ALPHASKILL: AGENTIC REINFORCEMENT LEARNING WITH SELF-EVOLVING SKILLS

Anonymous authors
Paper under double-blind review

ABSTRACT

Agent skills extract interaction experience into high-level, reusable knowledge that is essential for reliably solving complex tasks. Nevertheless, the acquisition of suitable skills remains challenging: manually designing them introduces a critical scalability bottleneck, while static skills may lead to a progressive misalignment between accumulated skills and the agent's evolving capabilities. In this work, we propose **AlphaSkill**, a reinforcement learning framework that enables self-curated skills to co-evolve alongside the agent's improving capabilities. In this paradigm, a single policy network functions as two roles: (1) a `Reasoner` that interacts with the environment to solve tasks, and (2) a `Skill Curator` that maintains a skills library by creating or refining skills based on the frontier of the Reasoner's capability. The Reasoner is rewarded for solving challenging tasks in environments, while the Skill Curator is optimized according to whether its curated skills contribute to the Reasoner's successes, thereby creating a virtuous loop of co-evolution between task solving and skills curation. Extensive experiments on various agentic environments and reasoning tasks demonstrate that AlphaSkill achieves consistent gains compared to its counterparts. Moreover, these self-curated skills can further improve test-time performance when used as in-prompt hints.

1 INTRODUCTION

Large Language Models (LLMs) (Achiam et al., 2023; Team et al., 2023; Yang et al., 2025) have demonstrated tremendous potential in perceiving, reasoning, and acting in open-ended environments, including embodied assistants that navigate simulated homes (Shridhar et al., 2020; Li et al., 2024; Huang et al., 2026), web navigators shopping online (Yao et al., 2022a; Zhou et al., 2023), and autonomous explorers in interactive games (Wang et al., 2023; Liu et al., 2024; Hu et al., 2024). These open-world tasks necessitate that LLMs autonomously interact with complex and dynamic environments, make long-horizon planning based on incomplete information, and deal with unexpected events. However, these advanced capabilities remain a challenge for LLMs, as agents must make a sequence of decisions with limited experience in such open-ended setting. Recent breakthroughs in Reinforcement Learning (RL) have proven that leveraging outcome reward signals can successfully facilitate the emergence of sophisticated capabilities for language-based agents to solve complex open-ended tasks (Lambert et al., 2024; Guo et al., 2025; Feng et al., 2025; Dong et al., 2025).

Despite the encouraging results, task execution in current agent systems is largely isolated, preventing LLMs from learning prior successful or failed experience across different trajectories and tasks (Xia et al., 2026). This, in effect, hinders their ability to evolve: policy updates rely solely on blind trial-and-error from scratch with sparse reward signals, while the reusable patterns remain unexploited. To address this, a growing line of work has explored the concept of *skills* (Anthropic, 2025; Xu & Yan, 2026; Li et al., 2026a; Ling et al., 2026), where previous interaction experiences are consolidated into reusable high-level procedures for reference. By organizing structured skills and retrieving them during training or inference, LLMs can transfer helpful behaviors across different tasks, which reduces redundant exploration and improves the generalization of open-ended reasoning and planning (Wu et al., 2025; Li et al., 2026b).

However, while effective, existing skill-based approaches often suffer from several fundamental limitations. Firstly, some methods (Anthropic, 2025; Ling et al., 2026; Han et al., 2026) rely on manually designed skills that require labor-intensive annotation tailored to specific environments,

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

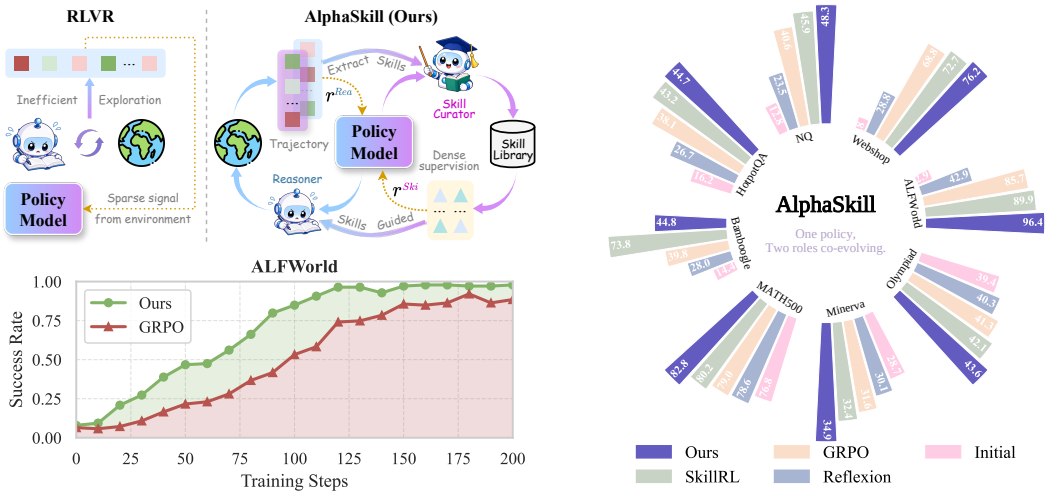

Figure 1: **(Left)** The comparison between RLVR and AlphaSkill. By training the policy to be a co-evolving `Reasoner` and `Skill Curator` that solves challenging tasks and curates helpful skills for itself, AlphaSkill outperforms the RLVR with faster improvements. **(Right)** Performance comparison with different methods on various agentic and reasoning tasks using Qwen2.5-7B-Instruct.

which introduces a critical scalability bottleneck. To alleviate this issue, subsequent studies (Yang et al., 2026; Xia et al., 2026; Ni et al., 2026) attempt to automatically extract skills using static LLMs. But as the policy's distribution shifts during training, such skills gradually become misaligned with the agent's improving capabilities. Finally, most current methods (Zhang et al., 2026a;b) treat skills as in-prompt hints, meaning the model is merely executing a skill contextually rather than memorizing the underlying knowledge and reusable competencies in skills. These limitations naturally raise a core question: ***Can the policy model self-curate skills to evolve and internalize them during training?*** If achievable, such a paradigm would transform RL from merely optimizing task-level outputs into explicitly learning the essence of how to accomplish relevant tasks.

In this work, we propose **AlphaSkill**, a unified framework to empower reinforcement learning with *self-curated* and *self-evolving* skills while simultaneously enhancing agentic reasoning. The core idea of AlphaSkill is to let a single policy play two roles that are jointly optimized: (1) a `Reasoner` that interacts with environments to solve open-ended tasks, and (2) a `Skill Curator` that reflects and distills reusable behavioral patterns based on successful and failed explorations. During co-evolving, the Reasoner is optimized with RL objective using task-level rewards from environmental feedback, whereas the Skill Curator is optimized with Supervise Fine-Tuning (SFT) objective over validated self-curated skills. We deem a skill valid if its injection into prompts leads to a higher success rate than before, and encourage the Skill Curator to internalize them for the generation of increasingly high-quality skills. To further facilitate the transition from explicit experience learning to implicit mastery in reasoning, we implement an inter-step decay to progressively decrease the weight of SFT loss during training. Overall, the reciprocal mechanism of AlphaSkill creates a virtuous self-evolving cycle, where the policy not only solves tasks but also continuously internalizes the underlying competencies grounded in real environments. Our main contributions are:

- Unlike prior methods that rely on manually designed or static skills, we explore a new paradigm with self-curated and self-evolving skills to strengthen agentic RL training, thereby co-evolving both sophisticated reasoning capability and reusable experience structures.
- We propose AlphaSkill, a unified framework that jointly optimizes LLMs to function as two roles: a `Reasoner` which interacts within environments to maximize the rewards, and a `Skill Curator` which extracts reusable patterns from explorations to further augment reasoning.
- Extensive experiments on diverse agentic environments and reasoning tasks demonstrate that AlphaSkill consistently improves task success rates over strong skill-based RL counterparts. Notably, when used as in-context hints, these self-curated skills can further reduce redundant exploration and improve test-time performance.

2 PRELIMINARIES

Let π_θ represent a policy model parameterized by θ. The post-training of Large Language Models (LLMs) aims to optimize the policy model to generate desirable responses for downstream tasks. Existing optimization paradigms can be primarily divided into two categories: Supervised Fine-Tuning (SFT), which learns from supervision on golden demonstrations; and Reinforcement Learning (RL), which learns from reward signals in environments.

2.1 SUPERVISED FINE-TUNING FOR MEMORIZATION

SFT adjusts the policy model to desired behaviors based on a set of golden demonstrations $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{N}$ with size N, where x denotes an input prompt and y denotes the corresponding desirable response. The policy model π_θ is optimized to minimize the negative log-likelihood (NLL) of golden demonstrations, which can be formulated as follows:

$$\mathcal{L}_{\text{SFT}}(\theta) = -\mathbb{E}_{(x,y)\sim\mathcal{D}}\left[\frac{1}{N}\sum_{t=1}^{N}\frac{1}{|y_i|}\sum_{t=1}^{T}\log \pi_\theta(y_{i,t} \mid x, y_{i,<t})\right], \tag{1}$$

where $y_{i,t}$ is the t-th token in the response y_i, and $y_{i,<t}$ represents the sequences before $y_{i,t}$. Minimizing the NLL objective encourages the model to mimic high-quality demonstrations and internalize structured behavioral patterns into the policy's parametric memory.

2.2 REINFORCEMENT LEARNING FOR REASONING

While SFT equips LLMs with rich knowledge, it does not necessarily enable the policy to effectively utilize such parametric knowledge for complex reasoning in open-ended environments. RL has therefore emerged as an effective paradigm for activating and refining latent capacity and knowledge in model parameters, enabling LLMs to progressively learn how to accomplish tasks through environmental feedback. The policy model π_θ is optimized to maximize the expected rewards of a group of G responses under question q. Specifically, we adopt Group Relative Policy Optimization (GRPO) (Shao et al., 2024) as our core RL algorithm, which can be formulated as follows:

$$\mathcal{L}_{\text{RL}}(\theta) = -\mathbb{E}_{\{y_i\}_{i=1}^{G}\sim\pi_{\theta_{\text{old}}}(\cdot|q)}\left[\frac{1}{G}\sum_{i=1}^{G}\frac{1}{|y_i|}\sum_{t=1}^{|y_i|}\min\left(\rho_{i,t}(\theta)\hat{A}_i,\ \text{clip}(\rho_{i,t}(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_i\right)\right], \tag{2}$$

where q denotes the task instruction, t denotes the token position within response y_i, $\rho_{i,t}(\theta) = \frac{\pi_\theta(y_{i,t}|q,y_{i,<t})}{\pi_{\theta_{\text{old}}}(y_{i,t}|q,y_{i,<t})}$ is the token-level importance ratio between the current policy and the previous policy. The group-relative advantage $\hat{A}_i$ for each response is computed by $\hat{A}_i = (r_i - \mu_G)/\sigma_G$, where μ_G and σ_G are the mean and standard deviation of rewards $\{r_i\}_{i=1}^{G}$ within the group respectively.

3 METHODOLOGY

In this section, we introduce **AlphaSkill**, a novel agentic reinforcement learning framework that bootstraps reasoning with self-evolving skills. In what follows, we first provide a high-level overview of the two roles co-evolving framework in Section 3.1. Then, we describe how the `Reasoner` interacts in environments and leverages skills to improve exploration in Section 3.2. After that, we discuss how the `Skill Curator` distills skills from previous attempts and self-evolves such skills in Section 3.3. Finally, we present the joint optimization of the policy under two roles in Section 3.4.

3.1 ONE POLICY WITH TWO ROLES CO-EVOLVING

The core idea of AlphaSkill is to go beyond blind trial-and-error from scratch in environments to explicitly guide its behaviors through self-evolving skills. Meanwhile, the capability of curating high-quality skills is evolved by tracking the actual utility of these skills during task execution.

To achieve this, we develop two complementary roles in one policy π_θ for co-evolution: a `Reasoner` π_θ^R for task execution, and a `Skill Curator` π_θ^S for extracting reusable high-level skills that can

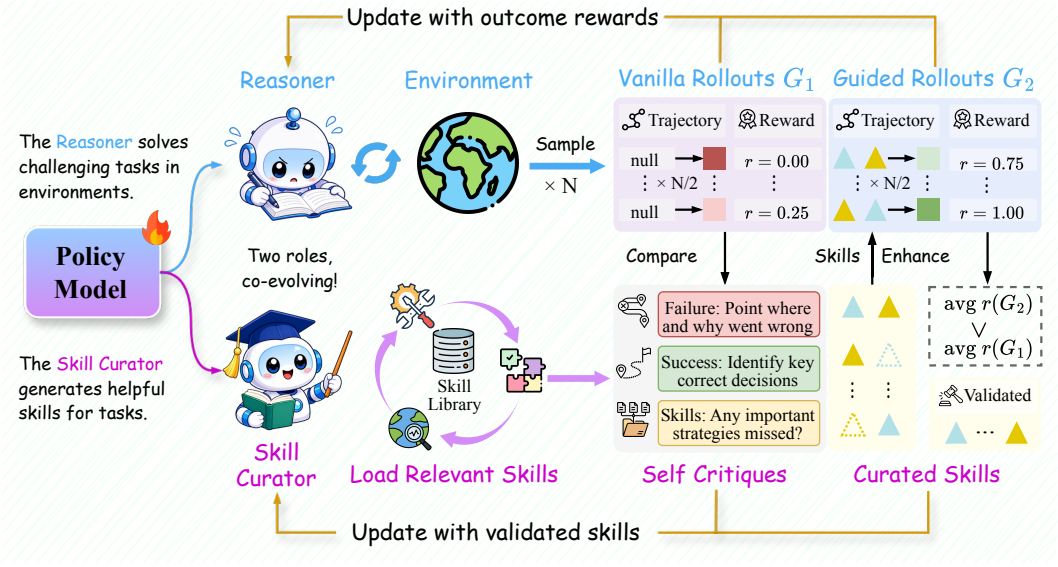

Figure 2: An overview of **AlphaSkill**, where a policy model π_θ acts in two roles: (1) a `Reasoner` π_θ^R which tries to successfully solve tasks in environments, and (2) a `Skill Curator` π_θ^S which refines or generates skills to help the task run more correctly based on self critiques. The Reasoner gets rewarded for solving tasks successfully, while the Skill Curator internalizes skills that improve the Reasoner's success rate, thereby enabling the co-evolution of the Reasoner and the Skill Curator.

facilitate future execution, as shown in Figure 2. Note that the Reasoner $\pi_\theta^R(\cdot) = \pi_\theta(\cdot|p^R)$ and the Skill Curator $\pi_\theta^S(\cdot) = \pi_\theta(\cdot|p^S)$ share the same policy parameter θ, while being distinguished through role-specific prompts p. In the following sections, we take a closer look at the different functions and optimization ways between the Reasoner π_θ^R and the Skill Curator π_θ^S within our framework.

3.2 Reasoner for Environment Exploration

The `Reasoner` π_θ^R is responsible for interacting with environments and solving tasks through sequential reasoning and acting. Different from traditional RL agents that rely solely on trial-and-error for exploration, it additionally augments with skills curated by the Skill Curator.

Rollouts. We follow the react (Yao et al., 2022b) framework, where LLMs alternates between reasoning and executing actions. For example, the system instruction guides the LLM to wrap its reasoning between tags `<think>` and action between tags `<action>`, respectively. The observation from environment is then enclosed within tags `<observation>` and appended to the rollout sequence, serving as additional context for the next generation step. The system prompts of different environments are in Appendix C.

For each task q, we will sample a group of N trajectories, denoted by $\mathcal{G}$. The original sampling group is evenly divided into a vanilla group $\mathcal{G}_1$ and a skills guided group $\mathcal{G}_2$, each expecting $N/2$ trajectories. The vanilla group first first samples $N/2$ trajectories $\{\tau_i\}_{i=0}^{N/2} \sim \pi_\theta^R(\cdot|q)$ without skills augmentation. Then trajectories in group $\mathcal{G}_1$ are subsequently analyzed by the Skill Curator, which curates several corresponding skills $\mathcal{S}_q$. After that, all curated skills are injected into the prompts of skill-guided group, enabling the Reasoner to explore conditioned on prior experience: $\{\tau_i\}_{i=N/2}^{N} \sim \pi_\theta^R(\cdot|q, \mathcal{S}_q)$. In this way, AlphaSkill establishes an explicit feedback loop between exploration and skill curation, where newly discovered experience can immediately facilitate future reasoning.

Optimization. Finally, Reasoner is optimized using GRPO loss $\mathcal{L}_{\text{RL}}^R$ in Eq. (2) based on task-level rewards to estimate advantages in a group. Notably, the two subgroups $\mathcal{G}_1$ and $\mathcal{G}_2$ are conditioned on different prompts, pooling them into a single group would mix rewards from different conditional distributions and bias the advantage baseline. Therefore, we compute advantages within each subgroup, ensuring that relative advantage is evaluated under consistent situation.

3.3 SKILL CURATOR FOR SELF-EVOLVING SKILLS

The `Skill Curator` π_θ^S is responsible for producing reusable skills from previous trajectories. Its goal is to explicitly extract the reusable behavioral pattern that will contribute to future successes.

Rollouts. The Skill Curator enables multi-turn interactions with actions `<load>`, `<reflect>`, `<modify>` and `<create>`. Specifically, for a received task q and its corresponding rollout group $\mathcal{G}_1$, the Skill Curator first loads at most $K = 3$ relevant skills $\mathcal{S} = \{s_1, \ldots, s_K\}$ by names within tag `<load>` from the skill library (initially empty) based on skills description. Then it performs diagnostic critiques $c \sim \pi_\theta^S(\cdot|q, \mathcal{S}, \tau^+, \tau^-)$ within tag `<reflect>` to identify the different strategies underlying successful trajectories τ^+ and failed trajectories τ^-. After that, it modifies loaded skills or creates a new skill within tags `<modify>` and `<create>` that can generalize across tasks to avoid failures based on self-critiques: $\mathcal{S}_q \sim \pi_\theta^S(\cdot|q, \mathcal{S}, \tau^+, \tau^-, c)$, where $\mathcal{S}_q$ denotes the set of skills newly created or updated according to task q. The prompt for Skill Curator is in Appendix C.

However, not all curated skills are beneficial. To ensure skills quality, we employ a validation mechanism based on downstream performance improvement. If the skills guided group $\mathcal{G}_2$ of the Reasoner under $\mathcal{S}_q$ achieves a higher success rate than its vanilla counterpart (*i.e.*, avg $r(\mathcal{G}_2) >$ avg $r(\mathcal{G}_1)$), the corresponding skills in $\mathcal{S}_q$ are regarded as valid and added into the skill library.

Optimization. While the two-role design enables a single policy to self-curate skills for behavioral guidance, we argue that such skills may get saturated: once the ability to self-curate skills lags behind the Reasoner's improving capability, it loses discriminative power over critical decision points in reasoning and no longer provide beneficial guidance. Therefore, we further propose to self-evolve the skill curation capability, so as to progressively produce more informative skills that mitigate saturation. Specifically, those validated skills also become supervision targets for the Skill Curator itself, where a fine-tuning based negative log-likelihood loss $\mathcal{L}_{\text{SFT}}^S$ in Eq. (1) is adopted to internalize the beneficial self-critiques and skills into the policy's parameters. Formally, the golden demonstration set of M validated skills is $\mathcal{D}_{\text{skills}} = \{(x_i, o_i)\}_{i=1}^M$, where the context $x = [q, \mathcal{S}, \tau^+, \tau^-]$ and the interaction trajectory $o = [c, \mathcal{S}_q]$. Such a supervision encourages the Skill Curator to continuously evolve its capability to keep pace with the Reasoner's dynamics, providing informative skills for guidance.

3.4 JOINT OPTIMIZATION OBJECTIVE

After computing the loss for the Reasoner and the Skill Curator, we jointly optimize them within a shared parameter space θ, where gradients from both roles are aggregated to learn from role-specific feedback while maintaining a unified policy. To further facilitate the transition from explicit experience learning to practical mastery in reasoning, we implement an inter-step decay to progressively decrease the weight of Skill Curator during training. With role-specific objective $\mathcal{L}_{\text{RL}}^R$ and $\mathcal{L}_{\text{SFT}}^S$, the final optimization objective can be formulated as $\mathcal{L}(\theta) = \mathcal{L}_{\text{RL}}^R + \alpha \cdot \mathcal{L}_{\text{SFT}}^S$, where the decay factor $\alpha = 0.95^s$ and s is training steps. Intuitively, the early training stage benefits from stronger skill supervision to internalize reusable behavioral priors distilled from interactions, whereas the later training stage gradually shifts toward incentivizing reasoning utilizing parametric knowledge, which facilitates the transition from explicit experience learning to practical mastery.

Overall, the joint optimization creates a virtuous self-evolving cycle: the policy not only improves task-solving capability but simultaneously producing increasingly informative skills, which enhance future exploration in turn. At inference time, the trained policy can solve tasks directly without explicit skill guidance. Alternatively, it can iteratively generate and refine skills from scratch based on trajectories collected during inference, thereby using these skills to guide subsequent rollouts. The whole algorithm procedure is presented in Appendix D.

4 EXPERIMENTS

4.1 EXPERIMENTAL SETUPS

Datasets and Benchmarks. We evaluate AlphaSkill on a variety of datasets that can be categorized into three different domains: (1) *Virtual world environments* including ALFWord (Shridhar et al., 2020) and Webshop (Yao et al., 2022a); (2) *Multi-hop QA* including NQ (Kwiatkowski et al., 2019), HotpotQA (Yang et al., 2018), and Bamboogle (Press et al., 2023); (3) *Mathematical reasoning* includ-

Table 1: Results overview of proposed AlphaSkill across different benchmarks. **Bold** represents the best performance, and Δ denotes the performance gain compared with GRPO.

Method	Virtual World		Multi-hop QA			Mathematical Reasoning			
	ALFWorld	Webshop	NQ	HotpotQA	Bamboogle	MATH500	Olympiad	Minerva	AIME25
Model I: Qwen2.5-7B-Instruct									
Initial	12.9	7.8	12.8	16.2	14.4	76.8	39.4	28.7	6.7
GRPO	85.7	68.8	40.6	38.1	39.8	79.0	41.3	31.6	10.0
AlphaSkill	**96.4**	**76.2**	**48.3**	**44.7**	**44.8**	**82.8**	**43.6**	**34.9**	**10.0**
Δ	+10.7	+7.4	+7.7	+6.6	+5.0	+3.8	+2.3	+3.3	+0.0
Model II: Qwen3-4B-Instruct-2507									
Initial	25.7	19.2	19.6	19.9	14.4	88.0	56.6	41.9	40.0
GRPO	92.9	74.4	42.3	39.6	40.8	**89.4**	58.2	43.4	40.0
AlphaSkill	**98.6**	**79.2**	**51.1**	**44.2**	**46.4**	88.8	**60.1**	**46.0**	**43.3**
Δ	+5.7	+4.8	+8.8	+4.6	+5.6	-0.6	+1.9	+2.6	+3.3

Table 2: Performance comparison with baseline methods using Qwen2.5-7B-Instruct in virtual world environments. Best results among all methods are marked as **bold**.

Method	ALFWorld							Webshop	
	Pick	Look	Clean	Heat	Cool	Pick2	All	Score	Succ.
Initial	31.4	21.6	18.5	6.90	2.80	3.20	12.9	26.4	7.80
Reflexion	62.0	41.6	44.9	30.9	36.3	23.8	42.9	58.1	28.8
Mem0	54.0	55.0	26.9	36.4	20.8	7.69	33.6	23.9	2.00
GRPO	94.3	84.6	88.9	81.3	84.0	75.0	85.7	81.5	68.8
SkillRL	97.9	71.4	90.0	90.0	95.5	87.5	89.9	85.2	72.7
Skill0	100	85.8	94.6	81.9	85.7	80.1	89.8	85.3	73.2
AlphaSkill (Ours)	**100**	**92.3**	**96.3**	**87.5**	**100**	**95.8**	**96.4**	**86.3**	**76.2**

Table 3: Performance comparison with baseline methods using Qwen2.5-7B-Instruct in multi-hop QA and mathematical reasoning. Best results among all methods are marked as **bold**.

Multi-hop QA				Mathematical Reasoning				
Method	NQ	HotpotQA	Bamboogle	Method	MATH500	Minerva	Olympiad	AIME25
Initial	12.8	16.2	14.4	Initial	76.8	28.7	39.4	6.7
Search-R1	39.3	37.0	36.8	Fewshot	77.6	29.0	40.3	6.7
ZeroSearch	43.6	34.6	28.0	AbsoluteZero	78.2	33.8	41.9	6.7
SkillRL	45.9	43.2	**73.8**	GRPO	79.0	31.6	41.3	10.0
Skill0	42.7	40.0	66.9	Critique-GRPO	81.2	33.1	42.4	**13.3**
AlphaSkill (Ours)	**48.3**	**44.7**	44.8	**AlphaSkill (Ours)**	**82.8**	**34.9**	**43.6**	10.0

ing MATH500 (Lightman et al., 2023), Olympiad Bench (He et al., 2024), MinervaMath (Lewkowycz et al., 2022) and AIME25[1]. During evaluation, all methods are provided only with the original task instructions, without injecting any skills into prompts. For training virtual worlds, we employ the training set of ALFWorld and Webshop respectively. For training multi-hop QA, we employ the training set of NQ and HotpotQA with e5 (Wang et al., 2022) as the retriever. For training mathematical reasoning, we employ DAPO-Math-17K (Yu et al., 2025) and NuminaMath-CoT (LI et al., 2024) as our training set. The detailed dataset information is provided in Appendix B.

Models and Baselines. We use two models from different series and parameter scales for our experiments, including the Qwen2.5 series (Qwen2.5-7B-Instruct) (Yang et al., 2024) and Qwen3 series (Qwen3-4B-Instruct-2507) (Yang et al., 2025). We include Qwen2.5 as a relatively earlier model family, which reduces the risk of contamination from these agentic benchmarks, thereby providing a cleaner testbed for evaluating the effectiveness of our method. We compare AlphaSkill against representative baseline methods in different domains. In virtual world environments, we include a prompt-based baseline Reflexion (Shinn et al., 2023), a memory-based baseline Mem0 (Chhikara et al., 2025), and RL-based baseline GRPO (Shao et al., 2024), SkillRL (Xia et al., 2026) and Skill0 (Lu et al., 2026). In multi-hop QA, we include the baseline Search-R1 (Jin et al., 2025), ZeroSearch (Sun

[1]https://artofproblemsolving.com/wiki/index.php/AIME

Table 4: Results of ablations on self-evolving mechanisms using Qwen2.5-7B-Instruct.

Method	Virtual World		Multi-hop QA			Mathematical Reasoning			
	ALFWorld	Webshop	NQ	HotpotQA	Bamboogle	MATH500	Minerva	Olympiad	AIME25
w/o evolving	90.0	72.8	45.2	43.5	43.2	77.6	31.6	41.9	10.0
w/o decaying	92.1	75.6	46.7	43.1	46.4	79.8	32.4	42.1	10.0
AlphaSkill	96.4	76.2	48.3	44.7	44.8	82.8	34.9	43.6	10.0

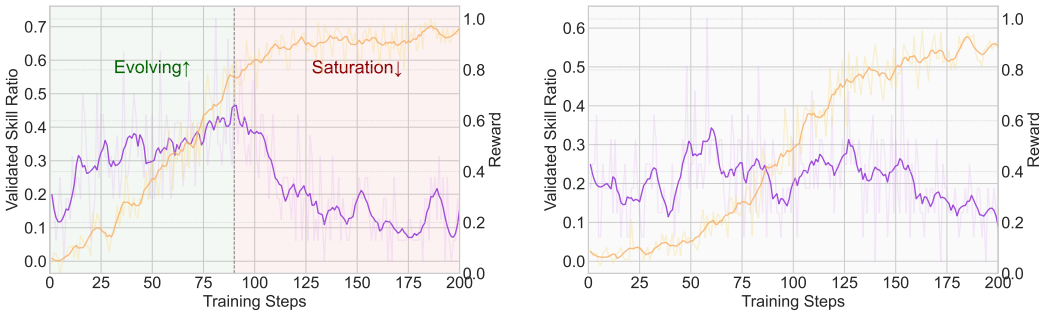

Figure 3: The training dynamics (Rewards and Validated skill percentage) of AlphaSkill (Left) and AlphaSkill w/o self-evolving skills (Right).

et al., 2025), SkillRL (Xia et al., 2026) and Skill0 (Lu et al., 2026). In mathematical reasoning, we include the baseline Fewshot-CoT, GRPO (Shao et al., 2024), and Critique-GRPO (Zhang et al., 2025). All models are trained based on the verl framework (Sheng et al., 2025). The detailed training hyperparameters are provided in Appendix A.

4.2 Main Results

We analyze the effectiveness of AlphaSkill by investigating the following research questions.

> **RQ1:** How does AlphaSkill enhance the agentic capabilities of LLMs?

To answer this question, we compare AlphaSkill with the vanilla GRPO that only employs the outcome reward, as well as several widely adopted baselines across different domains.

AlphaSkill consistently outperforms outcome-based GRPO across tasks. As shown in Table 1, AlphaSkill demonstrates promising gains across different training tasks and model scales, showcasing its effectiveness and broad generalization. Notably, the performance gains become more pronounced on larger models, with the 7B model exhibiting greater relative performance improvements than the 4B model. These results validate that self-curated skills can effectively accelerate policy optimization and improve learning efficiency in sparse-reward environments.

AlphaSkill yields superior results than domain-specific counterparts. As shown in Table 2 and Table 3, AlphaSkill outperforms prompt-based methods and training-based methods. On Qwen2.5-7B-Instruct, AlphaSkill consistently achieves performance gains of 2-4% over strongest counterparts across different domains, demonstrating the broad applicability of our self-evolving training loop.

> **RQ2:** How does the self-evolving enable the Skill Curator to gradually provide better skills?

To investigate this point, we conduct ablation studies on Qwen2.5-7B-Instruct by removing the skills evolving objective $\mathcal{L}^{S}_{\text{SFT}}$, and the policy is only optimized with the outcome reward using $\mathcal{L}^{R}_{\text{RL}}$. We report the training dynamics (*i.e.*, the ratio of validated skills and rewards) in Figure 3 and the performance across tasks in Table 4. Our key observation is that *self-evolving skills enable better performance and more informative skills*:

- **Observation 1**: By optimization with $\mathcal{L}^{S}_{\text{SFT}}$, AlphaSkill exhibits a faster reward learning curve in Figure 3 while outperforming the ablation version (*i.e.*, w/o evolving) across tasks as shown in Table 4. Such results validate the effectiveness of self-evolving skills in enhancing reasoning.

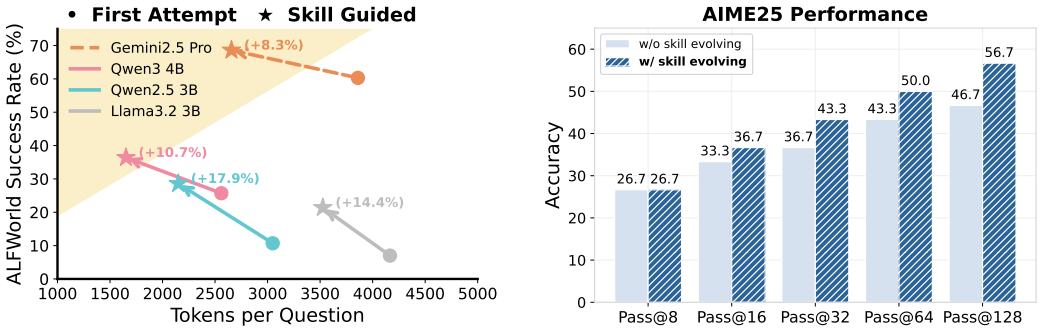

Figure 4: AlphaSkill can further facilitate test time self-evolving. **(Left)** When using curated skills during training as in-context hints, they can further induce effective task execution while improving token efficiency. **(Right)** The models trained by AlphaSkill can self-curated and self-evolving skills from zero, leading to higher capacity upper bound at test time.

- **Observation 2**: More interestingly, we find that in the later training phase, the proportion of validated skills in both the original and ablation versions exhibit a decreasing trend, as shown in Figure 3. This phenomenon likely arises as the policy model has already developed relatively strong reasoning abilities, thereby reducing the marginal utility of additional skills. However, AlphaSkill exhibits a steadily increasing validated skill ratio throughout the early-to-mid training phase, whereas the ablated variant continuously fluctuates around a lower level. These findings further highlight the critical role of self-evolving mechanisms in curating better skills.

As discussed in Observation 2, the guidance provided by skills gradually diminishes as the policy approaches performance saturation. This naturally motivates our design of the inter-step decay mechanism for the Skill Curator. To evaluate its effectiveness, we conduct an ablation study on this component. As shown in Table 4, removing the decay mechanism (*i.e.*, w/o decaying) leads to a slight degradation in final performance. Such results indicate that the decay mechanism plays a role in facilitating the model's transition from learning to better mastery.

RQ3: How can the self-evolving skills mechanism facilitate LLM reasoning at test time?

Here, we attempt to understand how the self-evolving skills can push the model toward a higher reasoning bound during test time. To this end, we solve the following two sub-questions.

How effective are accumulated skills during training when used as in-context hints at the test time? To this end, we explicitly employ these self-curated skills accumulated during training as in-context hints, and evaluate multiple initial models' (without training) reasoning performance under this setup. We demonstrate the success rate on ALFWorld using Qwen2.5-3B-Instruct (Yang et al., 2024), Llama-3.2-3B-Instruct (Dubey et al., 2024), Qwen3-4B-Instruct-2507 (Yang et al., 2025) and Gemini2.5-Pro (Comanici et al., 2025). As shown in Figure 4 left, when using accumulated skills as in-context hints, the performance improves with fewer generated tokens, suggesting that these skills can guide better reasoning and reduce redundant exploration.

Can LLMs trained by AlphaSkill self-evolve from zero skill at the test time? To this end, we use Qwen2.5-7B-Instruct trained after AlphaSkill to self-evolve skills from zero at the test time. Specifically, the model continuously interacts within environments and self-curate reusable skills from every eight rollouts of each task, similar to the training procedure. We compare this setting against the standard pass@k evaluation protocol. In standard pass@k, the model independently samples k trajectories from scratch for each task without generating or refining any reusable skills across tasks. In contrast, our test-time self-evolution setting allows the model to gradually accumulate experience across previous trajectories and iteratively evolve the skill library, which is subsequently injected to guide future explorations. As shown in Figure 4 right, AlphaSkill pushes the model toward higher pass@k performance than standard sampling. These results suggest that the model is capable of distilling online experience to progressively improve reasoning efficiency and task success without independent trial-and-error during inference, ultimately achieving test-time self-evolution.

4.3 Analysis and Discussion

Skills Visualization. To further analyze the characteristics of skills, we project the TF-IDF features of the curated skills throughout training on ALFWorld using UMAP, as illustrated in Figure 5 left. As expected, skills originating from semantically similar task categories form clear clusters in the embedding space. In contrast, tasks with low semantic relevance exhibit minimal overlap (*e.g.*, the task of *heat* and *look*). These results demonstrate both the diversity of the self-evolving skills and their strong alignment with underlying task semantics.

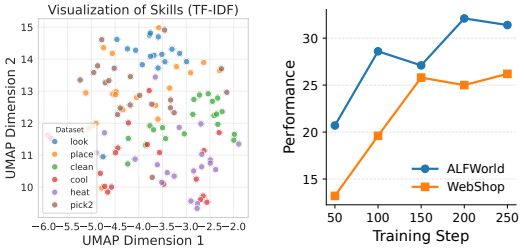

Figure 5: Left: UMAP projection of TF-IDF feature distributions for skills. Right: Performance with the evolving skill library over training steps.

Skills Quality Improvement. To examine whether the quality of the curated skills improves throughout training, we evaluate a frozen Qwen2.5-7B-Instruct model using snapshots of the skill library collected at different training steps. As shown in Figure 5 right, the skills in the later training steps provide greater utility than early skills at test time.

5 Related Work

Agentic Memory and Skills. Large Language Models (LLMs) have transitioned from static question-answering systems to autonomous agents capable of executing real-world tasks. However, they often struggle to generalize across tasks or retain information over extended interactions, as they treat every interaction independently and repeat trial-and-error from scratch. To overcome this, previous attempts augment LLMs with external memory modules that store interaction history to enable long-horizon reasoning (Packer et al., 2023; Zhong et al., 2024; Yan et al., 2025; Wang et al., 2025). However, such interactions are often redundant and noisy, making it difficult for LLMs to extract underlying principles that can enhance reasoning. To bridge this gap, a growing line of work explores the concept of *skills* (Anthropic, 2025; Xu & Yan, 2026; Li et al., 2026a; Ling et al., 2026), where previous interaction trajectories are distilled into reusable high-level procedures or behavioral patterns for reference. More recently, some works like SkillRL (Xia et al., 2026) and Skill0 (Lu et al., 2026) further demonstrate that incorporating skills into reinforcement learning framework can provide effective guidance. Despite these advances, most existing skill-based methods (Zhang et al., 2026a; Ni et al., 2026; Anthropic, 2025) maintain fixed skill libraries or use non-adaptive LLMs to automatically generate skills, which may not keep pace with the agent's improving capabilities.

Self-evolving methods. Self-evolving has revolutionized game-playing AI, such as AlphaGo's superhuman performance (Silver et al., 2017) and CICERO's comprehension of cooperative strategies (, FAIR). The paradigm of self-evolving, where models take on dual roles to create a self-improvement loop (Yuan et al., 2024; Chen et al., 2024) has recently been adapted to improve language models. Recent works in the self-evolving paradigm demonstrate that a single policy model in reinforcement learning can play different roles (like solver and questioner) to create a self-improvement loop from zero data (Liu et al., 2025; Zhao et al., 2025; Huang et al., 2025; Fang et al., 2025). These advances suggest the possibilities of self-curating skills for self-improvement, which remain largely unexplored.

6 Conclusion

This work introduces AlphaSkill, a unified framework to empower reinforcement learning with *self-curated* and *self-evolving* skills while simultaneously enhancing agentic reasoning. Technically, AlphaSkill enables one policy to function as a Reasoner that solves tasks in environments, and a Skill Curator that curates skills to benefit future successful execution. By co-evolving two roles during training, the reusable knowledge in self-curated skills is inherently fused into its exploration process, resulting in robust task execution. Experiments on various agentic and reasoning tasks demonstrate that AlphaSkill consistently outperforms RLVR across benchmarks and model backbones, highlighting that such self-curated and self-evolving skills can provide meaningful guidance in pushing the upper bound of agentic reasoning during training.

AI USE STATEMENT

In this work, we used AI to improve the readability and presentation of the manuscript, plus checks for compliance with submission formatting requirements. We have not used generative AI tools to generate datasets, develop theoretical or conceptual frameworks, propose or refine hypotheses, design the research methodology or experiments, or interpret experimental results. All AI-assisted text was carefully reviewed and revised by the authors. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

We provide experimental setups and implementation details in Section 4 and Appendix A, respectively. We further provide the algorithm procedure in Appendix D to facilitate reproducibility. Our code is available at https://anonymous.4open.science/r/Askill-81D0.

REFERENCES

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Anthropic. Agent skills overview. https://platform.claude.com/docs/en/agents-and-tools/agent-skills/overview, 2025.

Zixiang Chen, Yihe Deng, Huizhuo Yuan, Kaixuan Ji, and Quanquan Gu. Self-play fine-tuning converts weak language models to strong language models. *arXiv preprint arXiv:2401.01335*, 2024.

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.

Guanting Dong, Hangyu Mao, Kai Ma, Licheng Bao, Yifei Chen, Zhongyuan Wang, Zhongxia Chen, Jiazhen Du, Huiyang Wang, Fuzheng Zhang, et al. Agentic reinforced policy optimization. *arXiv preprint arXiv:2507.19849*, 2025.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Meta Fundamental AI Research Diplomacy Team (FAIR)†, Anton Bakhtin, Noam Brown, Emily Dinan, Gabriele Farina, Colin Flaherty, Daniel Fried, Andrew Goff, Jonathan Gray, Hengyuan Hu, et al. Human-level play in the game of diplomacy by combining language models with strategic reasoning. *Science*, 378(6624):1067–1074, 2022.

Wenkai Fang, Shunyu Liu, Yang Zhou, Kongcheng Zhang, Tongya Zheng, Kaixuan Chen, Mingli Song, and Dacheng Tao. Serl: Self-play reinforcement learning for large language models with limited data. *arXiv preprint arXiv:2505.20347*, 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.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Tingxu Han, Yi Zhang, Wei Song, Chunrong Fang, Zhenyu Chen, Youcheng Sun, and Lijie Hu. Swe-skills-bench: Do agent skills actually help in real-world software engineering? *arXiv preprint arXiv:2603.15401*, 2026.

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, et al. Olympiadbench: A challenging benchmark for promoting agi with olympiad-level bilingual multimodal scientific problems. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3828–3850, 2024.

Sihao Hu, Tiansheng Huang, Gaowen Liu, Ramana Rao Kompella, Fatih Ilhan, Selim Furkan Tekin, Yichang Xu, Zachary Yahn, and Ling Liu. A survey on large language model-based game agents. *arXiv preprint arXiv:2404.02039*, 2024.

Chengsong Huang, Wenhao Yu, Xiaoyang Wang, Hongming Zhang, Zongxia Li, Ruosen Li, Jiaxin Huang, Haitao Mi, and Dong Yu. R-zero: Self-evolving reasoning llm from zero data. *arXiv preprint arXiv:2508.05004*, 2025.

Helong Huang, Min Cen, Kai Tan, Xingyue Quan, Guowei Huang, and Hong Zhang. Graphcot-vla: A 3d spatial-aware reasoning vision-language-action model for robotic manipulation with ambiguous instructions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 18324–18332, 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.

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.

Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James V Miranda, Alisa Liu, Nouha Dziri, Shane Lyu, et al. Tulu 3: Pushing frontiers in open language model post-training. *arXiv preprint arXiv:2411.15124*, 2024.

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, et al. Solving quantitative reasoning problems with language models. *Advances in neural information processing systems*, 35:3843–3857, 2022.

Hao Li, Chunjiang Mu, Jianhao Chen, Siyue Ren, Zhiyao Cui, Yiqun Zhang, Lei Bai, and Shuyue Hu. Organizing, orchestrating, and benchmarking agent skills at ecosystem scale. *arXiv preprint arXiv:2603.02176*, 2026a.

Jia LI, Edward Beeching, Lewis Tunstall, Ben Lipkin, Roman Soletskyi, Shengyi Costa Huang, Kashif Rasul, Longhui Yu, Albert Jiang, Ziju Shen, Zihan Qin, Bin Dong, Li Zhou, Yann Fleureau, Guillaume Lample, and Stanislas Polu. Numinamath. https://huggingface.co/AI-MO/NuminaMath-CoT, 2024.

Manling Li, Shiyu Zhao, Qineng Wang, Kangrui Wang, Yu Zhou, Sanjana Srivastava, Cem Gokmen, Tony Lee, Li E Li, Ruohan Zhang, et al. Embodied agent interface: Benchmarking llms for embodied decision making. *Advances in Neural Information Processing Systems*, 37:100428–100534, 2024.

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*, 2026b.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *The twelfth international conference on learning representations*, 2023.

George Ling, Shanshan Zhong, and Richard Huang. Agent skills: A data-driven analysis of claude skills for extending large language model functionality. *arXiv preprint arXiv:2602.08004*, 2026.

Bo Liu, Chuanyang Jin, Seungone Kim, Weizhe Yuan, Wenting Zhao, Ilia Kulikov, Xian Li, Sainbayar Sukhbaatar, Jack Lanchantin, and Jason Weston. Spice: Self-play in corpus environments improves reasoning. *arXiv preprint arXiv:2510.24684*, 2025.

Shunyu Liu, Yaoru Li, Kongcheng Zhang, Zhenyu Cui, Wenkai Fang, Yuxuan Zheng, Tongya Zheng, and Mingli Song. Odyssey: Empowering minecraft agents with open-world skills. *arXiv preprint arXiv:2407.15325*, 2024.

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.

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.

Charles Packer, Vivian Fang, Shishir_G Patil, Kevin Lin, Sarah Wooders, and Joseph_E Gonzalez. Memgpt: towards llms as operating systems. *arXiv preprint arXiv:2310.08560*, 2023.

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.

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 *EuroSys*, pp. 1279–1297, 2025.

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, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, et al. Mastering chess and shogi by self-play with a general reinforcement learning algorithm. *arXiv preprint arXiv:1712.01815*, 2017.

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.

Gemini Team, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, Katie Millican, et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.

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.

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.

Yu Wang, Ryuichi Takanobu, Zhiqi Liang, Yuzhen Mao, Yuanzhe Hu, Julian McAuley, and Xiaojian Wu. Mem-{\alpha}: Learning memory construction via reinforcement learning. *arXiv preprint arXiv:2509.25911*, 2025.

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.

Renjun Xu and Yang Yan. Agent skills for large language models: Architecture, acquisition, security, and the path forward. *arXiv preprint arXiv:2602.12430*, 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, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

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*, 2025.

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*, 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 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.

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. *arXiv preprint arXiv:2503.14476*, 2025.

Weizhe Yuan, Richard Yuanzhe Pang, Kyunghyun Cho, Xian Li, Sainbayar Sukhbaatar, Jing Xu, and Jason Weston. Self-rewarding language models. *arXiv preprint arXiv:2401.10020*, 2024.

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*, 2026a.

Haozhen Zhang, Quanyu Long, Jianzhu Bao, Tao Feng, Weizhi Zhang, Haodong Yue, and Wenya Wang. Memskill: Learning and evolving memory skills for self-evolving agents. *arXiv preprint arXiv:2602.02474*, 2026b.

Xiaoying Zhang, Yipeng Zhang, Hao Sun, Kaituo Feng, Chaochao Lu, Chao Yang, and Helen Meng. Critique-grpo: Advancing llm reasoning with natural language and numerical feedback. *arXiv preprint arXiv:2506.03106*, 2025.

Andrew Zhao, Yiran Wu, Yang Yue, Tong Wu, Quentin Xu, Matthieu Lin, Shenzhi Wang, Qingyun Wu, Zilong Zheng, and Gao Huang. Absolute zero: Reinforced self-play reasoning with zero data. *arXiv preprint arXiv:2505.03335*, 2025.

Wanjun Zhong, Lianghong Guo, Qiqi Gao, He Ye, and Yanlin Wang. Memorybank: Enhancing large language models with long-term memory. In *Proceedings of the AAAI conference on artificial intelligence*, volume 38, pp. 19724–19731, 2024.

Shuyan Zhou, Frank F Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, et al. Webarena: A realistic web environment for building autonomous agents. *arXiv preprint arXiv:2307.13854*, 2023.

A Implementation Details

All experiments run on 8×A100-80GB GPUs. We use the verl (Sheng et al., 2025) framework for RL training. For ALFWorld and Webshop, we sample 16 different tasks per batch and 8 rollouts per prompt, with a maximum response length of 4,096 tokens. For multi-hop QA, we sample 128 tasks and 8 rollouts per prompt, with a maximum response length of 4,096 tokens. For mathematical reasoning, we sample 128 questions and 8 rollouts per prompt, with a maximum response length of 6,144 tokens. For other general hyperparameters, we present in Table 5.

Table 5: Training configuration of AlphaSkill.

Category	Configuration
Sampling	Temperature: 1.0 Top-P: 1.0, Top-K: -1 Rollout Samples per Prompt: 8
Training	Optimizer: Adam Learning Rate: 1×10^{-6} (Constant) Mini Batch Size: 16 KL Loss Coefficient: 1×10^{-3} Entropy Coefficient: 0 Steps: 200
Max Turns	35 (ALFWorld), 15 (Webshop), 4 (Multi-hop QA)

B Dataset Information

Detailed statistics regarding data splits and evaluation protocols for all benchmarks are summarized in Table 6. In the following, we introduce each benchmark in detail.

Table 6: Statistics of evaluation datasets across three domains.

Dataset	**# Train**	**# Test**	**Domain**
Virtual Environments			
ALFWorld Shridhar et al. (2020)	3,553	140	Virtual World
Webshop Yao et al. (2022a)	6000	500	Virtual World
Multi-hop Question Answering			
NQ Kwiatkowski et al. (2019)	87,925	3,610	Multi-hop QA
HotpotQA Yang et al. (2018)	90,447	7,405	Multi-hop QA
Bamboogle Press et al. (2023)	–	125	Multi-hop QA
Mathematical Reasoning			
MATH500 Lightman et al. (2023)	–	500	Mathematics
Olympiad Bench He et al. (2024)	–	675	Mathematics
MinervaMath Lewkowycz et al. (2022)	–	272	Mathematics
AIME2025[1]	–	30	Mathematics

Virtual Environments. These benchmarks evaluate agents in text-based interactive environments that simulate real-world tasks.

- **ALFWorld** Shridhar et al. (2020) is a unified simulator that enables agents to learn abstract text-based policies in a symbolic environment and subsequently execute the same goals in a visually grounded setting. The framework supports six household task types including picking up, examining, cleaning, heating, cooling, and placing objects, requiring agents to translate high-level plans into concrete, scene-adapted action sequences. We train on 3,553 tasks and evaluate on 134 seen and 140 unseen test environments to assess cross-domain

generalization. The modular design factors language understanding, planning, navigation, and visual perception into decoupled components, facilitating targeted improvements and fostering better generalization than training solely in visually grounded environments.

- **WebShop** Yao et al. (2022a) is a simulated e-commerce environment comprising 1.18 million real-world products and 12,087 crowd-sourced text instructions that specify diverse product requirements. Agents must navigate heterogeneous webpages, issue structured actions to search, filter, and compare items, and ultimately purchase products that satisfy compositional user constraints. The benchmark presents multiple language grounding challenges including query reformulation, reasoning over noisy webpage text, and strategic exploration under partial observability. We train on over 6,000 interactive sessions and evaluate on 500 held-out test queries to assess robust decision-making and sim-to-real transfer capability.

Multi-hop Question Answering. These datasets require models to retrieve and integrate information from multiple sources to answer complex questions.

- **NQ** Kwiatkowski et al. (2019) is an open-domain QA dataset constructed from real anonymized Google search queries. For each query, human annotators review a Wikipedia article from the top five search results to extract a paragraph-level long answer and an entity-level short answer, or label the instance as unanswerable. The official corpus comprises 307,373 singly annotated training examples alongside 7,830 development and 7,842 test examples with five-way annotations. Following standard preprocessing, we adopt the widely used subset containing 87,925 training and 3,610 test instances for evaluation.
- **HotpotQA** Yang et al. (2018) is a benchmark of 113,000 Wikipedia-based question-answer pairs designed to evaluate complex multi-hop reasoning. The questions span diverse topics without relying on predefined knowledge schemas, requiring models to synthesize information across multiple supporting documents. Each example includes sentence-level supporting facts that provide strong supervision and enable models to produce explainable predictions. The dataset further incorporates factoid comparison questions to test cross-document information extraction. We follow the official distractor setting and use the standard split with 90,447 training and 7,405 test examples.
- **Bamboogle** Press et al. (2023) is a diagnostic benchmark comprising 125 multi-hop questions designed to quantify the compositionality gap in language models. The tasks require models to synthesize answers from independent sub-problems using factual combinations unlikely to co-occur during pretraining. Each instance measures how frequently models correctly resolve all intermediate steps yet fail to produce the final composed answer, thereby distinguishing genuine compositional reasoning from pattern memorization. The dataset serves as the standard evaluation suite for elicitive prompting techniques such as chain-of-thought and the self-ask framework, which explicitly guide models to decompose and resolve complex queries.

Mathematical Reasoning. These benchmarks assess multi-step symbolic reasoning abilities, ranging from high-school mathematics to competition-level problems.

- **MATH500** Lightman et al. (2023) is a carefully curated subset of 500 mathematics problems drawn from the original MATH benchmark, covering algebra, geometry, number theory, and counting/probability. Each problem requires multi-step symbolic derivation with a precise final answer, making it a clean and efficient benchmark for evaluating chain-of-thought reasoning and process-level supervision. The subset is designed to be representative of the full test distribution while enabling rapid iteration during model development. We adopt the complete set of 500 examples for evaluation, following recent work on process supervision and reward modeling that has established MATH500 as a standard testbed for high-school to competition-level mathematical reasoning.
- **OlympiadBench** He et al. (2024) is a bilingual and multimodal scientific benchmark comprising Olympiad-level problems in mathematics and physics, including selections from the Chinese college entrance examination. Each problem is annotated with expert-level step-by-step reasoning traces, enabling fine-grained evaluation of both final answers and

intermediate derivations. The benchmark employs a comprehensive assessment methodology to accurately score model responses across open-ended and closed-form tasks. We evaluate on a curated subset of 675 test examples, which maintains the original distribution of difficulty and topic coverage while enabling efficient benchmarking of advanced reasoning capabilities.

- **MinervaMath** Lewkowycz et al. (2022) comprises undergraduate-level problems in mathematics and the sciences, including physics, chemistry, biology, and economics, that require quantitative reasoning and multi-step derivation. Each problem is paired with a solution expressed in formal mathematical notation, enabling rigorous evaluation of a model's ability to generate verifiable, step-by-step reasoning. The original benchmark contains over two hundred examples across diverse scientific domains. We evaluate on a curated subset of 272 test instances that preserves the distribution of topics and difficulty levels, following the protocol established in prior work on process-supervised reasoning.
- **AIME2025**[2] is drawn from the American Invitational Mathematics Examination, the second stage in the United States Mathematics Olympiad selection pipeline administered by the Mathematical Association of America. The examination consists of fifteen free-response problems per session to be solved within three hours without calculator assistance, where each answer is a unique integer between zero and nine hundred ninety-nine. Questions span arithmetic, algebra, counting, geometry, number theory, and probability, typically increasing in difficulty and requiring creative synthesis of secondary-school mathematical concepts. We evaluate on the full test set of 30 examples from the 2025 administration to assess robust multi-step reasoning under competition-level constraints.

C Prompt Templates

System prompt for ALFWorld

You are an agent interacting with a virtual text-based environment.

Notes:
At each step, you should first think then perform action to fulfill the instruction. You should ALWAYS wrap your thinking with the <think> </think> tag and wrap your action with the <action> </action> tag.
You should ALWAYS take one action each step.
DO NOT try to interact with the user at anytime. Finish the task by yourself.

Action Format:
Below are the available commands you can use:
look: look around your current location
inventory: check the object you are currently holding (you can only hold one)
go to (receptacle): move to a receptacle
open (receptacle): open a receptacle
close (receptacle): close a receptacle
take (object) from (receptacle): take an object from a receptacle
move (object) to (receptacle): place an object that you are holding in or on a receptacle
examine (something): examine a receptacle or an object to learn its properties
use (object): use an object
heat (object) with (receptacle): heat an object using a receptacle
clean (object) with (receptacle): clean an object using a receptacle
cool (object) with (receptacle): cool an object using a receptacle
slice (object) with (object): slice an object using a sharp object

For example your output should be like this:
<think> To solve the task, I need first to ... </think><action>go to cabinet 1</action>

[2] https://artofproblemsolving.com/wiki/index.php/AIME

System prompt for Webshop

You are an agent interacting with a virtual text-based web shopping environment to test out your ability. Your job is to follow the Instruction provided and mimic the steps to buy the item that are closest to the Instruction provided.

Action Format:
You should give both the action_name and action_arg like the format 'action_name[action_arg]'. You can execute two types of actions, search and click.
- When the button '[button] Search [button_]' is available in the current observation, you can execute the action <action>search[xxx]</action> (you should type the query you want to search in the square brackets here).
- You can click buttons '[button] xxx [button_]' that is available in the current observation, by executing the action <action>click[xxx]</action>.

Below are some examples of action formats.
- <action>search[white shoes]</action>
- <action>click[Buy Now]</action>

Notes:
At each step, you should first think then perform action to fulfill the instruction. You should ALWAYS wrap your thinking with the <think> </think> tag and wrap your action with the <action> </action> tag.
You should ALWAYS take one action each step
. You should finish the task and buy the item within 15 steps.
DO NOT try to interact with the user at anytime. Finish the task and buy the item by yourself.

System prompt for Multi-hop QA

Answer the given question. You must conduct reasoning inside <think> and </think> first every time you get new information. After reasoning, if you find you lack some knowledge, you can call a search engine by <tool_call> query </tool_call> and it will return the top searched results between <tool_response> and </tool_response>. You can search as many times as your want. If you find no further external knowledge needed, you can directly provide the answer inside <answer> and </answer>, without detailed illustrations. For example, <answer> Beijing </answer>. Question: {question}

System prompt for Mathematical Reasoning

Let's think step by step and output the final answer within boxed{}.

System prompt for Skill Curator

— User turn 1 —
You are an expert at distilling agent behavior patterns into concise, actionable skills.

Task
{task}

Existing Skills
{skills}

First load **1 to 3** relevant skills related to this task. You should use commas to separate each selected skill name and wrap them in <load> </load> tags. For example, your output should be like this:
<load>skill_name1, skill_name2, ...</load>

— User turn 2 —
Loaded Skills
{loaded}

SUCCEESSES
{success}
FAILURES
{failure}

Analyze these successful and failed trajectories from an agent operating in the environments and extract generalizable experiences. Consider these aspects:
- For successful steps: Identify key correct decisions and insights
- For errors: Pinpoint where and why the reasoning went wrong
- Note any important patterns or strategies used/missed
- Review why some trajectories fail? Do experiences provide enough guidance?

Please provide a detailed analysis in line with the above aspects within tags <reflect> </reflect>.

— User turn 3 —
Based on the analysis, you now need to update skills using two options: <modify> and <create>.

Instructions
modify: refine a certain skill to be more complete
create: create a new skill that is helpful

Note that each item in a skill must satisfy the following requirements:
1. Concise: It must be clear, generalizable lessons for this case.
2. Actionable: Focus on strategic patterns, not specific calculations/actions.
3. Transferable: Emphasize decision points that could apply to similar tasks.

Output format
Use XML tags as shown below. For example, your output should be like:

<modify><name>target skill name</name><content>revised content</content></modify>
...
<create><name> unique skill name (no duplication with existing skills) </name> <description> 1-2 sentences to clarify applicable scenarios </description> <content> your skill content </content> </create>

User prompt with skills guidance

{task_description}
Here are some relevant experiences for reference:
{skills}

D ALGORITHM PROCESS

The complete rollout and training procedure of AlphaSkill are detailed in Algorithm 1.

E BROADER IMPACT AND LIMITATIONS

This work presents AlphaSkill, a unified training framework that enables language agents to self-curate and internalize reusable skills through interaction with open-ended environments. By improving the efficiency and adaptability of agentic reasoning, our method may contribute to the development of

Algorithm 1 AlphaSkill: Agentic Reinforcement Learning with Self-evolving Skills

Require: Initial policy π_θ; Task q; Rollout number N; Training steps T; Initial skill library

for $t \leftarrow 1$ to T **do**
  // Reasoner: Solve tasks without guidance
  $\mathcal{G}_1 \leftarrow \{\tau_i\}_{i=0}^{N/2} \sim \pi_\theta^R(\cdot \mid q)$
  Skill Curator: Generate or modify skills based on original rollouts
  Load relevant skills $\mathcal{S}$ from skill library
  $o \leftarrow c, \mathcal{S}_q \sim \pi_\theta^S(\cdot \mid q, \mathcal{S}, \mathcal{G}_1)$
  // Reasoner: Solve tasks with skill guidance
  $\mathcal{G}_2 \leftarrow \{\tau_i\}_{i=N/2}^{N} \sim \pi_\theta^R(\cdot \mid q, \mathcal{S}_q)$
  // Update: Calculate unified loss
  $\{r_i^R\}_{i=0}^N = r(\mathcal{G}_1) \cup r(\mathcal{G}_2)$
  is_valid $= \mathbb{I}(\text{avg } r(\mathcal{G}_2) > \text{avg } r(\mathcal{G}_1))$
  Add validated skills to the library and construct $\mathcal{D}_{\text{skills}}$
  Compute $\mathcal{L}_{\text{RL}}^R$ with $\{r_i^R\}_{i=0}^N$
  Compute $\mathcal{L}_{\text{SFT}}^S$ with $\mathcal{D}_{\text{skills}}$
  Update policy π_θ via $\mathcal{L}(\theta)$
  $t \leftarrow t + 1$
end for
return Trained policy π_θ

more capable autonomous systems for applications such as embodied assistants, web automation, scientific discovery, and interactive decision-making. In particular, the ability to accumulate and reuse experience across tasks could reduce redundant exploration and improve the reliability of long-horizon reasoning in complex environments.

At the same time, increasingly capable autonomous agents may also introduce potential risks. Since AlphaSkill strengthens agents' ability to iteratively refine behaviors from environmental feedback, it could be misused to optimize undesirable strategies in unconstrained settings, including deceptive planning, unsafe automation, or large-scale autonomous interaction with online systems. Moreover, self-evolving skills may implicitly hack into some biases or harmful behavioral patterns present in training environments and reward signals, potentially amplifying unintended behaviors over time.

We emphasize that AlphaSkill is designed as a research framework for studying skill acquisition and reasoning in controlled environments, rather than for unrestricted deployment. Future work should further investigate safety-aware skill validation, interpretable skill representations, and alignment mechanisms that can prevent harmful or reward-hacking behaviors from being internalized during training. We hope this work encourages future research on building autonomous agents that are not only more capable, but also more transparent, reliable, and aligned with human values.

Limitations. Although AlphaSkill demonstrates promising gains in agentic reinforcement learning through self-evolving skills, several limitations remain. First, the effectiveness of the framework still depends on the quality of environmental reward signals and trajectory feedback. In sparse-reward or highly stochastic environments, the Skill Curator may learn noisy or suboptimal behavioral patterns that negatively affect subsequent reasoning. Second, while the proposed self-evolving mechanism alleviates the need for manually designed skills, the overall training pipeline introduces about $1.2\times$ in training times due to additional rollouts and optimization of the Skill Curator.

F Case Study

Here we present what the self-curated skills look like.

ALFWorld Example Skills

—
name: alfworld_object_transform_place

description: For tasks requiring finding an object, changing its state, and placing it in a target location.
—
Goal decomposition
For this kind of state-constrained tasks, first decompose the goal into final receptacle and required object state.
Action Guide
- Search efficiently, but when the target appears, take it immediately before continuing to explore.
- Maintain a state checklist: current location, container open/closed, whether the agent is already holding an object, which object is held, and which transformation is still missing.
- Apply required transformations in the goal order: acquire → change state such as clean/dry/heat/cool if required → place in target receptacle → verify/finish.
- Use only canonical action forms and valid object roles; if an action fails with "Nothing happens", diagnose syntax, location, inventory, or object eligibility, then change one thing—never repeat the same invalid command.
- If holding the target, do not 'take' again; go to the correct station and run the explicit clean/use action. After placing, confirm the target is in the receptacle and has the required state before finishing.