) receive larger penalties and tokens endorsed by the teacher ($\delta > 0$) receive smaller ones. The hint therefore does not act as an imitation target: the update still lowers the

probability of the sampled tokens, and the hint determines how strongly each is penalized. The correct action itself is reinforced through the successful rollouts of the same group, which keep their positive advantage. Because $\exp(\cdot) > 0$ and clipping preserves positivity, $\mathrm{sign}(\tilde{A}) = \mathrm{sign}(A^{\mathrm{ep}})$ holds for every token, so the environment reward determines the update direction.

The mixing coefficient λ_k decays linearly from $\lambda_0 = 0.5$ to 0 over D training steps. With the clip, the per-token multiplier therefore stays within $[0.9, 1.1]$, and after step D the update is exactly GRPO. Early training benefits from dense PI-based credit assignment, while later training returns to vanilla GRPO to avoid over-reliance on privileged information.

Objective. The final objective replaces A^{ep} with $\tilde{A}_{t,\ell}$ in the GRPO clipped surrogate:

$$\mathcal{L}(\theta) = -\mathbb{E}_{\tau,t,\ell}\Big[\min\Big(\rho_{t,\ell}\,\tilde{A}_{t,\ell},\ \hat{\rho}_{t,\ell}\,\tilde{A}_{t,\ell}\Big)\Big]. \tag{7}$$

Training–Inference Boundary. The LLM analyzer, corrective hints, and PI-augmented scoring are used only during training to construct the advantage. At inference time, the policy acts from h_t alone, without any PI, LLM calls, or environment feedback injection.

3 Experiments

3.1 Experimental Setting

Benchmarks. We conduct experiments on three agentic benchmarks that cover embodied reasoning, web navigation, and search-augmented question answering.

ALFWorld (Shridhar et al., 2020) is a text-based household environment where an agent must complete language-specified goals via sequential textual actions. It includes six task categories: Pick, Look, Clean, Heat, Cool, and Pick2. We use the training split from GiGPO (Feng et al., 2025).

WebShop (Yao et al., 2022) simulates an e-commerce website where an agent searches for and purchases products matching natural-language specifications. We evaluate on 128 fixed tasks following prior work (Feng et al., 2025), reporting both task-completion score and binary success rate.

Search-based QA (Jin et al., 2025) requires an agent to answer questions by issuing search queries and reading returned documents. The benchmark spans single-hop (NQ (Kwiatkowski et al., 2019), TriviaQA (Joshi et al., 2017), PopQA (Mallen et al., 2023)) and multi-hop (HotpotQA (Yang et al., 2018), 2WikiMultiHopQA (Ho et al., 2020), MuSiQue (Trivedi et al., 2022), Bamboogle (Press et al., 2023)) datasets. Following Search-R1 (Jin et al., 2025), we train on NQ and HotpotQA; the remaining datasets serve as out-of-domain evaluation. Full dataset and metric details are given in Appendix C.

Baselines. We compare against three categories of methods. (1) Training-free: Vanilla uses the base instruction-tuned model without post-training. Skill-Prompt* retrieves a task-relevant skill and prepends it to the prompt at inference time. (2) RL methods: GRPO (Shao et al., 2024) optimizes the policy with group-relative trajectory-level advantages. Skill-GRPO augments GRPO by injecting retrieved skills into training prompts; Skill-GRPO* additionally retains skills at validation time. (3) Hybrid methods combine RL with self-distillation or skill-conditioned supervision. OPSD (Zhao et al., 2026) distills from the same model conditioned on privileged information such as a reference solution. GRPO+OPSD adds OPSD as an auxiliary loss on top of GRPO. Skill-SD (Wang et al., 2026a) uses importance-weighted distillation with retrieved skills as privileged context. RLSD (Yang et al., 2026) re-weights GRPO advantages using the teacher-student log-probability gap. SDAR (Lu et al., 2026b) applies a sigmoid-gated auxiliary distillation loss that selectively distills teacher-endorsed tokens.

Implementation Details. We implement **AHEAD** in the codebase of Lu et al. (2026b) and follow their training and evaluation setup. We use Qwen2.5-3B/7B-Instruct (Yang et al., 2024) and Qwen3-1.7B (Yang et al., 2025) in non-thinking mode as backbone models. All models are trained for 150 steps on 8 H100 GPUs. For ALFWorld and WebShop, each batch samples 16 tasks with 8 rollouts per prompt. For Search-based QA, the batch size is 128 tasks. The maximum prompt length is 2,048 for ALFWorld and 4,096 for WebShop and Search-based QA. We use a GRPO clipping range of $\epsilon = 0.2$ and a reweight bound of $\varepsilon_w = 0.2$; the distillation coefficient decays linearly from $\lambda_0 = 0.5$

Method	ALFWorld							Search-based QA								WebShop	
	Pick	Look	Clean	Heat	Cool	Pick2	Avg	NQ	Triv	Pop	Hotp	2Wk	MuS	Bam	Avg	Score	Succ.
Qwen2.5-3B-Instruct																	
Vanilla	44.4	11.1	6.2	15.4	28.6	12.5	21.9	24.6	48.1	31.0	26.3	25.3	7.2	59.7	31.7	6.7	0.8
Skill-Prompt*	51.7	66.7	48.4	0.0	4.3	10.0	28.9	23.7	46.2	30.6	24.4	22.1	7.5	12.5	23.9	0.2	0.8
OPSD	48.8	41.7	16.7	0.0	15.8	16.7	28.1	0.1	0.1	0.1	0.0	0.0	0.0	0.0	0.0	11.3	3.1
GRPO†	87.1	75.0	87.5	46.7	**82.6**	56.5	75.0	39.3	60.6	41.1	37.4	34.6	15.4	26.4	36.4	79.8	63.3
Skill-GRPO	88.9	71.4	58.8	70.6	40.7	29.2	60.2	43.5	58.8	43.0	36.8	32.2	11.7	12.5	34.1	77.3	60.9
Skill-GRPO*	94.3	57.1	**100.0**	66.7	73.1	57.1	80.5	44.3	59.6	44.3	39.0	36.1	14.5	14.9	36.1	76.3	66.4
GRPO+OPSD	**100.0**	**82.4**	85.7	75.0	70.0	60.0	81.2	44.9	**61.2**	45.2	**40.4**	38.5	16.0	66.1	**44.6**	77.8	66.4
Skill-SD	88.2	50.0	96.2	52.4	65.0	57.9	73.4	44.4	60.4	44.0	39.5	40.4	15.4	64.9	44.1	75.9	64.0
RLSD	87.9	75.0	90.9	75.0	73.1	68.4	79.7	41.5	58.6	42.3	**40.4**	40.2	**16.8**	**66.9**	43.8	84.4	66.4
SDAR	97.1	62.5	**100.0**	61.9	75.0	**84.2**	**84.4**	44.8	58.1	44.3	38.6	36.2	15.7	66.1	43.4	85.0	68.0
AHEAD	96.8	58.3	87.5	**86.7**	78.3	78.3	83.6	**46.1**	**61.2**	**49.2**	40.3	**41.7**	12.7	59.2	44.3	**88.5**	**73.4**
Qwen2.5-7B-Instruct																	
Vanilla	36.1	22.2	3.1	0.0	0.0	0.0	12.5	25.2	50.8	29.5	29.0	29.0	10.4	63.7	33.9	5.9	1.6
Skill-Prompt*	51.7	50.0	32.3	5.3	4.3	0.0	23.4	30.9	52.1	32.7	32.7	27.9	12.7	66.1	36.4	1.7	0.8
OPSD	50.0	60.0	22.7	21.4	17.6	9.5	32.8	8.8	8.6	17.5	2.5	4.2	0.5	1.2	6.2	4.5	2.3
GRPO†	80.6	66.7	95.8	93.3	**82.6**	39.1	76.6	45.1	63.7	44.0	43.6	43.2	16.8	37.6	42.0	80.9	72.6
Skill-GRPO	88.5	66.7	65.2	61.1	57.7	73.1	69.5	45.2	63.7	45.7	43.1	43.3	19.6	21.4	40.3	80.4	71.9
Skill-GRPO*	**100.0**	83.3	96.4	83.3	75.0	78.9	88.3	44.8	63.0	45.1	43.7	43.7	20.5	71.4	47.5	87.0	81.2
GRPO+OPSD	91.4	61.5	**100.0**	87.5	76.5	52.2	80.4	**47.3**	64.5	46.9	43.8	39.3	18.0	69.4	47.0	86.8	76.5
Skill-SD	93.9	**93.8**	90.9	**100.0**	69.2	68.4	85.1	47.1	64.5	47.8	44.2	42.1	20.2	69.0	47.8	86.1	76.5
RLSD	**100.0**	87.5	92.3	58.8	80.0	65.2	82.0	46.8	63.0	44.4	**45.5**	**48.9**	**21.5**	**73.0**	**49.0**	87.4	77.3
SDAR	94.7	75.0	**100.0**	86.7	68.2	78.9	85.9	46.3	63.5	48.2	43.8	48.4	19.6	**73.0**	**49.0**	89.4	82.8
AHEAD	**100.0**	83.3	**100.0**	**100.0**	**82.6**	**91.3**	**93.8**	47.1	**64.7**	**48.3**	**45.5**	44.7	19.3	69.8	48.5	**89.9**	**83.6**
Qwen3-1.7B																	
Vanilla	25.0	22.2	3.1	0.0	21.4	4.2	12.5	29.4	46.9	37.0	23.5	19.6	6.4	10.5	24.8	46.5	4.7
Skill-Prompt*	10.3	50.0	16.1	0.0	0.0	5.0	9.4	29.4	46.5	36.2	22.9	20.8	4.3	10.1	24.3	23.0	2.3
OPSD	26.3	33.3	9.1	0.0	4.5	5.3	14.1	4.2	8.3	4.6	6.6	15.3	0.7	1.2	5.8	47.4	9.3
GRPO†	67.7	16.7	58.3	53.3	21.7	21.7	43.0	40.0	58.9	43.5	35.4	30.3	12.0	65.7	40.8	67.3	38.3
Skill-GRPO	27.6	54.5	22.7	27.3	0.0	19.2	21.1	39.2	58.6	43.9	35.2	28.2	11.5	66.1	40.4	73.4	46.1
Skill-GRPO*	31.4	42.9	51.9	8.3	11.5	7.1	28.1	38.0	58.4	43.9	36.3	29.0	12.5	**66.9**	40.7	80.4	50.0
GRPO+OPSD	38.2	50.0	30.8	28.6	30.0	21.1	32.0	40.7	58.9	45.0	37.0	34.6	**13.3**	65.7	**42.2**	70.7	38.3
Skill-SD	52.9	37.5	69.2	42.9	60.0	36.8	52.3	39.1	57.5	45.4	34.8	34.1	10.7	64.1	40.8	**81.8**	53.9
RLSD	50.0	37.5	61.5	19.0	50.0	21.1	42.2	38.6	57.3	43.0	34.5	34.1	11.5	65.3	40.6	74.0	50.8
SDAR	73.5	25.0	**76.9**	33.3	40.0	36.8	53.9	39.7	58.9	45.3	35.9	35.5	12.6	65.3	41.9	76.8	58.6
AHEAD	**80.6**	**58.3**	70.8	**66.7**	**69.6**	**39.1**	**65.6**	**43.3**	**59.0**	**48.2**	**39.0**	**36.8**	10.9	56.0	41.9	79.3	**64.8**

Table 1: **Performance comparison on ALFWorld, Search-based QA and WebShop.** We report the success rate (%) on ALFWorld, accuracy on Search-based QA (Search-Avg is the unweighted mean over the seven datasets), and task-completion score/success rate on WebShop. An asterisk (*) denotes validation with skills. Within each backbone block, the **best** result in a column is in bold and the second-best is underlined. ALFWorld results use Val-128 (Appendix C). †GRPO on ALFWorld is rerun in our codebase on the same seed-0 Val-128 as AHEAD (Appendix D); other baseline numbers are taken from Lu et al. (2026b).

to 0 over $D = 50$ training steps (Sec. 2.4). Error detection and corrective hint generation use Claude Opus 4.7 (Anthropic, 2026a) as the LLM analyzer. Full hyperparameters are provided in Appendix G.

Evaluation Protocol. We report ALFWorld results on **Val-128**, a fixed set of 128 seen-split tasks sampled without replacement (Appendix C), using for every method the checkpoint with the highest Val-128 success rate. We evaluate generalization to unseen room layouts on the **Unseen-134** split. Results on the full **Seen-140** and **Unseen-134** splits are provided in Appendix F.

3.2 Main Results

Overall Performance. AHEAD achieves the best or second-best result in most columns in Table 1. It improves over GRPO on ALFWorld (+8.6 points on 3B, +17.2 on 7B, +22.7 on 1.7B) and WebShop success rate (+10.1, +11.0 and +26.5 points respectively), and raises Search-Avg at every scale (+7.9, +6.5 and +1.1). The gains on ALFWorld and WebShop are largest for the smallest model, Qwen3-1.7B.

Comparison with Self-Distillation Baselines. On ALFWorld, AHEAD outperforms all self-distillation baselines at 7B and 1.7B and is within one point of SDAR at 3B (83.6% vs. 84.4%). At 7B, it reaches 93.8%, surpassing SDAR (85.9%), Skill-SD (85.1%), and RLSD (82.0%) by large margins. At 1.7B, AHEAD achieves 65.6% while SDAR reaches 53.9% and RLSD only 42.2%. On WebShop, AHEAD leads in success rate at 3B and 1.7B and matches SDAR at 7B (83.6% vs. 82.8%). On Search-based QA, AHEAD is comparable to the strongest baselines. Episodes there last at most four steps and the analyzer marks at most two error steps, so nearly every step is a decision step and the routine/error asymmetry that AHEAD exploits is small.

Analyzer	Val-128	Seen-140	Unseen-134
Opus 4.7 (default)	65.6	66.4	59.0
Sonnet 5	59.4	60.0	**61.2**
Kimi K2.5	57.0	57.1	50.7
GLM-5	**68.0**	**67.1**	56.7
Qwen3.6-27B	64.8	63.6	59.7
GRPO (no analyzer)	43.0	42.1	30.6

Table 2: ALFWorld success rate (%) of Qwen3-1.7B when the error-step analyzer is swapped, holding everything else fixed. GRPO, which uses no analyzer, is repeated from Tables 1, 6 and 7 as a floor. Qwen3.6-27B runs in non-thinking mode.

Method	7B	1.7B
Skill-SD	85.2	25.0
GRPO+OPSD	82.0	44.5
RLSD	79.7	39.8
SDAR	73.4	30.5
AHEAD	**93.8**	**65.6**

Table 3: ALFWorld success rate (%) on Val-128 with every baseline trained on skills generated by Opus 4.7. All runs use the checkpoint rule of Appendix C; AHEAD is repeated from Table 1.

	Components					ALFWorld			WebShop		
Configuration	Env. fb.	Hints	Multi	Fail-only	Decay	1.7B	3B	7B	1.7B	3B	7B
AHEAD	✓	✓	✓	✓	✓	**65.6**	**83.6**	**93.8**	64.8	**73.4**	**83.6**
w/o decay	✓	✓	✓	✓	×	58.6	67.2	80.5	54.7	66.4	**83.6**
w/o failure-only	✓	✓	✓	×	✓	43.0	71.1	89.1	64.1	72.7	75.8
w/o multi-step	✓	✓	×	✓	✓	57.8	75.8	88.3	57.0	47.7	75.0
w/o hints	✓	×	–	✓	✓	44.5	64.8	85.9	63.3	68.0	74.2
w/o env. feedback	×	✓	✓	✓	✓	57.8	77.3	83.6	**67.2**	72.7	72.7
Env. feedback only	✓	×	–	×	✓	46.9	68.0	91.4	60.2	71.9	78.9

Table 4: **Component ablation.** Success rate (%) on ALFWorld (Val-128) and WebShop. Ticks give the configuration of each run, so rows differing in a single tick are one-component comparisons. *Env. fb.*: environment feedback as base PI. *Hints*: the LLM-generated corrective hint Φ^{llm} on error steps. *Multi*: the analyzer may mark several error steps per trajectory rather than one. *Fail-only*: PI is applied to failed trajectories only (Sec. 2.3). *Decay*: the linear schedule on λ_k. The last row removes both the hints and the failure-only filter, leaving environment feedback applied uniformly.

Stability. Standalone OPSD collapses (near-zero accuracy on Search-based QA), and the naive GRPO+OPSD combination degrades on Qwen3-1.7B. AHEAD shows no such instability, since the reweighting (Sec. 2.4) can only adjust how strongly each token's advantage is applied, never its sign.

3.3 Analysis

The Gain Does Not Come from a Stronger LLM. AHEAD's corrective hints are written by Claude Opus 4.7, while the skills used by the skill-conditioned baselines in Table 1 were mostly generated by OpenAI o3, so a stronger generator alone could explain part of AHEAD's lead. To test this, we retrain Skill-SD, GRPO+OPSD, RLSD and SDAR on ALFWorld with skills generated by Opus 4.7 (Table 3). The gap stays large: at 7B the strongest baseline reaches 85.2 against 93.8 for AHEAD, and at 1.7B 44.5 against 65.6. AHEAD's gain therefore does not come from using a stronger LLM.

Ablation Study. Table 4 removes one component at a time; the full method leads nearly every column. Without the linear decay of λ_k, performance drops by 7.0–16.4 points on ALFWorld and up to 10.1 on WebShop, indicating that PI-based reweighting should fade as training progresses. Applying PI to all trajectories instead of only failed ones hurts most at 1.7B (-22.7 on ALFWorld), where PI on a successful trajectory pulls against an already correct gradient, while restricting RLSD to failed trajectories alone does not help (Appendix F.4). Restricting the analyzer to a single error step per trajectory costs up to 25.7 points (WebShop-3B), suggesting that failed episodes often contain more than one critical mistake. Both PI sources contribute on ALFWorld: removing corrective hints costs 7.8–21.1 points, so a failure signal without corrective direction is not sufficient, and removing environment feedback costs 6.3–10.2 points. Hints matter more at the smaller scales and environment feedback at 7B, and the relative importance of the two sources also varies across benchmarks.

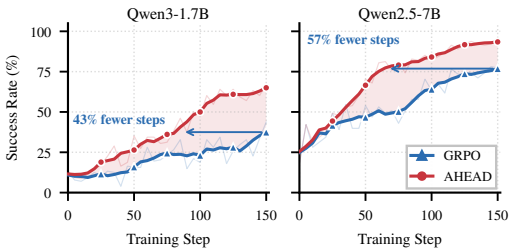


Figure 3: Sample efficiency on ALFWorld, measured on Val-128 during training, for Qwen3-1.7B (left) and Qwen2.5-7B (right). **AHEAD** converges faster and reaches a higher final success rate than GRPO at both scales; arrows mark the training-step saving to reach GRPO's final score.

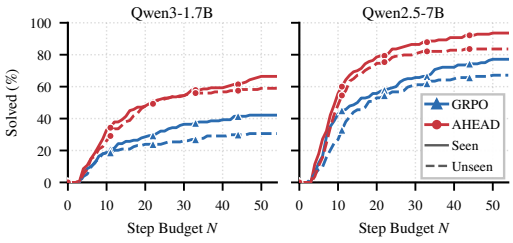


Figure 4: Fraction of ALFWorld tasks solved within N steps for Qwen3-1.7B (left) and Qwen2.5-7B (right). Solid and dashed lines denote seen and unseen splits, respectively. Beyond the first few steps ($N \geq 5$), **AHEAD** consistently outperforms GRPO across both splits and scales.

Choice of LLM Analyzer. **AHEAD** relies on an external LLM to select error steps and write corrective hints, so we test how much the gain depends on the choice of analyzer. Table 2 swaps it for four alternatives at 1.7B, including the open-weight Qwen3.6-27B in non-thinking mode, while holding the rest of the pipeline fixed. Every analyzer beats GRPO by a wide margin: the weakest, Kimi K2.5, scores 57.0 against GRPO's 43.0, so the gain is not contingent on one particular model. The choice still matters, since the five analyzers span 10.9 points (57.0–68.0) with GLM-5 highest, and the open-weight Qwen3.6-27B comes within one point of the default (64.8 vs. 65.6).

Sample Efficiency. As shown in Figure 3, **AHEAD** converges faster than GRPO. At 7B, it reaches GRPO's final performance level early in training and continues to improve to 93.8 against GRPO's 76.6. The same pattern holds at 1.7B, where **AHEAD** ends 22.7 points above GRPO (65.6 vs. 43.0). Beyond raising the final score, step-aware PI reaches a given score in fewer training steps. We attribute this to the corrective hints, which let the update single out the wrong decision at each error step instead of spreading the penalty over the whole trajectory. Although the reweight changes each token's advantage by at most 10% and vanishes after step 50, the gap to GRPO keeps growing afterwards, suggesting that early credit assignment changes which behaviors the policy settles into.

Step Efficiency at Test Time. The efficiency gains transfer to test time (Figure 4). Beyond the first few steps ($N \geq 5$), **AHEAD**'s curve lies above GRPO's on both the seen and unseen splits of ALFWorld, so the improvement is not an artifact of a generous interaction limit. The gap is largest at small step budgets: at 7B on the unseen split, **AHEAD** solves 74.6% of tasks within 20 steps where GRPO solves 53.0%, and **AHEAD** needs only 15 steps to match the success rate GRPO attains with its full 50-step budget. The margin is even larger at 1.7B: **AHEAD** solves 47.8% of tasks within 20 steps, compared with 23.9% for GRPO, and reaches GRPO's full-budget coverage in only 12 steps. These results suggest that **AHEAD**'s gains come from eliminating unnecessary steps rather than from using additional steps for exploration.

Qualitative Trajectory Comparison. Figure 5 contrasts a GRPO-trained policy with a **AHEAD**-trained policy on the same ALFWorld task, *put a hot plate in cabinet*. Both start from identical observations, but the two trajectories diverge immediately at the first plate-bearing countertop. The GRPO agent repeatedly overlooks the plates in front of it, cycles through the same countertops and empty cabinets, and never issues a single take, heat, or move action before hitting the 50-step limit, so the episode fails. The **AHEAD** agent instead grounds each decision in the current observation: it picks up the plate it sees, heats it in the microwave, and places it in the cabinet, completing the full find $\rightarrow$ pick $\rightarrow$ heat $\rightarrow$ place sequence in nine steps. Step-aware corrective hints target this failure mode during training, in which GRPO compounds an early perception error into an unrecoverable loop. The example is also consistent with the shorter and more successful trajectories that **AHEAD** produces (Appendix F). Excerpts are lightly condensed, as noted in the caption.

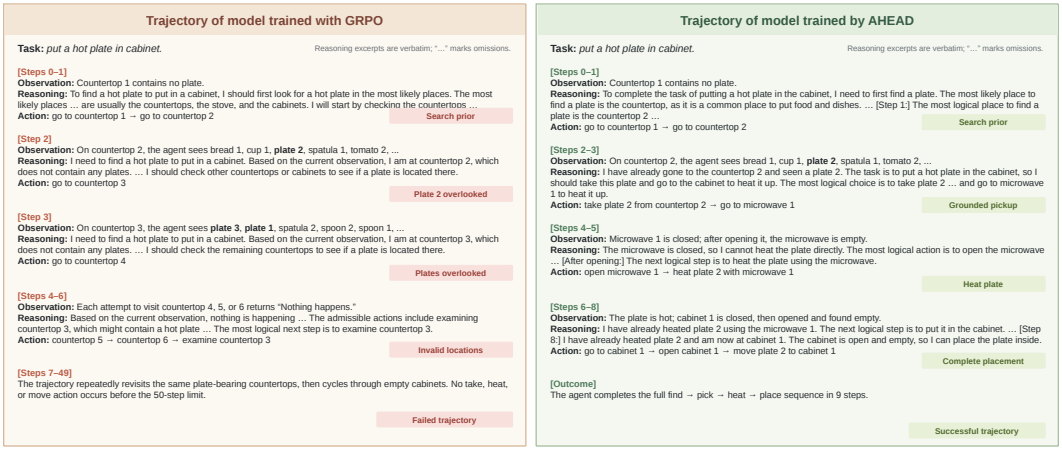


Figure 5: Trajectory comparison on an ALFWorld task *put a hot plate in cabinet*. **Left:** a GRPO-trained policy overlooks the visible plates, revisits the same countertops and empty cabinets, and never takes, heats, or moves an object before the 50-step limit (failed trajectory). **Right:** a **AHEAD**-trained policy grounds each action in the current observation and completes the find → pick → heat → place sequence in nine steps. Reasoning and observation excerpts are lightly condensed for space; "…" marks omissions and retained text is verbatim.

4 Related Work

On-Policy Self-Distillation for Agents. On-policy self-distillation provides an alternative source of dense supervision by letting the same model serve as both student and teacher under different contexts (Agarwal et al., 2024; Zhao et al., 2026; He et al., 2026). Several recent methods combine this idea with RL for multi-turn agents. SDAR (Lu et al., 2026b) uses a sigmoid gate on the teacher-student log-probability gap to selectively distill teacher-endorsed tokens while attenuating noisy signals. RLSD (Yang et al., 2026) re-weights GRPO advantages using the same gap without a separate distillation loss. Skill-SD (Wang et al., 2026a) augments the teacher with retrieved natural-language skills and applies importance-weighted distillation. These methods apply privileged information uniformly across steps or select among different granularities of the same PI source. Our work differs by using two heterogeneous PI sources and allocating them based on step type.

Privileged Information in Agent Training. Using PI during training while removing it at inference has roots in the learning-by-cheating paradigm (Chen et al., 2019). In LLM agent training, this idea appears as skill-conditioned learning, where natural-language skills are provided during training but removed at test time (Lu et al., 2026c; Xia et al., 2026; Shi et al., 2026). These approaches typically use a single type of privileged information applied uniformly across steps. Our work extends this line by combining two PI sources with different costs and informativeness, and allocating them based on step-level supervision needs. Privileged information can also come from an expert that corrects the student's trajectories (Choudhury & Sodhi, 2024) or a critic that sees training-time information (Zhou et al., 2025). Both use it through a separate model or as an imitation target, whereas **AHEAD** uses it only to reweight the policy's own advantage. Natural-language hindsight such as reflections (Shinn et al., 2023; Yao et al., 2023) is mostly used as context at inference time; **AHEAD** uses it only during training. Further related work is discussed in Appendix B.

5 Conclusion

We presented **AHEAD**, which constructs step-aware privileged information for multi-turn agent self-distillation. By combining environment feedback on all steps with LLM-generated corrective hints on identified error steps, **AHEAD** produces adaptive token-level supervision: weak confirmatory signals on routine steps and strong corrective signals on error steps. Across three benchmarks and three model scales, **AHEAD** improves over GRPO at every scale and over self-distillation baselines in most ALFWorld and WebShop settings. We discuss limitations in Appendix A.

USE OF LARGE LANGUAGE MODELS

An LLM assistant helped write the training scripts that implement **AHEAD** and the plotting scripts used for the figures, and was used to edit prose for readability. It was not used to generate or refine research ideas, formulate or prove claims, design the methodology or the experiments, generate synthetic data, or interpret results.

REFERENCES

Rishabh Agarwal, Nino Vieillard, Yongchao Zhou, Piotr Stanczyk, Sabela Ramos, Matthieu Geist, and Olivier Bachem. On-Policy Distillation of Language Models: Learning from Self-Generated Mistakes. In *International Conference on Learning Representations (ICLR)*, 2024. URL https://arxiv.org/abs/2306.13649.

Marcin Andrychowicz, Filip Wolski, Alex Ray, Jonas Schneider, Rachel Fong, Peter Welinder, Bob McGrew, Josh Tobin, Pieter Abbeel, and Wojciech Zaremba. Hindsight Experience Replay. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017. URL https://arxiv.org/abs/1707.01495.

Anthropic. Claude opus 4.7 system card, April 2026a. URL https://www.anthropic.com/claude-opus-4-7-system-card.

Anthropic. Claude sonnet 5 system card, June 2026b. URL https://www.anthropic.com/claude-sonnet-5-system-card.

Dian Chen, Brady Zhou, Vladlen Koltun, and Philipp Krähenbühl. Learning by Cheating. In *Conference on Robot Learning (CoRL)*, 2019. URL https://arxiv.org/abs/1912.12294.

Sanjiban Choudhury. Process Reward Models for LLM Agents: Practical Framework and Directions. *arXiv preprint arXiv:2502.10325*, 2025. URL https://arxiv.org/abs/2502.10325.

Sanjiban Choudhury and Paloma Sodhi. Better than Your Teacher: LLM Agents that learn from Privileged AI Feedback. *arXiv preprint arXiv:2410.05434*, 2024. URL https://arxiv.org/abs/2410.05434.

Ganqu Cui, Lifan Yuan, Zefan Wang, Hanbin Wang, Yuchen Zhang, Jiacheng Chen, Wendi Li, Bingxiang He, Yuchen Fan, Tianyu Yu, Qixin Xu, Weize Chen, Jiarui Yuan, Huayu Chen, Kaiyan Zhang, Xingtai Lv, Shuo Wang, Yuan Yao, Xu Han, Hao Peng, Yu Cheng, Zhiyuan Liu, Maosong Sun, Bowen Zhou, and Ning Ding. Process Reinforcement through Implicit Rewards. *arXiv preprint arXiv:2502.01456*, 2025. URL https://arxiv.org/abs/2502.01456.

Zeyun Deng, Jasorsi Ghosh, Fiona Xie, Yuzhe Lu, Katia Sycara, and Joseph Campbell. Energy-based transfer for reinforcement learning, 2025. URL https://arxiv.org/abs/2506.16590.

Guanting Dong, Hangyu Mao, Kai Ma, Licheng Bao, Yifei Chen, Zhongyuan Wang, Zhongxia Chen, Jiazhen Du, Huiyang Wang, Fuzheng Zhang, Guorui Zhou, Yutao Zhu, Ji-Rong Wen, and Zhicheng Dou. Agentic Reinforced Policy Optimization. *arXiv preprint arXiv:2507.19849*, 2025. URL https://arxiv.org/abs/2507.19849.

Lang Feng, Zhenghai Xue, Tingcong Liu, and Bo An. Group-in-Group Policy Optimization for LLM Agent Training. *arXiv preprint arXiv:2505.10978*, 2025. URL https://arxiv.org/abs/2505.10978.

GLM-5-Team, Aohan Zeng, et al. GLM-5: from vibe coding to agentic engineering. *arXiv preprint arXiv:2602.15763*, 2026.

Yinghui He, Simran Kaur, Adithya Bhaskar, Yongjin Yang, Jiarui Liu, Narutatsu Ri, Liam Fowl, Abhishek Panigrahi, Danqi Chen, and Sanjeev Arora. Self-Distillation Zero: Self-Revision Turns Binary Rewards into Dense Supervision. *arXiv preprint arXiv:2604.12002*, 2026. URL https://arxiv.org/abs/2604.12002.

Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. Constructing a Multi-hop QA Dataset for Comprehensive Evaluation of Reasoning Steps. In *Proceedings of the 28th International Conference on Computational Linguistics (COLING)*, pp. 6609–6625, 2020. doi: 10.18653/v1/2020.coling-main.580. URL https://arxiv.org/abs/2011.01060.

Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan Arik, Dong Wang, Hamed Zamani, and Jiawei Han. Search-R1: Training LLMs to Reason and Leverage Search Engines with Reinforcement Learning. *arXiv preprint arXiv:2503.09516*, 2025. URL https://arxiv.org/abs/2503.09516.

Mandar Joshi, Eunsol Choi, Daniel S. Weld, and Luke Zettlemoyer. TriviaQA: A Large Scale Distantly Supervised Challenge Dataset for Reading Comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 1601–1611, 2017. doi: 10.18653/v1/P17-1147. URL https://aclanthology.org/P17-1147/.

Amirhossein Kazemnejad, Milad Aghajohari, Eva Portelance, Alessandro Sordoni, Siva Reddy, Aaron Courville, and Nicolas Le Roux. VinePPO: Refining Credit Assignment in RL Training of LLMs. *arXiv preprint arXiv:2410.01679*, 2024. URL https://arxiv.org/abs/2410.01679.

Kimi Team et al. Kimi K2.5: Visual agentic intelligence. *arXiv preprint arXiv:2602.02276*, 2026.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. Natural Questions: A Benchmark for Question Answering Research. *Transactions of the Association for Computational Linguistics (TACL)*, 7:452–466, 2019. doi: 10.1162/tacl_a_00276. URL https://aclanthology.org/Q19-1026/.

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. URL https://arxiv.org/abs/2305.20050.

Zhengxi Lu, Jiabo Ye, Fei Tang, Yongliang Shen, Haiyang Xu, Ziwei Zheng, Weiming Lu, Ming Yan, Fei Huang, Jun Xiao, and Yueting Zhuang. UI-S1: Advancing GUI Automation via Semi-online Reinforcement Learning. *arXiv preprint arXiv:2509.11543*, 2025. URL https://arxiv.org/abs/2509.11543.

Zhengxi Lu, Yuxiang Chai, Yaxuan Guo, Xi Yin, Liang Liu, Hao Wang, Han Xiao, Shuai Ren, Guanjing Xiong, and Hongsheng Li. UI-R1: Enhancing Efficient Action Prediction of GUI Agents by Reinforcement Learning. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 2026a. URL https://arxiv.org/abs/2503.21620.

Zhengxi Lu, Zhiyuan Yao, Zhuowen Han, Zi-Han Wang, Jinyang Wu, Qi Gu, Xunliang Cai, Weiming Lu, Jun Xiao, Yueting Zhuang, and Yongliang Shen. Self-Distilled Agentic Reinforcement Learning. *arXiv preprint arXiv:2605.15155*, 2026b. URL https://arxiv.org/abs/2605.15155.

Zhengxi Lu, Zhiyuan Yao, Jinyang Wu, Chengcheng Han, Qi Gu, Xunliang Cai, Weiming Lu, Jun Xiao, Yueting Zhuang, and Yongliang Shen. Skill0: In-Context Agentic Reinforcement Learning for Skill Internalization. *arXiv preprint arXiv:2604.02268*, 2026c. URL https://arxiv.org/abs/2604.02268.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. Self-Refine: Iterative Refinement with Self-Feedback. *arXiv preprint arXiv:2303.17651*, 2023. URL https://arxiv.org/abs/2303.17651.

Alex Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Daniel Khashabi, and Hannaneh Hajishirzi. When Not to Trust Language Models: Investigating Effectiveness of Parametric and Non-Parametric Memories. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (ACL), Volume 1: Long Papers*, pp. 9802–9822. Association for Computational Linguistics, 2023. doi: 10.18653/v1/2023.acl-long.546. URL https://aclanthology.org/2023.acl-long.546/.

Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah A. Smith, and Mike Lewis. Measuring and Narrowing the Compositionality Gap in Language Models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 5687–5711. Association for Computational Linguistics, 2023. doi: 10.18653/v1/2023.findings-emnlp.378. URL https://aclanthology.org/2023.findings-emnlp.378/.

Qwen Team. Qwen3.6-27B: Flagship-level coding in a 27B dense model, April 2026. URL https://qwen.ai/blog?id=qwen3.6-27b.

Jérémy Scheurer, Jon Ander Campos, Tomasz Korbak, Jun Shern Chan, Angelica Chen, Kyunghyun Cho, and Ethan Perez. Training Language Models with Language Feedback at Scale. *arXiv preprint arXiv:2303.16755*, 2023. URL https://arxiv.org/abs/2303.16755.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y.K. Li, Y. Wu, and Daya Guo. DeepSeekMath: Pushing the Limits of Mathematical Reasoning in Open Language Models. *arXiv preprint arXiv:2402.03300*, 2024. URL https://arxiv.org/abs/2402.03300.

Yaorui Shi, Yuxin Chen, Zhengxi Lu, Yuchun Miao, Shugui Liu, Qi Gu, Xunliang Cai, Xiang Wang, and An Zhang. Skill1: Unified Evolution of Skill-Augmented Agents via Reinforcement Learning. *arXiv preprint arXiv:2605.06130*, 2026. URL https://arxiv.org/abs/2605.06130.

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. URL https://arxiv.org/abs/2303.11366.

Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. ALFWorld: Aligning Text and Embodied Environments for Interactive Learning. In *International Conference on Learning Representations (ICLR)*, 2020. URL https://arxiv.org/abs/2010.03768.

Vaishnavi Shrivastava, Piero Kauffmann, Ahmed Awadallah, and Dimitris Papailiopoulos. ECHO: Terminal Agents Learn World Models for Free. *arXiv preprint arXiv:2605.24517*, 2026. URL https://arxiv.org/abs/2605.24517.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. MuSiQue: Multihop Questions via Single-hop Question Composition. *Transactions of the Association for Computational Linguistics (TACL)*, 10:539–554, 2022. doi: 10.1162/tacl_a_00475. URL https://aclanthology.org/2022.tacl-1.31/.

Hao Wang, Guozhi Wang, Han Xiao, Yufeng Zhou, Yue Pan, Jichao Wang, Ke Xu, Yafei Wen, Xiaohu Ruan, Xiaoxin Chen, and Honggang Qi. Skill-SD: Skill-Conditioned Self-Distillation for Multi-turn LLM Agents. *arXiv preprint arXiv:2604.10674*, 2026a. URL https://arxiv.org/abs/2604.10674.

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*, 2023. URL https://arxiv.org/abs/2312.08935.

Yubo Wang, Xiang Yue, and Wenhu Chen. Critique Fine-Tuning: Learning to Critique is More Effective than Learning to Imitate. *arXiv preprint arXiv:2501.17703*, 2025. URL https://arxiv.org/abs/2501.17703.

Zhitong Wang, Songze Li, Hao Peng, Shuzheng Si, Yi Wang, Maosong Sun, and Juanzi Li. EnvRL: Learn from Environment Dynamics in Agentic Reinforcement Learning. *arXiv preprint arXiv:2606.17680*, 2026b. URL https://arxiv.org/abs/2606.17680.

Peng Xia, Jianwen Chen, Hanyang Wang, Jiaqi Liu, Kaide Zeng, Yu Wang, Siwei Han, Yiyang Zhou, Xujiang Zhao, Haifeng Chen, Zeyu Zheng, Cihang Xie, and Huaxiu Yao. SkillRL: Evolving Agents via Recursive Skill-Augmented Reinforcement Learning. *arXiv preprint arXiv:2602.08234*, 2026. URL https://arxiv.org/abs/2602.08234.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 Technical Report. *arXiv preprint arXiv:2412.15115*, 2024. URL https://arxiv.org/abs/2412.15115.

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*, 2025. URL https://arxiv.org/abs/2505.09388.

Chenxu Yang, Chuanyu Qin, Qingyi Si, Minghui Chen, Naibin Gu, Dingyu Yao, Zheng Lin, Weiping Wang, Jiaqi Wang, and Nan Duan. Self-Distilled RLVR. *arXiv preprint arXiv:2604.03128*, 2026. URL https://arxiv.org/abs/2604.03128.

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 (EMNLP)*, 2018. doi: 10.18653/v1/D18-1259. URL https://arxiv.org/abs/1809.09600.

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 35 (NeurIPS)*, 2022. URL https://arxiv.org/abs/2207.01206.

Weiran Yao, Shelby Heinecke, Juan Carlos Niebles, Zhiwei Liu, Yihao Feng, Le Xue, Rithesh Murthy, Zeyuan Chen, Jianguo Zhang, Devansh Arpit, Ran Xu, Phil Mui, Huan Wang, Caiming Xiong, and Silvio Savarese. Retroformer: Retrospective Large Language Agents with Policy Gradient Optimization. *arXiv preprint arXiv:2308.02151*, 2023. URL https://arxiv.org/abs/2308.02151.

Siyu Yuan, Zehui Chen, Zhiheng Xi, Junjie Ye, Zhengyin Du, and Jiecao Chen. Agent-R: Training Language Model Agents to Reflect via Iterative Self-Training. *arXiv preprint arXiv:2501.11425*, 2025. URL https://arxiv.org/abs/2501.11425.

Tianjun Zhang, Fangchen Liu, Justin Wong, Pieter Abbeel, and Joseph E. Gonzalez. The Wisdom of Hindsight Makes Language Models Better Instruction Followers. *arXiv preprint arXiv:2302.05206*, 2023. URL https://arxiv.org/abs/2302.05206.

Andrew Zhao, Daniel Huang, Quentin Xu, Matthieu Lin, Yong-Jin Liu, and Gao Huang. ExpeL: LLM Agents Are Experiential Learners. *arXiv preprint arXiv:2308.10144*, 2023. URL https://arxiv.org/abs/2308.10144.

Siyan Zhao, Zhihui Xie, Mengchen Liu, Jing Huang, Guan Pang, Feiyu Chen, and Aditya Grover. Self-Distilled Reasoner: On-Policy Self-Distillation for Large Language Models. *arXiv preprint arXiv:2601.18734*, 2026. URL https://arxiv.org/abs/2601.18734.

Yifei Zhou, Andrea Zanette, Jiayi Pan, Sergey Levine, and Aviral Kumar. ArCHer: Training Language Model Agents via Hierarchical Multi-Turn RL. *arXiv preprint arXiv:2402.19446*, 2024. URL https://arxiv.org/abs/2402.19446.

Yifei Zhou, Song Jiang, Yuandong Tian, Jason Weston, Sergey Levine, Sainbayar Sukhbaatar, and Xian Li. SWEET-RL: Training Multi-Turn LLM Agents on Collaborative Reasoning Tasks. *arXiv preprint arXiv:2503.15478*, 2025. URL https://arxiv.org/abs/2503.15478.

A LIMITATIONS

Cost of the LLM analyzer. AHEAD calls an external LLM on failed trajectories to identify error steps and write corrective hints. This adds the cost of LLM calls during training, which GRPO does not incur, and it makes the method's behavior depend on the analyzer's quality. The cost is confined to training. Inference uses no privileged information, no LLM calls, and no environment-feedback injection.

Dependence on error step identifiability. The effectiveness of AHEAD's corrective hints relies on the LLM analyzer's ability to correctly identify which steps were critical errors. Our evaluation focuses on environments with relatively discrete, identifiable mistakes such as picking up the wrong object, navigating to an irrelevant location, or issuing an unproductive search query. In environments where errors are subtle, cumulative, or arise from omissions rather than overt wrong actions, the analyzer may fail to pinpoint the true decision points, reducing the quality of step-aware PI. How well the approach transfers to such settings remains an open question.

Groups without a success. AHEAD redistributes credit within groups that already contain a successful rollout. When every rollout in a group fails, the advantage is zero and the corrective hints have no effect, so, like GRPO, AHEAD gains nothing from such groups. This limits its benefit on tasks the policy rarely solves.

An LLM-free control. Every hint that affects the update is written with a successful rollout of the same group as reference (Appendix H). Using that rollout directly as the teacher's PI, without an analyzer, is therefore a natural control, which we leave to future work.

B RELATED WORK

Reinforcement Learning for LLM Agents. Reinforcement learning is now widely used for post-training language model agents in interactive environments (Shao et al., 2024; Dong et al., 2025; Feng et al., 2025). Agents trained with RL interact with environments over many steps, making sequential decisions in settings such as embodied reasoning (Shridhar et al., 2020), web navigation (Yao et al., 2022), search-augmented QA (Jin et al., 2025), and GUI automation (Lu et al., 2025; 2026a). A central difficulty in these settings is credit assignment: outcome rewards indicate whether an episode succeeded but provide no information about which intermediate decisions were responsible (Deng et al., 2025). AHEAD adds a step-level signal to this trajectory-level reward.

Step-Level Credit Assignment. Several methods refine the outcome reward into step-level credit. GiGPO (Feng et al., 2025) groups repeated states across rollouts to compute step-level relative advantages, VinePPO (Kazemnejad et al., 2024) estimates the value of intermediate states with Monte Carlo rollouts, and ArCHer (Zhou et al., 2024) trains a turn-level critic for multi-turn agents. Process reward models score intermediate steps with human labels (Lightman et al., 2023), automatically generated labels (Wang et al., 2023), or implicit rewards learned from outcomes (Cui et al., 2025), and have been adapted to LLM agents (Choudhury, 2025). Closest to ours, SWEET-RL (Zhou et al., 2025) trains a step-level critic that sees training-time information unavailable to the policy. These methods derive step-level credit from rollout statistics or a learned critic or reward model. AHEAD instead gets it from environment feedback and LLM hints, scored by the policy itself, and trains no additional model. The two approaches could be combined.

Environment Feedback as Supervision. Standard agent RL discards environment observations during optimization, using them only as context for future actions. Recent work uses these observations as supervision. ECHO (Shrivastava et al., 2026) trains the policy to predict environment observations as an auxiliary task, learning an implicit world model at no additional rollout cost. EnvRL (Wang et al., 2026b) extends this with separate state prediction and inverse dynamics objectives. These methods treat environment feedback as prediction targets for representation learning. We instead use environment feedback as privileged information for the teacher.

Hindsight and Language Feedback. Hindsight relabeling turns failed episodes into supervision by reinterpreting them against goals or instructions they did achieve (Andrychowicz et al., 2017;

Zhang et al., 2023). For LLM agents, natural-language hindsight is more common. Reflexion (Shinn et al., 2023), Self-Refine (Madaan et al., 2023) and ExpeL (Zhao et al., 2023) use verbal reflections or extracted insights as context at inference time, without updating the policy. Retroformer (Yao et al., 2023) trains a separate model to write such reflections, and Agent-R (Yuan et al., 2025) trains the agent to recover from its own errors. Other work fine-tunes on outputs refined with language feedback (Scheurer et al., 2023) or with critiques (Wang et al., 2025), and LEAP (Choudhury & Sodhi, 2024) has a privileged expert correct the student's trajectories and imitates the corrections. In these methods the feedback becomes an imitation target or inference-time context. In **AHEAD** the hint never enters the policy's context at test time and is not imitated; it only changes how strongly each sampled token is penalized (Sec. 2.4).

C Dataset and Metric Details

Table 5 summarizes the datasets used in our experiments. Below we provide additional details on each benchmark and the corresponding evaluation metrics.

Benchmark	#Train	#Test
ALFWorld	2,400	128 (Val-128) / 140 (Seen-140) / 134 (Unseen-134)
WebShop	2,400	128
NQ, TriviaQA, PopQA, HotpotQA, 2WikiMultiHopQA, MuSiQue, Bamboogle	19,200	51,713

Table 5: Datasets used in our experiments: ALFWorld (embodied reasoning), WebShop (web navigation), and Search-based QA. For Search-based QA we train only on NQ and HotpotQA; the remaining five datasets are held out and never seen during training.

ALFWorld. ALFWorld (Shridhar et al., 2020) connects text-based interaction with the ALFRED household environment. The agent receives a natural-language goal and textual observations, and must issue a sequence of valid actions to complete the task. The benchmark covers six task categories: Pick, Look, Clean, Heat, Cool, and Pick2. We sample 2,400 training examples from the GiGPO split (Feng et al., 2025). Evaluation uses three sets. **Val-128** is a fixed subset of 128 seen-split tasks: we sort the 140 valid-seen tasks by path and draw 128 of them without replacement with NumPy's `RandomState(0)` (31 Pick, 12 Look, 24 Clean, 15 Heat, 23 Cool and 23 Pick2). **Seen-140** is the full seen split, and **Unseen-134** is the unseen split, which measures generalization to room layouts not encountered during training. Each task is run for one episode with at most 50 steps and sampling temperature 0.4, using the policy alone. For every run we evaluate on Val-128 every 5 training steps and report the checkpoint with the highest Val-128 success rate; Seen-140 and Unseen-134 use the same checkpoint. With 128 tasks, a success rate on Val-128 carries a binomial standard error of 2.1 to 4.4 points in the range we report, so differences of a few points should be read with caution. On each set we report the micro-averaged success rate, in which every task counts equally:

$$\text{ALFWorld-Avg} = \frac{\sum_{c=1}^{6} n_c \, \text{SR}_c}{\sum_{c=1}^{6} n_c}, \tag{8}$$

where n_c is the number of tasks in category c. Because the categories are not equally sized, this differs from the unweighted mean of the six per-category rates.

WebShop. WebShop (Yao et al., 2022) is a simulated e-commerce environment where an agent searches for products, navigates product pages, selects attributes, and makes a purchase to satisfy a natural-language request. We use 2,400 training examples and evaluate on 128 fixed tasks following Feng et al. (2025). The environment returns two metrics: a normalized task-completion score that gives partial credit for matching requested attributes, and a binary success indicator for exact task completion. We report Score (the mean normalized score multiplied by 100) and Succ. (the percentage of exactly successful tasks).

Search-based QA. Following Search-R1 (Jin et al., 2025), the agent interacts with a search engine to retrieve relevant documents before producing a final answer. The evaluation spans seven datasets:

three single-hop (NQ (Kwiatkowski et al., 2019), TriviaQA (Joshi et al., 2017), PopQA (Mallen et al., 2023)) and four multi-hop (HotpotQA (Yang et al., 2018), 2WikiMultiHopQA (Ho et al., 2020), MuSiQue (Trivedi et al., 2022), Bamboogle (Press et al., 2023)). We train on 19,200 examples drawn from NQ and HotpotQA; the remaining five datasets serve as out-of-domain evaluation. We compute answer accuracy on each dataset and report the unweighted macro-average:

$$\text{Search-Avg} = \frac{1}{7}\sum_{d=1}^{7} \text{Acc}_d. \tag{9}$$

Unlike ALFWorld, where the six categories partition a single task suite and are therefore pooled, the seven QA datasets are independently constructed benchmarks with test sets of very different sizes; averaging them without weights keeps a large dataset such as TriviaQA from dominating the score.

D Baseline Descriptions

Following Lu et al. (2026b), we adopt the same set of baselines and the same training and evaluation setup: backbone model, environment configuration, rollout budget, evaluation protocol, number of optimization steps, group size, and learning rate schedule, and **AHEAD** is implemented in the codebase of Lu et al. (2026b). Except for GRPO on ALFWorld, which we rerun in our codebase and evaluate on Val-128, the baseline numbers in Table 1, including GRPO on Search-based QA and WebShop, are taken from Lu et al. (2026b), since rerunning every baseline was beyond our compute budget. An asterisk (*) marks methods that retain the retrieved skill in the prompt at validation/test time.

D.1 Prompting-Only Methods

Vanilla. The base instruction-tuned model is evaluated as-is, with no post-training applied. It receives the environment prompt and interaction history as its only input.

Skill-Prompt*. No post-training is applied. Instead, a retrieved task-relevant skill is added to the prompt at validation/test time. Any performance gain therefore comes from using the skill in context.

D.2 Outcome-Based Reinforcement Learning

GRPO. GRPO (Shao et al., 2024) is a critic-free policy-gradient method. It generates multiple trajectories per task and computes advantages by normalizing their outcome rewards relative to the group. All tokens in a trajectory share the same sequence-level advantage. The policy is optimized with a clipped importance-ratio objective.

Skill-GRPO. This method trains with the same GRPO objective but additionally includes a task-relevant skill in the prompt during rollouts. At validation/test time the skill is removed. This evaluates whether the policy has learned from skill-guided exploration well enough to perform without the skill.

Skill-GRPO*. Training is the same as Skill-GRPO. The difference is that the skill remains in the prompt during validation/test time, so the model has consistent access to the skill in both training and evaluation.

D.3 Self-Distillation and Hybrid Methods

Skill-SD. Skill-SD (Wang et al., 2026a) applies self-distillation to multi-turn agent settings. It first distills successful trajectories into natural-language skills that describe effective strategies and common failure modes. During training, these skills serve as privileged context for the teacher, while the student operates with the standard task prompt only. Since the skills are not available at test time, the student must learn to reproduce the teacher's behavior from its parameters alone.

Algorithm 1 AHEAD

Require: Policy π_θ, task set $\mathcal{S}$, group size N, clipping range ϵ, reweight bound ε_w, initial mixing coefficient λ_0, decay horizon D, LLM analyzer $\mathcal{A}$

1: **for** each training iteration k **do**
2: $\lambda_k \leftarrow \max(0,\ \lambda_0 \cdot (1 - k/D))$ ▷ *Linear decay*
3: Sample a batch of tasks $\{q\}$ from $\mathcal{S}$
4: **for** each task q **do**
5: *// Stage 0: On-policy rollout*
6: Sample N trajectories $\{\tau^{(1)}, \ldots, \tau^{(N)}\} \sim \pi_\theta(\cdot \mid q)$
7: Compute $A_i^{\text{ep}} = (R(\tau^{(i)}) - \mu_q)/\sigma_q$ ▷ *Group-relative advantage*
8: **for** $i = 1, \ldots, N$ **do**
9: **if** $R(\tau^{(i)})$ indicates failure **then**
10: *// Stage 1: Error Step Detection*
11: $\mathcal{E}_\tau,\ \{\Phi_t^{\text{llm}}\}_{t \in \mathcal{E}_\tau} \leftarrow \mathcal{A}(\tau^{(i)})$ ▷ *LLM analyzer*
12: *// Stage 2: Step-Aware PI Construction*
13: **for** each step t in $\tau^{(i)}$ **do**
14: $\Phi_t^{\text{env}} \leftarrow o_{t+1}$ ▷ *Environment feedback*
15: **if** $t \in \mathcal{E}_\tau$ **then**
16: $\tilde{h}_t \leftarrow H(h_t,\ \Phi_t^{\text{env}},\ \Phi_t^{\text{llm}})$ ▷ *Env + Hint*
17: **else**
18: $\tilde{h}_t \leftarrow H(h_t,\ \Phi_t^{\text{env}})$ ▷ *Env only*
19: **end if**
20: **end for**
21: *// Stage 3: Token-Level Self-Distillation*
22: **for** each step t, token ℓ **do**
23: $\delta_{t,\ell} \leftarrow \log \pi_{\theta_{\text{old}}}(y_{t,\ell} \mid \tilde{h}_t, y_{t,<\ell}) - \log \pi_{\theta_{\text{old}}}(y_{t,\ell} \mid h_t, y_{t,<\ell})$
24: $w_{t,\ell} \leftarrow \text{clip}\big(\exp\big(\text{sgn}(A^{\text{ep}}) \cdot \delta_{t,\ell}\big),\ 1{-}\varepsilon_w,\ 1{+}\varepsilon_w\big)$
25: $\tilde{A}_{t,\ell} \leftarrow A^{\text{ep}} \cdot \big[(1 - \lambda_k) + \lambda_k \cdot w_{t,\ell}\big]$
26: **end for**
27: **else**
28: $\tilde{A}_{t,\ell} \leftarrow A^{\text{ep}}$ for all t, ℓ ▷ *Vanilla advantage*
29: **end if**
30: **end for**
31: *// Policy update*
32: Update θ by minimizing $\mathcal{L}(\theta) = -\mathbb{E}\Big[\min\Big(\rho_{t,\ell}\,\tilde{A}_{t,\ell},\ \hat{\rho}_{t,\ell}\,\tilde{A}_{t,\ell}\Big)\Big]$
33: **end for**
34: **end for**

RLSD. RLSD (Yang et al., 2026) modifies GRPO by using a privileged self-teacher to assign token-level importance weights. The log-probability gap between teacher and student at each token is mapped to a bounded scalar that adjusts the magnitude of that token's gradient update, while the sign still follows the outcome-level advantage. The teacher's contribution is scheduled to decay over training, so that later stages reduce to standard GRPO.

SDAR. SDAR (Lu et al., 2026b) augments GRPO with a gated distillation loss as an auxiliary objective. A teacher conditioned on privileged context (e.g., a retrieved skill) evaluates the student's on-policy tokens and produces token-level signals. These signals pass through a bounded gate that upweights reliable positive guidance and downweights noisy negative ones before entering the loss. The GRPO advantage is not modified; the gating operates solely on the distillation term.

E ALGORITHM

The full procedure of **AHEAD** is presented in Algorithm 1. Compared to standard GRPO (which corresponds to removing lines 9–28 and setting $\tilde{A}_{t,\ell} = A^{\text{ep}}$ everywhere), **AHEAD** adds three components: (1) an LLM analyzer call on failed trajectories, (2) a PI-augmented forward pass to compute $\delta_{t,\ell}$, and (3) a bounded reweight of the advantage. All three run only during training.

Method	Pick	Look	Clean	Heat	Cool	Pick2	Avg.
Qwen2.5-3B-Instruct							
GRPO	88.6	**76.9**	**88.9**	43.8	**76.0**	54.2	74.3
AHEAD	**97.1**	53.8	**88.9**	**87.5**	72.0	**79.2**	**82.9**
Δ	+8.6	−23.1	+0.0	+43.8	−4.0	+25.0	+8.6
Qwen2.5-7B-Instruct							
GRPO	82.9	61.5	96.3	93.8	**80.0**	41.7	77.1
AHEAD	**100.0**	**84.6**	**100.0**	**100.0**	**80.0**	**91.7**	**93.6**
Δ	+17.1	+23.1	+3.7	+6.3	+0.0	+50.0	+16.4
Qwen3-1.7B							
GRPO	68.6	15.4	55.6	50.0	20.0	20.8	42.1
AHEAD	**82.9**	**53.8**	**70.4**	**68.8**	**68.0**	**41.7**	**66.4**
Δ	+14.3	+38.5	+14.8	+18.8	+48.0	+20.8	+24.3

Table 6: Success rate (%) per task category on the ALFWorld **Seen-140** split. Δ is the absolute gain of AHEAD over GRPO in percentage points, and **Avg.** is the micro-average over all 140 tasks.

Method	Pick	Look	Clean	Heat	Cool	Pick2	Avg.
Qwen2.5-3B-Instruct							
GRPO	**91.7**	**77.8**	**83.9**	73.9	**95.2**	47.1	79.9
AHEAD	83.3	**77.8**	77.4	**87.0**	76.2	**94.1**	**82.1**
Δ	−8.3	+0.0	−6.5	+13.0	−19.0	+47.1	+2.2
Qwen2.5-7B-Instruct							
GRPO	62.5	38.9	**83.9**	65.2	**85.7**	52.9	67.2
AHEAD	**79.2**	**83.3**	**83.9**	**82.6**	**85.7**	**88.2**	**83.6**
Δ	+16.7	+44.4	+0.0	+17.4	+0.0	+35.3	+16.4
Qwen3-1.7B							
GRPO	50.0	0.0	29.0	47.8	33.3	11.8	30.6
AHEAD	**66.7**	**44.4**	**51.6**	**73.9**	**57.1**	**58.8**	**59.0**
Δ	+16.7	+44.4	+22.6	+26.1	+23.8	+47.1	+28.4

Table 7: Success rate (%) per task category on the ALFWorld **Unseen-134** split. Δ is the absolute gain of AHEAD over GRPO in percentage points, and **Avg.** is the micro-average over all 134 tasks.

F Additional Results

The main text reports the analysis of Sec. 3.3 on Qwen2.5-7B. This appendix gives the per-category numbers behind it and repeats each experiment on Qwen2.5-3B and Qwen3-1.7B. The trends observed at 7B carry over to 1.7B, where the gains are larger. At 3B the average gains are smaller ($+8.6$ on Seen-140 and $+2.2$ on Unseen-134) and the per-category changes are mixed.

F.1 Per-Category Breakdown

Table 6 and Table 7 report per-category success rates on Seen-140 and Unseen-134. The GRPO rows use the same rerun GRPO checkpoints as Table 1, so GRPO and AHEAD are trained and evaluated in the same codebase and are directly comparable.

The improvement is not uniform across task categories. On Unseen-134 at 7B, AHEAD leaves Clean and Cool untouched, categories that GRPO already solves at 83.9% and 85.7%, while lifting Look by 44.4 points and Pick2 by 35.3 points. These two categories require the longest action sequences, where a single misstep can derail the episode, consistent with hints helping most there. The same ordering holds at 1.7B: the largest gains fall on Pick2 ($+47.1$) and Look ($+44.4$, from a GRPO baseline that solves none of the 18 tasks).

At 7B and 1.7B the gain does not shrink on the unseen split. On Qwen2.5-7B, AHEAD improves the average by $+16.4$ points on Seen-140 (93.6 vs. 77.1) and by the same $+16.4$ points on Unseen-134 (83.6 vs. 67.2). On Qwen3-1.7B the gains are $+24.3$ (66.4 vs. 42.1) and $+28.4$ (59.0 vs. 30.6), larger on the unseen split than on the seen one. At 3B the gain is smaller on both splits and shrinks on the unseen one, from $+8.6$ on Seen-140 (82.9 vs. 74.3) to $+2.2$ on Unseen-134 (82.1 vs. 79.9).

F.2 Training Dynamics

Figure 6 tracks training at 7B and Figure 7 at 1.7B.

AHEAD also trains differently from GRPO. Success rate and train reward are consistently higher throughout optimization, with AHEAD reaching a final train reward of 4.40 compared to 3.47 for GRPO. Trajectory length also separates the two methods. Both start at an average of 46.8 steps, but by the end of training AHEAD has driven it down to 23.8 while GRPO only reaches 31.2. Higher reward thus comes with shorter episodes. A policy that only explored more would lengthen its episodes; AHEAD shortens them, consistent with the hints teaching the agent not to repeat a failed action, which in GRPO often extends a trajectory without progress.

The same picture holds at 1.7B. Both methods start from an average episode length of 47.8 steps; by step 150 AHEAD has driven it down to 29.8 while GRPO stalls at 40.2, and it does so at a higher train reward (3.62 vs. 1.36). As at 7B, success rises as trajectory length falls, and the effect is stronger. The final gap in episode length is 10.4 steps at 1.7B, compared with 7.4 steps at 7B.

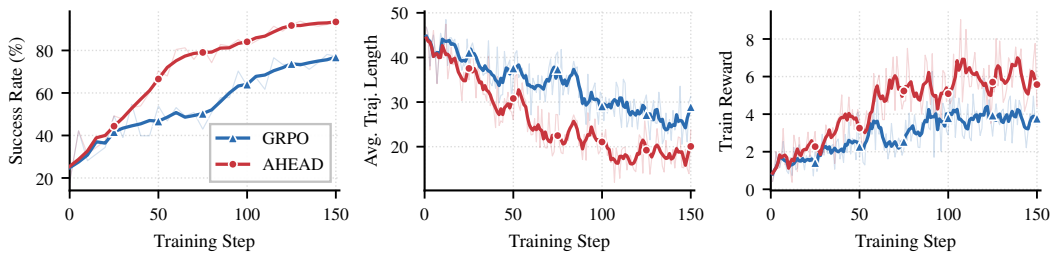


Figure 6: Training dynamics on ALFWorld (7B); the success-rate panel is measured on Val-128. **AHEAD** attains a higher success rate and train reward throughout training, while driving average trajectory length down faster.

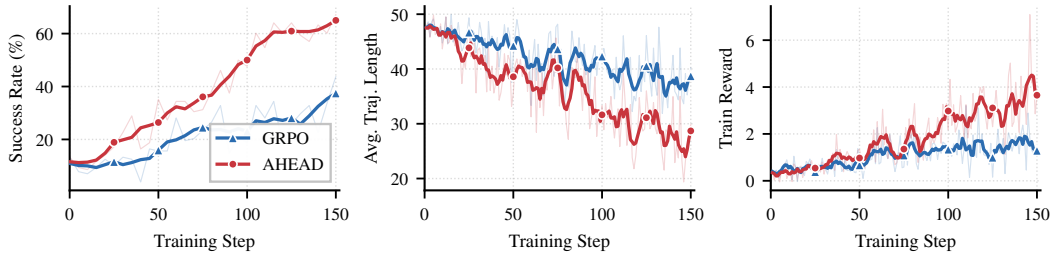


Figure 7: Training dynamics on ALFWorld (Qwen3-1.7B); the success-rate panel is measured on Val-128. Companion to Figure 6.

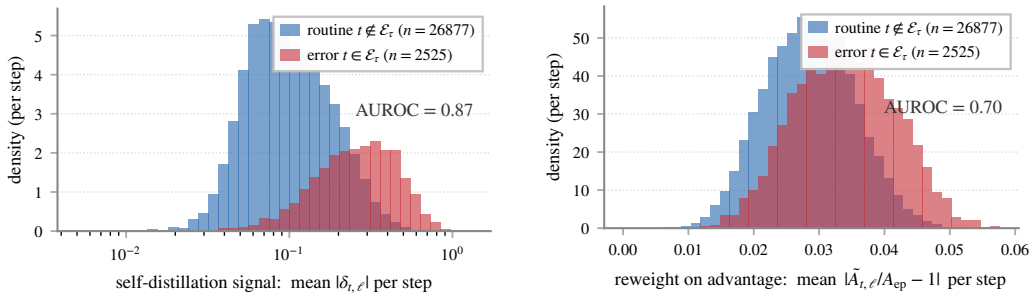
F.3 Where the Reweighting Signal Concentrates

Setup. **AHEAD** assumes the self-distillation gap $\delta_{t,\ell}$ is larger on error steps than on routine steps, so that reweighting the advantage puts more credit on the tokens that caused the failure. To test this, we freeze a mid-training checkpoint (step 25) on ALFWorld and collect $\delta_{t,\ell}$ on the reweighted steps $\mathcal{M}_\tau$ (Sec. 2.3): steps of failed trajectories with non-zero advantage, i.e. the steps that actually affect the loss. A step is an *error step* if it is in the LLM-identified set $\mathcal{E}_\tau$, and *routine* otherwise. This gives 2,525 error steps and 26,877 routine steps.

The signal is larger on error steps. Figure 8a shows the per-step $|\delta_{t,\ell}|$ for both groups. Error steps have a larger gap (mean 0.37 vs. 0.17), and the two distributions separate clearly: ranking steps by $|\delta_{t,\ell}|$ recovers the LLM error labels at AUROC 0.87. The richer PI on error steps (Eq. 4) is therefore enough to strengthen the signal there.

The reweight in the loss is smaller but still targets error steps. We do not apply the raw gap. The clip ($\varepsilon_w = 0.2$) and the mixing coefficient λ_k bound and scale it (Eq. 5–6). Figure 8b shows what the advantage is actually multiplied by, $|\tilde{A}_{t,\ell}/A^{\text{ep}} - 1|$. Error steps are still favored, but the gap shrinks (AUROC 0.70), leaving the reward in control of the update direction.

Token-level view. Figure 9 shows one failed trajectory (task: *find two statues and put them in the diningtable*). At the error step ($t \in \mathcal{E}_\tau$), the tokens that re-examine the diningtable instead of searching `sidetable 2` have the largest $|\delta_{t,\ell}|$. Since $A^{\text{ep}} < 0$ on failed trajectories, these are exactly the tokens amplified by $w_{t,\ell} > 1$ (Eq. 5). A routine step in the same trajectory stays near uniform.



(a) Self-distillation signal $|\delta_{t,\ell}|$.

(b) Reweight applied to the advantage.

Figure 8: **The reweighting signal targets error steps.** Per-step distributions on the reweighted steps $\mathcal{M}_\tau$ (ALFWorld, frozen step-25 checkpoint). **(a)** The gap $|\delta_{t,\ell}|$ is larger on error steps $t \in \mathcal{E}_\tau$ than on routine steps (AUROC 0.87). **(b)** After the clip and mixing λ_k, the reweight that actually scales the advantage, $|\tilde{A}_{t,\ell}/A^{\text{ep}} - 1|$, still favors error steps but with a smaller gap (AUROC 0.70), leaving the reward in control of the update direction.

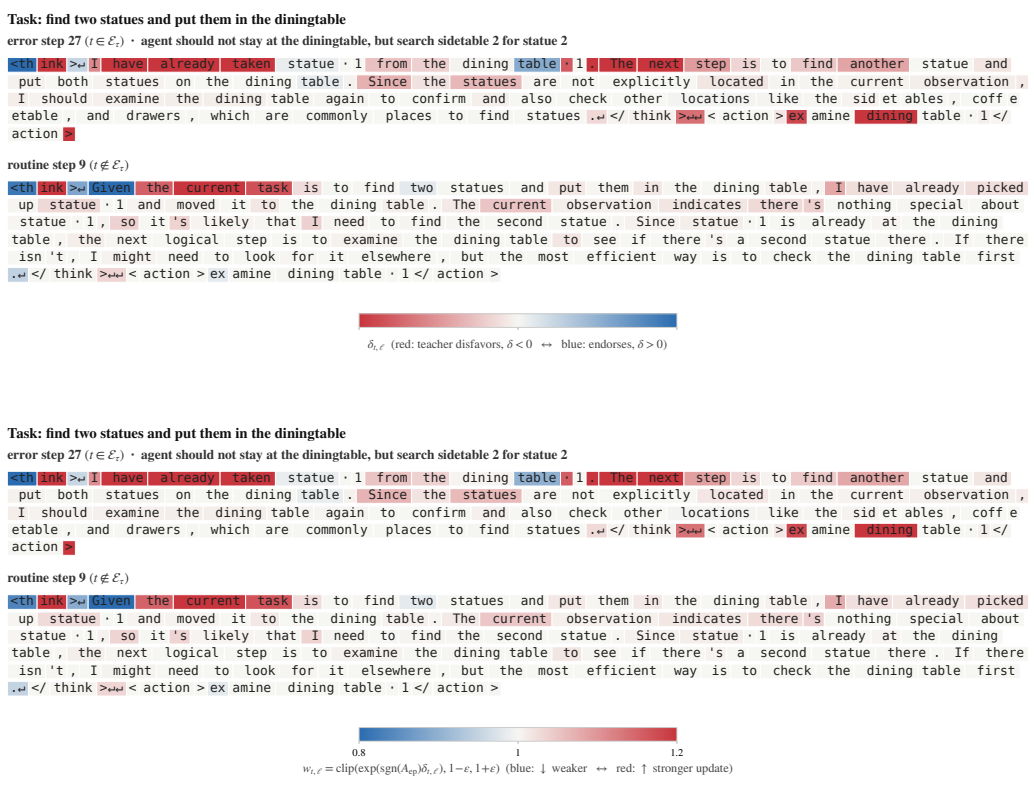


Figure 9: **Token-level credit on one failed trajectory** (ALFWorld). **Top:** signed gap $\delta_{t,\ell}$; red marks tokens the teacher disfavors ($\delta < 0$). At the error step ($t \in \mathcal{E}_\tau$) the tokens for the wrong decision have the largest $|\delta|$; the routine step ($t \notin \mathcal{E}_\tau$) stays near zero. **Bottom:** the weight $w_{t,\ell} = \text{clip}(\exp(\text{sgn}(A^{\text{ep}})\delta_{t,\ell}), 1-\varepsilon_w, 1+\varepsilon_w)$. Since $A^{\text{ep}} < 0$ on failed trajectories, these tokens get the largest amplification ($w > 1$).

Prompt for ALFWorld

You are an expert agent operating in the ALFRED Embodied Environment. Your task is to: {task_description}
Prior to this step, you have already taken {step_count} step(s). Below are the most recent {history_length} observations and the corresponding actions you took: {action_history}
You are now at step {current_step} and your current observation is: {current_observation}
Your admissible actions of the current situation are: [{admissible_actions}].

Now it's your turn to take an action.
You should first reason step-by-step about the current situation. This reasoning process MUST be enclosed within <think> </think> tags.
Once you've finished your reasoning, you should choose an admissible action for current step and present it within <action> </action> tags.

Figure 10: Prompt template for the ALFWorld environment.

F.4 Interaction of PI Source and Signal Scope

Table 8 crosses the two changes **AHEAD** makes to RLSD: what the teacher sees and which trajectories the signal applies to. Neither change helps much on its own. Restricting RLSD to failed trajectories leaves it unchanged (39.8 vs. 39.1 on Val-128), and giving the teacher **AHEAD**'s PI on all trajectories reaches only 43.0. Together they reach 65.6. One explanation is that each change fixes what the other lacks: a task-level skill document does not tell the teacher which step of a failed trajectory went wrong, while on successful trajectories, whose advantage already has the right sign, step-aware PI adds noise rather than direction (Sec. 2.3).

PI source	All trajectories	Failed only
RLSD (task-level skill document)	39.8 / 39.3	39.1 / 40.0
AHEAD (env. feedback + hints)	43.0 / 43.6	**65.6** / **66.4**

Table 8: ALFWorld success rate (%, Val-128 / Seen-140) of Qwen3-1.7B, crossing the two changes **AHEAD** makes to RLSD. All runs use skills or hints from Opus 4.7 and the checkpoint rule of Appendix C. The top-left cell is RLSD from Table 3, the bottom-left cell is "w/o failure-only" from Table 4, and the bottom-right cell is **AHEAD**.

G Hyperparameters

Table 9 lists the full configuration. All backbones share these settings. **AHEAD** adds no KL regularization term. Its method-specific hyperparameters are the reweight bound ε_w, the decay schedule of λ_k, and the maximum number of error steps {max_steps}. The first two start from the defaults of RLSD (Yang et al., 2026); all are selected on Val-128, and Unseen-134 is not used for any selection.

H Prompts

H.1 Environment Interaction Prompts

The ALFWorld and WebShop templates follow Feng et al. (2025), and the Search-based QA template follows Jin et al. (2025). Every method (**AHEAD** and all baselines alike) rolls out with the same template in a given environment. Braces mark slots filled at run time by the environment. The agent is shown only a recent window of the interaction history rather than the full trajectory, making the step-level history h_t in Sec. 2.2 a bounded context.

Hyperparameter	Value
Training steps	150
Training batch size	16 for ALFWorld and WebShop; 128 for Search-based QA
Rollout group size N	8
Learning rate	1×10^{-6}
Hardware	8 × H100
GRPO clipping range ϵ	0.2
Reweight bound ε_w	0.2
Initial distillation coefficient λ_0	0.5
Decay horizon D	50
LLM analyzer	Claude Opus 4.7
Max. error steps ({max_steps})	5 for ALFWorld and WebShop; 2 for Search-based QA
Maximum prompt length	2,048 for ALFWorld; 4,096 for WebShop and Search-based QA
Maximum interaction steps	50 for ALFWorld; 15 for WebShop; 4 for Search-based QA

Table 9: Training configuration. The block in the middle lists the parameters introduced by **AHEAD** (Sec. 2.4); everything else follows the GRPO setup of Lu et al. (2026b).

Prompt for WebShop

You are an expert autonomous agent operating in the WebShop e-commerce environment.
Your task is to: {task_description}.
Prior to this step, you have already taken {step_count} step(s). Below are the most recent {history_length} observations and the corresponding actions you took: {action_history}
You are now at step {current_step} and your current observation is: {current_observation}.
Your admissible actions of the current situation are:
[
{available_actions}
].

Now it's your turn to take one action for the current step.
You should first reason step-by-step about the current situation, then think carefully which admissible action best advances the shopping goal. This reasoning process MUST be enclosed within <think> </think> tags.
Once you've finished your reasoning, you should choose an admissible action for current step and present it within <action> </action> tags.

Figure 11: Prompt template for the WebShop environment.

H.2 Error-Step Analyzer Prompt

Figure 13 gives the prompt behind Stage 1 and Stage 2 of **AHEAD** (Sec. 2.2). A single call does both jobs: it selects the error steps $\mathcal{E}_\tau$ and, for each one, writes the corrective hint Φ_t^{llm} that is injected into the teacher context. It is invoked only on failed trajectories. The analyzers are Claude Opus 4.7 (the default; Anthropic, 2026a), Claude Sonnet 5 (Anthropic, 2026b), Kimi K2.5 (Kimi Team et al., 2026) and GLM-5 (GLM-5-Team et al., 2026), accessed through the Amazon Bedrock API, and Qwen3.6-27B (`Qwen/Qwen3.6-27B` on Hugging Face; Qwen Team, 2026), which we serve ourselves in non-thinking mode.

The analyzer receives the task specification in {task}, the failed trajectory in {failed}, with each step labeled by its index and shown as action and environment feedback, and a successful rollout of the same task in {success_block}. This reference is the first successful rollout among the N rollouts in the same GRPO group, shown with the policy's responses only and no environment feedback, and it is shared by every failed rollout in that group. Because the training reward is binary, a failed rollout has a non-zero advantage only if its group contains a success, so the analyzer is called only on such rollouts and every hint is written with a reference.

Two constraints in the prompt matter for the method. First, the analyzer is asked for *at most* {max_steps} key decision steps (Table 9) rather than a label for every step, which keeps $\mathcal{E}_\tau$ sparse so

Prompt for Search-based QA

You are an expert agent tasked with answering the given question step-by-step.
Your question: {task_description}

Prior to this step, you have already taken {step_count} step(s). Below is the interaction history where <search> </search> wrapped your past search queries and <information> </information> wrapped the corresponding search results returned by the external search engine. History:
{memory_context}

Now it's your turn to respond for the current step.
You should first conduct reasoning process. This process MUST be enclosed within <think> </think> tags.
After completing your reasoning, choose only one of the following actions (do not perform both):
(1) If you find you lack some knowledge, you can call a search engine to get more external information using format: <search> your query </search>.
(2) If you have enough knowledge to answer the question confidently, provide your final answer within <answer> </answer> tags, without detailed illustrations. For example, <answer>Beijing</answer>.

Figure 12: Prompt template for the Search-based QA environment.

	Qwen3-1.7B	**Qwen2.5-3B**	**Qwen2.5-7B**
Analyzer calls	1,274	2,074	1,574
$\lvert\mathcal{E}_\tau\rvert = 5$	77%	87%	70%
$\lvert\mathcal{E}_\tau\rvert = 4$	19%	12%	24%
$\lvert\mathcal{E}_\tau\rvert \leq 3$	4%	1%	6%
Mean $\lvert\mathcal{E}_\tau\rvert$	4.73	4.86	4.64

Table 10: Analyzer statistics on ALFWorld over the first 50 training steps ($\lambda_k > 0$), with {max_steps} $= 5$.

that most steps fall back to environment feedback alone; if it returns more, only the first {max_steps} in its output are kept. Second, the hint is required to be an imperative naming the correct object or query, and is explicitly forbidden from describing what went wrong. A hint that restated the failure would only repeat the environment feedback. Only the step indices and hints are used in training; the per-step diagnosis and the first_fatal_error index requested by the prompt are discarded.

Table 10 reports analyzer statistics. A run makes 1.3k–2.1k calls, and the analyzer usually uses the full budget of five error steps, a small fraction of a failed ALFWorld trajectory of up to 50 steps.

H.3 Teacher Context

Figure 14 shows the teacher context $\tilde{h}_t$ of Eq. (4). Both PI sources are appended to the end of the current user turn, and the scored response y_t is left unchanged.

Prompt for the Error-Step Analyzer

System: You are an expert agent trajectory analyzer.

User:

```
## Task
{task}

## Failed Trajectory
{failed}
{success_block}

## Instructions
```

You are given a FAILED trajectory. Each step is labeled `[step N] action: ... | env: ...`; N is that step's index. Indices are 0-based and contiguous (first step is `[step 0]`); valid indices are EXACTLY the N values printed above. {noref_note}
Read the ENTIRE trajectory (and the reference Successful Trajectory, if provided) and judge each step against the `## Task` goal. Select AT MOST {max_steps} KEY decision steps: the steps where the choice of action most determined whether the task would succeed. Prefer the steps that need correcting; do NOT list every step, only the few that mattered most. `first_fatal_error` = the earliest step that made failure unavoidable.
Rules: each `step_index` must be copied verbatim from a printed `[step N]` label (do NOT add/subtract 1).
Respond ONLY with JSON (no prose, no markdown fences):

```
{"error_steps": [{"step_index": <int>, "diagnosis": "<one sentence: why this step is a key decision>", "hint": "<provide one imperative, concise sentence telling the agent the correct action to take at this step; do NOT describe what went wrong or use the word 'agent'>"}, ...], "first_fatal_error": <int>}
```

Figure 13: Prompt template for the LLM error-step analyzer.

Teacher Context at Step t of a Failed Trajectory

```
{student_prompt}

[Immediate feedback]
{next_observation}

[Hint] {hint}
```

Correctly solve the current task.

Figure 14: Teacher context for step t of a failed trajectory. {student_prompt} is the student's prompt at step t (Figures 10–12), {next_observation} is the environment feedback $\Phi_t^{\text{env}} = o_{t+1}$, and {hint} is the corrective hint Φ_t^{llm}. The [Hint] line appears only for $t \in \mathcal{E}_\tau$. Successful trajectories are not reweighted, so no teacher context is built for them.