SIAR: SKILL INFLUENCE ATTRIBUTION FOR SKILL-AUGMENTED AGENTIC REINFORCEMENT LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Large language model (LLM) agents have shown strong potential in long-horizon interactive tasks, and recent skill-augmented agentic reinforcement learning (RL) methods further improve them by distilling past experience into reusable natural-language skills. However, existing methods remain coarse-grained in both skill updating and policy optimization: skills are updated according to weakly diagnostic outcome signals or fixed schedules, and all actions in a trajectory receive a uniform advantage regardless of the retrieved skill's influence. To address these limitations, we propose SIAR, a skill-augmented agentic RL framework centered on skill influence attribution. SIAR measures how strongly the retrieved skill drives the agent toward each executed action, producing a step-level attribution signal that constitutes the missing fine-grained evidence. Building on this signal, attribution-guided skill updating rewrites a skill only when accumulated attribution evidence suggests a mismatch between its influence and task outcomes, and attribution-calibrated advantage assignment augments the uniform trajectory-level advantage with additional credit or penalty for strongly skill-influenced actions. The two components allow skills and the agent policy to co-evolve throughout training. Experiments on three challenging agentic tasks (ALFWorld, WebShop, and search-augmented QA) show that SIAR outperforms existing skill-augmented agentic RL methods.

1 INTRODUCTION

Recent advances in large language models (LLMs) (Anthropic, 2026; OpenAI, 2026a; Google, 2026; Guo et al., 2025a; Qwen Team, 2026) have significantly enhanced capabilities such as instruction following, reasoning, tool utilization, and long-context processing. Building on these capabilities, LLM-powered agents can integrate perception, deliberation, and action in interactive environments, operating through natural-language interfaces with little human supervision. Consequently, LLM agents have demonstrated growing promise in long-horizon tasks, including GUI automation (Nguyen et al., 2025; Tang et al., 2025), software engineering (Guo et al., 2025b; Jimenez et al., 2024), and deep research (Zhang et al., 2025b; Shi et al., 2025). To further improve LLM agents, reinforcement learning (RL) (Sutton & Barto, 1998) provides a principled framework for optimizing agent policies using multi-turn interaction trajectories and trajectory-level feedback.

Agent skills, recently formalized by Anthropic (Anthropic, 2025), are self-contained packages of task-specific instructions that can be invoked on demand and encode procedural knowledge such as subgoals, action patterns, and common failure modes. Compared with raw interaction trajectories, skills facilitate experience transfer by distilling generalizable principles without retaining instance-specific details. Recent studies have further integrated skill construction and refinement into agentic RL, where skills and agent policies are optimized jointly, so that skills serve both as external guidance and as structured priors for policy improvement (Xia et al., 2026; Lu et al., 2026b).

Despite the potential benefits of jointly optimizing skills and policies, existing methods still rely on coarse-grained mechanisms for both skill updating and policy optimization. For skill updating, skills are typically rewritten based on trajectory-level signals, such as raw success or failure outcomes, or according to fixed schedules (Xia et al., 2026; Tu et al., 2026b). However, outcome signals are only weakly diagnostic, because a failed rollout does not necessarily imply that the retrieved skill itself is deficient. Failures may instead result from exploration noise, environmental difficulty, or execution

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.

errors made by the current policy. Fixed schedules are equally problematic, as they trigger updates regardless of whether the skill actually needs correction. Updating skills without sufficient evidence can therefore introduce redundant or conflicting content and overwrite guidance that is still reliable. For policy optimization, most existing skill-augmented RL methods still estimate advantages at the trajectory level (Xia et al., 2026; Lu et al., 2026b; Li et al., 2026). Consequently, all actions within a trajectory receive the same learning signal, even though the influence of the retrieved skill may vary substantially across actions. Such differences in skill influence may inform how the policy learns to incorporate skill guidance based on task outcomes. Both limitations can be attributed to a common deficiency: existing methods lack a mechanism for quantifying the influence of the retrieved skill on individual actions. Accordingly, two central questions emerge: when a skill should be updated, and how advantages should be assigned during skill-augmented policy optimization.

To address these questions, we propose **SIAR** (**S**kill **I**nfluence **A**ttribution for Skill-Augmented Agentic **R**einforcement Learning), a framework centered on step-level skill influence attribution. Specifically, SIAR first quantifies skill influence at each interaction step by comparing the log probabilities assigned to the executed action under skill-conditioned and skill-free contexts, which produces a step-level attribution signal that indicates how strongly each action is driven by the skill. Building on this signal, SIAR introduces two coupled components. First, **attribution-guided skill updating** aggregates the step-level influence scores with trajectory outcomes into task-level consistency evidence. This evidence links each update decision to the skill's measured influence in both successful and failed trajectories. A skill is rewritten only when the task-level consistency score is negative and sufficient trajectories of both outcomes have been collected. Second, **attribution-calibrated advantage assignment** augments the trajectory-level group relative policy optimization (GRPO) (Shao et al., 2024) advantage with a step-level advantage. The additional advantage enriches sparse trajectory-level feedback with targeted step-level signals. Actions with influence scores above the group mean receive additional credit in successful trajectories and additional penalties in failed trajectories. This design aims to help the policy learn from skill guidance associated with success and avoid reinforcing guidance associated with failure. By grounding both skill updating and policy optimization in a shared attribution signal, SIAR aims to maintain reliable skills through evidence-based updating and align policy updates with step-level skill influence throughout agentic RL training. Extensive experiments on ALFWorld (Shridhar et al., 2020b), WebShop (Yao et al., 2022), and seven search-augmented QA benchmarks (Jin et al., 2025) show that SIAR outperforms all compared baselines on the overall metrics.

2 Related Work

2.1 LLM Agents

LLMs are increasingly employed as autonomous agents that interact with external environments, building on paradigms such as reasoning and acting (ReAct) (Yao et al., 2023) and verbal self-reflection (Reflexion) (Shinn et al., 2023). To further enhance LLM agents beyond prompting alone, RL has emerged as a common post-training strategy (Zhang et al., 2025a). Existing approaches include actor-critic methods such as PPO (Schulman et al., 2017), preference-based optimization such as DPO (Rafailov et al., 2023), and critic-free, group-based estimators such as GRPO (Shao et al., 2024), DAPO (Yu et al., 2025), and Dr. GRPO (Liu et al., 2025). Among these methods, GRPO is well suited to agent training due to its critic-free formulation, which eliminates the need to learn an additional value function. Nevertheless, its reliance on trajectory-level advantage estimation exacerbates the challenge of credit assignment. In interactive agent environments, trajectories are often long and rewards are sparse and delayed. Assigning a single scalar advantage to all actions within a trajectory thus fails to distinguish their heterogeneous contributions. To address this limitation, recent studies have proposed finer-grained credit assignment methods that exploit the internal structure of rollouts, including anchor-state step grouping (Feng et al., 2025), context-aware hierarchical grouping (He et al., 2026a), and attribution over a unified state-transition graph (Cheng et al., 2026). By enabling more precise step-level advantage estimation, these methods further improve the training effectiveness and downstream performance of LLM agents. However, these methods derive step-level signals solely from the structure of the rollouts themselves and remain agnostic to the external knowledge that guides the agent.

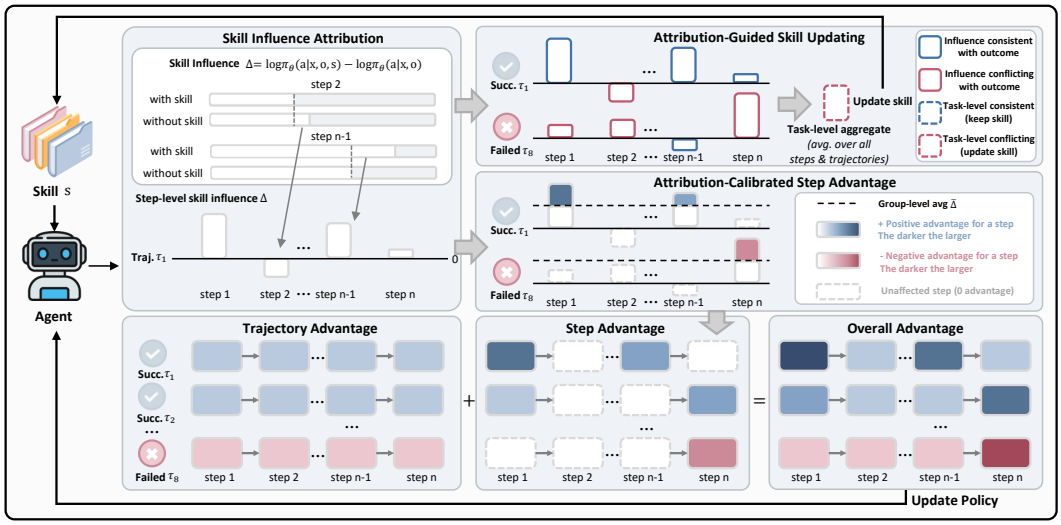

Figure 1: Overview of SIAR, our proposed framework centered on skill influence attribution. SIAR quantifies step-level skill influence and uses the resulting attribution signal to guide skill updating and calibrate advantage assignment during RL training, allowing skills and the agent policy to co-evolve in a mutually reinforcing loop.

2.2 Memory- and Skill-Augmented Agents

Early approaches to improving LLM agents relied on external memory mechanisms that retain information from prior interactions for retrieval at inference time without updating model parameters. Representative implementations include recalled trajectories paired with distilled natural-language insights (Zhao et al., 2024), consolidated long-term memory (Chhikara et al., 2025), and episodic or procedural memory (Fang et al., 2025; Zhang et al., 2026b; Liu et al., 2026). Moving beyond the retention of interaction histories, skill-based approaches distill experience into self-contained and reusable units of procedural knowledge (Anthropic, 2025; Yang et al., 2026; Vishe et al., 2026; Zhang et al., 2026a; Lu et al., 2026a). Skills may be instantiated as executable code modules or natural-language procedures, supporting retrieval and reuse across tasks. Skill-augmented agentic RL further integrates skill library construction and refinement with policy optimization, allowing the library and the policy to be optimized jointly during training. Existing skill-augmented agentic RL methods can be broadly divided into three categories. The first category constructs and maintains structured or hierarchical skill libraries, including recursively refined skill banks (Xia et al., 2026), typed dependency graphs (Li et al., 2026), and dual-granularity banks with utility-based pruning (Tu et al., 2026b). The second integrates skill selection, utilization, and distillation into a unified policy optimized with outcome-based rewards (Shi et al., 2026; Wang et al., 2025; Tu et al., 2026a). The third internalizes explicit skills into model parameters, thereby eliminating skill retrieval at inference time (Lu et al., 2026b; Lin et al., 2026; He et al., 2026b). Despite these advances, existing methods still update skills according to coarse trajectory-level outcomes or fixed schedules, and optimize the policy with trajectory-level advantages, leaving the influence of the retrieved skill on each executed action unexplored.

3 Preliminaries

3.1 Problem Setup

We consider a general setting where an LLM agent interacts with an external environment to complete a task specified by a task prompt x. At each interaction step $t = 1, 2, \ldots, T$, the agent receives the current observation o_t and produces an executable action a_t according to its policy π_θ. The environment then returns the next observation o_{t+1} and a scalar reward r_t. A complete trajectory is denoted by $\tau = \{(o_t, a_t, r_t)\}_{t=1}^{T}$, and for notational simplicity we use a common horizon T for all

trajectories. Compared with single-turn generation, such multi-turn interaction involves long horizons and delayed feedback, which makes agentic tasks more challenging.

3.2 RL FOR AGENTIC TRAINING

RL has become an established paradigm for training LLM agents, and GRPO (Shao et al., 2024; Guo et al., 2025a) has been widely adopted in recent agentic training frameworks. Specifically, GRPO is a critic-free policy optimization method that estimates relative advantages from a group of rollouts sampled for the same task. For a given task x, the policy rolls out G trajectories $\{\tau_i\}_{i=1}^{G}$ and receives rewards $\{R(\tau_i)\}_{i=1}^{G}$ from the environment, where $R(\tau_i) = \sum_{t=1}^{T} r_t^i$ is the total reward of trajectory τ_i. The relative advantage of trajectory τ_i can be estimated as

$$A^E(\tau_i) = \frac{R(\tau_i) - \text{mean}(\{R(\tau_j)\}_{j=1}^{G})}{\text{std}(\{R(\tau_j)\}_{j=1}^{G})}. \tag{1}$$

The policy model is then optimized with a PPO-style clipped objective:

$$J_{\text{GRPO}}(\theta) = \mathbb{E}_{x,\{\tau_i\}_{i=1}^{G}} \left[\frac{1}{GT} \sum_{i=1}^{G} \sum_{t=1}^{T} \min \Big(\rho_\theta(a_t^i)\, A^E(\tau_i),\ \text{clip}\big(\rho_\theta(a_t^i), 1 \pm \epsilon\big) A^E(\tau_i) \Big) \right] - \beta \mathbb{D}_{\text{KL}}(\pi_\theta \| \pi_{\text{ref}}), \tag{2}$$

where $\rho_\theta(a_t^i) = \frac{\pi_\theta(a_t^i | x, o_t^i)}{\pi_{\theta_{\text{old}}}(a_t^i | x, o_t^i)}$ is the importance sampling ratio and ϵ controls the clipping range. The term $\mathbb{D}_{\text{KL}}(\pi_\theta \| \pi_{\text{ref}})$ denotes the Kullback–Leibler (KL) divergence between the current policy π_θ and the reference policy π_{ref}. The coefficient β determines the weight of the KL penalty.

4 SIAR

Agentic tasks require an LLM agent to perform long-horizon decision making in an external environment. Without sufficient task knowledge, the agent may explore ineffectively, repeat past mistakes, or take irreversible actions. Skills mitigate these problems by injecting reusable procedural knowledge into the agent's prompt, which narrows the search space and guides the agent toward the task objective. However, a retrieved skill does not influence every action equally, as the agent may follow its guidance closely at some steps while relying on its own knowledge at other steps. Because this influence is not directly observable, it is difficult to judge whether a skill remains reliable, or to credit the actions that its guidance actually shaped. SIAR makes this influence measurable through a step-level skill influence score and uses it as a shared signal for both skill updating and policy optimization. Building on this signal, we develop two coupled components, **attribution-guided skill updating** and **attribution-calibrated advantage assignment**, which are detailed in the following subsections.

4.1 STEP-LEVEL SKILL INFLUENCE ATTRIBUTION

To attribute the agent's behavior to the retrieved skill s, we quantify its influence on each executed action by comparing the log probabilities assigned to the same action tokens under a skill-conditioned context (x, o_t^i, s) and a skill-free context (x, o_t^i), as shown in the *left part* of Figure 1. Specifically, consider the t-th step of trajectory τ_i, where the agent observes o_t^i under task prompt x and generates an action $a_t^i = (a_{t,1}^i, \ldots, a_{t,L}^i)$ of L tokens according to the skill-conditioned policy $\pi_\theta(\cdot \mid x, o_t^i, s)$. We then keep the generated action tokens fixed and recompute their log probabilities under the same task prompt and observation but without injecting the skill. The step-level skill influence score is defined as the average difference between the log probabilities assigned to the fixed action tokens under the two contexts:

$$\Delta_t^i(s) = \text{sg}\left[\frac{1}{L} \sum_{\ell=1}^{L} \Big(\log \pi_\theta(a_{t,\ell}^i \mid x, o_t^i, s, a_{t,<\ell}^i) - \log \pi_\theta(a_{t,\ell}^i \mid x, o_t^i, a_{t,<\ell}^i) \Big) \right], \tag{3}$$

where $\text{sg}[\cdot]$ denotes the stop-gradient operator, which prevents gradients from flowing through the attribution signal. A positive $\Delta_t^i(s)$ indicates that conditioning on the skill raises the probability

of the executed action relative to the skill-free context and thus encourages the action, whereas a negative $\Delta_t^i(s)$ indicates that the skill suppresses it. Consequently, $\Delta_t^i(s)$ quantifies the influence of the retrieved skill on the probability of each executed action and serves as the shared attribution signal for the following two components.

4.2 Attribution-Guided Skill Updating

The attribution signal enables a skill to be judged by its measured influence on the agent rather than by raw outcomes alone or by fixed schedules. Since a failed trajectory may reflect exploration noise, environmental difficulty, or policy execution errors rather than a deficient skill, we use accumulated attribution evidence from successful and failed trajectories to guide skill updating.

To quantify the required evidence, we compare the skill's step-level influence with the trajectory-level outcome, as depicted in the *upper-right* part of Figure 1. For each trajectory τ_i with total reward $R(\tau_i)$, we define the step-level influence-reward consistency score as

$$\kappa_t^i(s) = \Delta_t^i(s) \cdot \mathrm{sign}(R(\tau_i)), \tag{4}$$

where $\mathrm{sign}(R(\tau_i)) = +1$ if $R(\tau_i) > 0$, and $\mathrm{sign}(R(\tau_i)) = -1$ if $R(\tau_i) \leq 0$. The sign of $\kappa_t^i(s)$ indicates whether the skill's influence at step t is consistent with the final trajectory outcome. When $\kappa_t^i(s) > 0$, the skill increases the probability of an action executed in a successful trajectory or decreases it in a failed one. Conversely, when $\kappa_t^i(s) < 0$, the skill increases the probability of an action executed in a failed trajectory or decreases it in a successful one.

Since $\kappa_t^i(s)$ reflects consistency only at a single interaction step, we aggregate it across steps and trajectories to obtain task-level evidence for skill updating. For each task category q, let s_q denote its associated skill and $\mathcal{T}_q$ the set of trajectories collected for tasks of that category within the current update period. The resulting task-level consistency score is defined as

$$\kappa(s_q) = \frac{1}{|\mathcal{T}_q|\, T} \sum_{\tau_i \in \mathcal{T}_q} \sum_{t=1}^{T} \kappa_t^i(s_q), \tag{5}$$

which averages the step-level consistency scores over all steps of the trajectories collected for the category. A negative $\kappa(s_q)$ indicates that negative step-level consistency scores outweigh positive ones across the collected trajectories. We therefore treat the current skill s_q as a candidate for updating.

Given the task-level consistency score, we evaluate whether the current skill should be updated at scheduled checking steps. Let g denote the current training step, K the checking period, and n_q^+ and n_q^- the numbers of successful and failed trajectories collected for category q during the current update period. The update indicator for skill s_q is defined as

$$\mathrm{Update}(s_q) = \mathbb{I}\left[g \bmod K = 0 \ \wedge \ \kappa(s_q) < 0 \ \wedge \ n_q^+ \geq N^+ \ \wedge \ n_q^- \geq N^-\right], \tag{6}$$

where $\mathbb{I}[\cdot]$ denotes the indicator function, and N^+ and N^- are the minimum numbers of successful and failed trajectories required for an update. Thus, a skill is updated only when the scheduled check is reached, the task-level consistency score is negative, and sufficient contrastive evidence has been collected. Once an update is triggered, the skill is rewritten by an auxiliary LLM conditioned on both successful and failed trajectories collected for that category, following the updating rules detailed in Appendix C.

4.3 Attribution-Calibrated Advantage Assignment

Beyond guiding skill updating, the attribution signal also informs step-level advantage assignment during policy optimization. Standard GRPO assigns the same advantage to all actions in a trajectory. It therefore does not reflect differences in how the retrieved skill affects their probabilities. We use these differences to direct an additional learning signal toward actions with influence scores above the group-level baseline, as depicted in the *middle-right part* of Figure 1. Formally, we define the *attribution-calibrated step advantage* as

$$A^S(a_t^i) = \mathrm{sign}(R(\tau_i)) \cdot \max\left(0, \Delta_t^i(s) - \bar{\Delta}\right), \tag{7}$$

Table 1: Experimental results on ALFWorld and WebShop. Bold and underline indicate the best and the second-best overall results (All, Score, and Succ.), respectively.

Method	ALFWorld							WebShop	
	Pick	Look	Clean	Heat	Cool	Pick2	**All**	**Score**	**Succ.**
Closed-source LLMs									
GPT-4o	75.3	60.8	31.2	56.7	21.6	49.8	48.0	31.8	23.7
Gemini-2.5-Pro	92.8	63.3	62.1	69.0	26.6	58.7	60.3	42.5	35.9
Qwen2.5-7B-Instruct									
Qwen2.5	33.4	21.6	19.3	6.90	2.80	3.20	14.8	26.4	7.80
Prompt-based or Memory-based Methods									
ReAct	48.5	35.4	34.3	13.2	18.2	17.6	31.2	46.2	19.5
Reflexion	62.0	41.6	44.9	30.9	36.3	23.8	42.7	58.1	28.8
Mem0	54.0	55.0	26.9	36.4	20.8	7.69	33.6	23.9	2.00
ExpeL	21.0	67.0	55.0	52.0	71.0	6.00	46.3	30.9	11.2
MemP	54.3	38.5	48.1	56.2	32.0	16.7	41.4	25.3	6.40
SimpleMem	64.5	33.3	20.0	12.5	33.3	3.84	29.7	33.2	8.59
RL-based Methods									
RLOO	87.6	78.2	87.3	81.3	71.9	48.9	75.5	80.3	65.7
GRPO	90.8	66.1	89.3	74.7	72.5	64.7	77.6	79.3	66.1
Memory-Augmented or Skill-Augmented RL-based Methods									
MemRL	62.8	38.5	22.2	12.5	8.00	0.00	21.4	29.5	9.20
EvolveR	64.9	33.3	46.4	13.3	33.3	33.3	43.8	42.5	17.6
Mem0+GRPO	78.1	54.8	56.1	31.0	65.0	26.9	54.7	58.1	37.5
SimpleMem+GRPO	89.5	36.3	60.0	50.0	64.9	26.3	62.5	67.8	46.9
SkillRL	97.9	71.4	90.0	90.0	95.5	87.5	89.9	85.2	72.7
Skill0	100.0	85.8	94.6	81.9	85.7	80.1	89.8	85.1	74.2
D2Skill	97.1	87.5	100.0	100.0	75.0	78.6	<u>90.6</u>	<u>91.1</u>	<u>80.5</u>
SkillC	88.5	78.6	91.2	94.7	100.0	88.9	<u>90.6</u>	85.6	74.0
SIAR (Ours)	100.0	90.5	95.6	97.2	82.2	92.3	**93.4**	**92.0**	**82.4**

where the baseline $\bar{\Delta} = \frac{1}{GT}\sum_{i=1}^{G}\sum_{t=1}^{T}\Delta_t^i(s)$ denotes the group-level average skill influence, computed across all steps in the G trajectories sampled for task x. The rectification operator $\max(0,\cdot)$ ensures that only steps with influence scores above $\bar{\Delta}$ receive a nonzero additional advantage. The term $\text{sign}(R(\tau_i))$ determines its direction according to the trajectory outcome. The additional term reinforces these actions in successful trajectories and discourages them in failed trajectories. This design aims to help the policy adapt its use of skill guidance based on task outcomes. All other steps receive $A^S(a_t^i) = 0$ and retain their original GRPO advantage. We then define the overall advantage as a weighted sum of the trajectory-level GRPO advantage and the attribution-calibrated step advantage, as illustrated in the *bottom* part of the overview in Figure 1:

$$A(a_t^i) = A^E(\tau_i) + \omega \cdot A^S(a_t^i), \tag{8}$$

where ω is a hyperparameter that controls the contribution of the step-level term.

Finally, we formulate the attribution-calibrated policy objective by incorporating the overall advantage $A(a_t^i)$ defined in Eq. (8):

$$J_{\text{SIAR}}(\theta) = \mathbb{E}_{x,\{\tau_i\}_{i=1}^{G}}\left[\frac{1}{GT}\sum_{i=1}^{G}\sum_{t=1}^{T}\min\Big(\rho_\theta(a_t^i)\,A(a_t^i),\ \text{clip}\big(\rho_\theta(a_t^i),1\pm\epsilon\big)A(a_t^i)\Big)\right] - \beta\mathbb{D}_{\text{KL}}(\pi_\theta\|\pi_{\text{ref}}), \tag{9}$$

where the importance ratio $\rho_\theta(a_t^i)$ is computed under the skill-conditioned context (x, o_t^i, s).

In combination, attribution-guided skill updating and attribution-calibrated advantage assignment ground skill maintenance and policy optimization in the same step-level attribution signal. The two components form a mutually reinforcing loop during training. The calibrated advantage uses task feedback to help the policy learn how to use skill guidance. The skill update rule combines influence

Table 2: Experimental results on search-augmented QA. † and ⋆ indicate in-domain and out-of-domain benchmarks, respectively. Bold and underline indicate the best and the second-best average results (Avg.), respectively.

Method	Single-Hop QA			Multi-Hop QA				Avg.
	NQ†	TriviaQA⋆	PopQA⋆	HotpotQA†	2Wiki⋆	MuSiQue⋆	Bamboogle⋆	
Qwen2.5-7B-Instruct								
Qwen2.5	11.6	35.6	1.20	16.4	22.2	4.80	14.4	15.2
CoT	12.8	35.6	3.80	16.2	22.6	6.60	24.0	17.4
RAG	27.4	58.2	17.8	25.8	23.2	9.40	16.8	25.5
Search-o1	19.4	40.6	11.4	17.0	27.0	8.60	30.4	22.1
R1-Instruct	21.0	44.9	17.1	20.8	27.5	6.00	19.2	22.4
Search-R1	39.3	61.0	39.7	37.0	40.1	14.6	36.8	38.5
ZeroSearch	43.6	61.8	51.5	34.6	35.2	18.4	27.8	39.1
StepSearch	-	-	-	38.6	36.6	22.6	40.0	-
EvolveR	43.5	63.4	44.6	38.2	42.0	15.6	54.4	43.1
SkillRL	45.9	63.3	45.9	43.2	40.3	20.2	73.8	<u>47.1</u>
Skill0	42.7	61.1	45.3	40.0	38.3	16.4	66.9	44.4
SIAR (Ours)	45.1	64.3	47.1	44.7	45.0	20.4	75.4	**48.9**

scores with trajectory outcomes to select skills for updating. The updated policy and skills jointly shape subsequent rollouts, which provide new attribution evidence and task feedback for further updates. Through this loop, skills and the agent policy co-evolve throughout agentic RL training.

5 Experiments

5.1 Experimental Setup

Benchmarks. We evaluate our method SIAR on three agentic tasks: ALFWorld (Shridhar et al., 2020b), WebShop (Yao et al., 2022), and search-augmented QA (Jin et al., 2025). ALFWorld requires completing long-horizon household tasks through sequential actions in a text-based embodied environment, and WebShop requires finding and purchasing products that match natural-language instructions in a simulated e-commerce website. Search-augmented QA requires answering open-domain questions by iteratively issuing search queries and reasoning over the retrieved evidence. Further details of the three tasks are provided in Appendix A.

Baselines. On ALFWorld and WebShop, we compare SIAR against five categories of methods: **(1)** closed-source LLMs; **(2)** the backbone model Qwen2.5-7B-Instruct without additional agentic training; **(3)** prompt-based or memory-based methods; **(4)** RL-based methods; and **(5)** memory-augmented or skill-augmented RL methods. For search-augmented QA, the baselines comprise prompting and retrieval-augmented methods, RL-based search agents, and memory-augmented or skill-augmented RL methods. Each baseline is described in Appendix A.

Training Details. We use Qwen2.5-7B-Instruct (Yang et al., 2025) as the base model for SIAR. For attribution-guided skill updating, the minimum numbers of successful and failed trajectories, N^+ and N^-, are both set to 5, and the checking period K is set to 15 for ALFWorld and WebShop and to 20 for search-augmented QA. For attribution-calibrated advantage assignment, the weight ω is set to 1.5 for ALFWorld and WebShop and to 2.0 for search-augmented QA. More training details can be found in Appendix A.

5.2 Main Results

Table 1 shows that SIAR achieves the highest overall success rates on the ALFWorld and WebShop benchmarks among all compared methods. On ALFWorld, SIAR reaches 93.4%, exceeding SkillRL by 3.5 percentage points and the strongest baselines, D2Skill and SkillC, by 2.8 points. At the category level, SIAR achieves success rates above 90% in five of the six categories. It reaches a success rate of 100.0% on Pick and achieves the highest success rates on Look (90.5%) and Pick2 (92.3%). The difference between its highest and lowest category success rates is 17.8 percentage

points, compared with 19.9 to 26.5 points for the four skill-augmented baselines. Thus, SIAR combines a high overall success rate with a smaller performance gap across task categories. On WebShop, SIAR achieves both the highest task score of 92.0 and the highest success rate of 82.4%, exceeding D2Skill by 0.9 and 1.9 points, respectively. The improvement across both metrics further supports the effectiveness of SIAR in web shopping tasks.

Table 2 reports the results on search-augmented QA. According to Table 2, SIAR achieves the highest average success rate of 48.9%, exceeding SkillRL by 1.8 percentage points and Skill0 by 4.5 points. Compared with SkillRL, SIAR improves performance on six of the seven benchmarks, covering both single-hop and multi-hop QA. The largest improvement occurs on 2Wiki, where the success rate increases from 40.3% to 45.0%. SIAR also outperforms SkillRL on all five out-of-domain benchmarks: TriviaQA, PopQA, 2Wiki, MuSiQue, and Bamboogle. Since training is performed only on NQ and HotpotQA, these out-of-domain improvements suggest that SIAR learns search strategies that generalize beyond the training benchmarks.

5.3 Qualitative Analysis

Figure 2 compares a successful and a failed trajectory from the same group for a Cool & Place task in ALFWorld. We examine the highest-influence step in each trajectory and relate its additional advantage to the observed action. In the successful trajectory, the agent takes `cup 1` from `cabinet 2` (step 18), and the next observation confirms that the cup has been picked up. This action has an influence score of 0.54, above the group mean of 0.31. It receives the largest positive additional advantage in the trajectory. The additional credit therefore reinforces an action that supports subsequent cooling and placement. In the failed trajectory, the agent examines `countertop 1` while at `microwave 1` (step 34), and this invalid action has the highest influence score in the trajectory. The environment returns "Nothing happens.", showing that the action makes no progress in the observed state. Its score also exceeds the group mean, and the failed outcome results in the largest additional penalty in the trajectory. These cases illustrate the intended role of the additional advantage: under ideal conditions, it reinforces skill-guided behavior that supports task completion and discourages behavior that makes no progress. The two steps also contribute positive and negative consistency scores to the evidence used for skill updating, respectively. Additional cases are provided in Appendix F.

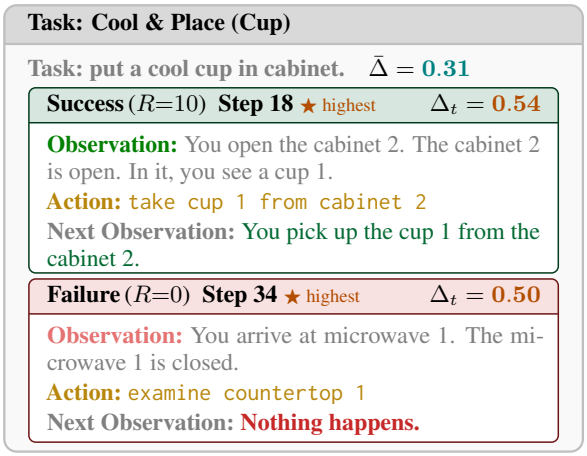

Figure 2: Examples of the highest-influence steps of a successful and a failed trajectory on an ALFWorld task.

5.4 Ablation Study

Table 3 reports the ablation results on ALFWorld and search-augmented QA. Equipping GRPO with static skills already outperforms vanilla GRPO (82.8% vs. 77.6% on ALFWorld), confirming that skills provide useful priors for agentic training. Updating the skills on a fixed schedule brings further yet limited benefits, as schedule-driven updates may also rewrite skills that are already reliable. Replacing scheduled updating with attribution-guided skill updating improves the success rate from 83.6% to 88.3% on ALFWorld and from 48.2% to 48.4% on search-augmented QA. This comparison

Table 3: Ablation study on ALFWorld and search-augmented QA.

Configuration	ALFWorld	Search QA
GRPO + static skills	82.8	47.1
GRPO + scheduled updating	83.6	48.2
GRPO + attribution-guided updating	88.3	48.4
SIAR	93.4	48.9
group → trajectory baseline	90.2	48.4

supports combining skill influence with task outcomes to determine when to update a skill. With attribution-guided skill updating retained, adding the attribution-calibrated advantage increases success rates from 88.3% to 93.4% on ALFWorld and from 48.4% to 48.9% on search-augmented QA. This improvement supports enriching sparse trajectory-level feedback with targeted step-level signals. The additional advantage reinforces actions with influence scores above the group mean in successful trajectories and discourages them in failed trajectories, helping the policy learn from skill guidance based on task outcomes. Finally, replacing the group-level baseline Δ with a trajectory-level baseline reduces success rates to 90.2% and 48.4%, respectively, suggesting that the group-level reference provides a more reliable estimate of skill influence for step-level advantage calibration.

5.5 Skill Updating Frequency

To examine the frequency of attribution-guided skill updating, Figure 3 reports the number of updates per skill during training. The scheduled updating baseline updates every skill 10 times under the same training budget. SIAR triggers fewer updates for every skill: all six ALFWorld skills are updated at most five times, and the multi-hop QA skill is updated only once. Together with the higher success rates of SIAR, these results support the value of attribution-guided skill updating. This mechanism is designed to concentrate updates on skills that need refinement and avoid unnecessary rewrites.

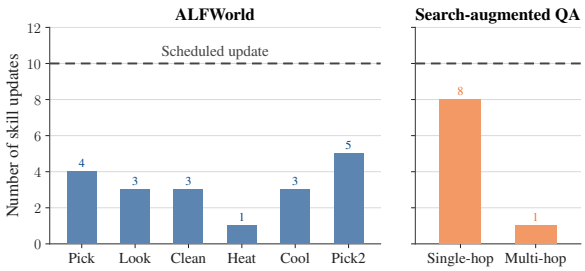

Figure 3: Number of updates per skill under SIAR (bars) compared with the scheduled updating baseline (dashed lines) on ALFWorld and search-augmented QA.

5.6 Hyperparameter Analysis

To examine the sensitivity of SIAR to the weight ω of the attribution-calibrated step advantage, we vary $\omega \in \{1.0, 1.5, 2.0, 2.5\}$ and report the average success rates on ALFWorld and search-augmented QA in Figure 4. The performance of SIAR remains stable on search-augmented QA, varying only from 48.2% to 48.9%, while on ALFWorld it peaks at $\omega = 1.5$ and stays above the vanilla GRPO baseline across the entire tested range. When ω is too small, the step-level signal is dominated by the trajectory-level advantage, whereas an overly large ω over-amplifies the step-level term and degrades policy optimization in agentic tasks.

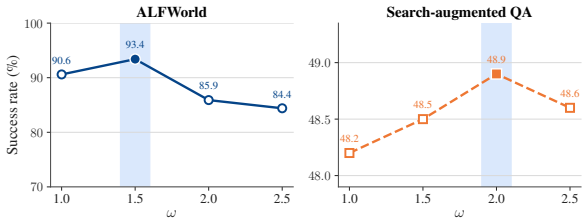

Figure 4: Hyperparameter sensitivity analysis of ω on ALFWorld and search-augmented QA.

6 Conclusion

In this paper, we presented SIAR, a skill-augmented agentic RL framework centered on skill influence attribution. By contrasting action log probabilities under skill-conditioned and skill-free contexts, SIAR obtains a shared step-level attribution signal for skill updating and policy optimization. Attribution-guided skill updating uses accumulated evidence to determine when to update skills, while attribution-calibrated advantage assignment provides additional credit or penalties based on skill influence and task outcomes. Together, these components form a feedback loop that allows skills and the agent policy to co-evolve throughout training. Experiments on ALFWorld, WebShop, and search-augmented QA show that SIAR achieves the best overall performance, reaching success rates of 93.4% on ALFWorld, 82.4% on WebShop, and 48.9% on average across seven QA benchmarks. Ablation studies further verify the contribution of each component.

AI USE STATEMENT

Generative AI tools were used in this work in three distinct roles. First, an LLM is a component of the proposed method itself: as described in Appendix C, GPT-5.4 acts as the auxiliary skill generator and skill updater that produces the initial skills and rewrites them when the update condition in Eq. (6) is triggered. This use is intrinsic to SIAR and is reported together with the prompts in Appendix C. We also report results with o3 (OpenAI, 2025) as the skill updater in Appendix B.4. Second, we used GPT-5.4-mini (OpenAI, 2026b) as an automatic judge to assess the steps selected in the attribution validation, as reported in Tables 5 and 6 of Appendix B.2. Third, beyond these uses, we used generative AI tools only as writing assistants, limited to grammar correction, style improvement, and wording adjustments for clarity. All AI-assisted text was reviewed and verified by the authors, and we take full responsibility for the final content of this work, including its text, claims, and artifacts.

ETHICS STATEMENT

All experiments in this work are conducted on publicly available benchmarks and simulated environments, including ALFWorld, WebShop, and seven search-augmented QA benchmarks, which involve no sensitive personal information or privacy concerns. Since our study is confined to simulated environments and does not involve real-world deployment, its immediate societal impact is limited. From a longer-term perspective, skill influence attribution may contribute positively to the transparency of agentic training, as it makes explicit how strongly each action is driven by external skill guidance and thus helps developers audit skill-augmented agents and detect misleading guidance.

REPRODUCIBILITY STATEMENT

The complete method is specified in Section 4, where Eq. (3) defines the step-level attribution signal, Eq. (6) defines the skill updating condition, and Eqs. (7) to (9) define the attribution-calibrated objective. Algorithm 1 in Appendix E summarizes the full training procedure. Appendix A reports the benchmark construction and all training hyperparameters for each of the three tasks, together with the underlying frameworks and the hardware used. Appendix A.6 describes the evaluation protocol and metrics. Appendix C documents the skill template, how the initial skills are generated, and the updating rules and updating prompt, and Appendix D provides the full agent prompt templates for all three tasks. The code will be publicly available after the acceptance of this paper.

REFERENCES

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

Anthropic. Introducing Agent Skills. https://claude.com/blog/skills, 2025. Official announcement.

Anthropic. Introducing Claude Opus 4.8. https://www.anthropic.com/news/claude-opus-4-8, 2026. Official announcement.

Xin Cheng, Shuo He, Lang Feng, HaiYang Xu, Ming Yan, Lei Feng, and Bo An. Beyond trajectory-level attribution: Graph-based credit assignment for agentic reinforcement learning. *arXiv preprint arXiv:2605.26684*, 2026.

Prateek Chhikara, Dev Khant, Saket Aryan, Taranjeet Singh, and Deshraj Yadav. Mem0: Building production-ready AI agents with scalable long-term memory. *arXiv preprint arXiv:2504.19413*, 2025.

Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, et al. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*, 2025.

Runnan Fang, Yuan Liang, Xiaobin Wang, Jialong Wu, Shuofei Qiao, Pengjun Xie, Fei Huang, Huajun Chen, and Ningyu Zhang. Memp: Exploring agent procedural memory. *arXiv preprint arXiv:2508.06433*, 2025.

Lang Feng, Zhenghai Xue, Tingcong Liu, and Bo An. Group-in-group policy optimization for LLM agent training. In *Advances in Neural Information Processing Systems*, volume 38, pp. 46375–46408, 2025.

Google. Gemini 3.1 Pro: Announcing our latest Gemini AI model. https://blog.google/innovation-and-ai/models-and-research/gemini-models/gemini-3-1-pro/, 2026. Official announcement.

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. *arXiv preprint arXiv:2501.12948*, 2025a.

Jiale Guo, Suizhi Huang, Mei Li, Dong Huang, Xingsheng Chen, Regina Zhang, Zhijiang Guo, Han Yu, Siu-Ming Yiu, Pietro Lio, et al. A comprehensive survey on benchmarks and solutions in software engineering of LLM-empowered agentic system. *arXiv preprint arXiv:2510.09721*, 2025b.

Shuo He, Lang Feng, Qi Wei, Xin Cheng, Lei Feng, and Bo An. Hierarchy-of-groups policy optimization for long-horizon agentic tasks. In *International Conference on Learning Representations (ICLR)*, 2026a.

Zhongyu He, Yuanfan Li, Fei Huang, Tianyu Chen, Siyuan Chen, Xingyang Li, Meng Hsuan Yu, Xiangrong Liu, Leyi Wei, Lu Pan, et al. SIRI: Self-internalizing reinforcement learning with intrinsic skills for LLM agent training. *arXiv preprint arXiv:2606.02355*, 2026b.

Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. Constructing a multi-hop QA dataset for comprehensive evaluation of reasoning steps. *arXiv preprint arXiv:2011.01060*, 2020.

Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. GPT-4o system card. *arXiv preprint arXiv:2410.21276*, 2024.

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 (ICLR)*, 2024.

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.

Mandar Joshi, Eunsol Choi, Daniel S Weld, and Luke Zettlemoyer. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. *arXiv preprint arXiv:1705.03551*, 2017.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, et al. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 2019.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Advances in Neural Information Processing Systems*, volume 33, pp. 9459–9474, 2020.

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. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 5420–5438, 2025.

Xiaoyuan Li, Moxin Li, Keqin Bao, Yubo Ma, Wenjie Wang, Dayiheng Liu, and Fuli Feng. SkillGraph: Skill-augmented reinforcement learning for agents via evolving skill graphs. *arXiv preprint arXiv:2605.12039*, 2026.

Hongxiang Lin, Zhirui Kuai, Erpeng Xue, and Lei Wang. SKILLC: Learning autonomous skill internalization in LLM agents via contrastive credit assignment. *arXiv preprint arXiv:2605.27899*, 2026.

Jiaqi Liu, Yaofeng Su, Peng Xia, Siwei Han, Zeyu Zheng, Cihang Xie, Mingyu Ding, and Huaxiu Yao. SimpleMem: Efficient lifelong memory for LLM agents. *arXiv preprint arXiv:2601.02553*, 2026.

Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. Understanding R1-Zero-like training: A critical perspective. *arXiv preprint arXiv:2503.20783*, 2025.

Zhengxi Lu, Zhiyuan Yao, Zhuowen Han, Zi-Han Wang, Jinyang Wu, Qi Gu, Xunliang Cai, Weiming Lu, Jun Xiao, Yueting Zhuang, et al. Self-distilled agentic reinforcement learning. *arXiv preprint arXiv:2605.15155*, 2026a.

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*, 2026b.

Alex Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Hannaneh Hajishirzi, and Daniel Khashabi. When not to trust language models: Investigating effectiveness and limitations of parametric and non-parametric memories. *arXiv preprint arXiv:2212.10511*, 2022.

Dang Nguyen, Jian Chen, Yu Wang, Gang Wu, Namyong Park, Zhengmian Hu, Hanjia Lyu, Junda Wu, Ryan Aponte, Yu Xia, et al. GUI agents: A survey. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 22522–22538, 2025.

OpenAI. Introducing OpenAI o3 and o4-mini. https://openai.com/index/introducing-o3-and-o4-mini/, 2025. Official announcement.

OpenAI. Introducing GPT-5.4. https://openai.com/index/introducing-gpt-5-4/, 2026a. Official announcement.

OpenAI. Introducing GPT-5.4 mini and nano. https://openai.com/index/introducing-gpt-5-4-mini-and-nano/, 2026b. Official announcement.

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.

Qwen Team. Qwen3.6-Plus: Towards real world agents. https://qwen.ai/blog?id=qwen3.6, 2026. Official release blog.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. In *Advances in Neural Information Processing Systems*, volume 36, pp. 53728–53741, 2023.

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

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

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. HybridFlow: A flexible and efficient RLHF framework. In *Proceedings of the Twentieth European Conference on Computer Systems*, pp. 1279–1297, 2025.

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.

Zhengliang Shi, Yiqun Chen, Haitao Li, Weiwei Sun, Shiyu Ni, Yougang Lyu, Run-Ze Fan, Bowen Jin, Yixuan Weng, Minjun Zhu, et al. Deep research: A systematic survey. *arXiv preprint arXiv:2512.02038*, 2025.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 36, pp. 8634–8652, 2023.

Mohit Shridhar, Jesse Thomason, Daniel Gordon, Yonatan Bisk, Winson Han, Roozbeh Mottaghi, Luke Zettlemoyer, and Dieter Fox. ALFRED: A benchmark for interpreting grounded instructions for everyday tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10740–10749, 2020a.

Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. ALFWorld: Aligning text and embodied environments for interactive learning. *arXiv preprint arXiv:2010.03768*, 2020b.

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

Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT Press, Cambridge, MA, 1998.

Fei Tang, Haolei Xu, Hang Zhang, Siqi Chen, Xingyu Wu, Yongliang Shen, Wenqi Zhang, Guiyang Hou, Zeqi Tan, Yuchen Yan, et al. A survey on (M)LLM-based GUI agents. *arXiv preprint arXiv:2504.13865*, 2025.

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

Songjun Tu, Chengdong Xu, Qichao Zhang, Yiwen Ma, Yaocheng Zhang, Linjing Li, Dong Li, Xiangyuan Lan, and Dongbin Zhao. UCOB: Learning to utilize and evolve agentic skills via credit-aware on-policy bidirectional self-distillation. *arXiv preprint arXiv:2606.29502*, 2026a.

Songjun Tu, Chengdong Xu, Qichao Zhang, Yaocheng Zhang, Xiangyuan Lan, Linjing Li, and Dongbin Zhao. Dynamic dual-granularity skill bank for agentic RL. *arXiv preprint arXiv:2603.28716*, 2026b.

Yash Vishe, Rohan Surana, Xunyi Jiang, Zihan Huang, Xintong Li, Nikki Lijing Kuang, Tong Yu, Ryan A Rossi, Jingbo Shang, Julian McAuley, et al. Skill-R1: Agent skill evolution via reinforcement learning. *arXiv preprint arXiv:2605.09359*, 2026.

Jiongxiao Wang, Qiaojing Yan, Yawei Wang, Yijun Tian, Soumya Smruti Mishra, Zhichao Xu, Megha Gandhi, Panpan Xu, and Lin Lee Cheong. Reinforcement learning for self-improving agent with skill library. *arXiv preprint arXiv:2512.17102*, 2025.

Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. Text embeddings by weakly-supervised contrastive pre-training. *arXiv preprint arXiv:2212.03533*, 2022.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, volume 35, pp. 24824–24837, 2022.

Rong Wu, Xiaoman Wang, Jianbiao Mei, Pinlong Cai, Daocheng Fu, Cheng Yang, Licheng Wen, Xuemeng Yang, Yufan Shen, Yuxin Wang, et al. EvolveR: Self-evolving LLM agents through an experience-driven lifecycle. *arXiv preprint arXiv:2510.16079*, 2025.

Peng Xia, Jianwen Chen, Hanyang Wang, Jiaqi Liu, Kaide Zeng, Yu Wang, Siwei Han, Yiyang Zhou, Xujiang Zhao, Haifeng Chen, et al. SkillRL: Evolving agents via recursive skill-augmented reinforcement learning. *arXiv preprint arXiv:2602.08234*, 2026.

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, 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*, 2025.

Cehao Yang, Xiaojun Wu, Honghao Liu, Xueyuan Lin, Chengjin Xu, Xuhui Jiang, Yuanliang Sun, Wenjie Zhang, Zhichao Shi, Yijie Xu, et al. A survey of agent skills: Toward procedural infrastructure for LLM agents. *Preprints*, 2026.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D Manning. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 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*, volume 35, pp. 20744–20757, 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 (ICLR)*, 2023.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, et al. DAPO: An open-source LLM reinforcement learning system at scale. In *Advances in Neural Information Processing Systems*, volume 38, pp. 113222–113244, 2025.

Guibin Zhang, Hejia Geng, Xiaohang Yu, Zhenfei Yin, Zaibin Zhang, Zelin Tan, Heng Zhou, Zhongzhi Li, Xiangyuan Xue, Yijiang Li, et al. The landscape of agentic reinforcement learning for LLMs: A survey. *arXiv preprint arXiv:2509.02547*, 2025a.

Hanrong Zhang, Shicheng Fan, Henry Peng Zou, Yankai Chen, Zhenting Wang, Jiayu Zhou, Chengze Li, Wei-Chieh Huang, Yifei Yao, Kening Zheng, et al. CoEvoSkills: Self-evolving agent skills via co-evolutionary verification. *arXiv preprint arXiv:2604.01687*, 2026a.

Shengtao Zhang, Jiaqian Wang, Ruiwen Zhou, Junwei Liao, Yuchen Feng, Zhuo Li, Yujie Zheng, Weinan Zhang, Ying Wen, Zhiyu Li, et al. MemRL: Self-evolving agents via runtime reinforcement learning on episodic memory. *arXiv preprint arXiv:2601.03192*, 2026b.

Wenlin Zhang, Xiaopeng Li, Yingyi Zhang, Pengyue Jia, Yichao Wang, Huifeng Guo, Yong Liu, and Xiangyu Zhao. Deep research: A survey of autonomous research agents. *arXiv preprint arXiv:2508.12752*, 2025b.

Andrew Zhao, Daniel Huang, Quentin Xu, Matthieu Lin, Yong-Jin Liu, and Gao Huang. ExpeL: LLM agents are experiential learners. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 19632–19642, 2024.

Xuhui Zheng, Kang An, Ziliang Wang, Yuhang Wang, and Yichao Wu. StepSearch: Igniting LLMs search ability via step-wise proximal policy optimization. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 21805–21830, 2025.

A Experimental Details

A.1 Details of ALFWorld

ALFWorld (Shridhar et al., 2020b) is a text-based embodied-agent benchmark built on household tasks from ALFRED (Shridhar et al., 2020a), in which agents complete long-horizon goals through sequential actions in simulated household environments. The benchmark includes six categories of household tasks: Pick & Place (Pick), Examine in Light (Look), Clean & Place (Clean), Heat & Place (Heat), Cool & Place (Cool), and Pick Two & Place (Pick2). For evaluation, we run 128 parallel environments for two rounds, and each environment samples a test task in every round. The reported success rate is averaged over the resulting 256 evaluation episodes. During the training phase, the maximum prompt length is 4096 tokens, and the maximum response length is 512 tokens. Each trajectory allows up to 50 environment steps. The learning rate is set to 1×10^{-6} for the actor. We adopt a rule-based reward mechanism, which assigns a reward of 10 for a successful trajectory and 0 for a failed trajectory. To handle invalid actions generated by the agent, we impose an additional reward penalty of -0.1. For our method, the group size is set to 8, and 16 different tasks are sampled per training step, resulting in a total of $16 \times 8 = 128$ environments. The rollout temperature is set to 1.0 during the training phase, while the temperature is set to 0.4 during validation. The mini-batch size is 256, and the KL-divergence loss coefficient is set to 0.01. The total number of training iterations is set to 150.

A.2 Details of WebShop

WebShop (Yao et al., 2022) is a simulated e-commerce web environment for grounded language agents, in which agents navigate webpages and issue actions to find, customize, and purchase products according to natural-language instructions. The benchmark includes five categories of shopping tasks: Beauty, Electronics, Fashion, Garden, and Grocery. For evaluation, we run 128 parallel environments for two rounds, and each round samples 128 instructions. The reported score and success rate are averaged over the resulting 256 evaluation episodes. During the training phase, the maximum prompt length is 8192 tokens, and the maximum response length is 512 tokens. Each trajectory allows up to 15 environment steps. The learning rate is set to 1×10^{-6} for the actor. We adopt a rule-based reward mechanism, which assigns a reward of 10 for a successful trajectory and 0 for a failed trajectory. To handle invalid actions generated by the agent, we impose an additional reward penalty of -0.5. For our method, the group size is set to 8, and 16 different tasks are sampled per training step, resulting in a total of $16 \times 8 = 128$ environments. The rollout temperature is set to 1.0 during the training phase, while the temperature is set to 0.4 during validation. The mini-batch size is 128, and the KL-divergence loss coefficient is set to 0.01. The total number of training iterations is set to 175.

A.3 Details of Search-Augmented QA

The search-augmented QA task covers the following seven benchmarks: NQ (Kwiatkowski et al., 2019), TriviaQA (Joshi et al., 2017), and PopQA (Mallen et al., 2022) for single-hop QA, and HotpotQA (Yang et al., 2018), 2Wiki (Ho et al., 2020), MuSiQue (Trivedi et al., 2022), and Bamboogle (Press et al., 2023) for multi-hop QA. Training is performed only on NQ and HotpotQA, while evaluation is conducted on all seven benchmarks, the remaining five of which are out of domain. The maximum prompt length is 8192 tokens, and the maximum response length is 512 tokens. Each trajectory allows up to 4 environment steps. For retrieval, we employ E5 (Wang et al., 2022) as the retriever. The learning rate is set to 1×10^{-6} for the actor. We adopt a rule-based reward mechanism, which assigns a reward of 1 for a successful trajectory and 0 for a failed trajectory. To handle invalid actions generated by the agent, we impose an additional reward penalty of -0.01. For our method, the group size is set to 5, and 256 different tasks are sampled per training step. The rollout temperature is set to 1.0 during the training phase, while the temperature is set to 0.4 during validation. The mini-batch size is 512, and the KL-divergence loss coefficient is set to 0.001. The total number of training iterations is set to 200.

A.4 Details of Baseline Methods

On ALFWorld and WebShop, we compare our method with closed-source LLMs, including GPT-4o (Hurst et al., 2024) and Gemini-2.5-Pro (Comanici et al., 2025); the backbone model Qwen2.5-

7B-Instruct (Yang et al., 2025), which serves as the base model reference without additional agentic training; prompt-based or memory-based methods, including ReAct (Yao et al., 2023), Reflexion (Shinn et al., 2023), Mem0 (Chhikara et al., 2025), ExpeL (Zhao et al., 2024), MemP (Fang et al., 2025), and SimpleMem (Liu et al., 2026); RL-based methods, including RLOO (Ahmadian et al., 2024) and GRPO (Shao et al., 2024); and memory-augmented or skill-augmented RL methods, including MemRL (Zhang et al., 2026b), EvolveR (Wu et al., 2025), Mem0+GRPO, SimpleMem+GRPO, SkillRL (Xia et al., 2026), Skill0 (Lu et al., 2026b), D2Skill (Tu et al., 2026b), and SkillC (Lin et al., 2026). For search-augmented QA, we compare our method with the backbone model Qwen2.5-7B-Instruct, Chain-of-Thought (CoT) (Wei et al., 2022), Retrieval-Augmented Generation (RAG) (Lewis et al., 2020), Search-o1 (Li et al., 2025), R1-Instruct (Jin et al., 2025), Search-R1 (Jin et al., 2025), ZeroSearch (Sun et al., 2025), StepSearch (Zheng et al., 2025), EvolveR, SkillRL, and Skill0. All baseline results are taken from the SkillRL paper (Xia et al., 2026), except those of Skill0, D2Skill, and SkillC, which are taken from their original papers.

A.5 Implementation Details

When computing the step-level skill influence score in Eq. (3), the token-averaged log-probability difference is clipped to $[-5, 5]$. Since GRPO samples G rollouts for the same task instance, the collected trajectories are deduplicated at the instance level: each instance keeps the successful rollout with the shortest episode and the failed rollout with the highest trajectory-level mean influence. Our code is built upon the verl (Sheng et al., 2025) and verl-agent (Feng et al., 2025) frameworks, which are also used by SkillRL, D2Skill, and Skill0. All experiments are conducted on four NVIDIA H200 GPUs, each with 140 GB of memory.

A.6 Evaluation Protocol and Metrics

For ALFWorld, we report the success rate, i.e., the percentage of evaluation episodes in which the task is completed. For WebShop, we report the task score (the environment reward in $[0, 1]$ scaled to 100) and the success rate (the percentage of episodes that achieve the full reward). For search-augmented QA, we report the success rate, i.e., the percentage of evaluation episodes in which the final answer exactly matches one of the reference answers after normalization, following the standard exact match (EM) criterion. We report SIAR results using the checkpoint from the final training iteration.

B Further Analysis

B.1 Statistical Analysis

Table 4 reports the mean and standard deviation of ALFWorld success rates over three evaluation seeds, using a fixed checkpoint for each method. We compare SIAR with GiGPO, a step-level credit assignment method, and SkillRL, a skill-augmented RL baseline. For GiGPO and SkillRL, we use the checkpoints publicly released by their authors, and for SIAR, we use the checkpoint from the final training iteration. We use pairwise t-tests at a significance level of 0.05. SIAR achieves an overall success rate of 93.2%, with a standard deviation of 0.4 percentage points, compared with 90.6% for GiGPO and 83.1% for SkillRL. Across the six task categories and two baselines, SIAR achieves statistically superior performance in 7 of the 12 comparisons. Specifically, SIAR significantly outperforms GiGPO on Pick and Heat, and SkillRL on five of the six task categories. SIAR also significantly outperforms both baselines in overall success rate, demonstrating a consistent advantage over a skill-augmented RL method and a step-level credit assignment method.

B.2 Validation of the Attribution Signal

We first examine whether the steps receiving the largest positive additional advantages in successful trajectories help complete the task. We select five task instances from each of the six ALFWorld task categories and generate eight trajectories per instance, for a total of 240 trajectories. For each task instance, we compute the group-level baseline $\bar{\Delta}$ using all steps in its eight trajectories. For each successful trajectory, we select the M steps with the highest influence scores among those satisfying $\Delta_t > \bar{\Delta}$, where $M \in \{2, 5\}$. If fewer than M steps satisfy this condition, we retain all eligible steps. Random indicates that we select the same number of steps at random from the same trajectory. To

Table 4: Mean success rates (%) and standard deviations over three evaluation runs on ALFWorld. •/○ indicates whether the success rates of SIAR are statistically superior/inferior to the compared algorithm according to the pairwise t-test at 0.05 significance level.

Method	Pick	Look	Clean	Heat	Cool	Pick2	**All**
GiGPO	$96.9_{(0.0)}$ •	$93.9_{(8.6)}$	$94.5_{(0.1)}$	$86.8_{(2.5)}$ •	$79.0_{(1.6)}$	$92.8_{(2.0)}$	$90.6_{(0.6)}$ •
SkillRL	$86.5_{(1.5)}$ •	$69.7_{(4.3)}$ •	$85.7_{(2.7)}$ •	$75.7_{(9.1)}$ •	$92.2_{(0.1)}$ ○	$78.3_{(3.5)}$ •	$83.1_{(1.0)}$ •
SIAR	$100.0_{(0.0)}$	$90.6_{(0.4)}$	$95.4_{(0.4)}$	$96.4_{(2.6)}$	$83.6_{(2.0)}$	$90.9_{(3.7)}$	$93.2_{(0.4)}$

Table 5: Percentage of selected steps judged useful by GPT-5.4-mini in successful ALFWorld trajectories.

M	SIAR (%)	Random (%)	Difference
2	84.32	72.50	+11.82
5	79.50	72.28	+7.22

Table 6: Percentage of selected steps judged unproductive by GPT-5.4-mini in failed ALFWorld trajectories.

M	SIAR (%)	Random (%)	Difference
2	83.53	71.47	+12.06
5	78.71	73.29	+5.42

assess each selected step, we provide GPT-5.4-mini (OpenAI, 2026b) with the complete interaction trajectory and the current step, and ask it to judge whether the step is useful or unproductive for task completion. We then compare the percentage of useful steps among those selected by SIAR and Random.

Table 5 shows that SIAR has a higher percentage of steps judged useful than Random for both values of M, with improvements of 11.82 and 7.22 percentage points for $M = 2$ and 5, respectively. At $M = 2$, 84.32% of the steps selected by SIAR are judged useful, compared with 72.50% for Random. Within each successful trajectory, these selected steps correspond to the largest positive additional advantages. These results provide quantitative evidence that actions receiving the most additional credit are more often judged useful than randomly selected actions. This supports using skill influence scores to prioritize additional credit for actions that advance the task.

Next, we examine the steps receiving the largest additional penalties in failed trajectories. A trajectory may fail when the retrieved skill does not fit the current task instance. In this case, actions strongly influenced by the skill may not contribute to task progress. Following the setting of Table 5, we select the M steps with the highest influence scores among those satisfying $\Delta_t > \bar{\Delta}$ in each failed trajectory. Table 6 shows that SIAR has a higher percentage of steps judged unproductive than Random for both values of M, with differences of 12.06 and 5.42 percentage points for $M = 2$ and 5, respectively. These results indicate that the steps receiving the largest additional penalties in failed trajectories are more often judged unproductive than randomly selected steps. This pattern is consistent with the situation in which the retrieved skill does not fit the current task instance. In this case, SIAR assigns the largest additional penalties to the actions most strongly influenced by the skill. The additional penalty thus reduces the policy's reliance on skill guidance that is associated with task failure.

B.3 Computation Time Analysis

Compared with skill-augmented GRPO training, the only additional per-step computation introduced by SIAR is recomputing the log probabilities of the executed actions under the skill-free context in Eq. (3). Table 7 reports the average computation time of this recomputation measured on ALFWorld, together with the skill-conditioned log-probability computation that GRPO already performs and the total time of a training step. Since the sampled actions are kept fixed, the recomputation is a single gradient-free forward pass, and the skill-free context is substantially shorter than the skill-conditioned one.

Table 7: Average per-step training time on ALFWorld. The skill-free log-probability recomputation and skill updating are the additional computations introduced by SIAR.

Computation	Time (s)	% of Total
Skill-conditioned log-prob.	35.4	7.1%
Skill-free log-prob. (Eq. (3))	14.4	2.9%
Skill updating	10.4	2.1%
Total training step	499.4	100%

As a result, it takes only about 41% of the time of the skill-conditioned computation, and accounts for merely 2.9% of the total per-step training time. Beyond this per-step cost, attribution-guided skill updating rewrites skills only a few times throughout training (see Figure 3); when its total time is amortized over all training steps, it accounts for only 2.1% of the per-step training time. Hence, the additional time cost introduced by SIAR is marginal.

B.4 Effect of the Auxiliary LLM

To examine the effect of the auxiliary model, we replace GPT-5.4 with o3 (OpenAI, 2025) for skill updating. Table 8 shows that SIAR achieves success rates of 93.0% with o3 and 93.4% with GPT-5.4 on ALFWorld, a difference of only 0.4 percentage points. These results show similar performance with the two auxiliary models. This is likely because the auxiliary LLM only rewrites skills according to the update decisions and trajectories provided by SIAR, rather than determining when skills are updated or how actions are credited. Instead, the performance gains are primarily driven by the attribution-guided skill updating and attribution-calibrated advantage assignment, which together improve the success rate on ALFWorld from 83.6% to 93.4% according to Table 3. Moreover, when both methods use o3 for skill updating, SIAR still outperforms SkillRL by 3.1 percentage points (93.0% vs. 89.9%), providing further evidence for the effectiveness of SIAR.

Table 8: ALFWorld success rates (%) with different auxiliary LLMs.

Method	**All**
SkillRL-o3	89.9
SIAR-o3	93.0
SIAR-GPT-5.4	93.4

C Details of Skills

C.1 Skill Template

As illustrated in Figure 5, each task skill is stored in a structured format. It begins with a task identifier, followed by three sections: a *Procedure* that lays out the ordered steps for solving the task, a set of *Key Patterns* that summarize recurring strategies, and a list of *Common Pitfalls* that records frequent failure modes. In our experiments, each task category is associated with a single skill shared by all tasks in that category. The corresponding skill is retrieved according to the task type and used throughout each trajectory. Accordingly, ALFWorld is equipped with six skills, one for each of its task categories (Pick, Look, Clean, Heat, Cool, and Pick2); WebShop with five skills, one for each shopping category (Beauty, Electronics, Fashion, Garden, and Grocery); and search-augmented QA with only two skills, one for single-hop QA and one for multi-hop QA. During training and validation, the agent is conditioned on the skill associated with its task category. Skill updates occur only during training and use only trajectories collected for the corresponding task category. During evaluation, skills remain fixed, and the auxiliary LLM is not invoked.

C.2 Skill Initialization

Before training, each task category is initialized with one skill generated by GPT-5.4 (OpenAI, 2026a). For each category, the generator receives five successful trajectories collected by an agent trained with GiGPO (Feng et al., 2025) and five failed trajectories collected by the backbone model Qwen2.5-7B-Instruct (Yang et al., 2025), and produces the skill in the structured format of Figure 5. All skill-based configurations in our experiments start from the same initial skills.

C.3 Updating Rules

The skill of each task category is checked periodically during training and rewritten only when the update condition in Eq. (6) is triggered. We employ GPT-5.4 (OpenAI, 2026a) as the skill updater, which is provided with the current skill together with both successful and failed trajectories and is prompted to update the skill in an evidence-grounded manner, where every modification must be supported by a specific trajectory step. Each existing item is assigned exactly one operation: *keep* it unchanged when successful trajectories still follow it and no failure contradicts it; *revise* it when failed trajectories reveal that its guidance is misleading or incomplete; *merge* several overlapping items into a single clearer rule; or *drop* it when its premise is invalid or it is never relied upon. In addition, a

Skill Template

TASK: {task_name}
Procedure
1. {An ordered step that grounds the goal and drives the agent toward task completion.}
2. · · · (the remaining ordered steps that outline the full task-solving workflow are omitted.)

Key Patterns
1. {pattern name}: a recurring strategy distilled from successful trajectories.
2. · · · (the remaining generalizable patterns are omitted.)

Common Pitfalls
1. {pitfall}: a frequent failure mode together with its corrective action.
2. · · · (the remaining pitfalls to avoid are omitted.)

Figure 5: The template of a task skill. Each skill is represented in a structured format consisting of a *Procedure*, a set of *Key Patterns*, and a list of *Common Pitfalls*.

Skill Updating Prompt

 # Role
You are revising a strategic skill file for an ALFWorld text-game agent on a specific task type. The skill file is organised into three sections — Procedure, Key Patterns, and Common Pitfalls — each a numbered list. You will be given the current file plus recent rollout trajectories. The failed trajectories shown are the most typical errors made under the current skill: cases where the current guidance led the agent astray.
The task is to (1) diagnose, item by item, what should happen to each old item, and (2) rewrite the full skill file such that items confirmed by the trajectories are preserved, items contradicted by the trajectories are rewritten or dropped, and gaps revealed by the trajectories are filled with new items.

 # Inputs
TASK NAME: {task_name}
CURRENT SKILL FILE ({task_name}.md): {current_content}
SUCCESSFUL TRAJECTORIES ({num_success}): {succ_block}
FAILED TRAJECTORIES — most typical errors under the current skill ({num_failure}): {fail_block}

 # Writing Rules
1. STRATEGIC, NOT SYNTACTIC. *(details omitted)*
2. NON-OBVIOUS ONLY. *(details omitted)*
3. POSITIVE PRESCRIPTION. *(details omitted)*
4. EVIDENCE-GROUNDED. *(details omitted)*
5. AVAILABLE OPERATIONS.
   — `keep:N` *(details omitted)*
   — `revise:N` *(details omitted)*
   — `merge:[N,M,...]` *(details omitted)*
   — drop (via `dropped_old_idxs`) *(details omitted)*
   — `new` *(details omitted)*
6. FLEXIBLE SECTION SHAPE. *(details omitted)*

 # Output Format
Return TWO fenced blocks in this exact order. No prose between or around them.
Block 1 — Diagnosis (fenced as `json`): *(contents omitted)*
Block 2 — Revised Skill File (fenced as `markdown`): *(contents omitted)*
Begin your response with the diagnosis block.

Figure 6: A condensed version of the skill updating prompt of ALFWorld.

Algorithm 1 Training procedure of SIAR

Require: Policy π_θ, task skills $\{s_q\}$, group size G, checking period K, minimum trajectory numbers N^+ and N^-, weight ω

for each training iteration g **do**
  Sample a task x of category q and retrieve its skill s_q.
  // Skill-conditioned rollout
  for $i = 1$ to G **do**
    for $t = 1$ to T **do**
      Sample action $a_t^i \sim \pi_\theta(\cdot \mid x, o_t^i, s_q)$.
      Execute the action and observe the next observation o_{t+1}^i and reward r_t^i.
    end for
  end for
  // Compute log probability with skill
  Compute $\log \pi_\theta(a_{t,\ell}^i \mid x, o_t^i, s_q, a_{t,<\ell}^i)$ for each token $a_{t,\ell}^i$.
  // Compute log probability without skill
  Compute $\log \pi_\theta(a_{t,\ell}^i \mid x, o_t^i, a_{t,<\ell}^i)$ for each token $a_{t,\ell}^i$.
  // Compute skill influence and influence-reward consistency
  Compute $\Delta_t^i(s_q)$ and $\kappa_t^i(s_q)$ for each action according to Eq. (3) and Eq. (4).
  Compute trajectory-level advantages according to Eq. (1).
  Compute attribution-calibrated advantages according to Eq. (7).
  Update policy θ by maximizing Eq. (9).
  if $\mathrm{Update}(s_q) = 1$ according to Eq. (6) **then**
    Rewrite the skill s_q with the auxiliary LLM.
  end if
end for

new item is added only as a last resort, when a failure mode recurs across multiple trajectories and no existing item addresses it. A condensed version of the updating prompt is shown in Figure 6.

D Prompt Template

The prompts we use for LLM agents on the ALFWorld, WebShop, and search-augmented QA tasks are presented in Figure 7, Figure 8, and Figure 9. Each prompt is assembled at every step from a fixed template whose placeholders are replaced with their current values before the policy is queried. These placeholders fall into two groups. The first group carries the interaction state, such as the goal {task_description}, the step counter {step_count}, the latest observation {current_observation}, and the recent action–observation history {action_history}. We maintain the two most recent steps for ALFWorld and WebShop and the entire history for search-augmented QA, where the past search queries and retrieved evidence are carried by the placeholder {memory_context}. The second group consists of a single placeholder {skill_context}. At each step, we populate this placeholder with the skill retrieved for the current task category.

Following the CoT paradigm (Wei et al., 2022), the agent first reasons within the <think> </think> tags before producing its response. The <action> </action> block indicates the agent's final action decision. The search agent submits search queries within <search> </search> and provides answers within <answer> </answer>. Evidence returned by the retriever for search-augmented QA is presented in <information> </information> tags.

E Pseudocode

Algorithm 1 summarizes the overall training procedure of SIAR. Each training iteration performs skill-conditioned rollouts, computes the influence score $\Delta_t^i(s)$ and the consistency score $\kappa_t^i(s)$ from the paired log probabilities, and uses them to calibrate the advantages for the policy update and to accumulate the evidence for skill updating.

Prompt Template of ALFWorld

You are an expert agent operating in the ALFRED embodied Environment.
Your task is to: {task_description}.
The following skills and guidelines may help you complete the task:{skill_context}.
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 7: The prompt template of ALFWorld.

Prompt Template of WebShop

You are an expert autonomous agent operating in the WebShop e-commerce environment.
Your task is to: {task_description}.
The following skills and guidelines may help you complete the task:{skill_context}.
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 8: The prompt template of WebShop.

Prompt Template of Search-Augmented QA

You are an expert agent tasked with answering the given question step-by-step.
Your question: {task_description}.
The following skills and guidelines may help you answer the question: {skill_context}
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 9: The prompt template of search-augmented QA.

Successful Trajectory: Heat & Place (Potato)

Task: -= Welcome to TextWorld, ALFRED! =- You are in the middle of a room. Looking quickly around you, you see a cabinet 9, a cabinet 8, a cabinet 7, a cabinet 6, a cabinet 5, a cabinet 4, a cabinet 3, a cabinet 2, a cabinet 1, a coffeemachine 1, a countertop 2, a countertop 1, a drawer 13, a drawer 12, a drawer 11, a drawer 10, a drawer 9, a drawer 8, a drawer 7, a drawer 6, a drawer 5, a drawer 4, a drawer 3, a drawer 2, a drawer 1, a fridge 1, a garbagecan 1, a microwave 1, a sinkbasin 1, a stoveburner 6, a stoveburner 5, a stoveburner 4, a stoveburner 3, a stoveburner 2, a stoveburner 1, and a toaster 1. Your task is to: **put a hot potato in fridge.**

Result: success (R=10) **Group-level average influence** $\bar{\Delta} = 1.48$

The two steps with the highest per-step skill influence Δ_t (Eq. (3)) are shown in temporal order, with zero-based step indices.

Step 2 ★ highest $\Delta_t = 3.25$ $\Delta_t - \bar{\Delta} = +1.78$

Observation: You pick up the potato 1 from the countertop 1.
Action: `go to microwave 1`
Next Observation: You arrive at microwave 1. The microwave 1 is closed.

Step 4 $\Delta_t = 2.57$ $\Delta_t - \bar{\Delta} = +1.09$

Observation: You open the microwave 1. The microwave 1 is open. In it, you see a bowl 1.
Action: `heat potato 1 with microwave 1`
Next Observation: You heat the potato 1 using the microwave 1.

Figure 10: Illustration of the two steps with the highest skill influence scores (Δ_t) in a successful *Heat & Place (Potato)* trajectory.

Successful Trajectory: Cool & Place (Potato)

Task: -= Welcome to TextWorld, ALFRED! =- You are in the middle of a room. Looking quickly around you, you see a cabinet 9, a cabinet 8, a cabinet 7, a cabinet 6, a cabinet 5, a cabinet 4, a cabinet 3, a cabinet 2, a cabinet 1, a coffeemachine 1, a countertop 3, a countertop 2, a countertop 1, a drawer 9, a drawer 8, a drawer 7, a drawer 6, a drawer 5, a drawer 4, a drawer 3, a drawer 2, a drawer 1, a fridge 1, a garbagecan 1, a microwave 1, a shelf 3, a shelf 2, a shelf 1, a sinkbasin 1, a stoveburner 4, a stoveburner 3, a stoveburner 2, a stoveburner 1, and a toaster 1. Your task is to: **cool some potato and put it in garbagecan.**

Result: success (R=10) **Group-level average influence** $\bar{\Delta} = 1.01$

The two steps with the highest per-step skill influence Δ_t (Eq. (3)) are shown in temporal order, with zero-based step indices.

Step 3 ★ highest $\Delta_t = 1.86$ $\Delta_t - \bar{\Delta} = +0.86$

Observation: You arrive at countertop 3. On the countertop 3, you see a apple 3, a book 1, a bread 2, a egg 1, a kettle 1, a lettuce 2, a potato 1, a spoon 1, a vase 3, and a vase 2.
Action: `take potato 1 from countertop 3`
Next Observation: You pick up the potato 1 from the countertop 3.

Step 6 $\Delta_t = 1.73$ $\Delta_t - \bar{\Delta} = +0.72$

Observation: You open the fridge 1. The fridge 1 is open. In it, you see a apple 2, a apple 1, a cup 1, and a tomato 1.
Action: `cool potato 1 with fridge 1`
Next Observation: You cool the potato 1 using the fridge 1.

Figure 11: Illustration of the two steps with the highest skill influence scores (Δ_t) in a successful *Cool & Place (Potato)* trajectory.

Failed Trajectory: Cool & Place (Pot)

Task: -= Welcome to TextWorld, ALFRED! =- You are in the middle of a room. Looking quickly around you, you see a cabinet 10, a cabinet 9, a cabinet 8, a cabinet 7, a cabinet 6, a cabinet 5, a cabinet 4, a cabinet 3, a cabinet 2, a cabinet 1, a coffeemachine 1, a countertop 1, a diningtable 1, a drawer 2, a drawer 1, a fridge 1, a garbagecan 1, a microwave 1, a sinkbasin 1, a stoveburner 4, a stoveburner 3, a stoveburner 2, a stoveburner 1, and a toaster 1. Your task is to: **cool some pot and put it in diningtable.**

Result: failure (R=0) **Group-level average influence** $\bar{\Delta} = 0.34$

The two steps with the highest per-step skill influence Δ_t (Eq. (3)) are shown in temporal order, with zero-based step indices.

Step 19 ★ highest $\Delta_t = 1.99$ $\Delta_t - \bar{\Delta} = +1.65$

Observation: On the stoveburner 2, you see a pot 1.
Action: `examine stoveburner 2`
Next Observation: On the stoveburner 2, you see a pot 1.

Step 38 $\Delta_t = 1.36$ $\Delta_t - \bar{\Delta} = +1.02$

Observation: On the stoveburner 2, you see a pot 1.
Action: `examine stoveburner 2`
Next Observation: On the stoveburner 2, you see a pot 1.

Figure 12: Illustration of the two steps with the highest skill influence scores (Δ_t) in a failed *Cool & Place (Pot)* trajectory.

Failed Trajectory: Pick Two & Place (Candle)

Task: -= Welcome to TextWorld, ALFRED! =- You are in the middle of a room. Looking quickly around you, you see a bathtubbasin 1, a garbagecan 1, a handtowelholder 1, a shelf 3, a shelf 2, a shelf 1, a sinkbasin 1, a toilet 1, a toiletpaperhanger 1, and a towelholder 1. Your task is to: **put two candle in shelf.**

Result: failure (R=0) **Group-level average influence** $\bar{\Delta} = 0.31$

The two steps with the highest per-step skill influence Δ_t (Eq. (3)) are shown in temporal order, with zero-based step indices.

Step 4 ★ highest $\Delta_t = 1.20$ $\Delta_t - \bar{\Delta} = +0.88$

Observation: You arrive at shelf 2. On the shelf 2, you see a soapbar 1, and a tissuebox 1.
Action: `examine shelf 1`
Next Observation: **Nothing happens.**

Step 31 $\Delta_t = 1.19$ $\Delta_t - \bar{\Delta} = +0.87$

Observation: On the shelf 1, you see a cloth 1, a soapbottle 1, a spraybottle 2, and a spraybottle 1.
Action: `examine shelf 2`
Next Observation: **Nothing happens.**

Figure 13: Illustration of the two steps with the highest skill influence scores (Δ_t) in a failed *Pick Two & Place (Candle)* trajectory.

F CASE STUDY

To complement the qualitative analysis in Section 5.3, Figures 10 and 11 show two successful ALFWorld trajectories together with their step-level skill influence scores. For each trajectory, we display the two highest-influence steps. Each step is annotated with its influence score Δ_t and its deviation $\Delta_t - \bar{\Delta}$ from the group mean. Both steps exceed the baseline and receive positive additional advantages according to $A^S = \text{sign}(R)\max(0, \Delta_t - \bar{\Delta})$. In the Heat & Place example, the agent navigates to the microwave while holding a potato (step 2) and heats it (step 4), preparing it for subsequent placement in the fridge. In the Cool & Place example, the agent picks up the potato from the countertop (step 3) and cools it in the fridge (step 6) before later placing it in the garbage can. For these successful trajectories, the largest positive additional advantages are therefore assigned to actions that directly support task completion.

Figures 12 and 13 show two failed trajectories. In the first case, the agent repeatedly examines `stoveburner 2` after discovering `pot 1`, without picking up the pot. In the second case, a Pick Two & Place task, the agent examines `shelf 1` while at `shelf 2` (step 4) and `shelf 2` while at `shelf 1` (step 31), and both actions return "Nothing happens.". The highlighted steps exceed the group mean, but their trajectories fail ($R = 0$). With $\text{sign}(R) = -1$, these steps receive negative additional advantages. The additional penalties discourage the repeated, unproductive behavior observed in these trajectories. Together, these cases illustrate the intended role of advantage calibration: under ideal conditions, it reinforces skill-guided behavior that supports task completion and discourages behavior that makes no progress. The highlighted steps also contribute positive consistency scores in the successful trajectories and negative consistency scores in the failed trajectories, providing evidence for skill updating.

G LIMITATIONS AND FUTURE WORK

While SIAR achieves strong performance across three agentic tasks, several limitations warrant discussion and suggest directions for future work. First, SIAR organizes skills at the granularity of predefined task categories, with each category associated with a single skill that is retrieved according to the task type. While this design is adequate for benchmarks with well-defined task types, open-ended settings require skills to be automatically discovered, indexed, and retrieved, and multiple skills may jointly influence a single trajectory. Extending the attribution mechanism to such dynamic skill libraries, in particular to settings where several skills are active simultaneously, is an important direction for future work. Second, SIAR uses the binary trajectory outcome to determine the direction of the additional advantage, with all steps in a trajectory sharing the same success or failure label. Since the true effectiveness of each individual step is difficult to verify without external assistance, such as evaluation by an LLM judge, this trajectory-level signal is inherently an approximation. Although SIAR aggregates attribution evidence across steps and trajectories, constructing finer-grained step-level outcome signals, such as process-level verification or intermediate rewards, remains a promising direction for future work.