SKILL-R1: AGENT SKILL EVOLUTION VIA REINFORCEMENT LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Agentic large language models increasingly rely on skills, reusable natural-language procedures that guide planning, action, and tool use. Current practice improves skills through prompt engineering or by fine-tuning the task LLM, both of which are costly, model-specific, and often infeasible for closed-source models. Effective skill optimization is moreover recurrent rather than one-shot: a useful skill must improve rollouts under the current conditioning, and a useful revision must turn observed outcomes into a better skill for the next round. We propose **Skill-R1**, a reinforcement learning framework that addresses skill optimization as a bi-level problem coupling within-generation execution quality with across-generation skill improvement. **Skill-R1** trains a lightweight skill generator conditioned on the task context, prior rollouts, and verified outcomes to produce skills that steer a frozen task LLM, preserving compatibility with both open- and closed-source models. The generator is optimized via a bi-level group-relative policy optimization objective: an *intra-generation* advantage compares skill–rollout pairs within each generation, and an *inter-generation* advantage rewards population-level improvement across generations. On agent benchmarks with verifiable rewards, Skill-R1 yields substantial gains over the evaluated baselines, with the largest improvements on complex multi-step tasks.

1 INTRODUCTION

Agentic large language models increasingly rely on external skills (Xu & Yan, 2026; Jiang et al., 2026c; Chen et al., 2026b; Nguyen et al., 2025; Huang et al., 2025b), which are reusable natural-language procedures that specify how an agent should decompose a task, invoke tools, and verify intermediate results. Since skills are designed as complementary modules of the task language model, they provide a practical interface for shaping LLM behavior without updating the task language model. This separation is practical in realistic deployments, where the task model may be proprietary (Gao et al., 2025; Huang et al., 2025a), expensive to adapt (Zhou et al., 2025; Wu et al., 2024a; 2025e; Wang et al., 2025c; Zhang et al., 2024), or shared across many downstream tasks (Li et al., 2026; Chen et al., 2026a; Jiang et al., 2026a).

However, many skill-based systems rely on well-engineered skills as predefined and fixed artifacts, or revise them only through prompting and single-round self-refinement (Jiang et al., 2026c; Xu & Yan, 2026; Chen et al., 2026b; Jiang et al., 2026a; Chen et al., 2026a; Xia et al., 2026; Zhang et al., 2026). These strategies can be misaligned and inefficient, since they rely on repeated prompt edits and are not directly aligned with downstream rewards (Wu et al., 2025c; Kveton et al., 2025; Xia et al., 2025a). On the other hand, some recent works adapt the task language model to the skill by fine-tuning the task language model on the skill (Wang et al., 2025a; Li et al., 2025b; Wang et al., 2024), but this approach can be computationally costly and limited in adaptability to many downstream tasks (Zhou et al., 2025; Xia et al., 2026; Jiang et al., 2026b; Wu et al., 2025e; Yao et al., 2024). Moreover, skill optimization should not be a single prompt-editing step, but a recurrent decision problem: a useful skill should improve rollout quality under the current context, and a useful revision should transform observed successes and failures into a better skill for the next round. This naturally leads to a bi-level optimization objective that couples execution quality within each generation with skill improvement across generations.

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.

To address this problem, we propose **Skill-R1**, a reinforcement learning framework for recurrent agent skill evolution from verifiable rewards. **Skill-R1** separates *task execution* from *skill improvement*. A frozen task LLM produces rollouts conditioned on the current skill, while a lightweight skill generator is trained to revise that skill based on the task context, prior rollouts, and their verified outcomes (Xia et al., 2025c; Wang et al., 2025b; Van Nguyen et al., 2025). By updating only the skill generator, the framework remains compatible with both open- and closed-source task models and can be more efficient than updating the task model itself (Van Nguyen et al., 2025; Xia et al., 2025a).

The core interaction loop of **Skill-R1** is multi-generation. For a given instance, each generation samples a population of skills from the generator; each skill induces a rollout from the frozen task model, a verifier scores the rollouts, and the resulting outcomes (Wu et al., 2025b; 2024b) are appended to an instance-specific history that conditions the next generation. This process yields an evolutionary view of skill improvement, in which each generation proposes a new population of skills, tests them through their rollouts, and passes forward evidence for subsequent refinement, supporting directional improvement (Xia et al., 2026; Sun et al., 2025; Zhou et al., 2025) on the same instance rather than selecting among isolated one-shot attempts.

Optimizing the skill generator requires both intra-generation and inter-generation credit assignment. To enable optimization of the recurrent skill generator, we introduce a bi-level group-relative policy optimization (Zhong et al., 2026; Shao et al., 2024; Mundada et al., 2026) objective that mirrors the recurrent structure above. The *intra-generation* term compares the skill candidates sampled within each generation by the rewards their rollouts achieve, capturing relative quality among contemporaneous skills. The *inter-generation* term measures whether the current generation's skill population improves the population-average reward relative to the previous generation. Together, these signals provide an objective for rewarding both strong individual skills and population-level progress across generations (Wu et al., 2025c; Surana et al., 2026; Li et al., 2025a; Huang et al., 2025c). Our experiments on three agent benchmarks with verifiable rewards (GAIA, WebWalker, and Geometry3k) show that Skill-R1 yields substantial improvements over no-skill, static-skill, prompt-evolution, and repeated-sampling baselines, as well as over a GRPO-style single-generation ablation on GAIA, with the largest gains on multi-step reasoning and tool-use tasks. We further observe a clear performance gap across skill generations, illustrating the effectiveness of recurrent skill evolution. We summarize the contributions of this work as follows:

- We formulate recurrent instance-level skill evolution as a bi-level group-relative policy optimization problem.
- We propose **Skill-R1**, a model-agnostic framework that trains a lightweight skill generator while keeping the task LLM frozen.
- We introduce a recurrent multi-generation rollout process together with a bi-level GRPO objective that combines *intra-* and *inter-generation* advantages.
- We evaluate **Skill-R1** on the GAIA, WebWalker, and Geometry3k benchmarks against agent-skill and prompt-evolution baselines. baselines, observing substantial gains in multi-step reasoning and tool use.

2 Preliminaries

2.1 Agent Skills

Recent LLM-agent systems commonly treat skills as reusable units of external procedural knowledge that can be retrieved, executed, and updated without modifying the underlying task model. Following this practice, let π_{task} denote a frozen task LLM, and let $\mathcal{S}$ be a space of explicit skill artifacts.

Definition 2.1 (Agent Skill)**.** An agent skill is a reusable procedural artifact $s \in \mathcal{S}$ that conditions the execution of the task model on an input instance. A skill may encode natural-language instructions, structured workflows, tool-use guidance, or other procedural constraints, while remaining external to the parameters of π_{task}.

Given an instance x and a skill s, the frozen task model induces a rollout distribution

$$\pi_{\text{task}}(\tau \mid x, s), \tag{1}$$

where τ denotes the resulting trajectory. In our setting, skills evolve recurrently on the same instance. We write $s_g \in \mathcal{S}$ for the skill used at generation g, and $\mathcal{H}_g$ for the instance-specific history available up to generation g, such as prior rollouts and their verified outcomes. These variables will be used to formalize skill evolution in later sections.

2.2 Group-Relative Policy Optimization

We briefly review Group-Relative Policy Optimization (GRPO) in the language-generation setting, where a policy $\pi_\theta(\cdot \mid x)$ defines a distribution over response rollouts y conditioned on a prompt or context $x \sim \mathcal{D}$. GRPO samples a group

$$\mathcal{G}(x) = \{y^{(i)}\}_{i=1}^{K}, \quad y^{(i)} \overset{\text{i.i.d.}}{\sim} \pi_\theta(\cdot \mid x), \tag{2}$$

assigns each rollout a verifier score $r(y^{(i)})$, and defines the group-relative advantage

$$A\Big(y^{(i)}; \mathcal{G}(x)\Big) = r\Big(y^{(i)}\Big) - \frac{1}{K}\sum_{j=1}^{K} r\Big(y^{(j)}\Big). \tag{3}$$

The GRPO objective is

$$\mathcal{L}_{\text{GRPO}}(\theta) = \mathbb{E}_{x\sim\mathcal{D},\,\mathcal{G}(x)}\Bigg[\sum_{i=1}^{K} A\Big(y^{(i)}; \mathcal{G}(x)\Big) \times \log \pi_\theta\Big(y^{(i)} \mid x\Big)\Bigg]. \tag{4}$$

This group-centered objective promotes rollouts that outperform their peers under the same input context.

3 Skill-R1: Agent Skill Evolution via Reinforcement Learning

We propose **Skill-R1** (illustrated in Figure 1), a reinforcement learning framework that improves a frozen task model by recurrently refining external skills without updating the task model itself. **Skill-R1** decouples *task execution* from *skill improvement*: a frozen task LLM generates rollout trajectories conditioned on a selected skill, while a separate editor policy updates skills based on verified execution outcomes.

3.1 Problem Setup

Let $x \in \mathcal{X}$ denote a task instance, and let $\mathcal{B}_x \subseteq \mathcal{S}$ denote the skill bank for instance x. Following Section 2.1, for any skill $s \in \mathcal{S}$ the frozen task model $\pi_{\text{task}}(\cdot \mid x, s)$ induces a rollout distribution (Equation (1)) over trajectories, and a verifier f_X assigns each rollout a scalar reward. At the initial generation, a population of K skills $\{s_0^{(i)}\}_{i=1}^{K} \subseteq \mathcal{B}_x$ is drawn from the bank, and each skill induces a single rollout under the task model. We collect the resulting (skill, rollout, reward) triples into the generation-wise history:

$$\begin{aligned} \mathcal{H}_0 &= \{(s_0^{(i)}, y_0^{(i)}, r_0^{(i)})\}_{i=1}^{K}, \\ y_0^{(i)} &\sim \pi_{\text{task}}(\cdot \mid x, s_0^{(i)}). \end{aligned} \tag{5}$$

Our goal is to improve performance on the given instance by revising skills so that later populations achieve higher verifier reward.

For each generation $g \geq 1$, a learnable skill generator π_θ samples a population of K skills conditioned on the current instance and the accumulated history, and each skill induces a single rollout under the frozen task model:

$$\begin{aligned} s_g^{(i)} &\sim \pi_\theta(\cdot \mid x, \mathcal{H}_{g-1}), \\ y_g^{(i)} &\sim \pi_{\text{task}}(\cdot \mid x, s_g^{(i)}), \quad i = 1, \ldots, K. \end{aligned} \tag{6}$$

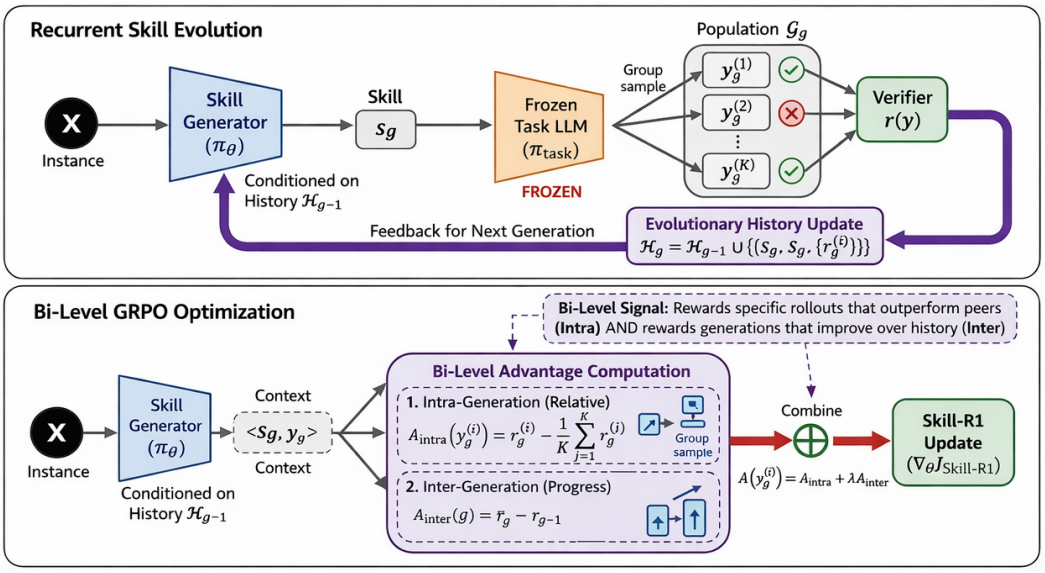


Figure 1: Overview of Skill-R1. **Recurrent Skill Evolution** iteratively generates skills, rolls out a frozen task LLM, and stores verifier-scored outcomes in an evolutionary history. **Bi-level GRPO Optimization** computes GRPO advantages from both intra-generation relative performance and inter-generation progress, and uses them to update the skill generator while keeping the task LLM frozen.

The resulting (skill, rollout, reward) triples are appended to the history, $\mathcal{H}_g = \mathcal{H}_{g-1} \cup \{(s_g^{(i)}, y_g^{(i)}, r_g^{(i)})\}_{i=1}^{K}$. This defines a recurrent skill-evolution process in which each generation's skill population is evaluated through the rollouts it induces under the frozen task model (Equation (1)), with verifier outcomes guiding subsequent revisions. Then the recurrent skill evolution problem can be defined as optimizing π_θ to maximize the discounted population mean verifier reward across the entire recurrent process, where $\gamma \in (0, 1]$ is a discount factor that controls how strongly later generations are weighted:

$$J(\theta) = \mathbb{E}_{x\sim p(x)}\, \mathbb{E}_{\substack{s_g^{(i)}\sim\pi_\theta(\cdot|x,\mathcal{H}_{g-1}) \\ y_g^{(i)}\sim\pi_{\text{task}}(\cdot|x,s_g^{(i)})}} \left[\sum_{g=1}^{G} \gamma^{g-1} \frac{1}{K}\sum_{i=1}^{K} r_g^{(i)}\right]. \tag{7}$$

Directly optimizing the objective in Equation (7) is challenging because the reward signal is evaluated on the task model's rollouts rather than directly on the generated skill. To address this structural decoupling between skill generation and task execution, we extend the standard GRPO framework to a bi-level setting. In the following section, we formulate a bi-level GRPO objective and derive *bi-level advantages*. This formulation allows us to properly assign credit to the skill generator π_θ based on the relative, aggregate performance of the rollout populations that each skill induces.

3.2 Bi-Level Group-Relative Advantages

The reward signal in Equation (7) is evaluated on rollouts from the frozen task model, yet the trainable parameters reside in the skill generator π_θ. To route rollout-level feedback to the skill that induced it, we decompose credit assignment into two complementary advantage signals. Within generation g, the K skills $\{s_g^{(i)}\}_{i=1}^{K}$ form a group of i.i.d. samples from π_θ, each paired with the single rollout it induced under the frozen task model. We apply the group-relative comparison from Equation (3) directly across the resulting K skill–rollout pairs:

$$A_{\text{intra}}(y_g^{(i)}) = r_g^{(i)} - \frac{1}{K}\sum_{j=1}^{K} r_g^{(j)}. \tag{8}$$

Algorithm 1 Skill-R1: Agent Skill Evolution via RL

Require: Task dist. $p(x)$, frozen LLM π_{task}, skill generator π_θ, skill bank $\mathcal{B}$, verifier f, generations G, group size K, coeff. λ

for each instance $x \sim p(x)$ with skill bank $\mathcal{B}_x$ **do**
  Draw initial population $\{s_0^{(i)}\}_{i=1}^K \subseteq \mathcal{B}_x$
  Sample $y_0^{(i)} \sim \pi_{\text{task}}(\cdot \mid x, s_0^{(i)})$ for $i = 1, \ldots, K$
  Score $r_0^{(i)} = f(y_0^{(i)})$; init $\mathcal{H}_0$
  for $g = 1, \ldots, G$ **do**
    Sample $s_g^{(i)} \sim \pi_\theta(\cdot \mid x, \mathcal{H}_{g-1})$ for $i = 1, \ldots, K$ ▷ Equation (6)
    Sample $y_g^{(i)} \sim \pi_{\text{task}}(\cdot \mid x, s_g^{(i)})$ for $i = 1, \ldots, K$
    Verify $r_g^{(i)} = f(y_g^{(i)})$ for each i
    Compute A_{intra}, A_{inter}, A ▷ Equations (8) to (10)
    $\mathcal{H}_g \leftarrow \mathcal{H}_{g-1} \cup \{(s_g^{(i)}, y_g^{(i)}, r_g^{(i)})\}_{i=1}^K$
  end for
end for
Update θ via $\mathcal{L}_{\text{Skill-R1}}(\theta)$ ▷ Equation (11)

Centering by the group mean isolates which skills within the generation outperformed their peers and assigns per-skill credit while removing the shared reward offset. However, A_{intra} is invariant to the level of reward in the generation and is therefore blind to whether the population as a whole improved on its predecessor. To capture cross-generation progress, we define the population mean reward and the inter-generation advantage

$$A_{\text{inter}}(g) = \bar{r}_g - \bar{r}_{g-1}, \quad \bar{r}_g = \frac{1}{K} \sum_{i=1}^{K} r_g^{(i)} \tag{9}$$

for $g \geq 1$, where $\bar{r}_0$ is the mean reward of the initial population drawn from the skill bank, so the first generation is compared against the warm start. A positive value indicates that the generation-g population produces higher mean reward than the previous generation; a negative value penalizes regressions. We combine the two signals into a single bi-level advantage assigned to each skill–rollout pair:

$$A(y_g^{(i)}) = A_{\text{intra}}(y_g^{(i)}) + \lambda\, A_{\text{inter}}(g), \tag{10}$$

where $\lambda \geq 0$ interpolates between pure within-generation GRPO (λ=0) and a regime that additionally rewards cross-generation improvement.

3.3 GRPO Objective for Skill Evolution

We optimize π_θ with a clipped GRPO surrogate that aggregates the bi-level advantage across all skill–rollout pairs in every generation. For each pair $(s_g^{(i)}, y_g^{(i)})$, let

$$\rho_g^{(i)} = \pi_\theta(s_g^{(i)} \mid x, \mathcal{H}_{g-1}) / \pi_{\theta_{\text{old}}}(s_g^{(i)} \mid x, \mathcal{H}_{g-1})$$

be the per-skill importance ratio between the current and data-collection policies. The **Skill-R1** objective is

$$\begin{aligned} \mathcal{L}_{\text{Skill-R1}}(\theta) = \mathbb{E}_{x \sim p(x)} \Bigg[\sum_{g=1}^{G} \Bigg(\sum_{i=1}^{K} \min\Big(& \rho_g^{(i)}\, A(y_g^{(i)}), \\ & \text{clip}(\rho_g^{(i)}, 1-\epsilon, 1+\epsilon)\, A(y_g^{(i)}) \Big) \\ & - \beta\, \text{KL}\big(\pi_\theta(\cdot \mid x, \mathcal{H}_{g-1}) \,\|\, \pi_{\text{ref}}(\cdot \mid x, \mathcal{H}_{g-1})\big) \Bigg) \Bigg], \end{aligned} \tag{11}$$

where $A(y_g^{(i)})$ is the bi-level advantage from Equation (10), $\epsilon > 0$ is the clipping radius, and $\beta \geq 0$ weights a KL penalty anchoring the policy near a reference π_{ref}. Because each skill $s_g^{(i)}$ in the population carries its own importance ratio $\rho_g^{(i)}$, the within-generation mean-centering in A_{intra} provides a non-degenerate gradient signal: skills whose rollouts score above the population mean are pushed up and those below are pushed down, recovering the standard GRPO behavior at the

skill level. Summing over g further couples the objective across the entire recurrent process in Equation (6), so π_θ is trained to produce improving skill populations rather than isolated single-step revisions. Because only π_θ is updated, the task LLM π_{task} remains frozen and can be open- or closed-source, with no gradients through π_{task} required. The full procedure is summarized in Algorithm 1.

4 EXPERIMENTS

4.1 EXPERIMENTAL SETUP

Tasks. We use GAIA (Mialon et al., 2024), WebWalker (Wu et al., 2025a), and Geometry3k (Lu et al., 2021) to evaluate Skill-R1 across multiple task types, including reasoning across heterogeneous information sources, multi-hop web navigation, and diagram-based geometry problem solving. For WebWalker and Geometry3k, we randomly sample tasks to cover a range of task types.

Baselines. Direct (no skills) uses the same tools and task-solving loop without skills. **ReAct + skills** (Yao et al., 2023) uses the fixed initial skill bank. **EvoPrompt** (Guo et al., 2024) evolves a prompt on development tasks for held-out evaluation. **Best-of-N sampling** samples 12 rollouts in three groups of four, reselecting fixed skills per group and returns the most frequent normalized answer (Wang et al., 2023).

Models and training. We train Qwen3-4B-Instruct-2507 (Yang et al., 2025) and Qwen3.5-4B (Qwen Team, 2026) editors with bi-level GRPO (G=3, K=4), using frozen Grok-4.20 for task execution across all benchmarks. Evaluation uses both GPT-4o-mini (OpenAI et al., 2024) and Grok-4.20. GPT-4o initializes skills from GPT-4o-mini execution traces. Tables denote Grok-4.20 as source and GPT-4o-mini as target; arrows indicate Skill-R1 transfer, with baselines evaluated on the destination model. See Sections A.1 and A.3 for skill initialization and compute/token costs.

Evaluation protocol. At evaluation time, Skill-R1 runs G=3 generations with K=1 rollout per generation. Direct (no skills), ReAct + skills, and EvoPrompt are each given three attempts per task. For all methodologies, a task counts as correct if any attempt is correct. The correctness labels and the model's final answers are stripped from the evidence given to the editor in Skill-R1, which ensures that revised skills never contain answer labels. Best-of-N sampling uses G=3 and K=4.

4.2 RESULTS

The best-performing Skill-R1 variant achieves the highest overall accuracy for each task model across all three benchmarks (Tables 1 to 3). With GPT-4o-mini, the Qwen3-4B (Yang et al., 2025) and Qwen3.5-4B (Qwen Team, 2026) editors raise GAIA accuracy from the strongest baseline's 8.0% to 32.0% and 40.0%, respectively. On WebWalker, they improve accuracy from 1.5% to 12.3% and 16.9%; on Geometry3k, they improve on the static-skill baseline's 46.2% to reach 50.8% and 55.4%. These gains show that revising skills from execution feedback can improve on both fixed skills and repeated sampling.

The gains persist with the more capable Grok-4.20 model. Skill-R1 reaches 62.0% on GAIA against the strongest baseline's 36.0%, and 30.8% on WebWalker against 27.7%. On Geometry3k, Grok already solves 61.5% of tasks without skills and reaches 80.0% with Best-of-N, yet Skill-R1 further raises accuracy to 84.6%. Learned skill revision therefore remains useful even when the task model is already a strong reasoner. For both task models, the largest gains over the strongest baseline occur on GAIA, suggesting that skill evolution is valuable for complex tasks that require coordinating multiple reasoning and tool-use steps. The gains on Geometry3k show that this benefit also extends to settings where the underlying model and sampling baseline are already competitive.

5 ANALYSIS

5.1 REWARD PROGRESSION ACROSS GENERATIONS

Figure 2 shows the running mean reward over 115 GAIA training tasks with the Qwen3-4B editor. Generation 0 uses skills inherited from earlier tasks; Generations 1–3 apply successive revisions on the current task.

Evaluation	Method	Level 1	Level 2	Level 3	Overall
Source → Target GPT-4o-mini	direct (no skills)	03/20 (15.0%)	01/23 (04.3%)	0/7 (00.0%)	04/50 (08.0%)
	ReAct + skills	02/20 (10.0%)	02/23 (08.7%)	0/7 (00.0%)	04/50 (08.0%)
	EvoPrompt	01/20 (05.0%)	00/23 (00.0%)	0/7 (00.0%)	01/50 (02.0%)
	Best-of-N sampling	03/20 (15.0%)	01/23 (04.3%)	0/7 (00.0%)	04/50 (08.0%)
	Skill-R1 (Qwen3-4B)	**08/20 (40.0%)**	07/23 (30.4%)	**1/7 (14.3%)**	16/50 (32.0%)
	Skill-R1 (Qwen3.5-4B)	**08/20 (40.0%)**	**11/23 (47.8%)**	**1/7 (14.3%)**	**20/50 (40.0%)**
Source → Source Grok-4.20	direct (no skills)	06/20 (30.0%)	05/23 (21.7%)	1/7 (14.3%)	12/50 (24.0%)
	ReAct + skills	10/20 (50.0%)	07/23 (30.4%)	1/7 (14.3%)	18/50 (36.0%)
	EvoPrompt	05/20 (25.0%)	03/23 (13.0%)	0/7 (00.0%)	08/50 (16.0%)
	Best-of-N sampling	10/20 (50.0%)	07/23 (30.4%)	1/7 (14.3%)	18/50 (36.0%)
	Skill-R1 (Qwen3-4B)	10/20 (50.0%)	10/23 (43.5%)	**3/7 (42.9%)**	23/50 (46.0%)
	Skill-R1 (Qwen3.5-4B)	**14/20 (70.0%)**	**15/23 (65.2%)**	2/7 (28.6%)	**31/50 (62.0%)**

Table 1: Evaluation results on GAIA by level.

Evaluation	Method	Easy	Medium	Hard	Overall
Source → Target GPT-4o-mini	direct (no skills)	00/15 (00.0%)	00/28 (00.0%)	00/22 (00.0%)	00/65 (00.0%)
	ReAct + skills	01/15 (06.7%)	00/28 (00.0%)	00/22 (00.0%)	01/65 (01.5%)
	EvoPrompt	01/15 (06.7%)	00/28 (00.0%)	00/22 (00.0%)	01/65 (01.5%)
	Best-of-N sampling	01/15 (06.7%)	00/28 (00.0%)	00/22 (00.0%)	01/65 (01.5%)
	Skill-R1 (Qwen3-4B)	01/15 (06.7%)	05/28 (17.9%)	**02/22 (09.1%)**	08/65 (12.3%)
	Skill-R1 (Qwen3.5-4B)	**03/15 (20.0%)**	**06/28 (21.4%)**	**02/22 (09.1%)**	**11/65 (16.9%)**
Source → Source Grok-4.20	direct (no skills)	03/15 (20.0%)	07/28 (25.0%)	01/22 (04.5%)	11/65 (16.9%)
	ReAct + skills	**04/15 (26.7%)**	06/28 (21.4%)	02/22 (09.1%)	12/65 (18.5%)
	EvoPrompt	03/15 (20.0%)	06/28 (21.4%)	06/22 (27.3%)	15/65 (23.1%)
	Best-of-N sampling	03/15 (20.0%)	09/28 (32.1%)	06/22 (27.3%)	18/65 (27.7%)
	Skill-R1 (Qwen3-4B)	**04/15 (26.7%)**	08/28 (28.6%)	05/22 (22.7%)	17/65 (26.2%)
	Skill-R1 (Qwen3.5-4B)	01/15 (06.7%)	**12/28 (42.9%)**	**07/22 (31.8%)**	**20/65 (30.8%)**

Table 2: Results on the WebWalker benchmark by difficulty.

Evaluation	Method	Overall
Source → Target GPT-4o-mini	direct (no skills)	24/65 (36.9%)
	ReAct + skills	30/65 (46.2%)
	EvoPrompt	29/65 (44.6%)
	Best-of-N sampling	27/65 (41.5%)
	Skill-R1 (Qwen3-4B)	33/65 (50.8%)
	Skill-R1 (Qwen3.5-4B)	**36/65 (55.4%)**
Source → Source Grok-4.20	direct (no skills)	40/65 (61.5%)
	ReAct + skills	50/65 (76.9%)
	EvoPrompt	45/65 (69.2%)
	Best-of-N sampling	52/65 (80.0%)
	Skill-R1 (Qwen3-4B)	**55/65 (84.6%)**
	Skill-R1 (Qwen3.5-4B)	53/65 (81.5%)

Table 3: Results on the Geometry3k benchmark.

Early fluctuations reflect a running mean computed over only a handful of tasks. Once these fluctuations settle, the revised generations consistently outperform the warm start through the first half of training. Between tasks 20 and 60, Generations 1 and 3 outperform the warm start by five to ten points, showing effective skill revision from current-task failures. The gap then narrows as the warm start rises from about 0.33 at task 35 to above 0.40 by task 65 and retains most of this gain, while Generations 1 and 3 decline from about 0.42 to 0.40. Because Generation 0 receives no revision on the current task, its improvement can only come from skills written on earlier tasks. The opposing trajectories therefore isolate improvement in the skill bank itself. The starting skill grows stronger even as the per-task gains from revision remain constant or decrease slightly. All four curves converge around 0.37 to 0.40, with Generation 2 slightly below the warm start, suggesting that the skill bank is nearing saturation.

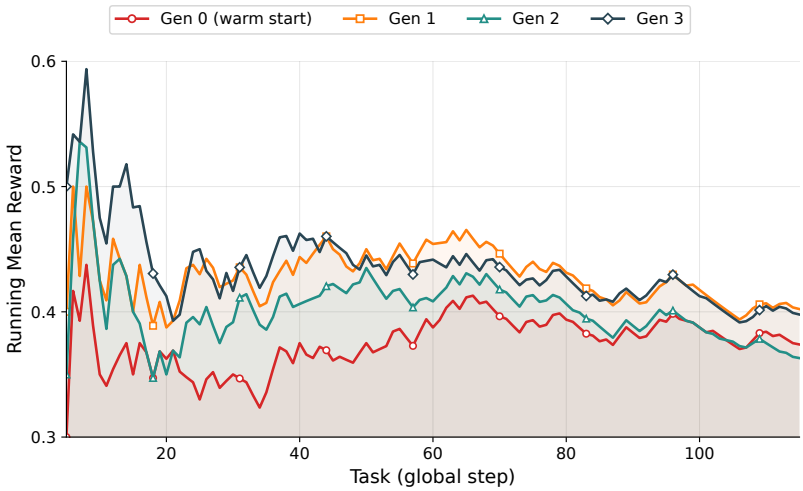


Figure 2: Running mean verifier reward across 115 GAIA training tasks for the Qwen3-4B editor. Each task contributes the mean of four rollouts in each generation.

5.2 Ablation of Training and Skill Revision

We compare two ablations on 50 held-out GAIA tasks with the Qwen3-4B editor. The *intra-generation* configuration (G=1) isolates GRPO-style finetuning through within-group rollout comparisons without subsequent revisions. The *inference-only* configuration uses the base editor to isolate prompted revision without training.

Accuracy rises from 8.0% for inference-only editing to 14.0% for intra-generation training and 32.0% for Skill-R1 (Table 4). This ordering holds on Levels 1 and 2, and only Skill-R1 solves a Level 3 task. The additional 18-point gain supports the contribution of the bi-level objective beyond GRPO-style finetuning alone. A qualitative comparison with the inference-only editor appears in Section A.2.

Configuration	Level 1	Level 2	Level 3	Overall
Intra-gen	25.0%	08.7%	00.0%	14.0%
Inference	15.0%	04.3%	00.0%	08.0%
Skill-R1	40.0%	30.4%	14.3%	32.0%

Table 4: Ablation on 50 held-out GAIA tasks with the Qwen3-4B editor.

5.3 Lambda and K Sensitivity

λ sensitivity

λ	Train (n=100)	Eval (n=65)
0.25	48.0%	50.8%
0.50	45.0%	50.8%
0.75	48.0%	50.8%

K sensitivity

K	Train (n=100)	Eval (n=65)
4	48.0%	50.8%
8	54.0%	52.3%
12	52.0%	49.2%

Table 5: Sensitivity to λ and K on Geometry3k.

Table 5 shows that Skill-R1 is robust to the choice of λ over the tested range. Held-out accuracy remains at 50.8% as λ varies from 0.25 to 0.75, while training accuracy ranges from 45.0% to 48.0%. This stability suggests that careful tuning of λ is unnecessary within this range.

Increasing K initially improves performance. Held-out accuracy rises from 50.8% at K=4 to 52.3% at K=8, and training accuracy increases from 48.0% to 54.0%. Increasing K further to 12, however, lowers held-out accuracy to 49.2% and training accuracy to 52.0%. This suggests that additional

rollouts provide useful feedback up to a point, but an overly long feedback context makes it harder for the skill generation model to focus on the specific edits needed to improve a skill.

6 Related Work

6.1 Agent Skill in Language Models

Recent work treats agent skills as reusable procedural abstractions that guide LLM behavior without modifying model parameters. Surveys (Xu & Yan, 2026; Jiang et al., 2026c) characterize skills as structured intermediates between tool usage and agent coordination. On the acquisition side, prior work has focused on constructing and organizing skill libraries (Nguyen et al., 2025; Huang et al., 2025a). SkillWeaver (Zheng et al., 2025) enables agents to autonomously discover and distill skills through web interaction, while ASI (Wang et al., 2025d) shows that programmatic skill representations improve reliability via verifiability. Similarly, CUA-Skill (Chen et al., 2026b) introduces a repository of parameterized execution graphs for computer-use agents. Adjacent agentic-LLM work treats reusable procedures and contextual conditioning as central design tools for tool use, document reasoning, and multimodal interaction (Xia et al., 2025b; Wu et al., 2025d;f; Wang et al., 2025b; Huang et al., 2025b). Li et al. (2026) study orchestration and system-level design, exploring ecosystem-scale skill coordination. More recent efforts move beyond static skill banks toward skill evolution. MemSkill (Zhang et al., 2026) models memory operations as skills refined through iterative control loops, while Memento-Skills (Zhou et al., 2026) enables continual skill improvement through reflective updates without modifying the base model. However, these approaches largely rely on heuristic refinement or implicit feedback signals.

6.2 Reinforcement Learning for Language Models

Reinforcement learning has become a central paradigm for aligning language models with task objectives and feedback signals. Methods such as PPO (Schulman et al., 2017) and its variants optimize policies using scalar rewards, while more recent approaches like GRPO (Shao et al., 2024) introduce group-relative comparisons to stabilize training and improve sample efficiency. Recent variants further extend GRPO to weakly supervised and reward-conditioned settings (Mundada et al., 2026; Zhong et al., 2026). Several works extend RL to agent settings and skill learning. SkillRL (Xia et al., 2026) jointly learns hierarchical skills and policies through recursive reinforcement learning, enabling agents to acquire reusable behaviors over time. Despite these advances, most existing methods focus on optimizing the task model itself or operate within a single-generation setting, limiting their ability to capture iterative improvement.

7 Conclusion

We presented Skill-R1, a framework for recurrent skill evolution that improves agent performance without modifying the underlying task model. By decoupling execution from skill refinement and introducing a bi-level optimization objective, Skill-R1 enables iterative, reward-driven improvement over multiple generations. Empirically, Skill-R1 consistently outperforms the evaluated baselines, with the largest gains observed on complex, multi-step tasks. Analysis shows that early generations drive most capability improvements and strengthen the skill bank. Overall, our results suggest that skill evolution is a practical alternative to model-level adaptation, offering a lightweight yet effective mechanism for improving agent behavior in both open- and closed-source settings.

AI use statement

We used LLM tools to assist with figure preparation and refine the writing.

References

Shiqi Chen, Jingze Gai, Ruochen Zhou, Jinghan Zhang, Tongyao Zhu, Junlong Li, Kangrui Wang, Zihan Wang, Zhengyu Chen, Klara Kaleb, et al. Skillcraft: Can llm agents learn to use tools skillfully? *arXiv preprint arXiv:2603.00718*, 2026a.

Tianyi Chen, Yinheng Li, Michael Solodko, Sen Wang, Nan Jiang, Tingyuan Cui, Junheng Hao, Jongwoo Ko, Sara Abdali, Leon Xu, et al. Cua-skill: Develop skills for computer using agent. *arXiv preprint arXiv:2601.21123*, 2026b.

Huan-ang Gao, Jiayi Geng, Wenyue Hua, Mengkang Hu, Xinzhe Juan, Hongzhang Liu, Shilong Liu, Jiahao Qiu, Xuan Qi, Yiran Wu, et al. A survey of self-evolving agents: What, when, how, and where to evolve on the path to artificial super intelligence. *arXiv preprint arXiv:2507.21046*, 2025.

Qingyan Guo, Rui Wang, Junliang Guo, Bei Li, Kaitao Song, Xu Tan, Guoqing Liu, Jiang Bian, and Yujiu Yang. Connecting large language models with evolutionary algorithms yields powerful prompt optimizers. In *International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/9156b0f6dfa9bbd18c79cc459ef5d61c-Abstract-Conference.html.

Chengkai Huang, Hongtao Huang, Tong Yu, Kaige Xie, Junda Wu, Shuai Zhang, Julian Mcauley, Dietmar Jannach, and Lina Yao. A survey of foundation model-powered recommender systems: From feature-based, generative to agentic paradigms. *arXiv preprint arXiv:2504.16420*, 2025a.

Chengkai Huang, Junda Wu, Yu Xia, Zixu Yu, Ruhan Wang, Tong Yu, Ruiyi Zhang, Ryan A. Rossi, Branislav Kveton, Dongruo Zhou, Julian McAuley, and Lina Yao. Towards agentic recommender systems in the era of multimodal large language models, 2025b. URL https://arxiv.org/abs/2503.16734.

Chengkai Huang, Junda Wu, Zhouhang Xie, Yu Xia, Rui Wang, Tong Yu, Subrata Mitra, Julian McAuley, and Lina Yao. Pluralistic off-policy evaluation and alignment. *arXiv preprint arXiv:2509.19333*, 2025c.

Guanyu Jiang, Zhaochen Su, Xiaoye Qu, et al. Xskill: Continual learning from experience and skills in multimodal agents. *arXiv preprint arXiv:2603.12056*, 2026a.

Pengcheng Jiang, Jiacheng Lin, Zhiyi Shi, Zifeng Wang, Luxi He, Yichen Wu, Ming Zhong, Peiyang Song, Qizheng Zhang, Heng Wang, Xueqiang Xu, Hanwen Xu, Pengrui Han, Dylan Zhang, Jiashuo Sun, Chaoqi Yang, Kun Qian, Tian Wang, Changran Hu, Manling Li, Quanzheng Li, Hao Peng, Sheng Wang, Jingbo Shang, Chao Zhang, Jiaxuan You, Liyuan Liu, Pan Lu, Yu Zhang, Heng Ji, Yejin Choi, Dawn Song, Jimeng Sun, and Jiawei Han. Adaptation of agentic ai: A survey of post-training, memory, and skills, 2026b. URL https://arxiv.org/abs/2512.16301.

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*, 2026c.

Branislav Kveton, Xintong Li, Julian McAuley, Ryan Rossi, Jingbo Shang, Junda Wu, and Tong Yu. Active learning for direct preference optimization. *arXiv preprint arXiv:2503.01076*, 2025.

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*, 2026.

Xintong Li, Chuhan Wang, Junda Wu, Rohan Surana, Tong Yu, Julian McAuley, and Jingbo Shang. Importance sampling for multi-negative multimodal direct preference optimization. *arXiv preprint arXiv:2509.25717*, 2025a.

Xintong Li, Junda Wu, Tong Yu, Rui Wang, Yu Wang, Xiang Chen, Jiuxiang Gu, Lina Yao, Julian McAuley, and Jingbo Shang. Commit: Coordinated multimodal instruction tuning. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 11533–11547, 2025b.

Pan Lu, Ran Gong, Shibiao Jiang, Liang Qiu, Siyuan Huang, Xiaodan Liang, and Song-Chun Zhu. Inter-gps: Interpretable geometry problem solving with formal language and symbolic reasoning, 2021. URL https://arxiv.org/abs/2105.04165.

Grégoire Mialon, Clémentine Fourrier, Thomas Wolf, Yann LeCun, and Thomas Scialom. Gaia: a benchmark for general ai assistants. In *International Conference on Learning Representations*, volume 2024, pp. 9025–9049, 2024.

Gagan Mundada, Zihan Huang, Rohan Surana, Sheldon Yu, Jennifer Yuntong Zhang, Xintong Li, Tong Yu, Lina Yao, Jingbo Shang, Julian McAuley, et al. Ws-grpo: Weakly-supervised group-relative policy optimization for rollout-efficient reasoning. *arXiv preprint arXiv:2602.17025*, 2026.

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

OpenAI, Aaron Hurst, Adam Lerer, Adam P. Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, Aleksander Mądry, Alex Baker-Whitcomb, Alex Beutel, Alex Borzunov, Alex Carney, Alex Chow, Alex Kirillov, Alex Nichol, Alex Paino, Alex Renzin, Alex Tachard Passos, Alexander Kirillov, Alexi Christakis, Alexis Conneau, Ali Kamali, Allan Jabri, Allison Moyer, Allison Tam, Amadou Crookes, Amin Tootoochian, Amin Tootoonchian, Ananya Kumar, Andrea Vallone, Andrej Karpathy, Andrew Braunstein, Andrew Cann, Andrew Codispoti, Andrew Galu, Andrew Kondrich, Andrew Tulloch, Andrey Mishchenko, Angela Baek, Angela Jiang, Antoine Pelisse, Antonia Woodford, Anuj Gosalia, Arka Dhar, Ashley Pantuliano, Avi Nayak, Avital Oliver, Barret Zoph, Behrooz Ghorbani, Ben Leimberger, Ben Rossen, Ben Sokolowsky, Ben Wang, Benjamin Zweig, Beth Hoover, Blake Samic, Bob McGrew, Bobby Spero, Bogo Giertler, Bowen Cheng, Brad Lightcap, Brandon Walkin, Brendan Quinn, Brian Guarraci, Brian Hsu, Bright Kellogg, Brydon Eastman, Camillo Lugaresi, Carroll Wainwright, Cary Bassin, Cary Hudson, Casey Chu, Chad Nelson, Chak Li, Chan Jun Shern, Channing Conger, Charlotte Barette, Chelsea Voss, Chen Ding, Cheng Lu, Chong Zhang, Chris Beaumont, Chris Hallacy, Chris Koch, Christian Gibson, Christina Kim, Christine Choi, Christine McLeavey, Christopher Hesse, Claudia Fischer, Clemens Winter, Coley Czarnecki, Colin Jarvis, Colin Wei, Constantin Koumouzelis, Dane Sherburn, Daniel Kappler, Daniel Levin, Daniel Levy, David Carr, David Farhi, David Mely, David Robinson, David Sasaki, Denny Jin, Dev Valladares, Dimitris Tsipras, Doug Li, Duc Phong Nguyen, Duncan Findlay, Edede Oiwoh, Edmund Wong, Ehsan Asdar, Elizabeth Proehl, Elizabeth Yang, Eric Antonow, Eric Kramer, Eric Peterson, Eric Sigler, Eric Wallace, Eugene Brevdo, Evan Mays, Farzad Khorasani, Felipe Petroski Such, Filippo Raso, Francis Zhang, Fred von Lohmann, Freddie Sulit, Gabriel Goh, Gene Oden, Geoff Salmon, Giulio Starace, Greg Brockman, Hadi Salman, Haiming Bao, Haitang Hu, Hannah Wong, Haoyu Wang, Heather Schmidt, Heather Whitney, Heewoo Jun, Hendrik Kirchner, Henrique Ponde de Oliveira Pinto, Hongyu Ren, Huiwen Chang, Hyung Won Chung, Ian Kivlichan, Ian O'Connell, Ian Osband, Ian Silber, Ian Sohl, Ibrahim Okuyucu, Ikai Lan, Ilya Kostrikov, Ilya Sutskever, Ingmar Kanitscheider, Ishaan Gulrajani, Jacob Coxon, Jacob Menick, Jakub Pachocki, James Aung, James Betker, James Crooks, James Lennon, Jamie Kiros, Jan Leike, Jane Park, Jason Kwon, Jason Phang, Jason Teplitz, Jason Wei, Jason Wolfe, Jay Chen, Jeff Harris, Jenia Varavva, Jessica Gan Lee, Jessica Shieh, Ji Lin, Jiahui Yu, Jiayi Weng, Jie Tang, Jieqi Yu, Joanne Jang, Joaquin Quinonero Candela, Joe Beutler, Joe Landers, Joel Parish, Johannes Heidecke, John Schulman, Jonathan Lachman, Jonathan McKay, Jonathan Uesato, Jonathan Ward, Jong Wook Kim, Joost Huizinga, Jordan Sitkin, Jos Kraaijeveld, Josh Gross, Josh Kaplan, Josh Snyder, Joshua Achiam, Joy Jiao, Joyce Lee, Juntang Zhuang, Justyn Harriman, Kai Fricke, Kai Hayashi, Karan Singhal, Katy Shi, Kavin Karthik, Kayla Wood, Kendra Rimbach, Kenny Hsu, Kenny Nguyen, Keren Gu-Lemberg, Kevin Button, Kevin Liu, Kiel Howe, Krithika Muthukumar, Kyle Luther, Lama Ahmad, Larry Kai, Lauren Itow, Lauren Workman, Leher Pathak, Leo Chen, Li Jing, Lia Guy, Liam Fedus, Liang Zhou, Lien Mamitsuka, Lilian Weng, Lindsay McCallum, Lindsey Held, Long Ouyang, Louis Feuvrier, Lu Zhang, Lukas Kondraciuk, Lukasz Kaiser, Luke Hewitt, Luke Metz, Lyric Doshi, Mada Aflak, Maddie Simens, Madelaine Boyd, Madeleine Thompson, Marat Dukhan, Mark Chen, Mark Gray, Mark Hudnall, Marvin Zhang, Marwan Aljubeh, Mateusz Litwin, Matthew Zeng, Max Johnson, Maya Shetty, Mayank Gupta, Meghan Shah, Mehmet Yatbaz, Meng Jia Yang, Mengchao Zhong, Mia Glaese, Mianna Chen, Michael Janner, Michael Lampe, Michael Petrov, Michael Wu, Michele Wang, Michelle Fradin, Michelle Pokrass, Miguel Castro, Miguel Oom Temudo de Castro, Mikhail Pavlov, Miles Brundage, Miles Wang, Minal Khan, Mira Murati, Mo Bavarian, Molly Lin, Murat Yesildal, Nacho Soto, Natalia Gimelshein, Natalie Cone, Natalie Staudacher, Natalie Summers,

Natan LaFontaine, Neil Chowdhury, Nick Ryder, Nick Stathas, Nick Turley, Nik Tezak, Niko Felix, Nithanth Kudige, Nitish Keskar, Noah Deutsch, Noel Bundick, Nora Puckett, Ofir Nachum, Ola Okelola, Oleg Boiko, Oleg Murk, Oliver Jaffe, Olivia Watkins, Olivier Godement, Owen Campbell-Moore, Patrick Chao, Paul McMillan, Pavel Belov, Peng Su, Peter Bak, Peter Bakkum, Peter Deng, Peter Dolan, Peter Hoeschele, Peter Welinder, Phil Tillet, Philip Pronin, Philippe Tillet, Prafulla Dhariwal, Qiming Yuan, Rachel Dias, Rachel Lim, Rahul Arora, Rajan Troll, Randall Lin, Rapha Gontijo Lopes, Raul Puri, Reah Miyara, Reimar Leike, Renaud Gaubert, Reza Zamani, Ricky Wang, Rob Donnelly, Rob Honsby, Rocky Smith, Rohan Sahai, Rohit Ramchandani, Romain Huet, Rory Carmichael, Rowan Zellers, Roy Chen, Ruby Chen, Ruslan Nigmatullin, Ryan Cheu, Saachi Jain, Sam Altman, Sam Schoenholz, Sam Toizer, Samuel Miserendino, Sandhini Agarwal, Sara Culver, Scott Ethersmith, Scott Gray, Sean Grove, Sean Metzger, Shamez Hermani, Shantanu Jain, Shengjia Zhao, Sherwin Wu, Shino Jomoto, Shirong Wu, Shuaiqi, Xia, Sonia Phene, Spencer Papay, Srinivas Narayanan, Steve Coffey, Steve Lee, Stewart Hall, Suchir Balaji, Tal Broda, Tal Stramer, Tao Xu, Tarun Gogineni, Taya Christianson, Ted Sanders, Tejal Patwardhan, Thomas Cunninghman, Thomas Degry, Thomas Dimson, Thomas Raoux, Thomas Shadwell, Tianhao Zheng, Todd Underwood, Todor Markov, Toki Sherbakov, Tom Rubin, Tom Stasi, Tomer Kaftan, Tristan Heywood, Troy Peterson, Tyce Walters, Tyna Eloundou, Valerie Qi, Veit Moeller, Vinnie Monaco, Vishal Kuo, Vlad Fomenko, Wayne Chang, Weiyi Zheng, Wenda Zhou, Wesam Manassra, Will Sheu, Wojciech Zaremba, Yash Patil, Yilei Qian, Yongjik Kim, Youlong Cheng, Yu Zhang, Yuchen He, Yuchen Zhang, Yujia Jin, Yunxing Dai, and Yury Malkov. Gpt-4o system card, 2024. URL https://arxiv.org/abs/2410.21276.

Qwen Team. Qwen3.5: Towards native multimodal agents, February 2026. URL https://qwen.ai/blog?id=qwen3.5.

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

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

Zeyi Sun, Ziyu Liu, Yuhang Zang, Yuhang Cao, Xiaoyi Dong, Tong Wu, Dahua Lin, and Jiaqi Wang. Seagent: Self-evolving computer use agent with autonomous learning from experience. *arXiv preprint arXiv:2508.04700*, 2025.

Rohan Surana, Junda Wu, Xintong Li, Sheldon Yu, Yiran Jenny Shen, Chuhan Wang, Tong Yu, Prithviraj Ammanabrolu, Jingbo Shang, and Julian McAuley. MASS-DPO: Multi-negative active sample selection for direct policy optimization, 2026. URL https://openreview.net/forum?id=gFtdK7pwHg.

Chien Van Nguyen, Xuan Shen, Ryan Aponte, Yu Xia, Samyadeep Basu, Zhengmian Hu, Jian Chen, Mihir Parmar, Sasidhar Kunapuli, Joe Barrow, et al. A survey on small language models. In *Proceedings of the 15th International Conference on Recent Advances in Natural Language Processing-Natural Language Processing in the Generative AI Era*, pp. 807–821, 2025.

Jianing Wang, Junda Wu, Yupeng Hou, Yao Liu, Ming Gao, and Julian McAuley. Instructgraph: Boosting large language models via graph-centric instruction tuning and preference alignment. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 13492–13510, 2024.

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.

Ruoyu Wang, Junda Wu, Yu Xia, Tong Yu, Ryan A Rossi, Julian McAuley, and Lina Yao. Dice: Dynamic in-context example selection in llm agents via efficient knowledge transfer. *arXiv preprint arXiv:2507.23554*, 2025b.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models, 2023. URL https://arxiv.org/abs/2203.11171.

Yu Wang, Xinshuang Liu, Xiusi Chen, Sean OBrien, Junda Wu, and Julian McAuley. Self-updatable large language models by integrating context into model parameters. In *International Conference on Learning Representations*, volume 2025, pp. 16961–16979, 2025c.

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

Jialong Wu, Wenbiao Yin, Yong Jiang, Zhenglin Wang, Zekun Xi, Runnan Fang, Linhai Zhang, Yulan He, Deyu Zhou, Pengjun Xie, et al. Webwalker: Benchmarking llms in web traversal. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 10290–10305, 2025a.

Junda Wu, Hanjia Lyu, Yu Xia, Zhehao Zhang, Joe Barrow, Ishita Kumar, Mehrnoosh Mirtaheri, Hongjie Chen, Ryan A Rossi, Franck Dernoncourt, et al. Personalized multimodal large language models: A survey. *arXiv preprint arXiv:2412.02142*, 2024a.

Junda Wu, Tong Yu, Xiang Chen, Haoliang Wang, Ryan Rossi, Sungchul Kim, Anup Rao, and Julian McAuley. Decot: Debiasing chain-of-thought for knowledge-intensive tasks in large language models via causal intervention. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 14073–14087, 2024b.

Junda Wu, Xintong Li, Ruoyu Wang, Yu Xia, Yuxin Xiong, Jianing Wang, Tong Yu, Xiang Chen, Branislav Kveton, Lina Yao, et al. Ocean: Offline chain-of-thought evaluation and alignment in large language models. In *International Conference on Learning Representations*, volume 2025, pp. 100570–100589, 2025b.

Junda Wu, Rohan Surana, Zhouhang Xie, Yiran Shen, Yu Xia, Tong Yu, Ryan A Rossi, Prithviraj Ammanabrolu, and Julian McAuley. In-context ranking preference optimization. In *Second Conference on Language Modeling*, 2025c.

Junda Wu, Yu Xia, Tong Yu, Xiang Chen, Sai Sree Harsha, Akash V Maharaj, Ruiyi Zhang, Victor Bursztyn, Sungchul Kim, Ryan A Rossi, et al. Doc-react: Multi-page heterogeneous document question-answering. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 67–78, 2025d.

Junda Wu, Yuxin Xiong, Xintong Li, Yu Xia, Ruoyu Wang, Yu Wang, Tong Yu, Sungchul Kim, Ryan A Rossi, Lina Yao, et al. Mitigating visual knowledge forgetting in mllm instruction-tuning via modality-decoupled gradient descent. *arXiv preprint arXiv:2502.11740*, 8, 2025e.

Junda Wu, Yuxin Xiong, Xintong Li, Sheldon Yu, Zhengmian Hu, Tong Yu, Rui Wang, Xiang Chen, Jingbo Shang, and Julian McAuley. Ctrls: Chain-of-thought reasoning via latent state-transition. *arXiv preprint arXiv:2507.08182*, 2025f.

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.

Yu Xia, Subhojyoti Mukherjee, Zhouhang Xie, Junda Wu, Xintong Li, Ryan Aponte, Hanjia Lyu, Joe Barrow, Hongjie Chen, Franck Dernoncourt, Branislav Kveton, Tong Yu, Ruiyi Zhang, Jiuxiang Gu, Nesreen K. Ahmed, Yu Wang, Xiang Chen, Hanieh Deilamsalehy, Sungchul Kim, Zhengmian Hu, Yue Zhao, Nedim Lipka, Seunghyun Yoon, Ting-Hao Kenneth Huang, Zichao Wang, Puneet Mathur, Soumyabrata Pal, Koyel Mukherjee, Zhehao Zhang, Namyong Park, Thien Huu Nguyen, Jiebo Luo, Ryan A. Rossi, and Julian McAuley. From selection to generation: A survey of LLM-based active learning. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 14552–14569, Vienna, Austria, July 2025a. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.708. URL https://aclanthology.org/2025.acl-long.708/.

Yu Xia, Yiran Jenny Shen, Junda Wu, Tong Yu, Sungchul Kim, Ryan A Rossi, Lina Yao, and Julian McAuley. Sand: Boosting llm agents with self-taught action deliberation. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 3062–3077, 2025b.

Yu Xia, Junda Wu, Sungchul Kim, Tong Yu, Ryan A Rossi, Haoliang Wang, and Julian McAuley. Knowledge-aware query expansion with large language models for textual and relational retrieval. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 4275–4286, 2025c.

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

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

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models, 2023. URL https://arxiv.org/abs/2210.03629.

Yuhang Yao, Jianyi Zhang, Junda Wu, Chengkai Huang, Yu Xia, Tong Yu, Ruiyi Zhang, Sungchul Kim, Ryan Rossi, Ang Li, et al. Federated large language models: Current progress and future directions. *arXiv preprint arXiv:2409.15723*, 2024.

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*, 2026.

Zhehao Zhang, Ryan A Rossi, Branislav Kveton, Yijia Shao, Diyi Yang, Hamed Zamani, Franck Dernoncourt, Joe Barrow, Tong Yu, Sungchul Kim, et al. Personalization of large language models: A survey. *arXiv preprint arXiv:2411.00027*, 2024.

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.

Haitian Zhong, Jixiu Zhai, Lei Song, Jiang Bian, Qiang Liu, and Tieniu Tan. Rc-grpo: Reward-conditioned group relative policy optimization for multi-turn tool calling agents. *arXiv preprint arXiv:2602.03025*, 2026.

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*, 2025.

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*, 2026.

A Appendix

A.1 Skill initialization

Base skills are constructed via a two-stage distillation procedure applied per benchmark. First, GPT-4o-mini is run on a randomly selected set of 20 tasks to collect reasoning traces, including both successful and failed trajectories. Second, these traces are provided to a stronger LLM (GPT-4o), which abstracts recurring patterns into concise, reusable skill guides. For reproducibility, each Skill-R1 run uses an isolated copy of the skill directory, while baselines share a read-only version.

A.2 SKILL CASE STUDY

This appendix provides excerpts of the `pdb` skill at three points in time, as referenced in Section 5: (1) the original skill from the shared skill library, (2) the version produced by the GRPO-trained Qwen3-4B editor after three generations of evolution (run `20260320_131744_0.25`), and (3) the version produced by the untrained base Qwen3-4B model operating in inference mode (run `20260321_185812`).

A.2.1 ORIGINAL PDB SKILL

The original skill is a general-purpose reference covering BioPython's full API. We reproduce the header description and the Quick Start section to illustrate the skill's initial breadth; the remainder of the document (Dependencies, PDB File Format Basics, Python Libraries, Manual Parsing, Common Tasks, Quick Reference table) is analogous in scope.

````
name: pdb
description: Use this skill whenever the
  user wants to read, parse, analyze, or
  manipulate PDB (Protein Data Bank) files.
  This includes extracting atomic
  coordinates, analyzing protein structures,
  identifying residues, chains, ligands,
  computing distances between atoms,
  visualizing molecular structures, and
  converting between molecular file formats.
  If the user mentions a .pdb file or asks
  about protein/molecular structure data,
  use this skill.

# PDB File Processing Guide

## Quick Start

```python
from Bio.PDB import PDBParser

parser = PDBParser(QUIET=True)
structure = parser.get_structure(
    "protein", "structure.pdb")

# Iterate: Structure -> Model -> Chain
#       -> Residue -> Atom
for model in structure:
    for chain in model:
        n = len(list(chain.get_residues()))
        print(f"Chain {chain.id}: "
              f"{n} residues")
```

## PDB File Format Basics

PDB files are fixed-width text files.
Key record types:
| ATOM   | Standard residue coordinates   |
| HETATM | Non-standard (ligands, water) |
| SEQRES | Full sequence of residues     |
| REMARK | Comments and metadata         |

[... BioPython parsing examples, distance
     computation, ligand extraction,
     sequence extraction, manual parsing,
     and quick-reference table omitted
     for brevity ...]
````

A.2.2 BI-LEVEL GRPO-TRAINED EDITOR: EDITED PDB SKILL

````
name: pdb
description: [unchanged from original]

# PDB File Processing Guide

## Quick Start

```python
from Bio.PDB import PDBParser
import numpy as np

# Parse a local PDB file path
parser = PDBParser(QUIET=True)

try:
    structure = parser.get_structure(
        "structure", "5wb7.pdb")
except FileNotFoundError:
    print("Error: PDB file not found. "
          "Verify the file name.")
    raise
except ImportError:
    print("Error: BioPython not installed."
          " pip install biopython")
    raise

# Access the first model and chain
models = list(structure)
if not models:
    raise ValueError(
        "No models found in PDB structure.")
model = models[0]

chains = list(model.get_chains())
if not chains:
    raise ValueError(
        "No chains found in the model.")
chain = chains[0]

residues = list(chain.get_residues())
if not residues:
    raise ValueError(
        "No residues found in the chain.")
```
````

A.2.3 BASE INFERENCE EDITOR: EDITED PDB SKILL

The untrained base Qwen3-4B model replaces the entire skill with the following. No executable code remains. The "Key Requirements" section mistakenly names the GAIA task UUID `7dd30055-0198-452e-8c25-f73dbe27dcb8` as an invalid PDB identifier, and the "Common Misuse" section is wholly devoted to handling a calculus homework audio recording (`Homework.mp3`) that is unrelated to PDB processing.

```
name: pdb
description: [unchanged from original]

# PDB File Processing Guide

## Key Requirements and Best Practices

- Always ensure the PDB ID is valid and
  accessible via the RCSB PDB database.
```

```
- Use the correct PDB ID (e.g., "5wb7",
  not "7dd30055-0198-452e-8c25-
  f73dbe27dcb8") -- the latter is a UUID
  or invalid identifier.
- This skill does NOT handle audio files.
  If a user provides an audio file (e.g.,
  Homework.mp3) and asks for page numbers,
  this skill is NOT applicable.

## Common Misuse and Clarifications

- Do not use this skill to extract page
  numbers from audio recordings.
- Do not attempt to parse audio files as
  if they were PDB files.
- If a user asks about study materials
  from a lecture, state that the skill
  cannot process audio content.
- Always verify the nature of the input:
  if it is .mp3 or .wav, respond with:
  "I cannot extract page numbers from
  audio files."

## Example of Incorrect Use

User: "I have a file called Homework.mp3
       with my professor reading the
       recommended pages. Can you tell me
       the page numbers?"
Correct response: "I cannot extract page
  numbers from audio files. This skill is
  designed for processing PDB files, not
  audio recordings."

[--- No code examples remain ---]
```

The contrast is stark. The GRPO-trained editor adds *code* that targets real failure modes, catching a missing PDB file and adding an installation hint for BioPython, while preserving the entire original reference. The `ImportError` handler, reproduced verbatim from the editor's output, cannot take effect because the import precedes the `try` block, but it shows the editor attending to the missing-library failure observed in rollouts. The base inference editor removes all code and fills the skill with content copied from two specific benchmark questions, leaving it useless for any unseen PDB task.

A.3 Compute Budget

We measure Skill-R1's compute requirements using the main training and evaluation runs on GAIA, WebWalker, and Geometry3k. All runs use NVIDIA A6000 GPUs with 48 GB of memory. Training uses two GPUs, and evaluation uses one GPU. Across all three benchmarks and both editors, Skill-R1 averages 24.7 hours per training run and 10.0 hours per evaluation run.

These averages give equal weight to each benchmark. Training covers six runs, one for each combination of benchmark and editor model, using Qwen3-4B and Qwen3.5-4B. Evaluation covers twelve configurations, pairing each trained editor with GPT-4o-mini and Grok-4.20. The separately executed WebWalker subsets are combined into one complete evaluation for each configuration.

Estimated token usage. Table 6 reports the total tokens consumed per run. Averaged across benchmarks, a training run uses approximately 217M tokens and an evaluation run approximately 34M tokens. Most of this budget is spent by the task model solving tasks, since each ReAct step resends the growing conversation, which makes WebWalker, with its long web pages, the most expensive benchmark. Skill revision by the editor accounts for under 1% of training tokens and

under 3% of evaluation tokens, so the overhead of skill evolution is small relative to the cost of task execution.

Benchmark	Training	Evaluation
GAIA	192M	17M
WebWalker	420M	77M
Geometry3k	39M	8M
Average	217M	34M

Table 6: Estimated total token usage per run for Skill-R1.