SAPO: SKILL-AUGMENTED POLICY OPTIMIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Skill-augmented agents leverage reusable procedural knowledge to improve performance on complex agentic tasks. Despite promising progress, existing methods often rely on strong language models to analyze past experience, generate skills, and directly insert them into a retrievable skill bank. However, we find that even skills generated by proprietary frontier LLMs exhibit highly mixed utility, with many having little or even negative effect on performance. Although prior work tracks skill utility from later rollouts where the skill is retrieved, this feedback is delayed and entangled: once inserted, low-quality skills mislead the agent and slow down policy learning before their negative effects are recognized; meanwhile, because multiple skills are usually retrieved together, the observed performance change reflects the aggregate effect of the skill set rather than the marginal contribution of any individual skill. To address these limitations, we propose Skill-Augmented Policy Optimization (SAPO), a novel online reinforcement learning framework that enforces a validate-before-store paradigm for skill curation. Given a task, SAPO splits the standard rollout budget into base and skill-augmented halves: it generates base rollouts with the currently retrieved skills, induces a candidate skill from these trajectories, and evaluates it through skill-augmented rollouts under the same retrieval context. The performance gap between the two halves estimates the candidate skill's marginal contribution beyond existing skills, enabling validation before long-term storage without additional rollout overhead. This utility signal further trains the policy itself as a stronger skill generator, whose skill-generation likelihood serves as a skill score for retrieval-time reranking and outdated-skill pruning. Extensive experiments show that SAPO outperforms prior skill-augmented RL methods and avoids costly API calls to proprietary models.

1 INTRODUCTION

Agentic large language models (LLMs) (Zhou et al., 2026a;b; Yu et al., 2026; Xu et al., 2026; Buyya et al., 2026; Zhang et al., 2025b; Plaat et al., 2025) are increasingly applied to complex tasks reasoning (Wei et al., 2026; Chen et al., 2026; Hao et al., 2026; Feng et al., 2026; Li et al., 2026b; Wu et al., 2025a; Zhao et al., 2025), tool use (Xu & Yan, 2026; Jiang et al., 2026; Wang et al., 2026a; Xia et al., 2026; Yang et al., 2026b; Hu et al., 2026; Lin et al., 2026), or interaction with external environments. To improve agents in such settings, recent work has introduced *skills*: reusable procedural knowledge that equips LLMs with domain-specific capabilities and guides their behavior across related tasks (Jiang et al., 2026; Xu & Yan, 2026; Zhang et al., 2026a). Skills typically encode reusable procedures, such as action patterns or decision rules, and can be represented in different forms, including natural-language guidance, executable programs, or memory units (Zhang et al., 2026a; Huang et al., 2025b; Zheng et al., 2025; Yang et al., 2025b; Zhou et al., 2025b; Fang et al., 2025; Cao et al., 2025). They may be manually designed or automatically induced from agent experience, and can be retrieved to condition future decisions on useful procedural knowledge.

Recent studies have explored different ways to use skills for agent improvement. One line of work keeps the base model fixed and improves external skill memory or task-specific context. For example, MCE (Ye et al., 2026) uses a stronger model to refine skills from trajectory-level feedback. Another line of work studies skill evolution together with policy optimization. SkillRL (Xia et al., 2026) updates a retrievable skill bank during online RL. Skill0 (Lu et al., 2026) focuses on skill internalization, progressively transferring skill knowledge into the policy so that the agent can rely less on runtime skill retrieval. D2Skill (Tu et al., 2026) maintains skills at multiple granularities and updates them with utility signals from rollout outcomes.

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.

Despite this progress, we identify a critical issue in existing skill-augmented RL methods: *Newly generated skills are often inserted into the skill bank before their **usefulness** is explicitly validated.* In particular, many existing methods (Xia et al., 2026; Lu et al., 2026; Tu et al., 2026) rely on proprietary frontier LLMs to analyze trajectories and generate skills, implicitly assuming that skills produced by stronger models are reliable. However, our preliminary experiments in Sec. 3 show that even skills generated by GPT-5.4 exhibit highly mixed utility. While a small subset improves performance, many provide little benefit or even mislead the agent, and the average utility stays close to zero throughout training. This is especially concerning because costly frontier-LLM API calls still produce many skills that fail to help or even interfere with policy learning. Although some prior methods (Tu et al., 2026; Zhou et al., 2026b) attempt to track skill utility, their feedback is delayed. For example, D2Skill (Tu et al., 2026) estimates skill usefulness from later rollouts after a skill has already been stored and retrieved. By the time a low-quality skill is identified, it has already misled the agent and slowed down policy learning.

Furthermore, skill usefulness changes as the policy evolves. As the policy becomes stronger during online RL, some previously useful skills may become outdated or redundant. Keeping such skills in the bank can introduce retrieval noise, because they may occupy slots that could otherwise be assigned to more useful skills. Effective skill maintenance therefore requires identifying skills whose value has decreased over training. A natural way to measure the current value of a skill is to compare rollouts with and without that skill, but doing so naively would require additional counterfactual rollouts. Prior work on memory or skill valuation (Tu et al., 2026; Zhou et al., 2026b) has explored learned scoring functions or delayed utility estimates, but these signals often fail to isolate the marginal contribution of an individual skill. In retrieval-based skill augmentation, skills are usually used jointly, so the observed performance reflects the aggregate effect of multiple co-retrieved skills rather than the value of any single skill. This matters because an ideal skill-utility measure should be context-dependent. A skill that is useful in isolation may become redundant when similar skills are already retrieved, while another skill may be valuable because it complements the current retrieval context. Thus, both skill maintenance and retrieval-time selection require estimating whether a skill provides marginal benefit beyond the skills already in the bank or already selected for the current query.

These observations reveal three key challenges for skill-augmented online RL: **(1)** how to estimate the marginal utility of newly generated skills without incurring additional rollout cost; **(2)** how to score a skill by its marginal contribution beyond the currently retrieved skill context, rather than by its standalone relevance; and **(3)** how to reduce reliance on costly proprietary LLM APIs for skill generation without sacrificing skill quality. To tackle these challenges, we propose **S**kill-**A**ugmented **P**olicy **O**ptimization (SAPO), a novel online RL framework that validates skills before storage and uses their context-dependent utility to improve skill generation, maintenance, and retrieval.

SAPO first splits the standard rollout budget into two matched groups for prospective skill validation. During training, SAPO retrieves existing skills for a task and uses the first part of the rollout budget to generate base rollouts. These rollouts show how the current policy behaves under the existing skill context and provide evidence for inducing a candidate skill. SAPO then uses the remaining rollout budget to generate skill-augmented rollouts under the same task and retrieved skill context, with only the candidate skill added. Because the two rollout groups share the same task and retrieved skill context and differ only in the presence of the candidate skill, their reward gap estimates the candidate's marginal contribution beyond the retrieved skills, without requiring additional rollouts. SAPO uses this context-dependent marginal utility to curate the skill bank before storage: useful and non-redundant skills are promoted into long-term memory, while low-utility skills are discarded before they can affect future retrieval and policy learning. The same utility signal further trains the policy itself as a stronger skill generator. This avoids repeated dependence on proprietary frontier LLMs for skill generation. Once trained with utility feedback, the policy's probability of generating a skill serves as a quality score. SAPO uses this context-dependent score for long-term maintenance and retrieval-time selection, pruning outdated skills and reranking candidate skills during retrieval.

We evaluate SAPO across diverse interactive decision-making and search-augmented question answering tasks. SAPO consistently outperforms prior skill-augmented RL methods while avoiding costly API calls to proprietary models. Further analyses show improved training dynamics, increasing skill utility, and the effectiveness of SAPO's major components. Our main contributions are: **(1)** we reveal the critical issue that prior methods overlook the mixed utility of generated skills, allowing low-quality skills to mislead agent learning; **(2)** we propose SAPO, a novel online RL framework

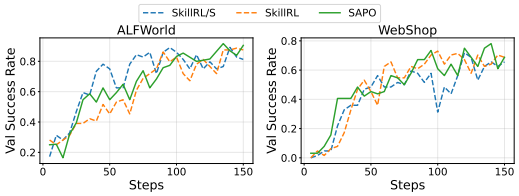


Figure 1: Validation success rate on ALFWorld and WebShop.

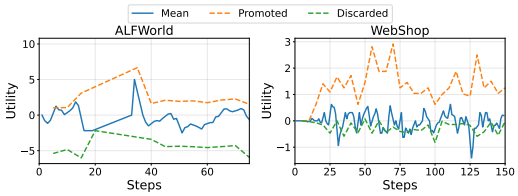


Figure 2: Utility of GPT-generated skills on ALFWorld and WebShop during training.

that provides skill utility estimation without additional overhead and trains the policy as both an agent and a skill generator; and **(3)** extensive experiments show that SAPO outperforms existing methods while avoiding costly API calls to proprietary models.

2 Related Work

Agent Skills. In autonomous agents, skills are reusable procedural knowledge for temporally extended, goal-directed behaviors beyond one-step action generation (Jiang et al., 2026; Xu & Yan, 2026). They encode action patterns, decision procedures, or tool-use workflows across related tasks (Zhang et al., 2026a; Wang et al., 2023), and may appear as natural-language instructions (Liu et al., 2024), executable programs (Wang et al., 2025c; Huang et al., 2025b), APIs (Zheng et al., 2025), trajectories (Yang et al., 2025b; Zhou et al., 2025b), or memory units (Fang et al., 2025; Cao et al., 2025). These forms connect high-level task intents to low-level actions, making skills an operational form of procedural memory (Wu & Zhang, 2026; Zhou et al., 2026a; Zhang et al., 2026d). Prior work obtains skills through manual design (Zhang et al., 2026a), demonstrations (Liu et al., 2024), repository mining (Bi et al., 2026), knowledge-base construction (Wang et al., 2026a; Shen et al., 2026), exploration (Yang et al., 2026c), or trajectory/feedback distillation (Ni et al., 2026; Xia et al., 2026). Recent efforts further extend skill repositories through external policy guidance (Wang et al., 2025a; Li et al., 2026c), parameter internalization (Lu et al., 2026), co-evolutionary verification (Zhang et al., 2026b), collective evolution (Ma et al., 2026), and lifelong self-evolution (Yang et al., 2026c). As repositories grow, the challenge shifts from skill generation to skill valuation (Li et al., 2026a; Zhong et al., 2026; Liu et al., 2026b; Wang et al., 2026b; Zhang et al., 2026e).

Agent Memory. Agent memory stores past experience in editable and retrievable forms for future use (Silver & Sutton, 2025; Du, 2026). Prior work studies memory at different levels: *case-based memory* stores raw trajectories, solutions, or examples for similar future tasks (Zhou et al., 2025a; Chen et al., 2025; Zhang et al., 2025e; 2026c; Fang et al., 2026); *strategy-based memory* summarizes interactions into reusable insights, workflows, or reasoning patterns (Ouyang et al., 2025; Huang et al., 2025a; Suzgun et al., 2026; Cai et al., 2025b; Xu et al., 2026); and *skill-based memory* stores callable knowledge such as code, functions, and APIs that map high-level plans to concrete actions (Zhang et al., 2025d; Zheng et al., 2025; Fang et al., 2025; Wang et al., 2025c; Han et al., 2025; Yang et al., 2026c; Ni et al., 2026). Beyond storing experience, recent work studies how memory should be organized and updated over time, including unified memory pipelines (Tang et al., 2025; Huang et al., 2025a; Zhang et al., 2025a; Wu et al., 2025b; Yu et al., 2026), graph-structured memory (Yang et al., 2026a), and memory management or evolution through consolidation, updating, forgetting, and RL-based construction (Zhang et al., 2025c; Zhai et al., 2025; Cai et al., 2025a; Yan et al., 2025; Khanda et al., 2026).

3 Preliminary

In this section, we present two preliminary findings that motivate our method design.

Experimental Setup. We follow the experimental setting of SkillRL (Xia et al., 2026) and report results on ALFWorld(Shridhar et al., 2020) and WebShop(Yao et al., 2022a). The base model used in all experiments is `Qwen2.5-7B-Instruct` (Yang et al., 2025a).

Skill evolution brings only marginal improvement over a no-evolution variant. We compare SkillRL with SkillRL/S, a no-evolution variant with a fixed skill bank. As shown in Fig. 1, SkillRL provides only limited gains on both ALFWORLD and WEBSHOP, and even reaches its best ALFWORLD performance more slowly than SkillRL/S. This suggests that naively evolving the skill bank does not necessarily improve learning efficiency.

Even GPT-generated skills are mixed, with near-zero average utility. To understand this limited gain, we estimate the marginal utility of GPT-generated skills during training. For each prompt, we sample G base rollouts, ask GPT-5.4 to induce a skill from the observed failures, and then sample another G rollouts with the generated skill added. The reward gap between the two groups is used as the skill's marginal utility, which we average across prompts at each training step. Every 5 training steps, we further split generated skills into *promoted* skills, defined as positive-utility skills in the top 20%, and *discarded* skills, containing the rest.

As shown in Fig. 2, the mean marginal utility of GPT-generated skills stays close to zero on both benchmarks, despite a clear gap between promoted and discarded skills. This indicates that generated skills are highly mixed: a small subset is useful, while many are ineffective or harmful. The limited improvement of SkillRL may therefore stem from admitting low-utility skills into the bank, where they can mislead retrieval and slow learning. Appendix D.1 shows a similar pattern with Claude-Opus-4.6 as the skill generator.

Based on the preliminary findings above and our analysis of existing work, we identify three limitations of existing skill-augmented agents in online reinforcement learning settings:

- **Direct Injection of Unvalidated Skills.** Existing methods often insert generated skills into the bank without validation, allowing low-quality or harmful skills to be stored and later retrieved. Although some methods (Zhou et al., 2026b; Tu et al., 2026) track skill utility after storage, this feedback is delayed: a low-quality skill has already misled the agent and harmed policy learning before being identified.
- **Lack of Context-Dependent Attribution for Individual Skills.** Ideally, skill utility should be context-dependent: a skill may be redundant when similar skills are retrieved, or useful when it complements the current context. However, existing methods either estimate skill utility (Tu et al., 2026) or train utility functions (Zhou et al., 2026b; Zhang et al., 2026c) from set-level signals, where the observed return reflects all retrieved skills jointly. Thus, they cannot determine an individual skill's marginal value in that context.
- **High API Cost of Skill Generation.** Existing methods (Xia et al., 2026; Tu et al., 2026; Lu et al., 2026) rely on proprietary frontier LLMs to generate skills during training, incurring high costs across repeated online RL updates, while many skills have negative utility. Appendix E analyzes API costs of existing methods.

4 SKILL-AUGMENTED POLICY OPTIMIZATION

As shown in Figure 3, we propose Skill-Augmented Policy Optimization (SAPO), an online reinforcement learning framework that jointly performs *skill induction*, *skill validation*, and *skill maintenance*. SAPO optimizes the agent policy with rewards from collected rollouts while estimating the marginal utility of newly induced skills online. This utility signal curates the skill bank and trains the same policy as a stronger skill generator, whose skill-generation likelihood is further reused as a skill score for maintenance and retrieval-time reranking. Next, we describe each component in detail.

4.1 ONLINE ROLLOUTS AND SKILL INDUCTION

SAPO operates on a *skill induction unit* $\mathcal{X}_i = \{x_{i,j}\}_{j=1}^{K}$, which contains either a single query ($K = 1$) or a small group of semantically similar queries ($K > 1$). Each induction unit produces one candidate skill $\hat{s}_i$. SAPO maintains two skill banks: a long-term bank $\mathcal{B}_{\text{long}}$ for validated skills and a temporary bank $\mathcal{B}_{\text{temp}}$ for candidates awaiting promotion. For each query $x_{i,j} \in \mathcal{X}_i$, SAPO first retrieves a set of relevant existing skills, $\mathcal{S}_{i,j} \leftarrow \text{Retr}(x_{i,j}, \mathcal{B}_{\text{long}} \cup \mathcal{B}_{\text{temp}})$, and allocates a

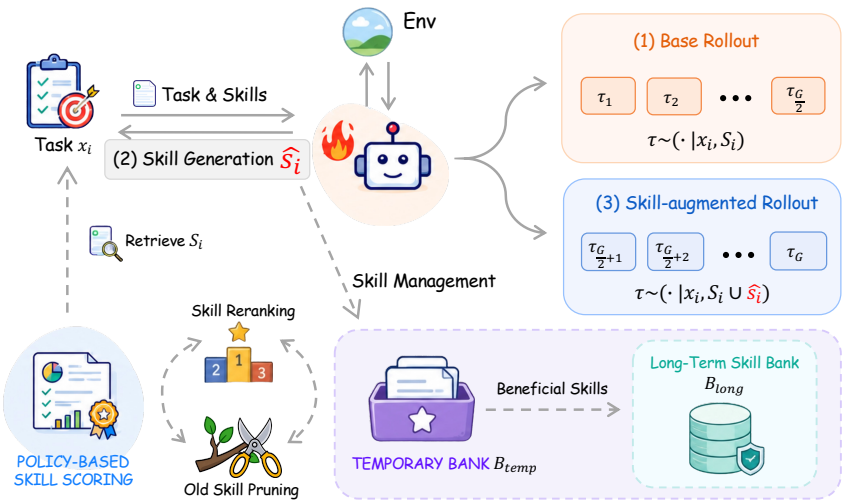


Figure 3: Framework of SAPO. SAPO first obtains base rollouts $\mathcal{Y}_{i,j}^{\text{base}}$, then generates a candidate skill $\hat{s}_i$, and validates it with skill-augmented rollouts $\mathcal{Y}_{i,j}^{\text{skill}}$.

rollout budget of G trajectories, split into two matched halves. The first half consists of *base rollouts* generated under the retrieved skill context alone:

$$\mathcal{Y}_{i,j}^{\text{base}} \sim \pi_\theta(\cdot \mid x_{i,j}, \mathcal{S}_{i,j}). \tag{1}$$

Based on the induction evidence $\{(x_{i,j}, \mathcal{S}_{i,j}, \mathcal{Y}_{i,j}^{\text{base}})\}_{j=1}^{K}$, the policy then induces a candidate skill

$$\hat{s}_i \sim \pi_\theta\big(\cdot \mid \mathcal{X}_i,\ \{\mathcal{S}_{i,j}\}_{j=1}^{K}, X_{\text{skill}},\ \{\mathcal{Y}_{i,j}^{\text{base}}\}_{j=1}^{K}\big), \tag{2}$$

where X_{skill} is a skill-generation prompt template (Appendix B). Intuitively, the base trajectories reveal both the policy's current capabilities under the retrieved context and the remaining failure patterns, enabling $\hat{s}_i$ to encode reusable guidance for future rollouts. However, as shown in Sec. 3, even GPT-generated skills are mixed in quality. SAPO therefore validates each candidate skill before storage by using the remaining rollout budget to generate *skill-augmented rollouts*:

$$\mathcal{Y}_{i,j}^{\text{skill}} \sim \pi_\theta(\cdot \mid x_{i,j}, \mathcal{S}_{i,j} \cup \{\hat{s}_i\}), \tag{3}$$

so that $(\mathcal{Y}_{i,j}^{\text{base}}, \mathcal{Y}_{i,j}^{\text{skill}})$ forms a matched rollout pair under the same query and retrieved context, differing only in whether the induced skill is included. The collected base and skill-augmented rollouts are used to optimize the agent policy with GRPO (Shao et al., 2024).

4.2 Skill Utility Estimation and Bank Update

Given the base rollouts $\mathcal{Y}_{i,j}^{\text{base}}$ and the skill-augmented rollouts $\mathcal{Y}_{i,j}^{\text{skill}}$, SAPO estimates the marginal utility of the candidate skill $\hat{s}_i$ by comparing the rewards of the two rollout sets.

Intra-prompt Utility. For a candidate skill $\hat{s}_i$ and a source query $x_{i,j}$, we define its prompt-specific marginal utility as

$$\begin{aligned} u(x_{i,j}, \hat{s}_i) = &\frac{1}{|\mathcal{Y}_{i,j}^{\text{skill}}|} \sum_{y \in \mathcal{Y}_{i,j}^{\text{skill}}} r(x_{i,j}, y) \\ &- \frac{1}{|\mathcal{Y}_{i,j}^{\text{base}}|} \sum_{y \in \mathcal{Y}_{i,j}^{\text{base}}} r(x_{i,j}, y). \end{aligned} \tag{4}$$

where $r(x, y)$ denotes the task reward. This quantity estimates the conditional marginal effect of adding $\hat{s}_i$ under the query $x_{i,j}$ and the retrieved skill context $\mathcal{S}_{i,j}$.

Cross-prompt Utility. When $K > 1$, the candidate skill $\hat{s}_i$ is induced from an induction unit $\mathcal{X}_i = \{x_{i,j}\}_{j=1}^{K}$. We therefore aggregate its utilities over the queries in the same unit:

$$U_{\hat{s}_i} = \frac{1}{K} \sum_{j=1}^{K} u(x_{i,j}, \hat{s}_i). \tag{5}$$

This cross-prompt utility measures whether the shared skill generalizes across related queries, and reduces to $U_{\hat{s}_i} = u(x_{i,1}, \hat{s}_i)$ when $K = 1$. SAPO then stores $(\hat{s}_i, U_{\hat{s}_i})$ in $\mathcal{B}_{\text{temp}}$ for later promotion.

Skill Promotion and Bank Update. At the end of each temporary horizon, SAPO promotes a temporary skill to the long-term bank only if it has positive utility, ranks among the top fraction of temporary skills, and is sufficiently distinct from existing long-term skills:

$$U_s > 0, \quad s \in \text{Top}_\rho(\mathcal{B}_{\text{temp}}; U), \quad \max_{s' \in \mathcal{B}_{\text{long}}} \text{Sim}(s, s') < \gamma. \tag{6}$$

Here, $\text{Top}_\rho(\mathcal{B}_{\text{temp}}; U)$ selects the top ρ fraction by validation utility, and γ controls the novelty threshold against the long-term bank. Promoted skills are added to $\mathcal{B}_{\text{long}}$, while the remaining temporary skills are discarded before the next horizon.

4.3 Policy as a Skill Generator and Scorer

The skill utility in Eq. (4) provides direct supervision for each newly induced skill without extra rollout budget, and measures the skill's marginal contribution beyond the existing retrieved skills. However, using this utility only to filter or track low-quality skills leaves two issues unresolved. First, existing methods often rely on costly closed-source LLMs for skill generation. Second, evaluating stored or retrieved skills for outdated-skill pruning and retrieval-time selection through additional counterfactual rollouts would be computationally expensive. SAPO therefore reuses the utility signal to train the policy itself as both a skill generator and a skill scorer.

Skill Generator Training. The utilities in Eq. (4) provide direct supervision for the skill generator. Inspired by W-REINFORCE (Zhu et al., 2025), SAPO trains the generator with an asymmetric utility-weighted objective: positive-utility skills are reinforced, while negative-utility skills are suppressed more strongly. Specifically, let $\ell_\theta(\hat{s}_i; x_i) = \log \pi_\theta(\hat{s}_i \mid x_i, \mathcal{S}_i, X_{\text{skill}}, \mathcal{Y}_i^{\text{base}})$ denote the log-probability of generating $\hat{s}_i$ from the retrieved skills and base rollouts of x_i. SAPO optimizes

$$\begin{aligned} \mathcal{L}_{\text{gen}}(\theta) = &-\mathbb{E}_{x_i} \left[\lambda [u(x_i, \hat{s}_i)]_+ \, \ell_\theta(\hat{s}_i; x_i) \right] \\ &+ \mathbb{E}_{x_i} \left[[-u(x_i, \hat{s}_i)]_+ \, \ell_\theta(\hat{s}_i; x_i) \right]. \end{aligned} \tag{7}$$

where $[a]_+ = \max(a, 0)$ and $\lambda \in [0, 1]$ down-weights positive reinforcement. We use the intra-prompt utility rather than a cross-prompt utility to supervise the generator. First, even when $K > 1$ and the generator conditions on multiple queries and their corresponding rollouts, the skill-generation interface remains the same as in the single-query case: the grouped setting only provides more task descriptions and base rollouts, while the prompt structure and output format are unchanged. More details are provided in Appendix B. Thus, intra-prompt utility teaches the generator to extract a useful skill from the evidence available in the current prompt, and this ability naturally extends to grouped-query skill induction. The effectiveness of this design is further supported by Sec. 5.4. Second, skill maintenance and retrieval-time reranking are query-specific decisions. A cross-prompt utility can blur this query-specific value: a skill that is unhelpful for the current query may still receive a high score because it is useful for other queries. Intra-prompt utility therefore provides a more faithful supervision signal for both generating and selecting skills.

Policy-Based Skill Scoring. Eq. (7) trains the policy to assign higher likelihood to positive-utility skills and lower likelihood to negative-utility skills. Thus, the skill-generation likelihood itself can serve as a learned skill-usefulness score. Given a task x, skill context $\mathcal{S}$, optional rollout evidence $\mathcal{Y}$, and candidate skill s, SAPO defines

$$\texttt{Score}_{\pi_\theta}(x, \mathcal{S}, X_{\text{skill}}, [\mathcal{Y}], s) = \frac{1}{|s|} \sum_{l=1}^{|s|} \log \pi_\theta\Big(s^{(l)} \mid x, \mathcal{S}, X_{\text{skill}}, [\mathcal{Y}], s^{(<l)}\Big). \tag{8}$$

Table 1: Main results on ALFWorld and WebShop. **Best** and Second-Best results are highlighted.

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 Agentic 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 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	95.6	80.4	**100**	86.7	78.7	75.2	87.9	83.2	71.9
D2Skill	93.8	**94.7**	95.5	77.8	95.0	72.0	87.8	83.4	73.4
SAPO	**98.7**	73.9	98.1	**92.6**	85.0	**89.2**	**92.2**	**90.5**	**78.1**

where $[\mathcal{Y}]$ denotes an optional rollout input. During training, the full-mode score is conditioned on the collected base rollouts $\mathcal{Y}_i^{\text{base}}$. At maintenance or retrieval time, however, collecting rollouts for every candidate skill would be costly, so SAPO omits $[\mathcal{Y}]$. To transfer the preferences learned in the full mode to this efficient reduced mode, SAPO distills the full distribution into the reduced-input distribution:

$$\mathcal{L}_{\text{KD}}(\theta) = \mathbb{E}_{x_i}\Big[\text{KL}\Big(\text{sg}\Big[\pi_\theta(\cdot \mid x_i, \mathcal{S}_i, X_{\text{skill}}, \mathcal{Y}_i^{\text{base}})\Big] \Big\| \pi_\theta(\cdot \mid x_i, \mathcal{S}_i, X_{\text{skill}})\Big)\Big]. \tag{9}$$

where $\text{sg}[\cdot]$ denotes stop-gradient.

Long-Term Skill Maintenance. When the long-term skill bank reaches its capacity limit, SAPO removes outdated or low-value skills using the reduced-input skill-likelihood score in Eq. (8). For each old skill s, SAPO constructs a set of relevant evaluation prompts $\mathcal{P}(s)$. For each prompt $x \in \mathcal{P}(s)$, SAPO retrieves a contextual skill sequence $\mathbf{S}^{\text{ctx}}(x, s)$ from the current bank while excluding s, and scores the old skill with $\texttt{Score}_{\pi_\theta}(x, \mathbf{S}^{\text{ctx}}(x, s), X_{\text{skill}}, s)$. The scores are averaged over $\mathcal{P}(s)$, and skills with low average scores are removed from the long-term bank.

Retrieval-Time Skill Selection. Whenever skill retrieval is needed, SAPO first retrieves a candidate pool $\mathcal{C}(x)$ using similarity-based retrieval and then reranks it with the reduced-input skill-likelihood score in Eq. (8). Skills are selected sequentially: at step k, given already selected skills $\mathbf{S}_{k-1} = (s_1, \ldots, s_{k-1})$ with $\mathbf{S}_0 = \emptyset$, SAPO scores each remaining candidate $s \in \mathcal{C}(x) \setminus \{s_1, \ldots, s_{k-1}\}$ using $\texttt{Score}_{\pi_\theta}(x, \mathbf{S}_{k-1}, X_{\text{skill}}, s)$ and selects the highest-scoring one as s_k. The process repeats until the retrieval budget is reached. The full SAPO algorithm is provided in Appendix A.

Table 2: Results on single-hop and multi-hop QA benchmarks.

Method	Single-Hop QA			Multi-Hop QA				Avg.
	NQ	TriviaQA	PopQA	Hotpot.	2Wiki	MuSiQue	Bamboog.	
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	61.2	45.9	43.9	39.9	17.2	45.6	45.5
SAPO	**48.4**	62.9	46.7	**45.0**	**45.2**	18.3	46.4	**47.8**

5 EXPERIMENTS

5.1 EXPERIMENTAL SETUP

Benchmarks. We evaluate SAPO on three benchmark families: ALFWORLD (Shridhar et al., 2020), WEBSHOP (Yao et al., 2022a), and search-augmented question answering. ALFWORLD tests embodied household tasks through text-based interaction, while WEBSHOP evaluates web navigation for product search and purchase under user-specified constraints. For search-augmented QA, we consider both single-hop datasets, including NQ (Kwiatkowski et al., 2019), TriviaQA (Joshi et al., 2017), and PopQA (Mallen et al., 2023), and multi-hop datasets, including HotpotQA (Yang et al., 2018), 2Wiki (Ho et al., 2020), MuSiQue (Trivedi et al., 2022), and Bamboogle (Press et al., 2023).

Baselines. We compare SAPO against five categories of baselines. First, we include strong closed-source LLMs, including GPT-4o and Gemini-2.5-Pro (Comanici et al., 2025), which represent state-of-the-art general-purpose reasoning models. Second, we consider prompt-based and memory-based agentic methods, including ReAct (Yao et al., 2022b), Reflexion (Shinn et al., 2023), Mem0 (Chhikara et al., 2025), ExpeL (Zhao et al., 2024), and MemP (Fang et al., 2025). Third, we include general RL baselines, including RLOO (Ahmadian et al., 2024) and GRPO (Shao et al., 2024). Fourth, we compare against memory-augmented RL methods, including EvolveR (Wu et al., 2025b), MemRL (Zhang et al., 2026c), and SimpleMem (Liu et al., 2026a). Fifth, we compare with prior skill-augmented RL methods, including SkillRL (Xia et al., 2026), Skill0 (Lu et al., 2026), and D2Skill (Tu et al., 2026). For search-augmented QA, we additionally compare with search-oriented reasoning baselines, including Search-o1 (Li et al., 2025), Search-R1 (Jin et al., 2025), ZeroSearch (Sun et al., 2025), and StepSearch (Wang et al., 2025b).

Implementation Details. We use `Qwen2.5-7B-Instruct` and `Qwen3-4B-Instruct` (Yang et al., 2025a) as backbones. For ALFWorld, we follow GiGPO (Feng et al., 2025); for Search-QA, we follow Search-R1 (Jin et al., 2025) with E5 retrieval (Wang et al., 2022), training on NQ and HotpotQA and evaluating on the remaining QA datasets. The skill bank is initialized from SkillRL (Xia et al., 2026). Unless otherwise specified, we set the induction-unit size $K = 4$, promotion ratio $\rho = 20\%$, and novelty threshold $\gamma = 0.8$. Full implementation details are in Appendix C. A hyperparameter analysis of K is provided in Sec. D.4.

5.2 MAIN RESULTS

Results on ALFWorld and WebShop. As shown in Table 1, SAPO achieves the best overall performance among the compared methods. The consistent improvement over SkillRL, Skill0, and D2Skill indicates that performance in skill-augmented RL depends not only on expanding the skill

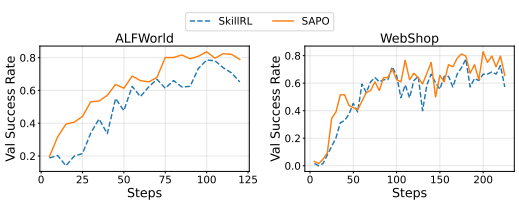


Figure 4: Validation success rate of SAPO and SkillRL on ALFWorld and WebShop.

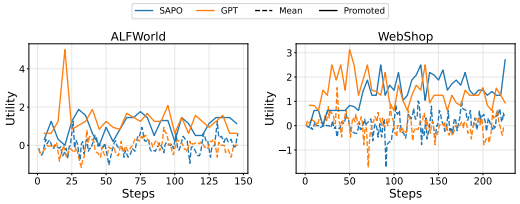


Figure 5: Skill utility of SAPO and SkillRL on ALFWorld and WebShop during training.

bank, but also on controlling the quality of the skills that enter it. Sec. D.2 shows similar gains when using `Qwen3-4B-Instruct` as the base model.

Results on Search-Augmented QA. As shown in Table 2, SAPO achieves the best average performance among the compared methods. Its gains on both training-domain and out-of-domain QA benchmarks suggest that marginal-utility feedback helps the skill generator learn transferable search and reasoning strategies rather than dataset-specific shortcuts.

5.3 Training Curves

We compare the training dynamics of SAPO and SkillRL (Xia et al., 2026) using validation success rate, with `Qwen3-4B-Instruct` as the base model. As shown in Figure 4, we make two observations. **(1)** SAPO improves faster than SkillRL on both benchmarks, suggesting that utility-based skill filtering provides more reliable guidance when the base policy is still weak. **(2)** SAPO achieves higher best validation performance, consistent with the main results and indicating that its gains generalize across backbones. Appendix D.3 provides training curves on individual ALFWorld subtasks.

5.4 Skill Utility

In this section, we analyze the training dynamics of generated skill utility. We compare SAPO with SkillRL, where SAPO uses `Qwen3-4B-Instruct` as the skill generator, while SkillRL uses GPT-5.4. The results are shown in Figure 5. From the figure, we make two observations. **(1)** In the early stage of training, SAPO generates skills with lower utility than those generated by GPT. This is expected because SAPO relies on a smaller open-weight model as its skill generator. However, as training progresses, the SAPO skill generator learns to produce more beneficial skills. Its skill utility becomes comparable to, and in some cases slightly better than, that of the GPT-based generator used in SkillRL. This demonstrates the benefit of training the skill generator with utility feedback. **(2)** Even when the policy becomes stronger in the later stage of training, the SAPO skill generator can still consistently produce skills with positive utility. This suggests that useful skills can continue to provide additional gains beyond an improved base policy.

6 Conclusion

We presented SAPO, an online RL framework that validates generated skills before storage and uses marginal utility to improve skill generation, maintenance, and retrieval. SAPO derives skill-utility signals from the same rollouts used for agent learning, enabling skill generator training without extra rollout cost or repeated proprietary LLM calls.

AI Use Statement

Generative AI tools were used to assist with language editing, improving manuscript readability, refining the presentation of the paper, and limited code editing. They were not used to design the research methodology or experiments, formulate or refine hypotheses, or interpret experimental results. All AI-assisted content was reviewed and verified by the authors. The authors take full responsibility for the final content of this work.

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.

Shuzhen Bi, Mengsong Wu, Hao Hao, Keqian Li, Wentao Liu, Siyu Song, Hongbo Zhao, and Aimin Zhou. Automating skill acquisition through large-scale mining of open-source agentic repositories: A framework for multi-agent procedural knowledge extraction. *arXiv preprint arXiv:2603.11808*, 2026.

Rajkumar Buyya et al. Agentic artificial intelligence (ai): Architectures, taxonomies, and evaluation of large language model agents. *arXiv preprint arXiv:2601.12560*, 2026.

Yuxuan Cai, Yipeng Hao, Jie Zhou, Hang Yan, Zhikai Lei, Rui Zhen, Zhenhua Han, Yutao Yang, Junsong Li, Qianjun Pan, et al. Building self-evolving agents via experience-driven lifelong learning: A framework and benchmark. *arXiv preprint arXiv:2508.19005*, 2025a.

Zhicheng Cai, Xinyuan Guo, Yu Pei, Jiangtao Feng, Jinsong Su, Jiangjie Chen, Ya-Qin Zhang, Wei-Ying Ma, Mingxuan Wang, and Hao Zhou. Flex: Continuous agent evolution via forward learning from experience. *arXiv preprint arXiv:2511.06449*, 2025b.

Zouying Cao, Jiaji Deng, Li Yu, Weikang Zhou, Zhaoyang Liu, Bolin Ding, and Hai Zhao. Remember me, refine me: A dynamic procedural memory framework for experience-driven agent evolution. *arXiv preprint arXiv:2512.10696*, 2025.

Wei-Lin Chen, Liqian Peng, Tian Tan, Chao Zhao, Blake JianHang Chen, Ziqian Lin, Alec Go, and Yu Meng. Think deep, not just long: Measuring llm reasoning effort via deep-thinking tokens. *arXiv preprint arXiv:2602.13517*, 2026.

Zhaorun Chen, Zhuokai Zhao, Kai Zhang, Bo Liu, Qi Qi, Yifan Wu, Tarun Kalluri, Sara Cao, Yuanhao Xiong, Haibo Tong, et al. Scaling agent learning via experience synthesis. *arXiv preprint arXiv:2511.03773*, 2025.

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.

Pengfei Du. Memory for autonomous llm agents: Mechanisms, evaluation, and emerging frontiers. *arXiv preprint arXiv:2603.07670*, 2026.

Gaodan Fang, Vatche Isahagian, KR Jayaram, Ritesh Kumar, Vinod Muthusamy, Punleuk Oum, and Gegi Thomas. Trajectory-informed memory generation for self-improving agent systems. *arXiv preprint arXiv:2603.10600*, 2026.

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. *arXiv preprint arXiv:2505.10978*, 2025.

Yingchaojie Feng, Qiang Huang, Xiaoya Xie, Zhaorui Yang, Jun Yu, Wei Chen, and Anthony KH Tung. Idrbench: Interactive deep research benchmark. *arXiv preprint arXiv:2601.06676*, 2026.

Dongge Han, Camille Couturier, Daniel Madrigal Diaz, Xuchao Zhang, Victor Rühle, and Saravan Rajmohan. Legomem: Modular procedural memory for multi-agent llm systems for workflow automation. *arXiv preprint arXiv:2510.04851*, 2025.

Guangfu Hao, Yuming Dai, Xianzhe Qin, and Shan Yu. Brain-inspired graph multi-agent systems for llm reasoning. *arXiv preprint arXiv:2603.15371*, 2026.

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*, pp. 6609–6625, 2020.

Jinchao Hu, Meizhi Zhong, Kehai Chen, Xuefeng Bai, and Min Zhang. Agentic tool use in large language models. *arXiv preprint arXiv:2604.00835*, 2026.

Tenghao Huang, Kinjal Basu, Ibrahim Abdelaziz, Pavan Kapanipathi, Jonathan May, and Muhao Chen. R2d2: Remembering, replaying and dynamic decision making with a reflective agentic memory. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 30318–30330, 2025a.

Xu Huang, Junwu Chen, Yuxing Fei, Zhuohan Li, Philippe Schwaller, and Gerbrand Ceder. Cascade: Cumulative agentic skill creation through autonomous development and evolution. *arXiv preprint arXiv:2512.23880*, 2025b.

Yanna Jiang, Delong Li, Haiyu Deng, Baihe Ma, Xu Wang, Qin Wang, and Guangsheng Yu. Sok: Agentic skills–beyond tool use in llm agents. *arXiv preprint arXiv:2602.20867*, 2026.

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

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 (Volume 1: Long Papers)*, pp. 1601–1611, 2017.

Rajat Khanda, Mohammad Baqar Sambuddha Chakrabarti, and Satyasaran Changdar. Adaptive memory crystallization for autonomous ai agent learning in dynamic environments. *arXiv preprint arXiv:2604.13085*, 2026.

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*, 7:453–466, 2019.

Xiangyi Li, Wenbo Chen, Yimin Liu, Shenghan Zheng, Xiaokun Chen, Yifeng He, Yubo Li, Bingran You, Haotian Shen, Jiankai Sun, et al. Skillsbench: Benchmarking how well agent skills work across diverse tasks. *arXiv preprint arXiv:2602.12670*, 2026a.

Xiaochuan Li, Ryan Ming, Pranav Setlur, Abhijay Paladugu, Andy Tang, Hao Kang, Shuai Shao, Rong Jin, and Chenyan Xiong. Benchmark test-time scaling of general llm agents. *arXiv preprint arXiv:2602.18998*, 2026b.

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.

Yu Li, Rui Miao, Zhengling Qi, and Tian Lan. Arise: Agent reasoning with intrinsic skill evolution in hierarchical reinforcement learning. *arXiv preprint arXiv:2603.16060*, 2026c.

Xiaoqiang Lin, Jun Hao Liew, Silvio Savarese, and Junnan Li. W&d: Scaling parallel tool calling for efficient deep research agents. *arXiv preprint arXiv:2602.07359*, 2026.

Anthony Zhe Liu, Jongwook Choi, Sungryull Sohn, Yao Fu, Jaekyeom Kim, Dong-Ki Kim, Xinhe Wang, Jaewon Yoo, and Honglak Lee. Skillact: Using skill abstractions improves llm agents. In *ICML 2024 Workshop on LLMs and Cognition*, 2024.

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*, 2026a.

Yujian Liu, Jiabao Ji, Li An, Tommi Jaakkola, Yang Zhang, and Shiyu Chang. How well do agentic skills work in the wild: Benchmarking llm skill usage in realistic settings. *arXiv preprint arXiv:2604.04323*, 2026b.

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*, 2026.

Ziyu Ma, Shidong Yang, Yuxiang Ji, Xucong Wang, Yong Wang, Yiming Hu, Tongwen Huang, and Xiangxiang Chu. Skillclaw: Let skills evolve collectively with agentic evolver. *arXiv preprint arXiv:2604.08377*, 2026.

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 (volume 1: Long papers)*, pp. 9802–9822, 2023.

Jingwei Ni, Yihao Liu, Xinpeng Liu, Yutao Sun, Mengyu Zhou, Pengyu Cheng, Dexin Wang, Xiaoxi Jiang, and Guanjun Jiang. Trace2skill: Distill trajectory-local lessons into transferable agent skills. *arXiv preprint arXiv:2603.25158*, 2026.

Siru Ouyang, Jun Yan, I Hsu, Yanfei Chen, Ke Jiang, Zifeng Wang, Rujun Han, Long T Le, Samira Daruki, Xiangru Tang, et al. Reasoningbank: Scaling agent self-evolving with reasoning memory. *arXiv preprint arXiv:2509.25140*, 2025.

Aske Plaat, Max van Duijn, Niki Van Stein, Mike Preuss, Peter van der Putten, and Kees Joost Batenburg. Agentic large language models, a survey. *Journal of Artificial Intelligence Research*, 84, 2025.

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, 2023.

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.

Shuaike Shen, Wenduo Cheng, Mingqian Ma, Alistair Turcan, Martin Jinye Zhang, and Jian Ma. Skillfoundry: Building self-evolving agent skill libraries from heterogeneous scientific resources. *arXiv preprint arXiv:2604.03964*, 2026.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. *Advances in neural information processing systems*, 36:8634–8652, 2023.

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*, 2020.

David Silver and Richard S Sutton. Welcome to the era of experience. *Google AI*, 1:11, 2025.

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.

Mirac Suzgun, Mert Yuksekgonul, Federico Bianchi, Dan Jurafsky, and James Zou. Dynamic cheatsheet: Test-time learning with adaptive memory. In *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 7080–7106, 2026.

Xiangru Tang, Tianrui Qin, Tianhao Peng, Ziyang Zhou, Daniel Shao, Tingting Du, Xinming Wei, Peng Xia, Fang Wu, He Zhu, et al. Agent kb: Leveraging cross-domain experience for agentic problem solving. *arXiv preprint arXiv:2507.06229*, 2025.

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

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*, 2026.

Chenxi Wang, Zhuoyun Yu, Xin Xie, Wuguannan Yao, Runnan Fang, Shuofei Qiao, Kexin Cao, Guozhou Zheng, Xiang Qi, Peng Zhang, et al. Skillx: Automatically constructing skill knowledge bases for agents. *arXiv preprint arXiv:2604.04804*, 2026a.

Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An open-ended embodied agent with large language models. *arXiv preprint arXiv:2305.16291*, 2023.

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*, 2025a.

Leye Wang, Zixing Wang, and Anjie Xu. Skilltester: Benchmarking utility and security of agent skills. *arXiv preprint arXiv:2603.28815*, 2026b.

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.

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

Zora Zhiruo Wang, Apurva Gandhi, Graham Neubig, and Daniel Fried. Inducing programmatic skills for agentic tasks. *arXiv preprint arXiv:2504.06821*, 2025c.

Tianxin Wei, Ting-Wei Li, Zhining Liu, Xuying Ning, Ze Yang, Jiaru Zou, Zhichen Zeng, Ruizhong Qiu, Xiao Lin, Dongqi Fu, et al. Agentic reasoning for large language models. *arXiv preprint arXiv:2601.12538*, 2026.

Junde Wu, Jiayuan Zhu, Yuyuan Liu, Min Xu, and Yueming Jin. Agentic reasoning: A streamlined framework for enhancing llm reasoning with agentic tools. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 28489–28503, 2025a.

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*, 2025b.

Yaxiong Wu and Yongyue Zhang. Agent skills from the perspective of procedural memory: A survey. *Authorea Preprints*, 2026.

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.

Renjun Xu and Yang Yan. Agent skills for large language models: Architecture, acquisition, security, and the path forward. *arXiv preprint arXiv:2602.12430*, 2026.

Wujiang Xu, Jiaojiao Han, Minghao Guo, Kai Mei, Xi Zhu, Han Zhang, and Dimitris N Metaxas. Ael: Agent evolving learning for open-ended environments. *arXiv preprint arXiv:2604.21725*, 2026.

Sikuan Yan, Xiufeng Yang, Zuchao Huang, Ercong Nie, Zifeng Ding, Zonggen Li, Xiaowen Ma, Jinhe Bi, Kristian Kersting, Jeff Z Pan, et al. Memory-r1: Enhancing large language model agents to manage and utilize memories via reinforcement learning. *arXiv preprint arXiv:2508.19828*, 2025.

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

Chang Yang, Chuang Zhou, Yilin Xiao, Su Dong, Luyao Zhuang, Yujing Zhang, Zhu Wang, Zijin Hong, Zheng Yuan, Zhishang Xiang, et al. Graph-based agent memory: Taxonomy, techniques, and applications. *arXiv preprint arXiv:2602.05665*, 2026a.

Shuo Yang, Soyeon Caren Han, Yihao Ding, Shuhe Wang, and Eduard Hoy. Tooltree: Efficient llm agent tool planning via dual-feedback monte carlo tree search and bidirectional pruning. *arXiv preprint arXiv:2603.12740*, 2026b.

Yongjin Yang, Sinjae Kang, Juyong Lee, Dongjun Lee, Se-Young Yun, and Kimin Lee. Automated skill discovery for language agents through exploration and iterative feedback. *arXiv preprint arXiv:2506.04287*, 2025b.

Yutao Yang, Junsong Li, Qianjun Pan, Bihao Zhan, Yuxuan Cai, Lin Du, Jie Zhou, Kai Chen, Qin Chen, Xin Li, et al. Autoskill: Experience-driven lifelong learning via skill self-evolution. *arXiv preprint arXiv:2603.01145*, 2026c.

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 2018 conference on empirical methods in natural language processing*, pp. 2369–2380, 2018.

Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. Webshop: Towards scalable real-world web interaction with grounded language agents. *Advances in Neural Information Processing Systems*, 35:20744–20757, 2022a.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629*, 2022b.

Haoran Ye, Xuning He, Vincent Arak, Haonan Dong, and Guojie Song. Meta context engineering via agentic skill evolution. *arXiv preprint arXiv:2601.21557*, 2026.

Yi Yu, Liuyi Yao, Yuexiang Xie, Qingquan Tan, Jiaqi Feng, Yaliang Li, and Libing Wu. Agentic memory: Learning unified long-term and short-term memory management for large language model agents. *arXiv preprint arXiv:2601.01885*, 2026.

Yunpeng Zhai, Shuchang Tao, Cheng Chen, Anni Zou, Ziqian Chen, Qingxu Fu, Shinji Mai, Li Yu, Jiaji Deng, Zouying Cao, et al. Agentevolver: Towards efficient self-evolving agent system. *arXiv preprint arXiv:2511.10395*, 2025.

Barry Zhang, Keith Lazuka, and Mahesh Murag. Equipping agents for the real world with agent skills, october 2025. *URL https://www.anthropic.com/engineering/equipping-agents-for-the-real-world-with-agent-skills.Accessed*, pp. 01–28, 2026a.

Guibin Zhang, Muxin Fu, Guancheng Wan, Miao Yu, Kun Wang, and Shuicheng Yan. G-memory: Tracing hierarchical memory for multi-agent systems. *arXiv preprint arXiv:2506.07398*, 2025a.

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*, 2025b.

Guibin Zhang, Haotian Ren, Chong Zhan, Zhenhong Zhou, Junhao Wang, He Zhu, Wangchunshu Zhou, and Shuicheng Yan. Memevolve: Meta-evolution of agent memory systems. *arXiv preprint arXiv:2512.18746*, 2025c.

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

Jenny Zhang, Shengran Hu, Cong Lu, Robert Lange, and Jeff Clune. Darwin godel machine: Open-ended evolution of self-improving agents. *arXiv preprint arXiv:2505.22954*, 2025d.

Kai Zhang, Xiangchao Chen, Bo Liu, Tianci Xue, Zeyi Liao, Zhihan Liu, Xiyao Wang, Yuting Ning, Zhaorun Chen, Xiaohan Fu, et al. Agent learning via early experience. *arXiv preprint arXiv:2510.08558*, 2025e.

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*, 2026c.

Xing Zhang, Guanghui Wang, Yanwei Cui, Wei Qiu, Ziyuan Li, Bing Zhu, and Peiyang He. Experience compression spectrum: Unifying memory, skills, and rules in llm agents. *arXiv preprint arXiv:2604.15877*, 2026d.

Ziao Zhang, Kou Shi, Shiting Huang, Avery Nie, Yu Zeng, Yiming Zhao, Zhen Fang, Qishen Su, Haibo Qiu, Wei Yang, et al. Skillflow: Benchmarking lifelong skill discovery and evolution for autonomous agents. *arXiv preprint arXiv:2604.17308*, 2026e.

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.

Bingxi Zhao, Lin Geng Foo, Ping Hu, Christian Theobalt, Hossein Rahmani, and Jun Liu. Llm-based agentic reasoning frameworks: A survey from methods to scenarios. *arXiv preprint arXiv:2508.17692*, 2025.

Boyuan Zheng, Michael Y Fatemi, Xiaolong Jin, Zora Zhiruo Wang, Apurva Gandhi, Yueqi Song, Yu Gu, Jayanth Srinivasa, Gaowen Liu, Graham Neubig, et al. Skillweaver: Web agents can self-improve by discovering and honing skills. *arXiv preprint arXiv:2504.07079*, 2025.

Shanshan Zhong, Yi Lu, Jingjie Ning, Yibing Wan, Lihan Feng, Yuyi Ao, Leonardo FR Ribeiro, Markus Dreyer, Sean Ammirati, and Chenyan Xiong. Skilllearnbench: Benchmarking continual learning methods for agent skill generation on real-world tasks. *arXiv preprint arXiv:2604.20087*, 2026.

Chenyu Zhou, Huacan Chai, Wenteng Chen, Zihan Guo, Rong Shan, Yuanyi Song, Tianyi Xu, Yingxuan Yang, Aofan Yu, Weiming Zhang, et al. Externalization in llm agents: A unified review of memory, skills, protocols and harness engineering. *arXiv preprint arXiv:2604.08224*, 2026a.

Huichi Zhou, Yihang Chen, Siyuan Guo, Xue Yan, Kin Hei Lee, Zihan Wang, Ka Yiu Lee, Guchun Zhang, Kun Shao, Linyi Yang, et al. Memento: Fine-tuning llm agents without fine-tuning llms. *arXiv preprint arXiv:2508.16153*, 2025a.

Huichi Zhou, Siyuan Guo, Anjie Liu, Zhongwei Yu, Ziqin Gong, Bowen Zhao, Zhixun Chen, Menglong Zhang, Yihang Chen, Jinsong Li, et al. Memento-skills: Let agents design agents. *arXiv preprint arXiv:2603.18743*, 2026b.

Yifei Zhou, Qianlan Yang, Kaixiang Lin, Min Bai, Xiong Zhou, Yu-Xiong Wang, Sergey Levine, and Li Erran Li. Proposer-agent-evaluator (pae): Autonomous skill discovery for foundation model internet agents. In *Forty-second International Conference on Machine Learning*, 2025b.

Xinyu Zhu, Mengzhou Xia, Zhepei Wei, Wei-Lin Chen, Danqi Chen, and Yu Meng. The surprising effectiveness of negative reinforcement in llm reasoning. *arXiv preprint arXiv:2506.01347*, 2025.

REPRODUCIBILITY STATEMENT

We provide the full algorithmic details and prompts in Appendix A–C, additional experimental results and hyperparameter analyses in Appendix D, and implementation and cost details in Appendix E.

A ALGORITHMIC DETAILS

In this section, we provide the full algorithmic procedure of SAPO in Algorithm 1.

Algorithm 1 Skill-Augmented Policy Optimization (SAPO)

Inputs: policy π_θ, long-term skill bank $\mathcal{B}_{\text{long}}$, temporary skill bank $\mathcal{B}_{\text{temp}}$, training distribution $\mathcal{D}$, rollout budget G, unit size K, temporary horizon T_{prom}, skill-generation prompt X_{skill}
for training step $t = 1, 2, \ldots$ **do**
  Sample a batch of skill induction units $\{\mathcal{X}_i\} \sim \mathcal{D}$, where each unit is $\mathcal{X}_i = \{x_{i,j}\}_{j=1}^{K}$
  for each induction unit $\mathcal{X}_i$ **do**
    for each query $x_{i,j} \in \mathcal{X}_i$ **do**
      Retrieve skills $\mathcal{S}_{i,j} \leftarrow \text{Retr}(x_{i,j}, \mathcal{B}_{\text{long}} \cup \mathcal{B}_{\text{temp}})$ and rerank them with $\texttt{Score}_{\pi_\theta}$ defined in Eq. (8)
      Sample base trajectories $\mathcal{Y}_{i,j}^{\text{base}} \sim \pi_\theta(\cdot \mid x_{i,j}, \mathcal{S}_{i,j})$
    end for
    Generate one candidate skill from the induction unit using Eq. (2)
    for each query $x_{i,j} \in \mathcal{X}_i$ **do**
      Sample skill-augmented trajectories $\mathcal{Y}_{i,j}^{\text{skill}} \sim \pi_\theta(\cdot \mid x_{i,j}, \mathcal{S}_{i,j} \cup \{\hat{s}_i\})$
      Compute prompt-specific utility $u(x_{i,j}, \hat{s}_i)$ using Eq. (4)
    end for
    Aggregate cross-prompt utility $U_{\hat{s}_i} = \frac{1}{K} \sum_{j=1}^{K} u(x_{i,j}, \hat{s}_i)$ using Eq. (5)
    Store $(\hat{s}_i, U_{\hat{s}_i})$ in $\mathcal{B}_{\text{temp}}$
  end for
  Update π_θ for the agent task with GRPO using all collected base and skill-augmented rollouts
  if $t \bmod T_{\text{prom}} = 0$ **then**
    Update π_θ as the skill generator with Eq. (7) and distill the reduced-input skill-likelihood score with Eq. (9)
    Promote high-utility and non-redundant skills from $\mathcal{B}_{\text{temp}}$ to $\mathcal{B}_{\text{long}}$ using Eq. (6)
    Discard unpromoted temporary skills and reset $\mathcal{B}_{\text{temp}}$
    if $|\mathcal{B}_{\text{long}}|$ exceeds the bank capacity **then**
      Remove low-scoring old skills using $\texttt{Score}_{\pi_\theta}$ defined in Eq. (8)
    end if
  end if
end for

B PROMPTS

In this section, we provide the system prompt used for skill generation in SAPO. The same prompt template is used for both single-query induction ($K = 1$) and grouped-query induction ($K > 1$). In implementation, the grouped case does not introduce a new skill-generation interface. Instead, SAPO formats the inputs in the same slots as the single-query case. When $K = 1$, $\{\texttt{task_description}\}$ contains one task and $\{\texttt{base_trajectories_text}\}$ contains its corresponding base rollouts. When $K > 1$, SAPO simply concatenates the K related tasks and their base rollouts into the same fields, for example,

$$\{\texttt{task_description}\} = \texttt{Task 1:}\ \ x_{i,1} \parallel \cdots \parallel \texttt{Task K:}\ x_{i,K}.$$

and

$$\{\texttt{base_trajectories_text}\} = \texttt{Trajectories for Task 1:}\ \mathcal{Y}_{i,1}^{\text{base}} \ \|\ \cdots\ \|\ \texttt{Trajectories for Task K:}\ \mathcal{Y}_{i,K}^{\text{base}}.$$

Thus, grouped-query induction only provides richer evidence to the same prompt template. The model is still asked to return exactly one JSON object, so the output remains a single shared skill $\hat{s}_i$ for the whole induction unit $\mathcal{X}_i = \{x_{i,j}\}_{j=1}^{K}$.

System Prompt for Skill Generation Agent of SAPO

SKILL GENERATION AGENT SYSTEM PROMPT
You are an expert agent trainer. Based on the following task and trajectory rollouts (with rewards), generate a NEW actionable skill that would improve the agent's performance.

Task
{task_description}

{context_skills_text}

Trajectories
{base_trajectories_text}

Instructions
Analyze the trajectories above. Identify patterns: which actions led to success vs failure? Generate exactly ONE new skill that would help the agent succeed on this type of task.
The skill must be a JSON object with these fields:

- `"skill_id": "candidate"`
- `"title"`: 4–8 words clearly naming the strategy
- `"principle"`: 3–5 sentences describing WHAT to do, HOW to do it step-by-step, and WHY it works. Be specific and actionable.
- `"when_to_apply"`: 2–3 sentences specifying exact conditions that trigger this skill.

Return ONLY a valid JSON object (not an array), no other text.

C Implementation Details

We train `Qwen2.5-7B-Instruct` and `Qwen3-4B-Instruct` on one node with 8 H200 GPUs. All RL experiments are implemented with the `verl` actor-rollout-reference framework using GRPO as the advantage estimator. The actor is optimized with AdamW using learning rate 1×10^{-6}. Rollouts are served with `vLLM` using tensor parallel size 1, and validation responses are sampled with temperature 0.4.

Benchmarks and Training Setup. For ALFWorld, we use the GiGPO training split (Feng et al., 2025). For WebShop, we use the standard WebShop environment. For Search-QA, we follow Search-R1 (Jin et al., 2025), use E5 (Wang et al., 2022) as the retriever, train on NQ and HotpotQA, and evaluate out-of-domain on TriviaQA, PopQA, 2Wiki, MuSiQue, and Bamboogle. The Search-QA policy is initialized from `Jianwen/Search-7B-SFT`. The main training hyperparameters are summarized in Table 3.

Setting	ALFWorld	WebShop	Search-QA
Train batch size	16	16	512
Validation batch size	256	64	1,024
Rollouts per prompt	8	8	8
Max prompt length	4,096	6,000	5,000
Max response length	512	768	700
Max environment steps	50	15	4
KL coefficient	0.01	0.01	0.001
Training epochs	150	300	1
Evaluation frequency	5	5	50

Table 3: Main training hyperparameters for SAPO.

SkillBank and Skill-Generator Settings. Across all environments, skills are retrieved by embedding similarity with top-$k = 3$, and SAPO maintains a dual-bank memory with a temporary candidate bank and a long-term bank. Candidate skills are promoted every 5 epochs after utility-based validation. We use a maximum long-term bank size of 45 and deduplication threshold 0.8. The main SkillBank hyperparameters are shown in Table 4.

Setting	ALFWorld	WebShop	Search-QA
Promotion ratio ρ	0.2	0.2	0.1
Promotion interval	5	5	5
Prompt examples per skill	4	4	16
Top-k retrieved skills	3	3	3
Reranking pool size	6	10	6
Max long-term bank size	45	45	45
Deduplication threshold	0.8	0.8	0.8

Table 4: SkillBank and auxiliary skill-generator hyperparameters.

Hyperparameter Selection. Hyperparameters are selected based on validation performance and compute constraints. We set $K = 4$ queries per skill induction unit to balance utility-estimation stability and rollout cost. We use small promotion ratios ($\rho = 20\%$ for ALFWorld/WebShop and $\rho = 10\%$ for Search-QA) to keep the SkillBank compact. The deduplication threshold is set to 0.8 to filter near-duplicate skills.

D Additional Experimental Results

D.1 Utility of Claude-Generated Skills

To examine whether the mixed utility of generated skills is specific to GPT-generated skills, we repeat the preliminary utility analysis in Sec. 3 using Claude-Opus-4.6 as the skill generator. We follow the same experimental protocol: for each prompt, we first collect base rollouts, ask Claude-Opus-4.6 to generate a candidate skill from the observed failures, and then collect skill-augmented rollouts with the generated skill added. The reward gap between the two rollout groups is used as the marginal utility of the generated skill.

As shown in Fig. 6, Claude-generated skills also exhibit highly mixed utility on both ALFWorld and WebShop. Although promoted skills achieve clearly positive utility, the mean utility remains close to zero during training, indicating that many generated skills provide limited or even negative benefit. This result further supports our main observation that relying on frontier LLMs does not guarantee high-quality skills, and motivates SAPO's validate-before-store design.

Table 5: ALFWorld performance using `Qwen3-4B-Instruct-2507` as the base model.

Method	Pick	Clean	Cool	Look	Heat	Pick2	All
Origin	50.0	9.5	0.0	2.1	11.1	4.8	17.2
GRPO	73.5	46.6	48.0	61.1	62.5	20.0	53.9
SkillRL (O3)	90.0	92.3	52.0	63.6	42.9	40.9	67.2
D2Skill (Gemini-3-Flash)	88.6	75.0	54.2	66.7	60.0	52.6	69.6
D2Skill (O3)	89.4	72.4	66.7	54.5	60.0	50.0	72.7
SAPO	86.9	93.1	82.2	77.7	89.6	67.3	82.0

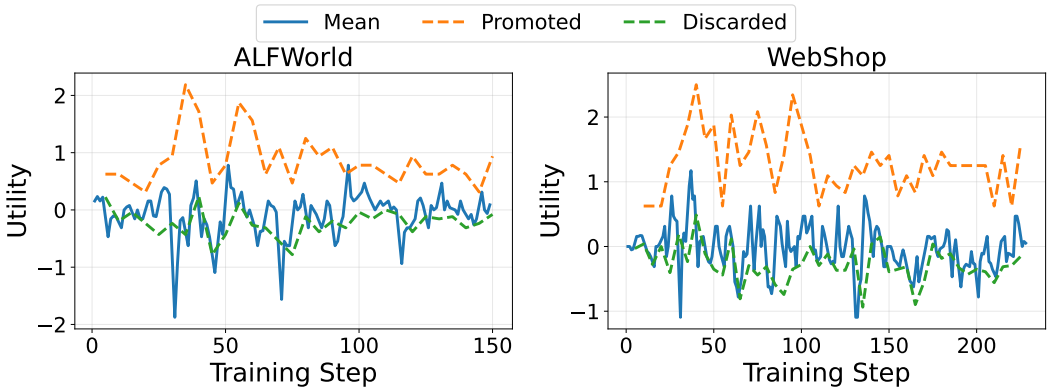


Figure 6: Utility of Claude-generated skills on ALFWorld and WebShop during training.

D.2 Main Results with Qwen3-4B-Instruct

To further evaluate SAPO under a smaller backbone, we follow the experimental setting in Sec. 5 and replace the base model with `Qwen3-4B-Instruct-2507`. Table 5 reports the results on ALFWorld. SAPO achieves the best overall performance, improving the All score from 72.7 to 82.0 over the strongest baseline. These results show that SAPO's gains are not limited to the 7B backbone used in the main experiments and remain effective with a smaller open-weight policy.

D.3 Training Dynamics on ALFWorld Subtasks

We further analyze the training dynamics of SAPO and SkillRL on individual ALFWorld subtasks. Following the experimental setting in Sec. 5, we use `Qwen2.5-7B-Instruct` as the base model and report validation performance throughout training.

As shown in Fig. 7, SAPO exhibits steady performance improvement across ALFWorld subtasks and shows smaller fluctuations in the later stage of training. In contrast, SkillRL often declines after reaching its peak performance on several subtasks. This suggests that directly evolving the skill bank without explicit skill curation can introduce low-quality or outdated skills, which may mislead the agent and destabilize later-stage learning.

D.4 Hyperparameter analysis

We study the effect of K, the number of queries used to induce each candidate skill. Table 7 reports results with different values of K and includes SkillRL as the baseline. SAPO consistently outperforms SkillRL across all tested settings, showing that its gains are not sensitive to a specific choice of K. Using multiple related queries generally improves over single-query induction, suggesting that grouped induction provides richer evidence for generating reusable skills. We use $K = 4$ as the default setting, which achieves the best WebShop score and strong overall performance across both benchmarks.

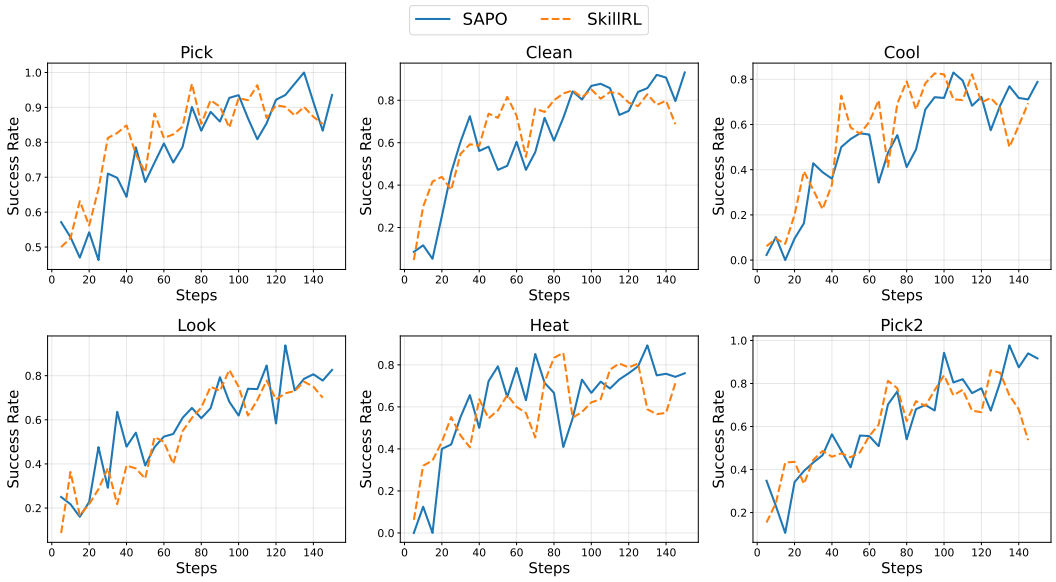


Figure 7: Training dynamics of SAPO and SkillRL on ALFWorld subtasks using `Qwen2.5-7B-Instruct` as the base model.

Table 6: Hyperparameter analysis on ALFWorld and WebShop.

Method	ALFWorld	WebShop	
	All.	Score	Succ.
SkillRL (Baseline)	89.9	85.2	72.7
SAPO ($K = 1$)	91.4	91.2	76.5
SAPO ($K = 8$)	93.3	88.8	79.6
SAPO ($K = 4$)	92.2	90.5	78.1

Table 7: Hyperparameter analysis of K, the number of queries used to induce each candidate skill.

Method	ALFWorld	WebShop	
	All.	Score	Succ.
SkillRL (Baseline)	89.9	85.2	72.7
SAPO ($K = 1$)	91.4	91.2	76.5
SAPO ($K = 8$)	93.3	88.8	79.6
SAPO ($K = 4$)	92.2	90.5	78.1

E API Cost of Baseline Skill Generation

SkillRL-style skill evolution relies on proprietary frontier LLM calls to generate new skills during training. In our ALFWorld runs, using GPT-5.4 for skill generation costs roughly $30 for a single run, even before accounting for environment rollout, retrieval, or policy-optimization cost. While this cost may appear modest at small scale, it becomes non-trivial for individual researchers when repeated across benchmarks, seeds, ablations, and hyperparameter settings. In contrast, SAPO trains the policy itself as the skill generator and derives skill-utility signals from the same rollouts used for agent learning, thereby avoiding repeated proprietary LLM calls during online RL.