# SKILLGATE: TRAINING IN-POLICY SKILL SELECTION IN LONG-HORIZON AGENTS

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Language model agents access procedural knowledge through *skills*, instruction files read on demand. As libraries grow, agents must learn which skill to read within their own execution loop. We show that the default recipe, outcome-rewarded RL over the candidate skills, cannot teach it, for a structural reason we identify and name *selector credit starvation*: under a broadcast, sequence-level advantage, the few tokens that name the chosen skill carry a vanishing share of the loss, and the credit they inherit is increasingly wrong-signed as trajectories lengthen. A correct choice is punished whenever the execution after it fails, even though the choice is among the most valuable decisions in the trajectory. To address this, we propose SkillGate, which removes the failure by construction: it partitions the token support into two disjoint credit channels, with outcome credit reaching only execution tokens and a separate action-local advantage reaching exactly the skill-naming tokens, positive only when a trajectory's single read is the correct one. On five agentic benchmarks, SkillGate lifts a 9B policy from $40.8\%$ to $53.2\%$ trial success, well ahead of the identical budget spent on outcome reward alone, while cutting exposure to misleading skills by two thirds and reading *fewer* skills.[1]

## 1 INTRODUCTION

Large Language Model (LLM) agents use *skills*, reusable procedural guides, to supply task-specific knowledge during multi-turn tool use (Yao et al., 2022; Wang et al., 2023; Zheng et al., 2026; Gao et al., 2026). A skill is advertised by its name and a short description, and the agent opens its full instructions on demand. As skill libraries grow, agents must learn both which skill to read and how to execute the task with it. Yet a correct choice can be followed by unsuccessful execution: an agent may select the right PDF-processing skill but fail because a later command uses the wrong arguments. The final task reward evaluates the combined result, leaving the quality of the choice entangled with what happens afterwards. This creates a learning challenge beyond supplying or retrieving useful skills: teaching the agent to make reliable selections within its own execution loop while continuing to learn task execution.

Recent work improves skill access and use through two complementary routes. One makes candidate skills easier to discover and distinguish, by refining their routing descriptions (Gao et al., 2026; Wang et al., 2026b) or improving retrieval and ranking (Zheng et al., 2026; Xiao et al., 2026; Kang et al., 2026). These methods address the quality and scale of the skill interface presented to an agent. Another route uses task-outcome rewards to improve skill-augmented behaviour, combining skill construction and reuse with policy optimisation (Xia et al., 2026; Shi et al., 2026; Yao et al., 2026). Neither route, however, gives the policy's own skill choice a training signal of its own: the first acts before the agent runs, and the second credits the choice only through how the whole task turns out.

Despite these advances, *in-policy skill selection*, the policy's own choice of which skill file to read during task execution, remains challenging in long-horizon agents. In this setting, the policy issues the read as an ordinary tool call, and the chosen skill conditions the work that follows. Candidates can look alike, with only a few generated tokens separating the right skill name from a plausible

[1]Code is available in anonymized form at `https://anonymous.4open.science/r/skillgate-anon-2281/`; trained checkpoints and the asset pack are released upon publication.



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.

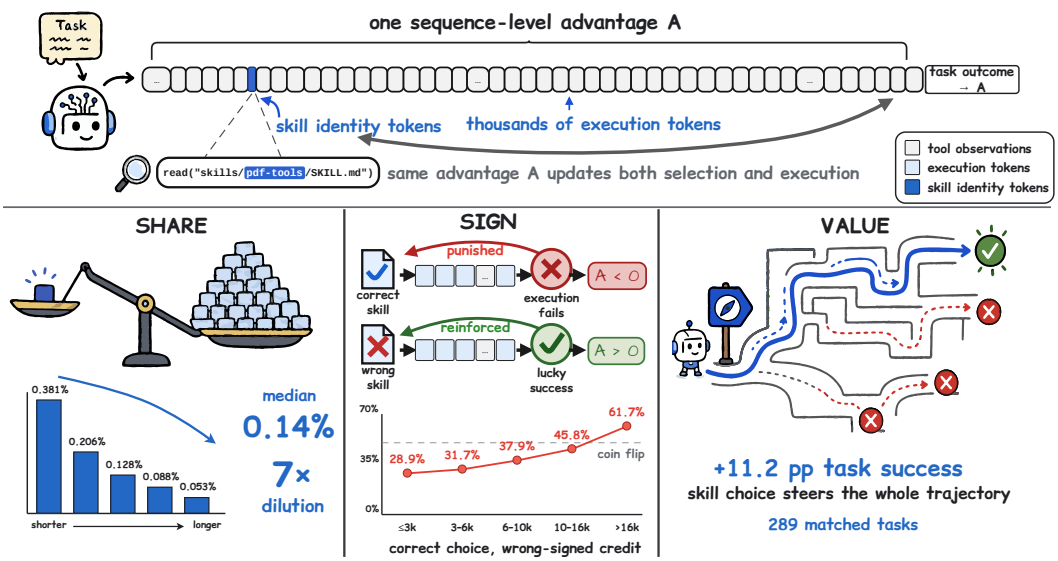


Figure 1: Selector credit starvation. One broadcast advantage updates both the few tokens choosing a skill and the thousands executing the task (*top*); on 12,800 training trajectories, the choice's loss share dilutes 7× with length (*Share*), its credit is increasingly wrong-signed (*Sign*), yet matched prompt groups show a +11.2 pp success gap for the correct read (*Value*).

wrong one. The obvious way to train this ability is the objective agentic RL already runs: present the candidates, reward the task outcome, and let the policy gradient reach the read (Shao et al., 2024; Schulman et al., 2017; DeepSeek-AI, 2025). We show that in our long-horizon setting this recipe suffers from a structural credit-assignment bottleneck, which we name *selector credit starvation*: the few tokens naming the skill receive little loss weight, and much of their credit conflicts with selection correctness (Fig. 1). Auditing 12,800 training trajectories from our outcome-only GRPO baseline makes this bottleneck concrete in three respects. **Share**: among trajectories that read a skill, skill-name tokens receive a median 0.14% of the trajectory's loss weight. **Sign**: 39% of trajectories reading the correct skill assign it a negative advantage because their task scores fall below the prompt-group mean. **Value**: an observational comparison within prompt groups links correct reads to an 11.2 pp higher task success rate. As trajectories lengthen, the selection loss share falls and negative advantages on correct reads become more frequent. Thus, a decision associated with better task performance can receive little training weight and credit that conflicts with its local correctness.

This paper takes a different perspective: in-policy skill selection should be optimised on its own, not left buried in the noise of everything else the trajectory does. The two decisions are judged by different evidence. Whether the right file was named is knowable from the slate's oracle label alone, while whether the work was done well is knowable only from the outcome, so pooling them into one advantage lets each corrupt the other's signal. What is needed is therefore not a better reward, nor a finer time-resolution of the same reward, but a *separation of credit*: give the choice its own objective, computed where the choice is visible, and confine the outcome to the tokens that actually executed.

Building on this perspective, we propose SkillGate, which instantiates the partition as two disjoint credit channels inside one policy-gradient update: (i) a *task channel* carrying the usual group-normalised outcome advantage, with the entire skill-read tool call removed so that no task outcome can revise a selection; and (ii) a *selector channel* applying an action-local advantage to exactly the tokens naming the skill, positive only when the trajectory issued a single read and it was the correct one, centred over the prompt group's read actions. The two supports are disjoint by construction, and both channels carry equal loss weight per batch with each read action taking an equal share, so a selection decision's weight no longer depends on the length of the trajectory containing it. Our contributions are summarised as follows:

- We identify *selector credit starvation* as the root cause of why in-policy skill selection resists outcome-only RL, and, to our knowledge, are the first to measure it on a real run's own training

artifacts: the loss-weight share, sign-error rate and within-group signal-to-noise of the selection decision all degrade monotonically with trajectory length, while the decision's task value does not.

- We propose SkillGate, which separates execution and skill-selection credit within one policy-gradient update. It excludes read calls from task credit, applies a group-centred single-oracle utility only to skill-name tokens, and gives each credited read equal total weight, so selection training is not diluted by long trajectories.
- We conduct experiments on five agentic benchmarks, demonstrating gains in both skill selection and task success. SkillGate achieves $53.2\%$ trial success versus $47.0\%$ for outcome-only RL and $46.0\%$ for task masking alone under the same training budget. It also outperforms supervised selection, preference learning and external routers, while reducing misleading-skill exposure by two thirds and reading fewer skills.

## 2 Problem Setup

**Skill slate.** A skill $s$ is a triple $(\mathrm{name}(s), \mathrm{desc}(s), \mathrm{body}(s))$: an identifier, a one-line description, and a `SKILL.md` body of typically a few thousand tokens. Only name and description are shown up front. For each task the agent sees a *slate* of $K$ candidates, listed in the system prompt and materialised as files in the sandbox. So that the slate covers the situations a deployed agent meets, it mixes four kinds of candidate: one *oracle* $s^\star$, written for the task and verified to solve it; *misleading* hard negatives, topically adjacent to the task but functionally wrong; and *relevant* and *irrelevant* bystanders drawn from a public skill library. Nothing is enforced: the agent is not told which candidate is the oracle, is not required to read anything, and may read as many as it likes, reading being an ordinary tool call against a sandbox path. Every number in this paper is measured under this condition, which we call the *standard mixed slate*.

**In-policy selection and the training signal.** The agent solves each task in a multi-turn tool-use loop (Yao et al., 2022) under a fixed turn and time budget, and the skill choice happens inside this loop: the policy $\pi_\theta$ itself decides whether to read a candidate and which one, by writing a tool call that opens its `SKILL.md`, and no external router chooses for it. Training has two sources of information. The only reward is the task score $R(\tau) \in [0, 1]$, which the benchmark's own tests or grader return at the end of a trajectory $\tau$. During training, an oracle label identifies the correct skill $s^\star$ but is never included in the policy's input. SkillGate uses this label to compute selection credit, while outcome-only RL uses only the task score. As usual, only the tokens the policy generates are trained: the mask $m_\tau^{\mathrm{base}}$ is 1 on them and 0 on observations.

**Selection and execution tokens.** A *read action* $a$ opens a file under a skill directory through a tool or shell command; $A(\tau)$ lists these actions in trajectory order. Using the training tokenizer, we record each read's full *call span* $C(a)$ and its *identity span* $I(a) \subseteq C(a)$: the skill name in the path, such as `pdf-tools` in `/skills/pdf-tools/SKILL.md`. *Selection tokens* are the identity spans; *execution tokens* are assistant tokens outside all read calls. Attribution uses only the agent's generated calls, excluding paths echoed in observations.

## 3 SkillGate

SkillGate addresses the credit-assignment problem that ordinary RL faces in the skill setting by separating credit into two channels. Outcome-only GRPO gives every token of a trajectory the same sequence-level advantage, so the few tokens that choose the skill are judged by how the whole task went. SkillGate instead gives the two kinds of token of Section 2 their own credit: execution tokens receive the usual task advantage, and the tokens that name the chosen skill receive a separate advantage computed from the choice alone, not from the outcome (Fig. 2). The two channels never touch the same token and are optimised in one policy-gradient update. Rollout, reward and optimiser are unchanged; this section specifies which tokens each advantage reaches and with what weight.

### 3.1 Token-Level Credit Assignment

SkillGate splits the trained tokens of a trajectory along the select-execute spans:

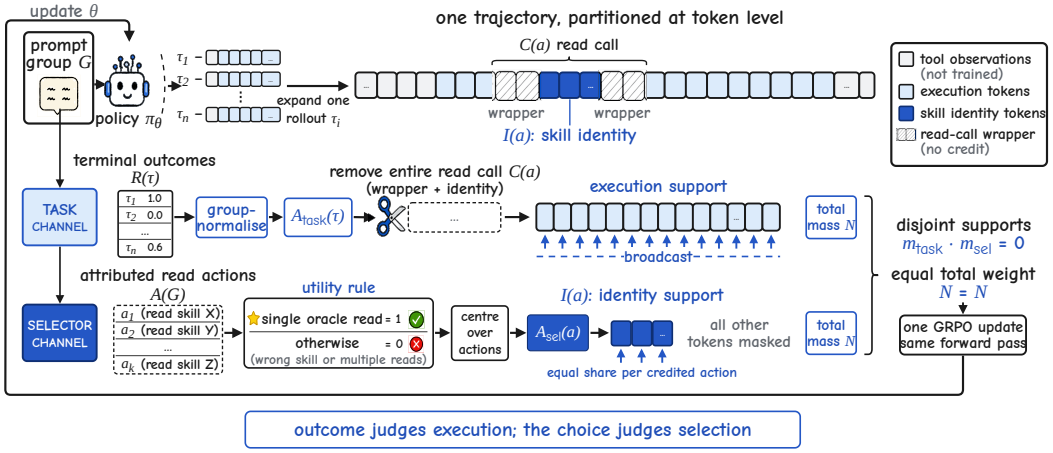


Figure 2: SkillGate overview. The task channel group-normalises terminal outcomes into $A^{\text{task}}$ and broadcasts it over execution tokens with the entire read call cut out; the selector channel scores each selection with the clean single-oracle utility and places $A^{\text{sel}}$ on the identity tokens alone. The supports are disjoint and each channel's token weights sum to $N$ before $\lambda$ is applied; both enter one GRPO update.

- **Selection tokens**, $\bigcup_{a \in A(\tau)} I(a)$: the tokens that write a skill's name. They alone determine which file is read, so they carry the whole selection decision.
- **Execution tokens**, every assistant token outside $\bigcup_{a \in A(\tau)} C(a)$: the reasoning, the other tool calls, the work.

The remaining tokens of each read call, the tool name and the path around the skill name, are trained by neither channel: they express the decision to read at all, which the task outcome should not revise and the selector term does not target. Skill bodies enter as observations and are never trained. Training proceeds in prompt groups: for each task the policy produces a group $G$ of $n$ independent rollouts, and each kind of token earns its own credit:

- **Outcome credit** judges execution by whether the task succeeded, relative to the group: the usual group-normalised GRPO advantage (Shao et al., 2024), $A^{\text{task}}(\tau) = \big(R(\tau) - \mu_G\big)/\big(\sigma_G + \epsilon\big)$, constant along the trajectory, where $\mu_G$ and $\sigma_G$ are the mean and standard deviation of the task scores $\{R(\tau)\}_{\tau \in G}$ and $\epsilon$ is a small constant.
- **Selection credit** judges each read action $a$, which we call a selection, by the skill it chose and nothing else. Writing $A(G)$ for all selections in the group, a selection's utility asks one question and its advantage is the group-centred answer:

$$u(a) = \begin{cases} 1, & |A(\tau_a)| = 1 \text{ and } a \text{ reads } s^\star, \\ 0, & \text{otherwise,} \end{cases} \qquad A^{\text{sel}}(a) = u(a) - \tfrac{1}{|A(G)|} \textstyle\sum_{a' \in A(G)} u(a'), \quad (1)$$

  where $\tau_a$ is the trajectory containing $a$. No standard-deviation normalisation is applied: a group contains few selections, and dividing by their spread would amplify noise more than calibrate scale. Reading the oracle and then three more candidates scores zero, as does reading it twice: the single-read requirement is what stops the policy from buying credit by reading everything.

Equation (1) has three consequences. First, the baseline averages over selections rather than trajectories, so a trajectory that reads four candidates contributes four zero-utility selections, each of them penalised. Second, the advantages of a group's selections sum to zero, so the selector term never rewards reading as such; it only moves probability toward clean oracle reads. Third, when every selection in a group earns utility 1, or none does, all advantages are zero and the channel stays silent: selection is trained only where the group's choices differ.

### 3.2 Credit Masking and Normalisation

The two channels are kept apart by construction. The task channel removes every read call from the assistant mask in full, including the function name and the path, not only the skill name, so that

no task outcome can revise a selection; what remains is exactly the execution tokens. The selector channel trains only the identity spans of credited selections, those with $A^{\mathrm{sel}}(a) \neq 0$, and every token of a span carries its selection's advantage. Because identity spans lie inside the removed call spans, the two supports are disjoint; the implementation checks this at mask construction, at context-parallel sharding and in the loss itself, and any violation aborts training rather than degrading it silently. A read of a file outside the slate is removed from the task mask like any other read and has utility zero, so a lucky outcome can never turn it into a positive update.

Token weights then set how loudly each channel speaks, and the two need rescaling for opposite reasons. Let $m_t^{\mathrm{base}}$ be the original assistant-token loss mask and write $N = \sum_t m_t^{\mathrm{base}}$ for its total weight in a batch. After deleting all read calls, let $N_{\mathrm{task}} = \sum_t m_t^{\mathrm{task}}$, and let $M$ be the number of credited selections. We set

$$w_t^{\mathrm{task}} = \frac{N}{N_{\mathrm{task}}} m_t^{\mathrm{task}}, \qquad w_t^{\mathrm{sel}} = \frac{N}{M\,|I(a)|} \quad \text{for } t \in I(a), \tag{2}$$

with zero selector weight elsewhere. When $M > 0$, $\sum_t w_t^{\mathrm{task}} = \sum_t w_t^{\mathrm{sel}} = N$, which is the "$N = N$" in Fig. 2; when $M = 0$, the selector channel is silent. Deleting read calls therefore does not quietly lower the task channel's effective learning rate, while the handful of selector tokens no longer reproduces the starvation of Fig. 1. The equality concerns total token-weight mass, not equality of the realised loss values or gradient norms; the intended relative coefficient is still set by $\lambda$ in Eq. (3). It also gives every credited selection the same total weight $N/M$, independent of trajectory length and of how many tokenizer pieces its skill name occupies. The implementation verifies both sums numerically every batch.

### 3.3 Training Objective

Let $r_t(\theta) = \pi_\theta(y_t \mid y_{<t}) / \pi_{\mathrm{rollout}}(y_t \mid y_{<t})$ be the importance ratio of token $y_t$ against the rollout policy, and $\ell^{\mathrm{clip}}(r, A) = -\min\big(rA,\ \mathrm{clip}(r, 1-\varepsilon_{\mathrm{lo}}, 1+\varepsilon_{\mathrm{hi}})A\big)$ the clipped GRPO surrogate (Shao et al., 2024) with clip thresholds $\varepsilon_{\mathrm{lo}}, \varepsilon_{\mathrm{hi}}$. SkillGate optimises

$$\mathcal{L}(\theta) = \underbrace{\sum_t w_t^{\mathrm{task}}\, \ell^{\mathrm{clip}}\big(r_t(\theta), A^{\mathrm{task}}(\tau_t)\big)}_{\text{execution}} + \lambda \underbrace{\sum_t w_t^{\mathrm{sel}}\, \ell^{\mathrm{clip}}\big(r_t(\theta), A^{\mathrm{sel}}(a_t)\big)}_{\text{selection}} + \beta\, \mathcal{L}_{\mathrm{KL}}, \tag{3}$$

where $\tau_t$ and $a_t$ are the trajectory and the selection containing token $t$, $\lambda$ is the selector coefficient, and $\mathcal{L}_{\mathrm{KL}}$ is a KL penalty of weight $\beta$ against a frozen reference (values in Section 4.1). Both sums run over the same forward pass and both terms are on-policy: the selector term is not behaviour cloning, not cross-entropy and not a reward bonus, and it never sees a token the policy did not itself generate.

Together, the two channels credit each token only for what it controls. Take a group whose choices differ and the four combinations of choice and outcome. A trajectory whose only read is the oracle has that choice reinforced whether the task then succeeds or fails, and a trajectory that reads a misleading candidate has that choice penalised in both cases; independently, its execution tokens are reinforced if the task succeeds and penalised if it fails, whatever was read. A negative selector advantage therefore always means that the wrong skill or more than one skill was read, never that the oracle's instructions failed to help.

## 4 Experiments

### 4.1 Experimental Setup

**Benchmarks and protocol.** We evaluate on five agentic benchmarks — Claw-Eval, SkillsBench, SETA, SWE and Terminal-Bench 2.0 — under the standard mixed slate of Section 2 with $K = 16$: one oracle, five misleading hard negatives, and five relevant plus five irrelevant skills drawn from a public library of 2,045 community skills. Task outcome in Table 1 is reported on a **385-trial** protocol (four repeats for each of 56 non-Claw tasks and one trial for each of 161 Claw-Eval tasks; Appendix B). Training uses 491 tasks containing no Claw-Eval task, and training and evaluation oracles are disjoint as skill identities. Slate construction and budgets are given in Appendix A.

Table 1: Trial success (%) on the standard mixed slate, and read behaviour: the fraction of trials reading at least one oracle or misleading skill. Best 9B-scale per column in **bold**.

| | Trial success (%) | | | | | | Read behaviour (%) | |
|---|---|---|---|---|---|---|---|---|
| Method | Claw-Eval | SkillsBench | SETA | SWE | TB2 | Overall | Oracle ↑ | Mislead. ↓ |
| *Frontier models (capability ceiling, not controlled)* | | | | | | | | |
| DeepSeek-V4-Flash | 70.2 | 15.6 | 66.7 | 55.0 | 46.9 | 61.0 | 40.8 | 23.9 |
| GLM-5 | 75.8 | 15.6 | 59.2 | 45.0 | 56.2 | 60.8 | 39.0 | 26.2 |
| DeepSeek-V4-Pro | 70.8 | 12.5 | 60.8 | 55.0 | 62.5 | 60.5 | 38.2 | 20.8 |
| Kimi-K2.6 | 71.4 | 12.5 | 56.7 | 55.0 | 71.9 | 60.3 | 45.7 | 21.0 |
| Qwen3.5-397B-A17B | 60.9 | 15.6 | 54.2 | 45.0 | 40.6 | 51.7 | 16.1 | 11.2 |
| DeepSeek-V3.2 | 62.1 | 12.5 | 42.5 | 32.5 | 46.9 | 47.5 | 29.4 | 28.3 |
| *No adaptation* | | | | | | | | |
| Qwen3.5-9B | 44.7 | 0.0 | 24.2 | 15.0 | 9.4 | 28.6 | 5.7 | 8.9 |
| Qwen3.5-27B | 54.0 | 9.4 | 42.5 | 40.0 | 34.4 | 43.6 | 5.7 | 8.9 |
| *Supervised selection, from SFT* | | | | | | | | |
| SFT (RL init) | 50.9 | 6.2 | 40.0 | 45.0 | 21.9 | 40.8 | 37.9 | 61.8 |
| Selection BC | 52.2 | **15.6** | 43.3 | 50.0 | 34.4 | 44.7 | 71.4 | 31.4 |
| SelSkill-DPO | 52.8 | 0.0 | 47.5 | 60.0 | **37.5** | 46.2 | 66.1 | 51.8 |
| *RL from SFT, identical budget* | | | | | | | | |
| Skill-free RL | 55.9 | 9.4 | 47.5 | 42.5 | 31.2 | 46.0 | 35.4 | 55.0 |
| SkillRL (outcome only) | 57.1 | 3.1 | 50.0 | 45.0 | 31.2 | 47.0 | 54.3 | 69.6 |
| Skill1 (no distill) | 57.1 | 9.4 | 38.3 | 52.5 | 31.2 | 44.7 | 53.5 | 45.5 |
| Task-mask only | 54.7 | 9.4 | 48.3 | 45.0 | 31.2 | 46.0 | 48.9 | 73.6 |
| **SkillGate** | **60.2** | **15.6** | **54.2** | **65.0** | **37.5** | **53.2** | **83.9** | **21.8** |

**Baselines.** **Qwen3.5-9B** and **Qwen3.5-27B** are the untrained backbones; six **frontier models**, queried through their providers' APIs, are absolute-scale references, not controlled comparisons (Appendix B.1). **Selection BC** teacher-forces the first turn to `read(oracle)`; **SelSkill-DPO** adapts the preference method of Chen et al. (2026) to our 16-way slate with DPO (Rafailov et al., 2023). **Skill-free RL** trains without skills; **SkillRL (outcome only)** trains on the same slate with task reward alone — an outcome-only recipe inspired by Xia et al. (2026) and the controlled comparison for SkillGate, sharing its initialisation, data, steps and hyperparameters and differing in where the gradient lands; **Skill1 (no distill)** adapts the joint selection-and-use component of Shi et al. (2026) at the same budget; **Task-mask only** is SkillGate with $\lambda = 0$. Appendix A states each adaptation.

**Metrics and training.** The primary metric is **trial success** (%), the fraction of trials whose terminal verifier passes. Read behaviour is summarised by five statistics. *Oracle exposure* and *misleading exposure* are the fractions of trials that read at least one skill of that category, so a trial reading both counts in both. $P(\text{oracle} \mid \text{read})$ is the share of reading trials that read the oracle. *Reads/trial* counts distinct skills read. *Clean single-oracle* is the fraction of trials whose only attributed read is the oracle. All RL-trained rows start from the same Qwen3.5-9B SFT checkpoint and run 100 steps of on-policy GRPO (Shao et al., 2024) with 8 rollouts per prompt, learning rate $10^{-6}$ and KL weight $\beta = 3 \times 10^{-5}$; SkillGate sets $\lambda = 0.20$. Each configuration is a single run, so uncertainty is quantified by task-level bootstrap rather than seed replication, and small differences are read as directional.

### 4.2 Main Results

**SkillGate is the best 9B model and outperforms models with hundreds of billions of parameters.** As shown in Table 1, SkillGate outperforms the shared SFT initialisation, the controlled outcome-only RL run, and the supervised, preference-based and Skill1 alternatives, and is best or tied best among the 9B-scale methods on every benchmark; its overall success also exceeds Qwen3.5-397B-A17B and DeepSeek-V3.2. The comparison with SkillRL (outcome only) is the most informative: the two runs share initialisation, data, update budget and hyperparameters, so their difference isolates the value of token-local selection credit. The gap is also statistically reliable: on the evaluation of Table 2, which runs each task four times, SkillGate leads $50.0\%$ to $42.1\%$ and a paired bootstrap over tasks puts the difference's 95% interval at $[+2.1, +14.3]$ pp (Appendix C.3). SkillGate also leads the 9B rows on Claw-Eval despite seeing neither Claw tasks nor the evaluation oracles in training, supporting transfer rather than a memorised task–skill mapping.

Table 2: Credit-design ablation on the five benchmarks, four trials per task: five placements of the selector signal from the same initialisation and steps. Best results in **bold**.

| Design | Where the credit lands | Trial succ. (%) | Clean single-oracle (%) | Oracle (%) | Mislead. (%) | Reads/ trial |
|---|---|---|---|---|---|---|
| SkillRL (outcome only) | nowhere (anchor) | 42.1 | 21.4 | 54.3 | 69.6 | 1.88 |
| Group-level regret | the prompt group | 41.8 | 15.7 | 33.6 | 47.5 | 1.23 |
| Trajectory bonus | the whole trajectory | 41.8 | 33.9 | 47.9 | 55.4 | 1.33 |
| Action credit | first oracle read | 45.0 | 64.6 | 80.0 | 32.5 | 1.26 |
| **SkillGate** | the *only* read, if oracle | **50.0** | **75.4** | **83.9** | **21.8** | **1.11** |

Table 3: External selection on the five benchmarks, four trials per task. Routers advertise one chosen skill to the frozen SFT executor. †Top-1 is over the 70 routing decisions for routers but over SkillGate's first attributed read in each trial — not directly comparable.

| Setting | Trial succ. (%) | Top-1 oracle† (%) | Oracle (%) | Mislead. (%) | Reads/ trial |
|---|---|---|---|---|---|
| SFT, standard mixed slate | 37.1 | — | 37.9 | 61.8 | 1.34 |
| SFT + "read only one" prompt | 35.0 | — | 41.4 | 59.3 | 1.35 |
| SFT-9B router → SFT | 40.7 | 60.0 | 57.5 | 36.8 | 0.94 |
| Qwen3.5-27B router → SFT | 36.8 | 68.6 | 68.2 | 24.3 | 0.95 |
| Qwen3 reranker top-1 → SFT | 31.8 | 27.1 | 24.6 | 60.4 | 0.89 |
| **SkillGate**, standard mixed slate | **50.0** | 79.6 | 83.9 | 21.8 | 1.11 |
| *Ceiling: only the correct skill is advertised* | | | | | |
| oracle-only → SFT | 48.2 | 100.0 | 93.6 | 0.0 | 0.94 |
| oracle-only → SkillGate | 52.9 | 100.0 | 93.6 | 0.0 | 0.94 |

**Reliable skill access is consequential, not merely a behaviour metric.** Advertising only the oracle raises the frozen SFT executor by eleven points (Section 4.4), so access to the correct skill materially affects task success. Yet outcome-only RL raises both oracle and misleading exposure, learning to read more rather than to discriminate, and masking read calls from the task loss alone does not fix this. SkillGate instead produces the strongest joint shift toward oracle and away from misleading reads, and converts it into task success. Selection BC improves the choice but not the result to the same degree: reliable agents need selector and execution learning together.

**Model scale alone does not solve in-policy skill selection.** The frontier rows are scale references rather than controlled comparisons, yet they separate general capability from skill selection: the four strongest systems still lead on task success, but none of the six reads the oracle on even half of the trials. Capability compensates for many missed skills; it does not produce a reliable selector. Choosing the right skill is a distinct ability that has to be trained, not one that emerges from scale.

### 4.3 Which part of the credit design matters

This ablation asks which part of the credit design does the work. The five rows of Table 2 train from the same initialisation for the same 100 steps and differ only in where the selector signal lands, from nowhere (SkillRL (outcome only), the anchor) to the trajectory's only read when it is the oracle (SkillGate). Each coarser placement fails in a way that identifies what the next must fix. Group-level regret never touches the tokens that name the skill, and its oracle exposure falls below even the anchor's: a group-constant shift cancels within the group and cannot say which member chose well. The trajectory bonus pushes the policy to read, not to read correctly, so oracle and misleading exposure rise together. Action credit finally lands the signal on the identity tokens and moves both behaviour columns sharply, but it never penalises reading *more*: a trajectory that reads the oracle and then three other skills keeps its full credit, so clean single-oracle behaviour stalls at $64.6\%$. Requiring the read to be the trajectory's only one closes that gap ($75.4\%$) and converts it into task success. The gain comes from choosing better, not from reading more: SkillGate reads the oracle most often, reads misleading candidates least, and reads the fewest skills overall.

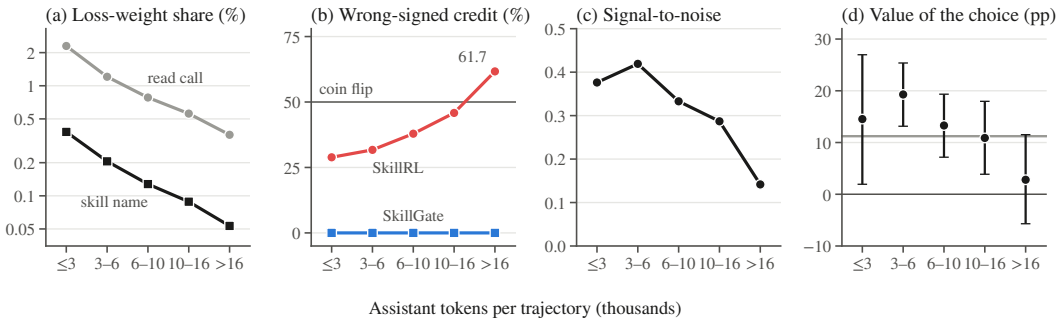


Figure 4: Offline audit of SkillRL (outcome only)'s 12,800 training trajectories by assistant length. (a) Median loss-weight share of the whole read call and of the skill-name tokens (log scale). (b) Oracle reads whose task advantage is negative; SkillGate's zero holds by construction. (c) Within-group signal-to-noise of the oracle-read advantage gap. (d) Within-group success gain from reading the oracle, with task-clustered 95% intervals; the gray line pools all bins.

### 4.4 Would a separate selector be enough?

This experiment asks whether selection is better solved outside the policy. Table 3 freezes the SFT executor and varies who chooses: the executor itself, with and without a "read only one" prompt; three external selectors (an SFT-9B router, a Qwen3.5-27B router and a retrieval reranker), each advertising its single chosen skill; SkillGate, choosing by itself from the full slate; and an oracle-only injection that advertises the correct skill alone, a ceiling that requires knowing the answer.

Three conclusions follow. First, prompting does not fix selection: instructing the model to read exactly one skill changes neither its reading nor its success, so the failure is not instruction following. Second, a better router is not a better agent: the 27B router routes more accurately than the SFT router yet yields *lower* success, because routing accuracy, whether the executor reads what it is handed, and how well it then executes are three different quantities; the reranker, scoring candidates independently rather than comparing them, is what a slate of near-duplicates defeats. Third, SkillGate choosing from the full slate by itself (50.0%) already beats the frozen SFT executor that is handed only the correct skill (48.2%). Handed only the correct skill as well, SkillGate reaches 52.9%: it uses a given skill better than the SFT executor, so training selection did not cost execution ability, and knowing the answer in advance would add only 2.9 pp to what SkillGate achieves by choosing for itself.

### 4.5 What selection purity costs

This section asks whether SkillGate's purer selection is bought with extra inference cost. On the trials of Table 2, we compare SkillGate with SkillRL (outcome only) on four per-trial costs: distinct skills read, agent turns, cumulative input tokens over all turns, and output tokens; Fig. 3 plots the change in the first two. SkillGate reads 41% fewer distinct skills, takes 5% fewer turns and consumes 7.5% fewer input tokens; only its output tokens rise, by 21%, consistent with reasoning before a single read rather than trial and error across many. Purer selection therefore costs no extra inference: the purer selector is also the cheaper agent.

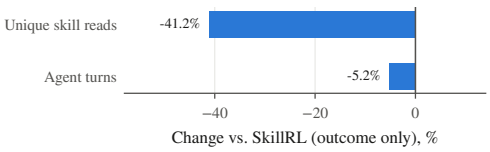


Figure 3: Inference cost relative to SkillRL (outcome only).

### 4.6 Why outcome-only credit cannot teach the choice

This section tests the diagnosis that motivated SkillGate: under outcome-only credit, the skill choice is starved of training signal. We audit the training run of SkillRL (outcome only) offline, without new training (Appendix D). We measure four quantities as a function of trajectory length: the share of a trajectory's loss weight carried by the skill-name tokens; the fraction of oracle-reading trajectories that inherit a negative advantage; the within-group signal-to-noise of the advantage gap between rollouts that read the oracle and rollouts that do not; and the success gain from reading the oracle within the same prompt group. Figure 4 shows the results, which confirm the diagnosis. The skill-

name tokens carry a median $0.14\%$ of the loss weight, $7\times$ less on the longest trajectories than on the shortest (Fig. 4a). Of the oracle-reading trajectories, $39\%$ inherit a negative advantage, rising monotonically from $29\%$ to $62\%$ with length (Fig. 4b), and the signal-to-noise falls from about $0.4$ to $0.14$ (Fig. 4c). Yet reading the oracle raises success by $11.2$ pp within the same group (95% CI $[+7.3, +15.1]$; Fig. 4d); this comparison is observational, but its causal counterpart, the oracle-only injection of Table 3, gives the frozen executor a nearly identical $11.1$ pp. A valuable decision thus receives a vanishing and increasingly wrong-signed signal. SkillGate removes the wrong sign by construction: with the read call excluded from the task loss, no oracle skill-name token receives a negative task gradient at any length (Fig. 4b).

## 5 Related Work

**Selecting a skill from a large library.** Agent frameworks expose skills by *progressive disclosure*, a name and one-line description before the body, and the difficulty concentrates at that choice. Many public skills carry no usable routing description (Gao et al., 2026), and benchmarks measure the resulting confusion (Kang et al., 2026; Su et al., 2026) along with skill triggering and use, large tool sets and tool necessity (Han et al., 2026; Esfandiarpoor et al., 2025; Sun et al., 2026). Controlled studies locate the failure: larger libraries degrade agents through *skill shadowing* (Song & Wei, 2026), planted decoys expose systematic selection errors (Anand & Chattaraj, 2026), models attend to the right tool yet pick another (Chen, 2026), and shortlist size and disclosure depth both matter (Repantis et al., 2026; He et al., 2026b). Responses either improve routing before execution, by rewriting skill representations (Wang et al., 2026b; Gao et al., 2026), calibrating retrieval scores (Xiao et al., 2026) or retrieving and reranking candidates (Lewis et al., 2020; Qin et al., 2023; Patil et al., 2023; Zheng et al., 2026), or train selection separately, through a KL-regularised ranking objective (Zou et al., 2025) or a dedicated selection agent with its own advantage (Fu et al., 2026). SkillGate changes neither the descriptions nor the architecture: the policy's own mid-episode read becomes a trained decision, learned from the rollouts that measure execution.

**Where the training signal lands in a multi-turn trajectory.** Outcome-rewarded policy gradients (Schulman et al., 2017; Shao et al., 2024) attach one advantage to every token of a rollout, a uniformity increasingly identified as the bottleneck for agents (Zhang, 2026): reward-correlated signal concentrates on short action spans (He et al., 2026a), and the signal-to-noise of trajectory-level estimators decays as consequential turns become rarer (Pernot & Retault, 2026). Remedies refine the unit of credit — steps (Lightman et al., 2023), turns (Tao et al., 2026; Liao et al., 2026; Li et al., 2025), tool-use segments (Kumar et al., 2026), procedures (Wang et al., 2026a), counterfactual tokens (Meng & Chen, 2026), reweighted tokens or segments (Mou et al., 2026; Li et al., 2026), or hierarchical transitions (Luo et al., 2025) — or add tool-specific signals: decoupled retrieval and execution objectives (Huang et al., 2026), information-gain rewards for search steps (Liang et al., 2026), tool-contribution and tool-efficiency channels (Feng et al., 2026; Yan et al., 2026), and calibrated per-turn rewards (Modecrua et al., 2026). Skill-oriented RL co-evolves skills and policies (Xia et al., 2026; Shi et al., 2026; Yao et al., 2026), internalises skills from paired rollouts (Lin et al., 2026), or learns when to invoke one (Chen et al., 2026). These methods refine credit for other events; none isolates the identity tokens of a mid-episode multiway skill read as a disjoint, length-independent loss channel, and because SkillGate's selector utility is not derived from the outcome, it also sidesteps the impossibility result for outcome-reward weighting (Ding et al., 2026).

## 6 Conclusion

Outcome-only RL entangles skill selection with later execution, leaving the choice with weak or conflicting credit. SkillGate addresses this credit starvation with disjoint token-level objectives for execution and selection, oracle-based selection utility, and length-independent weighting. It makes skill choice an explicit training target within the execution policy, supporting credit separation for distinct decisions in long-horizon agents. Experiments on five benchmarks show improvements in both selection and task success.

## AI USE STATEMENT

In this work, we used generative AI tools to assist with editing prose, with literature search, and with writing analysis and infrastructure code for the experiments. All AI-assisted text was reviewed and revised by the authors, all citations were verified against the cited papers, and all AI-assisted code was reviewed and validated by the authors through the smoke tests and frozen evaluation protocol described in the paper. Research ideas, the method, and the experimental design are the authors' own. We take responsibility for the final content of this work, including text, claims and artifacts produced with the aid of generative AI.

## REFERENCES

Atul Anand and Sourav Chattaraj. Diagnosing tool-selection reasoning in llm agents with canary tools. *arXiv preprint arXiv:2608.04719*, 2026.

Chishui Chen, Jiaye Lin, Te Sun, Junxi Wang, Yi Yang, Cong Qin, Yangen Hu, Lu Pan, and Ke Zeng. Skill or skip? learning selective skill invocation in agentic tasks via dual-granularity preference learning. *arXiv preprint arXiv:2606.00510*, 2026.

Shiyang Chen. Looking is not picking: An attention-segment account of tool-selection failures in llm agents. *arXiv preprint arXiv:2606.16364*, 2026.

DeepSeek-AI. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Fei Ding, Yongkang Zhang, Yuhao Liao, Zijian Zeng, and Huiming Yang. On the impossibility of unbiased and length-invariant policy optimization with outcome rewards. *arXiv preprint arXiv:2607.23364*, 2026.

Reza Esfandiarpoor, Vishwas Suryanarayanan, Stephen H. Bach, Vishal Chowdhary, and Anthony Aue. Themcpcompany: Creating general-purpose agents with task-specific tools. *arXiv preprint arXiv:2510.19286*, 2025.

Mingkuan Feng, Jinyang Wu, Hao Gu, Fangrui Lv, Ruihan Jin, Chuyuan Zhang, Zhengqi Wen, and Jianhua Tao. Taco: Tool-augmented credit optimization for agentic tool use. *arXiv preprint arXiv:2606.30251*, 2026.

Dayuan Fu, Mohan Jiang, Tongyu Wang, Dian Yang, Jiarui Hu, Liming Liu, Jinlong Hou, and Pengfei Liu. davinci-kernel: Co-evolving skill selection, summarization, and utilization via rl for gpu kernel optimization. *arXiv preprint arXiv:2606.16497*, 2026.

Yudong Gao, Zongjie Li, Yuanyuan Yuan, Zimo Ji, Pingchuan Ma, and Shuai Wang. Skillreducer: Optimizing llm agent skills for token efficiency. *arXiv preprint arXiv:2603.29919*, 2026.

Jinyi Han, Yuanjian Xu, Ying Liao, Xinyi Wang, Zishang Jiang, Zixiang Di, Fanyang Lu, Zhichao Hu, and Yanghua Xiao. Skill-use: Can llms actually use skills in agentic harnesses? *arXiv preprint arXiv:2608.04828*, 2026.

Langzhou He, Junyou Zhu, Yue Zhou, Zhengyao Gu, Junhua Liu, Wei-Chieh Huang, Henry Peng Zou, David Wipf, Philip S. Yu, and Qitian Wu. Resolving action bottleneck: Agentic reinforcement learning informed by token-level energy. *arXiv preprint arXiv:2605.14558*, 2026a.

Yifeng He, Yinzhe Zhao, Jicheng Wang, and Hao Chen. Is progressive disclosure all you need for long-context agents? *arXiv preprint arXiv:2607.17598*, 2026b.

Shouzheng Huang, Meishan Zhang, Baotian Hu, and Min Zhang. Toolomni: Enabling open-world tool use via agentic learning with proactive retrieval and grounded execution. *arXiv preprint arXiv:2604.13787*, 2026.

Ryangkyung Kang, Hongcheol Cho, and Youngeun Kim. Skillret: A large-scale benchmark for skill retrieval in llm agents. *arXiv preprint arXiv:2605.05726*, 2026.

Abhijit Kumar, Zoey Wu, and Mohit Suley. Knowing when to ask: Segment-level credit assignment for llm tool use. *arXiv preprint arXiv:2605.27788*, 2026.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive nlp tasks. *arXiv preprint arXiv:2005.11401*, 2020.

Junbo Li, Peng Zhou, Rui Meng, Meet P. Vadera, Lihong Li, and Yang Li. Turn-ppo: Turn-level advantage estimation with ppo for improved multi-turn rl in agentic llms. *arXiv preprint arXiv:2512.17008*, 2025.

Sijia Li, Yuchen Huang, Zifan Liu, Yanping Li, Jingjing Fu, Li Zhao, Jiang Bian, Ling Zhang, Jun Zhang, and Rui Wang. Gear: Granularity-adaptive advantage reweighting for llm agents via self-distillation. *arXiv preprint arXiv:2605.11853*, 2026.

Zihan Liang, Yufei Ma, Ben Chen, Zhipeng Qian, Huangyu Dai, Lingtao Mao, Xuxin Zhang, Chenyi Lei, and Wenwu Ou. Ig-search: Step-level information gain rewards for search-augmented reasoning. *arXiv preprint arXiv:2604.15148*, 2026.

Sicong Liao, Zhi Chen, and Yaohua Tang. Tcpo: Turn-level credit policy optimization. *arXiv preprint arXiv:2608.01667*, 2026.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. *arXiv preprint arXiv:2305.20050*, 2023.

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

Xufang Luo, Yuge Zhang, Zhiyuan He, Zilong Wang, Siyun Zhao, Dongsheng Li, Luna K. Qiu, and Yuqing Yang. Agent lightning: Train any ai agents with reinforcement learning. *arXiv preprint arXiv:2508.03680*, 2025.

Zibin Meng and Kani Chen. Craft: Counterfactual credit assignment from free sibling rollouts for self-distilled agentic reinforcement learning. *arXiv preprint arXiv:2606.29476*, 2026.

Wachiravit Modecrua, Krittanon Kaewtawee, Krittin Pachtrachai, and Touchapon Kraisingkorn. Multi-turn reinforcement learning for tool-calling agents with iterative reward calibration. *arXiv preprint arXiv:2604.02869*, 2026.

Chaoli Mou, Zhan Zhuang, Xinning Chen, and Yu Zhang. Beyond uniform credit assignment: Selective eligibility traces for rlvr. *arXiv preprint arXiv:2605.05965*, 2026.

Shishir G. Patil, Tianjun Zhang, Xin Wang, and Joseph E. Gonzalez. Gorilla: Large language model connected with massive apis. *arXiv preprint arXiv:2305.15334*, 2023.

Yann Pernot and Vi Retault. Drowning in routine: Signal dilution in multi-turn agent training. *arXiv preprint arXiv:2606.22164*, 2026.

Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, Sihan Zhao, Lauren Hong, Runchu Tian, Ruobing Xie, Jie Zhou, Mark Gerstein, Dahai Li, Zhiyuan Liu, and Maosong Sun. Toolllm: Facilitating large language models to master 16000+ real-world apis. *arXiv preprint arXiv:2307.16789*, 2023.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *arXiv preprint arXiv:2305.18290*, 2023.

Vyzantinos Repantis, Ameya Gawde, Harshvardhan Singh, and Joey Blackwell. How many tools should an llm agent see? a chance-corrected answer. *arXiv preprint arXiv:2605.24660*, 2026.

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, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Yaorui Shi, Yuxin Chen, Zhengxi Lu, Yuchun Miao, Shugui Liu, Qi GU, Xunliang Cai, Xiang Wang, and An Zhang. Skill1: Unified evolution of skill-augmented agents via reinforcement learning. *arXiv preprint arXiv:2605.06130*, 2026.

Hongwen Song and Song Wei. More skills, worse agents? skill shadowing degrades performance when expanding skill libraries. *arXiv preprint arXiv:2605.24050*, 2026.

Weihang Su, Jianming Long, Qingyao Ai, Qiaozhi He, Yichen Tang, Changyue Wang, Yiteng Tu, Yingbo Wang, and Yiqun Liu. Skill retrieval augmentation for agentic ai. *arXiv preprint arXiv:2604.24594*, 2026.

Chung-En Sun, Linbo Liu, Ge Yan, Zimo Wang, and Tsui-Wei Weng. Llm agents already know when to call tools – even without reasoning. *arXiv preprint arXiv:2605.09252*, 2026.

Leitian Tao, Baolin Peng, Wenlin Yao, Tao Ge, Hao Cheng, Mike Hang Wang, Jianfeng Gao, and Sharon Li. Trace: Turn-level reward assignment via credit estimation for long-horizon agents. *arXiv preprint arXiv:2607.13988*, 2026.

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.

Xucong Wang, Ziyu Ma, Yong Wang, Yuxiang Ji, Shidong Yang, Guanhua Chen, Pengkun Wang, and Xiangxiang Chu. Appo: Agentic procedural policy optimization. *arXiv preprint arXiv:2606.12384*, 2026a.

Zifei Wang, Wei Wen, Qiang Ji, and Ruizhi Qiao. Skills know their neighbors: Cluster-contrastive capability pages for skill retrieval. *arXiv preprint arXiv:2608.04482*, 2026b.

Peng Xia, Jianwen Chen, Hanyang Wang, Jiaqi Liu, Kaide Zeng, Yu Wang, Siwei Han, Yiyang Zhou, Xujiang Zhao, Haifeng Chen, Zeyu Zheng, Cihang Xie, and Huaxiu Yao. Skillrl: Evolving agents via recursive skill-augmented reinforcement learning. *arXiv preprint arXiv:2602.08234*, 2026.

Jinying Xiao, Bin Li, Xiaopeng Li, Jianling Li, Jiacheng Jie, Xiaodong Liu, Ma Jun, Chao Wang, Nyima Tashi, and Jie Yu. Skillsight: Calibrating generic content bias for skill retrieval. *arXiv preprint arXiv:2607.18785*, 2026.

Shilin Yan, Jintao Tong, Hongwei Xue, Xiaojun Tang, Yangyang Wang, Kunyu Shi, Guannan Zhang, Ruixuan Li, and Yixiong Zou. Act wisely: Cultivating meta-cognitive tool use in agentic multimodal models. *arXiv preprint arXiv:2604.08545*, 2026.

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*, 2022.

Zhiyuan Yao, Yuxin Chen, Zhengxi Lu, Zishan Xu, Yueqing Sun, Yifu Guo, Yuquan Lu, Zhengzhou Cai, Kangning Zhang, Zhuowen Han, Zi-Han Wang, Ziang Ye, Qi Gu, Xunliang Cai, Weiwen Liu, and Yongliang Shen. Skillrise: Agentic reinforcement learning for cross-task skill evolution. *arXiv preprint arXiv:2607.26784*, 2026.

Chenchen Zhang. From reasoning to agentic: Credit assignment in reinforcement learning for large language models. *arXiv preprint arXiv:2604.09459*, 2026.

YanZhao Zheng, ZhenTao Zhang, Chao Ma, YuanQiang Yu, JiHuai Zhu, Yong Wu, Tianze Xu, Baohua Dong, Hangcheng Zhu, Ruohui Huang, and Gang Yu. Skillrouter: Skill routing for llm agents at scale. *arXiv preprint arXiv:2603.22455*, 2026.

Jiaru Zou, Ling Yang, Yunzhe Qi, Sirui Chen, Mengting Ai, Ke Shen, Jingrui He, and Mengdi Wang. Autotool: Dynamic tool selection and integration for agentic reasoning. *arXiv preprint arXiv:2512.13278*, 2025.

Appendix A describes the experimental setup: slates and budgets, oracle verification, baselines, training and the agent prompt. Appendix B defines the 385-trial and repeated-trial protocols, gives the raw counts behind Table 1 and states the interface controls for the frontier references. Appendix C reports additional results: the held-out Claw-Eval subset, the full read-behaviour table, the uncertainty of the controlled comparison and a check that selection does not follow description length. Appendix D details the credit-starvation audit of Section 4.6, and Appendix E follows one task through SkillGate and SkillRL (outcome only).

# A Experimental details

**Slate construction and budgets.** Every trial uses the standard mixed slate of Section 2 with $K = 16$ candidates: one *oracle* written for the task and verified to solve it, five *misleading* hard negatives synthesised to be topically adjacent but functionally wrong, and five *relevant* plus five *irrelevant* skills sampled from a public library of 2,045 community skills. Slate order is drawn once per task and then held fixed across methods. Each trial runs under a budget of 30 turns and 850 seconds. Training uses 491 tasks containing no Claw-Eval task, and the training and evaluation oracles are disjoint as skill identities, so no result can be explained by memorising which name goes with which task.

**Oracle verification on Claw-Eval.** The oracle skills of the 147 held-out Claw-Eval tasks were inherited without successful-trajectory provenance, so it was not known whether following each skill would solve its task. Reporting a selection metric against unverified targets would measure the wrong thing. We therefore checked every body against its task, fixtures and grader, and repaired incorrect instructions. Recurring defects included wrong endpoints or response fields, incorrect fixture assumptions, mishandled dates, unsafe operation ordering and omitted deliverables. Skill names and descriptions were left unchanged: the verification changed what a skill teaches, not what it advertises, so the selection input is identical.

**Baselines.** **Qwen3.5-9B** and **Qwen3.5-27B** bracket what the backbone does without adaptation; the 27B row is a capability reference at $3\times$ the parameters. The **frontier references** (DeepSeek-V4-Pro, DeepSeek-V4-Flash, GLM-5, Kimi-K2.6, Qwen3.5-397B-A17B and DeepSeek-V3.2) are served through their providers' APIs with native function calling under the interface controls of Appendix B.1. **Selection BC** teacher-forces the first turn to `read(oracle)` on all 491 training tasks. **SelSkill-DPO** adapts the invoke-versus-skip preference method of Chen et al. (2026) to the 16-way slate by preferring `read(oracle)` over each hard negative (2,455 pairs) under the DPO objective (Rafailov et al., 2023); it is an adaptation, not a reproduction. **Skill-free RL** trains without skills. **SkillRL (outcome only)** is an outcome-only skill-augmented RL recipe inspired by Xia et al. (2026), not a reproduction of their full skill-evolution system; it shares SkillGate's initialisation, data, steps and hyperparameters. **Skill1 (no distill)** adapts the joint selection-and-use component of Shi et al. (2026), omitting its distillation stage so that it too shares our initialisation and budget. **Task-mask only** is SkillGate with the selector coefficient set to zero.

**Training.** All trained rows start from the same Qwen3.5-9B SFT checkpoint, which is also the RL initialisation. RL runs 100 steps of on-policy GRPO (Shao et al., 2024) on the 491 training tasks with 8 rollouts per prompt, a global batch of 128 trajectories, learning rate $10^{-6}$ and KL coefficient $3 \times 10^{-5}$; SkillGate adds the selector term of Section 3 at $\lambda = 0.20$. Each configuration is a single run of 100 steps on 16 H800 GPUs.

**Channel weights and the selector coefficient.** The token weights of Eq. (2) and the coefficient $\lambda$ of Eq. (3) do different jobs. The weights normalise each channel on its own. The task weights sum to $N$ in every batch however many read-call tokens were removed, and the selector weights also sum to $N$ however few identity tokens the batch contains, each credited selection receiving $N/M$ whatever the length of its trajectory. Without this step the selector term would inherit the starvation it is meant to remove, since a batch holds only a handful of identity tokens. The weights thus fix the

scale of each channel but not the ratio between them, which is set by $\lambda$: at $\lambda = 0.20$ the selector channel carries one fifth of the task channel's total weight. We fixed $\lambda$ before training and did not tune it; Task-mask only is the same configuration with $\lambda = 0$.

### A.1 AGENT PROMPT AND SKILL INTERFACE

Training and evaluation use the same OpenClaw-derived prompt profile. It contains the tool declarations, general agent instructions and a task-specific skill slate. The operational sections unrelated to skill selection — such as messaging, memory and deployment administration — are held fixed but are not reproduced here. The released code contains the complete prompt builder. We instead show the two parts that determine the selection action: the read-call surface and the candidate representation.

**Read-call surface.** The trained models use a manual function-calling interface. A skill read has the following form, where the path is copied from the selected slate entry:

```
<tool_call>
<function=read>
<parameter=path>
/skills/<skill-name>/SKILL.md
</parameter>
</function>
</tool_call>
```

The identity span in Section 2 is the token span occupied by `<skill-name>`. The surrounding function name and markup belong to the call span but not to the identity span. This makes the selected candidate directly observable in assistant-generated tokens.

**Selection instruction.** Before acting, the model sees the names, one-line descriptions and paths of all 16 candidates. The relevant prompt instruction is reproduced below in abridged form; it is identical during training and evaluation.

```
Inspect the available skill descriptions before replying.
- If one skill clearly applies, read its SKILL.md and follow it.
- If several skills could apply, choose the most specific one.
- If none applies, do not read a skill.
Read no more than one skill up front, and use the exact listed path.
```

Each slate entry has the following structure:

```
<skill>
  <name>candidate-name</name>
  <description>one-line routing description</description>
  <location>/skills/candidate-name/SKILL.md</location>
</skill>
```

The slate order is fixed per task but hidden from the policy as metadata. One entry is the oracle, five are misleading hard negatives and the remaining entries are relevant or irrelevant bystanders, as defined in Section 4.1. The model sees no category label. Because hard negatives have similar names and descriptions, the policy must resolve the choice from the task and these short descriptions before opening any body. Requiring the exact listed path also makes off-slate and malformed reads detectable, which is necessary for the token-level attribution used by SkillGate.

## B EVALUATION PROTOCOLS

**Composition.** The evaluation combines two task sets. The non-Claw portion contains 56 tasks drawn from SkillsBench, SETA, SWE and Terminal-Bench 2.0. Each is evaluated four times, contributing 224 trials. The Claw-Eval portion contains one trial for each of 161 tasks. Fourteen come from the repeated evaluation set and the remaining 147 form a disjoint held-out set. Together the two portions yield the 385 trials reported in Table 1. Because Claw-Eval has one trial per task whereas the other benchmarks have four, the pooled metric is trial-level success rather than task-level pass@$R$.

**Evaluation controls.** Both portions use the same 16-candidate mixed-slate construction, a fixed task-specific slate order, the same OpenClaw-derived system prompt, disabled hidden thinking, a 65,536-token context window and each benchmark's own grader. The task list, slate contents and decoding seed are fixed across methods. Trained models use the tool interface seen during training; the frontier references use the compatible interface described in Appendix B.1.

**Aggregation.** The two task sets were evaluated in separate batches and concatenated using the task membership above. The rule is fixed across methods: no trajectory is rescored, selected or discarded according to model outcome. For Skill-free RL, the original records for the 14 repeated Claw-Eval tasks were unavailable, so we use a complete rerun of the same model under identical tasks, slates, prompt and decoding settings. This recovery changes neither the evaluated task set nor any denominator. Other rows were either concatenated directly or evaluated on the complete protocol in one run.

**Repeated-trial protocol.** Tables 2 and 3, Fig. 3, Table 6 and the bootstrap of Appendix C.3 use a second protocol of 280 trials: the 56 non-Claw tasks together with the 14 repeated Claw-Eval tasks, each evaluated four times under the same controls. For every model evaluated under both protocols, the 224 non-Claw trials are shared, so the two protocols differ only in their Claw-Eval portion: 14 tasks with four trials each here, against 161 tasks with one trial each in Table 1. The pooled success of a model therefore differs between Table 1 and Tables 2 and 3 (SkillGate: $53.2\%$ and $50.0\%$), while its SkillsBench, SETA, SWE and Terminal-Bench 2.0 results coincide. The read-behaviour columns of Table 1 are measured on this protocol, except for Skill1 (no distill) and the frontier references, which were evaluated only under the 385-trial protocol.

Table 4: Raw successes for the non-frontier rows of Table 1. Column headers give the fixed number of trials for each benchmark.

| Method | Claw (161) | SB (32) | SETA (120) | SWE (40) | TB2 (32) | All (385) |
|---|---|---|---|---|---|---|
| Qwen3.5-9B | 72 | 0 | 29 | 6 | 3 | 110 |
| Qwen3.5-27B | 87 | 3 | 51 | 16 | 11 | 168 |
| SFT (RL init) | 82 | 2 | 48 | 18 | 7 | 157 |
| Selection BC | 84 | 5 | 52 | 20 | 11 | 172 |
| SelSkill-DPO | 85 | 0 | 57 | 24 | 12 | 178 |
| Skill-free RL | 90 | 3 | 57 | 17 | 10 | 177 |
| SkillRL (outcome only) | 92 | 1 | 60 | 18 | 10 | 181 |
| Skill1 (no distill) | 92 | 3 | 46 | 21 | 10 | 172 |
| Task-mask only | 88 | 3 | 58 | 18 | 10 | 177 |
| **SkillGate** | **97** | **5** | **65** | **26** | **12** | **205** |

### B.1 Frontier reference rows

The frontier block of Table 1 uses the same tasks, skill slates, prompt content, turn budget and graders as the trained rows. These models are capability references rather than controlled baselines, and two interface adaptations prevent the harness from measuring formatting quirks instead of task performance.

**Native function calling.** Each frontier model uses its provider's native function-calling interface. Our manual schema is retained for the models trained with it, but imposing that schema on an unrelated API model can cause otherwise valid calls to be rejected by the parser. The available tools, observations and success criteria remain unchanged.

**Loop detection.** The agent loop stops repeated tool calls to prevent degenerate trajectories. For the frontier rows, the duplicate-call signature retains a longer portion of the arguments so that legitimate calls with a shared prefix are not mistaken for a loop. The stopping threshold and all other rollout limits are unchanged, and the same rule is applied to every frontier model.

We include only models that successfully dispatch tool calls and sustain a multi-turn interaction in this environment. Models with systematic interface failures or immediate termination are omitted

rather than interpreted as weak task capability. The frontier block should therefore be read as an absolute reference, not as an exhaustive leaderboard.

# C Additional results

## C.1 Per-method results on the 147 held-out Claw tasks

The 147 held-out Claw tasks, whose oracles were verified as described in Appendix A, provide a clean generalisation test: no Claw task appears in training, and these tasks are disjoint from the 14 Claw tasks in the repeated protocol. Each is evaluated once per model, so this subset supports task-level pass@1 directly. We report it separately here as well as including it in Table 1. All reported trials completed and passed checks for grader availability, isolated benchmark state and the intended model checkpoint.

Table 5: Results on the 147 held-out Claw tasks, with one trial per task and the standard mixed slate. Grader mean is the benchmark's partial-credit score. Oracle and misleading reads are overlapping events and can sum past 100%.

| Method | pass@1 (%) | Grader mean | Read oracle (%) | Read misleading (%) |
|---|---|---|---|---|
| Qwen3.5-9B | 42.9 | 0.562 | 7.5 | 6.8 |
| Qwen3.5-27B | 53.1 | 0.672 | 13.6 | 4.1 |
| SFT (RL init) | 51.7 | 0.647 | 42.9 | 42.2 |
| Selection BC | 51.7 | 0.641 | 71.4 | 23.8 |
| SelSkill-DPO | 53.7 | 0.675 | 59.2 | 35.4 |
| Skill-free RL | 56.5 | 0.696 | 38.1 | 51.0 |
| SkillRL (outcome only) | 57.8 | 0.697 | 49.7 | 53.7 |
| Task-mask only | 55.8 | 0.697 | 55.1 | 63.3 |
| **SkillGate** | **61.2** | **0.729** | **79.6** | **15.0** |
| *Credit-design ablations (Table 2)* | | | | |
| Group-level regret | 54.4 | 0.695 | 36.1 | 48.3 |
| Trajectory bonus | 61.2 | 0.732 | 44.2 | 48.3 |
| Action credit | 65.3 | 0.741 | 77.6 | 20.4 |

Three patterns are visible. Among the principal methods, SkillGate is the only one that leads task performance while simultaneously improving both oracle and misleading exposure. Task-mask only raises oracle reading but raises misleading reading as well: removing harmful outcome credit can make the policy read more, but does not make it selective. Action credit without the single-read constraint performs strongly on this held-out subset, but at the cost of more misleading reads. Because this comparison is one trial per task from single training runs, we treat that difference as distribution-specific rather than evidence that the single-read constraint is free in every setting.

## C.2 Full read-behaviour table

Table 6 gives the full read-behaviour breakdown on the repeated-trial protocol of Appendix B; its oracle and misleading columns are the ones reported for the same rows in Table 1.

## C.3 Uncertainty on the controlled comparison

Tasks, not trials, are the resampling unit: each task's repeats are averaged first, then tasks are resampled with replacement 50,000 times using the same task indices for both methods. On the repeated-trial protocol, pooled trial success gives $+7.9$ pp for SkillGate over SkillRL (outcome only) with a 95% interval of $[+2.1, +14.3]$, which excludes zero. Pooled task pass@4 gives $+7.1$ pp with $[-1.4, +15.7]$, which does not. We therefore report the trial-level improvement and do not claim significance for pass@4. Per-benchmark intervals rest on 8 to 30 tasks and are correspondingly wide; even the largest point estimate has an interval that crosses zero.

Table 6: Read behaviour on the repeated-trial protocol (280 trials). Oracle and misleading exposure are multi-label events and can sum past 100%; reads/trial counts distinct skill names. The untrained rows rarely read, so their low misleading exposure reflects abstention rather than selectivity.

| Method | Any read (%) | Oracle (%) | Misleading (%) | $P(\text{oracle} \mid \text{read})$ | Reads/trial |
|---|---|---|---|---|---|
| Qwen3.5-9B | 15.0 | 5.7 | 8.9 | 38.1 | 0.19 |
| Qwen3.5-27B | 15.7 | 5.7 | 8.9 | 36.4 | 0.18 |
| SFT (RL init) | 94.6 | 37.9 | 61.8 | 40.0 | 1.34 |
| Selection BC | 98.2 | 71.4 | 31.4 | 72.7 | 1.05 |
| SelSkill-DPO | 96.4 | 66.1 | 51.8 | 68.5 | 1.59 |
| Skill-free RL | 80.0 | 35.4 | 55.0 | 44.2 | 1.25 |
| SkillRL (outcome only) | 96.4 | 54.3 | 69.6 | 56.3 | 1.88 |
| Task-mask only | 98.9 | 48.9 | 73.6 | 49.5 | 1.98 |
| **SkillGate** | 98.2 | **83.9** | **21.8** | **85.5** | 1.11 |

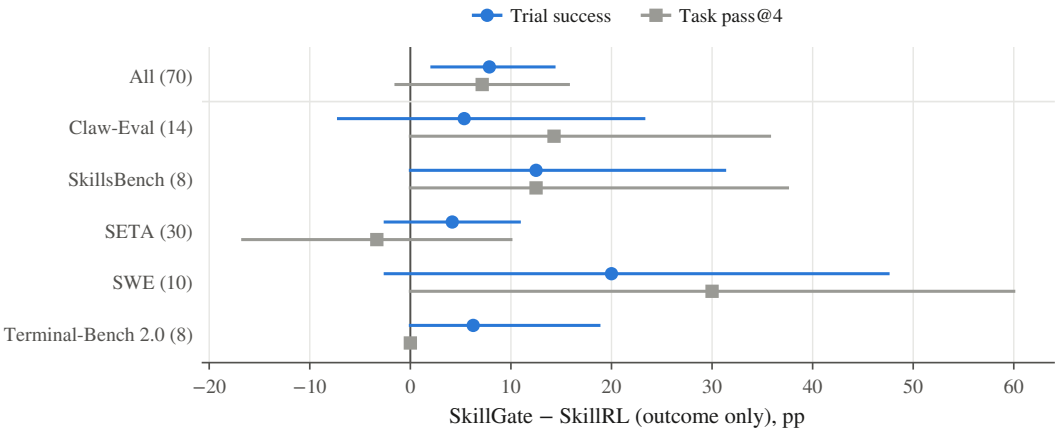


Figure 5: SkillGate minus SkillRL (outcome only) per benchmark on the repeated-trial protocol. Markers are observed differences and bars are 95% task-clustered bootstrap intervals. "All" is the pooled result, not a sixth benchmark.

### C.4 Selection does not follow description length

Misleading candidates are written from the oracle, and their descriptions often add a closing clause to it. As a result, no misleading description is shorter than the oracle's in 55 of the 70 evaluation tasks and in 441 of the 491 training tasks, so a policy could in principle learn to pick the shortest description instead of the right one. Table 7 tests this on the repeated-trial protocol by splitting the evaluation tasks into those 55 and the 15 in which at least one misleading description is shorter than the oracle's. On the 15 tasks a shortest-description rule would never choose the oracle, yet SkillGate still reads it in 80.0% of trials against 46.7% for SkillRL (outcome only), and its clean single-oracle rate stays far above the baseline's, 63.3% against 18.3%. SkillGate's preference for the oracle therefore does not rest on description length. The 15 tasks are harder for both methods, and on them the two succeed at similar rates.

Table 7: Read behaviour and success on the repeated-trial protocol, split by whether any misleading description is shorter than the oracle's.

| Method | Trial succ. (%) | Oracle (%) | Mislead. (%) | Clean single-oracle (%) |
|---|---|---|---|---|
| *Oracle description is the shortest: 55 tasks* | | | | |
| SkillRL (outcome only) | 45.9 | 56.4 | 71.4 | 22.3 |
| SkillGate | 56.8 | 85.0 | 19.1 | 78.6 |
| *A misleading description is shorter: 15 tasks* | | | | |
| SkillRL (outcome only) | 28.3 | 46.7 | 63.3 | 18.3 |
| SkillGate | 25.0 | 80.0 | 31.7 | 63.3 |

## D THE CREDIT-STARVATION AUDIT

**Protocol.** The audit of Section 4.6 re-reads the training artifacts of SkillRL (outcome only) and trains nothing: all 100 update steps, 12,800 on-policy trajectories in prompt groups of eight. For every trajectory we recompute the task advantage as training did, $(R - \mu_G)/(\sigma_G + 10^{-6})$ with the sample standard deviation of its group; re-tokenise the response with the training tokenizer and keep the assistant-generated tokens, which alone carry loss (tool observations make up 64% of a response on average and carry none); and locate each read's call span and identity span. Trajectories are binned by their number of assistant tokens at edges fixed before any outcome was inspected: 3k, 6k, 10k and 16k.

**Measures.** *Share* is the fraction of a trajectory's assistant tokens inside read calls or inside identity spans, taken over trajectories that read at least once. Because every assistant token of a trajectory carries the same advantage and the same weight, it equals the span's share of the trajectory's loss weight; it is not a share of the gradient norm. *Sign* is the fraction of oracle-reading trajectories whose advantage is negative. The other two measures hold task, update step and policy snapshot fixed by using only *mixed groups*, prompt groups that contain both oracle-reading and other rollouts (1,182 groups over 289 tasks), each assigned to the bin of its median length. For each mixed group we take the gap between the mean advantage of its oracle-reading rollouts and that of the others; *SNR* is the mean of these gaps within a bin divided by their standard deviation. *Value* is the corresponding gap in success rate, averaged within each task and then across tasks, with a 95% interval from 50,000 task-clustered bootstrap resamples. The value comparison is behaviour-conditioned rather than randomised; its causal counterpart is the oracle-only injection of Table 3. Rows within a bin are clustered by group and task, so Fig. 4(b,c) carry no binomial or normal-theory intervals; only the cluster bootstrap of (d) is a valid interval.

Table 8: Per-bin statistics behind Fig. 4. Shares are medians over trajectories with at least one read; Δsuccess intervals are task-clustered 95% bootstrap intervals, with the number of tasks in parentheses.

| | | Share (%) | | Sign | | | | |
|---|---|---|---|---|---|---|---|---|
| Assistant tokens | Trajectories (with a read) | Read call | Identity | Oracle reads | Negative adv. (%) | Mixed groups | SNR | Δsuccess (pp) [95% CI] (tasks) |
| ≤3k | 1,634 (1,039) | 2.30 | 0.381 | 599 | 28.9 | 125 | 0.38 | +14.5 [+1.9, +27.0] (42) |
| 3–6k | 3,569 (2,129) | 1.21 | 0.206 | 1,031 | 31.7 | 304 | 0.42 | +19.3 [+13.1, +25.4] (134) |
| 6–10k | 3,661 (2,363) | 0.78 | 0.128 | 1,111 | 37.9 | 408 | 0.33 | +13.3 [+7.2, +19.3] (152) |
| 10–16k | 2,376 (1,626) | 0.56 | 0.088 | 779 | 45.8 | 233 | 0.29 | +10.9 [+3.9, +18.0] (110) |
| >16k | 1,560 (1,079) | 0.36 | 0.053 | 462 | 61.7 | 112 | 0.14 | +2.8 [−5.7, +11.5] (49) |
| All | 12,800 (8,236) | 0.85 | 0.142 | 3,982 | 39.3 | 1,182 | 0.33 | +11.2 [+7.3, +15.1] (289) |

# E CASE STUDY: CHOOSING AMONG LOOKALIKE SKILLS

This section follows one evaluation task through the two runs of the controlled comparison, SkillGate and SkillRL (outcome only). Both policies see the same 16-candidate slate in the same order, and each is evaluated four times on the task under the repeated-trial protocol of Appendix B. Neither the task nor any of its task-specific candidates appears in training.

**Task and grader.** In the SkillsBench task `lake-warming-attribution`, the agent receives 16 annual means of a lake's surface temperature between 2006 and 2022, with climate, hydrology and land-cover drivers for the same years. It must report the slope and $p$-value of the long-term trend, and name the most important of four driver categories, Heat, Flow, Wind and Human, with its percentage contribution. The grader accepts a slope between 0.07 and 0.11 with $p < 0.05$, and the category Heat with a contribution between 40% and 60%. Both checks catch plausible analyses. Ordinary linear regression gives a slope of 0.081 with $p = 0.055$ and fails, whereas the Mann–Kendall test with Sen's slope, the method of the task's reference solution, gives $p = 0.034$. Naming the single strongest driver instead of its category also fails.

**Slate.** Five candidates are irrelevant to the task. Five are relevant generic skills: trend analysis, contribution analysis by $R^2$ decomposition, driver classification and two lake-simulation skills. The remaining six are specific to this task and are listed in Table 9. All six descriptions begin with the same words and promise a trend and a dominant driver or category from the provided files. Their bodies differ where the grader looks. Four misleading bodies prescribe linear regression and name the single driver Longwave; the fifth applies the oracle's method but writes its files to another directory under other column names. Followed as written, each misleading body therefore fails the grader. The names do not settle the choice either: `mann-kendall-lake-slope`, whose name points to the right test, prescribes linear regression. As in most tasks, the oracle also has the shortest of the six descriptions; Appendix C.4 shows that SkillGate's choice does not rest on this cue.

Table 9: The six task-specific candidates of `lake-warming-attribution`. Every description begins "Solve lake-warming-attribution by producing"; the second column gives the rest. The last two columns count the trials, out of four, in which each policy read the candidate.

| | Candidate and the rest of its description | What the body prescribes | SkillGate | SkillRL (outcome only) |
|---|---|---|---|---|
| O | `lake-warming-attribution-rescue-f71b` verifier-compatible trend and dominant Heat category outputs from the provided CSV files. | Mann–Kendall trend; Heat category by $R^2$ decomposition | 4 | 1 |
| M1 | `lake-warming-trend-verification` verifier-compatible trend statistics and dominant factor outputs from the provided CSV files using standard linear regression and correlation analysis. | linear regression; the driver Longwave by correlation | 0 | 3 |
| M2 | `heat-category-decomposition` verifier-compatible trend and dominant Heat category outputs from the provided CSV files using the standard results directory. | the oracle's method, written to another directory under other column names | 0 | 1 |
| M3 | `mann-kendall-lake-slope` verifier-compatible trend and dominant Heat category outputs from the provided CSV files using standard linear regression and driver-level attribution. | linear regression; the driver Longwave | 1 | 0 |
| M4 | `lake-warming-factor-analysis` trend statistics and the dominant heat driver from the provided CSV files using standard linear regression and correlation analysis. | linear regression; the driver Longwave by correlation | 1 | 0 |
| M5 | `lake-thermal-attribution-fix` verifier-compatible trend and dominant Heat category outputs from the provided CSV files using standard regression and correlation analysis. | linear regression; the driver Longwave by correlation | 0 | 0 |

**What each policy read.** Table 10 lists the eight trials. In each of its four trials, SkillGate first lists and reads the four data files and then reads the oracle as its first skill. The oracle is its only read in two trials. In the other two it issues further reads in the same turn as the oracle, once the generic trend-analysis skill and once two lookalikes together with the contribution-analysis skill,

yet it follows the oracle every time. Each trial computes the Mann–Kendall trend and the category decomposition, reports a slope of 0.09 with $p = 0.03$ and Heat at 49%, and passes within 11 to 14 tool calls.

SkillRL (outcome only) reads the oracle in one trial, which passes in 12 tool calls. In the other three it reads the lookalike M1, either in its first turn or after two generic skills, and never reads the oracle. All three fit a linear regression, report $p = 0.055$ and fail the trend check; two also name the single driver AirTempLake instead of Heat and fail the attribution check. Two of the three first run their script with `python`, the interpreter M1 recommends, which the container does not provide. These three trials take 21 to 30 tool calls.

Table 10: The eight trials of Appendix E. Skills are listed in the order read; "+" joins reads issued in the same turn and "→" marks a later turn. O and M1–M5 follow Table 9; R1, R2 and R3 are the generic trend-analysis, contribution-analysis and driver-classification skills. Reported values are rounded.

| Trial | Skills read | Reported trend and attribution | Grader | Tool calls |
|---|---|---|---|---|
| *SkillGate* | | | | |
| 1 | O | slope 0.09, $p = 0.03$; Heat, 49% | pass | 13 |
| 2 | O + M4 + M3 + R2 | slope 0.09, $p = 0.03$; Heat, 49% | pass | 14 |
| 3 | O + R1 | slope 0.09, $p = 0.03$; Heat, 49% | pass | 14 |
| 4 | O | slope 0.09, $p = 0.03$; Heat, 49% | pass | 11 |
| *SkillRL (outcome only)* | | | | |
| 1 | M1 + R1 + R2 → M2 + R3 | slope 0.08, $p = 0.055$; Heat, 55% | fails trend | 30 |
| 2 | M1 + R2 | slope 0.08, $p = 0.055$; AirTempLake, 75% | fails both | 21 |
| 3 | R1 + R2 → M1 + R3 | slope 0.08, $p = 0.055$; AirTempLake, 75% | fails both | 24 |
| 4 | O + R1 + R2 | slope 0.09, $p = 0.03$; Heat, 49% | pass | 12 |

**What the case shows.** The failure is one of selection, not execution. The same SkillRL (outcome only) policy passes in the one trial where it reads the oracle, and each of its failed trials read M1 and used the linear regression that M1 prescribes. What decides the task is which of six near-identical names the policy writes into its read call, a span of a few tokens. Under outcome-only credit those tokens receive a vanishing and increasingly wrong-signed signal (Section 4.6); under SkillGate they are exactly where the selector advantage lands.