GÖDEL ECOLOGY: PRODUCTIVITY-GUIDED CONTINUOUS CO-EVOLUTION OF AGENTS AND EVALUATORS FOR COLLECTIVE RECURSIVE SELF-IMPROVEMENT

Anonymous authors
Paper under double-blind review

ABSTRACT

Self-improving coding agents rely on evolutionary search over their own implementations, with evaluators determining which variants survive. Recent work has begun to co-evolve agents and evaluators, but evaluator updates remain periodic and anchored to fixed criteria. We introduce **Gödel Ecology**, a framework of collective recursive self-improvement via productivity-guided continuous co-evolution of self-improving agents and evaluators. Evaluators are scored jointly with Co-Clade Metaproductivity of the agent lineages they induce, and a continuously updated soft mixture of evaluator candidates forms an adaptive evaluation landscape. Delayed credit assignment and adversarial interventions are additionally adopted to counter self-reinforcing evaluator drift with reliable external evidence. Together, these interacting populations form an evolving ecology in which agents and evaluation criteria actively interplay and shape one another. Across experiments on coding, math, and writing domains, Gödel Ecology produces stronger evaluator signals, better downstream search decisions, and improvements that transfer across benchmarks and model backbones. These results suggest that continuously adapting the evaluation criteria that determine which directions of improvement are worth pursuing helps make better recursive self-improvement.

1 INTRODUCTION

Recursive self-improvement (RSI) asks agents not only to solve tasks, but also to improve the mechanisms by which they solve them. Formalized as utility-driven self-referential modification (Schmidhuber, 2007), this idea has recently found practical realization in LLM agents that empirically search over prompts, tools, workflows, and execution scaffolds (Yin et al., 2025; Zhang et al., 2026). Such evolutionary search ultimately depends on an evaluator whose judgments determine which mutations survive and which regions of the search space are further explored.

Most existing self-improving agent systems keep this evaluation criterion permanently fixed throughout evolution (Zhang et al., 2026; Wang et al., 2026; Liu et al., 2026a). The Red Queen Gödel Machine (RQGM; Iacob et al., 2026) takes an important step beyond by co-evolving agents and evaluators in a shared workspace, periodically replacing the incumbent evaluator when a challenger performs sufficiently better on a held-out anchor set. Related work has likewise explored adapting rubrics or jointly evolving generators and evaluators, suggesting that the evaluation criterion itself can be treated as an object of optimization (Ding et al., 2026; Liu et al., 2026b). However, this line of work still largely relies on checkpoint-based and anchor-centered evaluator evolution, which leaves a severe mismatch with population-based RSI, where evaluators directly affect which descendants could emerge. A potentially improved evaluator discovered midway through an epoch cannot influence search until the next checkpoint. More importantly, assessing evaluators solely by agreement with anchor reference may favor evaluators that accurately reproduce external judgments without effectively steering search toward productive future agents. Evaluators with similar anchor fidelity can nevertheless induce radically different evolutionary trajectories by preserving or eliminating intermediate agents whose value becomes apparent only through their descendants. This yields an evaluator-side analogue of the distinction between immediate fitness and descendant productivity emphasized in clade-level search (Wang et al., 2026), and echoes recent findings that stronger evaluation does not necessarily translate into stronger self-improvement (Shi et al., 2026).

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.

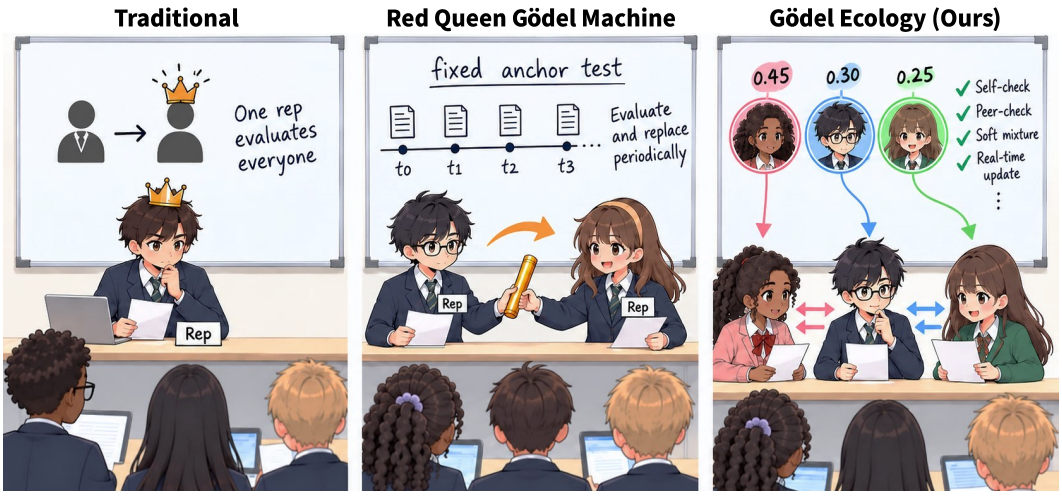


Figure 1: **Continuous co-evolution of agents and evaluators.** Traditional approaches rely on a single fixed evaluator for all agents. RQGM allows evaluators to evolve, but uses one evaluator at a time and replaces it only periodically based on performance over a fixed anchor set. Gödel Ecology maintains a continuously updated mixture of evaluator candidates, allowing evaluation landscape to adapt in real time with agents and evaluators co-evolve as mutually interacting populations through self- and peer-evaluation.

We introduce **Gödel Ecology**, a framework of collective RSI via productivity-guided continuous co-evolution. As illustrated in Figure 1, each archive node contains both an agent scaffold and an evaluator scaffold, and we maintain a soft mixture over active evaluator candidates instead of a single epoch-frozen judge, which induces an adaptive evaluation landscape coupled to a population of evolving agent lineages. Evaluators are scored with downstream productivity of the agent lineages they help select, captured by evaluator-side *Co-Clade Metaproductivity*, which attributes credit according to the productivity of the descendants their judgments help preserve. Together, agents and evaluators form an evolving ecology in which each population shapes the selection pressure acting on the other. To counter circular self-reinforcement in co-evolutionary systems (Ficici & Pollack, 1998), the productivity term is computed from a delayed snapshot of the agent archive, and we additionally introduce regularization, significance-gated updates, adversarial interventions, and an immutable verification floor to constrain evaluator drift. A value-of-information scheduler allocates evaluation effort between agent and evaluator candidates according to expected gain per unit cost.

Experiments on Polyglot (Aider-AI, 2024) show that Gödel Ecology not only yields better evaluators, but also brings this improvement into better search decisions. Continuous co-evolution makes stronger evaluators influence search much effectively, with strong evaluators explored midway through search dominating in 96% of runs, compared with 29% under baselines, and raising the in-loop evaluator AUC from 0.57 to 0.90. The agent evolved with `qwen3.6-35b-a3b` transfers seamlessly to `qwen3.8-flash` and `deepseek-v4.1-flash`, solving 98.3% held-out tasks versus 93.3% for baseline. The evaluator signal of Gödel Ecology consistently outperforms baseline when generalize to benchmarks beyond coding, including PeerRead (Kang et al., 2018), IMO-GradingBench (Luong et al., 2025), SummEval (Fabbri et al., 2021), and Change My View (Tan et al., 2016), accepting more beneficial revisions (39 vs. 23) and mistakenly rejecting fewer revisions preferred by independent judges (31 vs. 54). Together, these results suggest that continuously evolving the evaluation landscape can help produce more reliable selection pressure and more transferable collective self-improvement.

Our contributions are:

- We formulate Gödel Ecology, a framework for collective RSI with continuous co-evolution of agent and evaluator populations through an adaptive evaluation landscape.
- We introduce Co-Clade Metaproductivity for evaluator evolution and replace checkpoint-based update with continuously evolved soft mixture of evaluators.
- We demonstrate that Gödel Ecology improves both evaluator quality and downstream search performance, with robust generalizability across model backbones and across domains.

2 Background and Preliminaries

We follow the tree-search formulation of the Huxley-Gödel Machine (HGM; Wang et al., 2026). The search maintains an archive tree $\mathcal{T}$, where each node a represents an agent scaffold and $\mathcal{C}(a)$ denotes the descendant clade rooted at a. At each step t the system either expands $\mathcal{T}$ by turning a node into a self-modified child, or evaluates an existing node on a task. For a binary task outcome $o \in \{0, 1\}$, each node a accumulates success and failure counts $n_{\mathrm{s}}(a)$ and $n_{\mathrm{f}}(a)$. HGM evaluates a node by the empirical productivity of its entire lineage clade $\mathcal{C}(a)$,

$$\mathrm{CMP}(a) = \frac{\sum_{a' \in \mathcal{C}(a)} n_{\mathrm{s}}(a')}{\sum_{a' \in \mathcal{C}(a)} \big(n_{\mathrm{s}}(a') + n_{\mathrm{f}}(a')\big)}, \tag{1}$$

so that a node can receive credit not only for its own performance but also for productive descendants. Parent selection uses Thompson sampling over the corresponding clade-level posterior, while the returned agent is selected by the α-best-belief score $u_\alpha(a)$, the α-quantile of the Beta posterior $\mathrm{Beta}(1+n_{\mathrm{s}}(a), 1+n_{\mathrm{f}}(a))$. The expansion operation is available when the UCB-Air gate $N_t \geq |\mathcal{T}|^\rho$ is open, where N_t counts the evaluations run so far. Like the Darwin Gödel Machine (DGM; Zhang et al. (2026)), HGM keeps the scoring criterion fixed for the whole search.

RQGM (Iacob et al., 2026) extends this setting by making the evaluator itself evolvable. Coding agent and evaluator scaffolds share one workspace, and at any moment a single incumbent evaluator scores the population. At discrete checkpoints placed at powers of two of the evaluation count, the challenger with the highest ϵ-best-belief score on a held-out anchor set $\mathcal{D}_{\mathrm{ext}}$ replaces the incumbent if its score is strictly higher. Between checkpoints, the evaluation signal remains fixed.

In our setting, we consider a unified archive in which each node $a \in \mathcal{T}$ is a workspace that holds an agent scaffold π_a, an evaluator scaffold ξ_a, and shared infrastructure, all of which a meta-agent may edit during expansion. The search is given an initial seed workspace a_0, validation tasks $\mathcal{V}$, an anchor set $\mathcal{D}_{\mathrm{ext}}$, and a finite budget B. At any search step, the system can choose among three actions. `Expand` proposes a self-modified child workspace; $\texttt{Eval}_\pi$ gathers evidence about an agent on a task τ; and $\texttt{Eval}_\xi$ calibrates an evaluator against $\mathcal{D}_{\mathrm{ext}}$. The search returns $a^\star = \arg\max_a u_\alpha(a)$ over the nodes admitted by the dual acceptance gate (see Section 3.3). Agent utility combines the verifiable outcome $v(a \mid \tau) \in \{0, 1\}$ of each validation task with the calibrated score of the evaluator mixture (details in Appendix A.13). Appendix A.1 collects all the notation used in this paper.

3 Methodology

Gödel Ecology maintains two coupled populations within the same archive: evolving agent lineages and an active pool of evaluator candidates. At search step t, a mixture w_t over evaluators determines the learned evaluation signal used to score agents. These judgments affect which agent lineages survive and expand; the productivity subsequently observed in those lineages is then used, with a delay, to update the evaluators that helped shape them. External anchor evidence and verifiable task outcomes remain outside this feedback loop and provide grounding. Algorithm 1 describes the complete search procedure.

3.1 Co-Clade Metaproductivity

Under the current evaluator mixture w_t, agent evaluations contribute mixture-weighted success and failure evidence $\tilde{n}_s(a)$ and $\tilde{n}_f(a)$. We extend Eq. 1's clade-level productivity to Co-Clade Metaproductivity (Co-CMP) for agents

$$\widehat{\text{Co-CMP}}_\pi(a) = \frac{\sum_{a' \in \mathcal{C}(a)} \tilde{n}_{\mathrm{s}}(a')}{\sum_{a' \in \mathcal{C}(a)} \big(\tilde{n}_{\mathrm{s}}(a') + \tilde{n}_{\mathrm{f}}(a')\big)}. \tag{2}$$

As in CMP, credit propagates through descendants rather than being assigned only to the current node. Here, the evidence entering is produced under a co-evolving evaluator population.

An evaluator should be useful not only because it agrees with an anchor, but because the selections it induces lead to productive agent lineages. For evaluator candidate ξ_a, Co-CMP scores it by

$$\widehat{\text{Co-CMP}}_\xi(a) = \lambda_t\, u_\alpha^{\mathrm{ext}}(a) + (1 - \lambda_t) \max_{a \in \mathcal{S}(a)} \widehat{\text{Co-CMP}}_\pi^{(t-\Delta)}(a), \tag{3}$$

Algorithm 1: Gödel Ecology search loop.

Initialize $\mathcal{T} \leftarrow \{a_0\}$, $\mathcal{C}_{\text{ev}} \leftarrow \{a_0\}$, $\mathcal{P}_{\text{pend}}, \mathcal{P}_{\text{cool}} \leftarrow \emptyset$, uniform w, $\lambda_t \leftarrow \lambda_{\text{start}}$;
while *budget remains* **do**
  if *UCB-Air gate open* **then**
    sample parent $a \sim \text{TS}(\widehat{\text{Co-CMP}}_\pi)$ over $\mathcal{T} \setminus \mathcal{P}_{\text{cool}}$; the meta-agent edits $\text{workspace}(a)$ into a child c, which is evaluated on one task;
    if *the edit and the evaluation succeed* **then**
      add c to $\mathcal{P}_{\text{pend}}$;
    else
      $f(a)$ += 1; **if** $f(a) \geq \tau_{\text{cool}}$ **then** add a to $\mathcal{P}_{\text{cool}}$;
  else
    pick the evaluation with the largest VoI, children in $\mathcal{P}_{\text{pend}}$ first (Section 3.3);
    if *it is* $\texttt{Eval}_\pi$ *on* (a, τ) **then**
      run π_a on τ, score with the top-k evaluators under w_t, update $\tilde{n}_{\text{s}}(a), \tilde{n}_{\text{f}}(a)$;
      foreach $c \in \mathcal{P}_{\text{pend}}$ *with* ν_{gate} *evaluations* **do**
        the dual gate (Section 3.3) moves c to $\mathcal{T}$ and its calibrated evaluator to $\mathcal{C}_{\text{ev}}$, or prunes c with its records;
    else
      $\texttt{Eval}_\xi$: calibrate ξ_n on $\mathcal{D}_{\text{ext}}$, update $\widehat{\text{Co-CMP}}_\xi(n)$ against the Δ-delayed $\widehat{\text{Co-CMP}}_\pi$, update w_t with entropy floor and jump test;
      if $\|w_t - w_{t-1}\|_1 > \delta$ **then** mark records of down-weighted evaluators stale;
  every κ steps: probationary weight ϕ for a designated candidate, significance test at window end;
  anneal λ_t toward $\lambda_{\min}$;
return $a^\star = \arg\max_{a \in \mathcal{T}} u_\alpha(a)$

where $u_\alpha^{\text{ext}}(a)$ is the α-best-belief lower bound estimate on ξ_a's accuracy on $\mathcal{D}_{\text{ext}}$, and $\mathcal{S}(a)$ contains agent nodes on which ξ_a had non-negligible weight influence. Importantly, the lineage term is read from a delayed archive snapshot $t - \Delta$. This breaks the immediate feedback path in which an evaluator could raise its own score by changing the very agent judgments used to evaluate it. As evidence accumulates, the weight λ_t anneals from λ_{start} to $\lambda_{\min}$, gradually shifting evaluator credit from external fidelity toward observed downstream productivity.

Continuous co-evolution is useful only if evaluator adaptation remains tied to the task being optimized. We use two complementary sources of grounding. In our implementation of Eq. 3, for the first term, besides an external code-review corpus (Zhang, 2025), we additionally build $\mathcal{D}_{\text{ext}}$ from the search domain itself, labeling real agent patches on tasks that the search never uses by whether they pass the tests (see Appendix A.14). For the second term, evaluators receives the task statement, candidate output, and task specification available to the agent; for coding tasks this includes test files but never their outcomes, so that the evaluators never observe the verifiable reward itself.

3.2 Continuous evaluator landscape

Let $\mathcal{C}_{\text{ev}}$ denote the active evaluator population. Rather than selecting one incumbent, Gödel Ecology maintains the distribution

$$w_t(a) \propto \exp\big(\eta \cdot \widehat{\text{Co-CMP}}_\xi(a)\big), \qquad a \in \mathcal{C}_{\text{ev}}, \quad \eta > 0. \tag{4}$$

Each agent output is scored by the top-k evaluators under the current w_t, evaluator influence changes immediately as new calibration and lineage evidence arrives. The mixture therefore defines a continuously changing evaluation landscape rather than the piecewise-constant criterion induced by checkpoint-wise incumbent replacement. Because the evaluator population changes online, abrupt reweighting can invalidate historical scores or amplify transient noise. We introduce an entropy floor ζ, a minimum evaluator weight floor $w_{\min}$, and a significance test on large weight jumps to address these issues. When the mixture moves by more than δ in ℓ_1 norm, records tied to down-weighted evaluators are marked stale and re-scored at their next visit. For debiasing, every κ steps a designated candidate evaluator receives a probationary weight ϕ, and we refer to this probation as an *adversarial intervention*. At the end of the window, a significance test on shared-anchor evidence promotes the candidate only if its anchor lower bound strictly exceeds that of the reference evaluator, and otherwise the boost is withdrawn. More details are provided in Appendix A.9 and A.10.

3.3 Archive Maintenance and Evolution Scheduling

Because the evaluation landscape changes over time, a child node that appears favorable under the current mixture may no longer do so after evaluator weights move. We decouple evolutionary priority from archive admission. After accumulating ν_{gate} evaluations, a child is admitted only when

$$v(a_{\text{child}}) \geq v(a_{\text{parent}}) - \varepsilon_{\text{gate}} \quad \text{and} \quad L_{\bar{w}_t}(a_{\text{child}}) \geq L_{\bar{w}_t}(a_{\text{parent}}), \tag{5}$$

where the first condition is an immutable verifiable floor on the pass rate v over deterministic unit tests with tolerance $\varepsilon_{\text{gate}}$, whose verdict stays valid however the evaluators evolve; and the second compares learned utility under a smoothed evaluator mixture $\bar{w}$ to be at least the parent's. Together the two tests form a Pareto comparison in verifiable and learned score. Once admitted, a node remains in the archive; subsequent mixture changes alter only their search priority through Co-CMP.

During the evolution, the search alternates between expansion and evidence collection. We retain HGM's UCB-Air mechanism to regulate archive expansion. When expansion is not selected, a value-of-information (VoI) rule allocates the next evaluation to either an agent or an evaluator according to expected posterior uncertainty reduction per unit model cost, with a cooldown set $\mathcal{P}_{\text{cool}}$ that holds parents after τ_{cool} failed expansion attempts and priority for children awaiting the gate until they accumulate the evidence required by the archive admission. Detailed posterior formulas, cooldown handling, and asynchronous execution are provided inAppendices A.7 and A.12).

4 Experiments

Here we systematically evaluate whether Gödel Ecology's design for continuous co-evolution of agents and evaluators helps improve evaluator signal available to evolution, whether this improvement delivers better evolved agent performance, and whether the resulting gains could transfer across domains or model backbones. We implement separately controlled experiments on offline evaluator updating, complete closed-loop co-evolution, and generalization experiments to isolate these effects.

4.1 Experimental Setup

Benchmarks. We primarily experiment on **Polyglot** (Aider-AI, 2024), which contains 225 coding tasks in six programming languages, among which 34 tasks containing 635 real agent patches are divided into disjoint 17-task external evidence and evaluation halves, containing 317 and 318 patches respectively. The external half is used only to calibrate evaluators; agent search operates on the evaluation half. We further sample 60 tasks from the remaining 177 valid tasks that are disjoint from both halves to serve as a strict out-of-search evaluation set. To test whether the evaluator-update mechanism extends beyond software engineering, we consider four judgment-based domains: **PeerRead** (Kang et al., 2018), **IMO-GradingBench** (Luong et al., 2025), **SummEval** (Fabbri et al., 2021), and **Change My View** (Tan et al., 2016). Each domain provides an external target signal and a pool of six evaluator personas. We evaluate both fixed-judgment evaluator updating and closed-loop writer-revision loops. Dataset sizes, splits, and labels are detailed in Appendix A.14. Appendix A.16 additionally reports results on SWE-bench Verified (Jimenez et al., 2024).

Baselines. Our principal baseline is RQGM (Iacob et al., 2026). To isolate evaluator evolution, RQGM and Gödel Ecology share every component including the same initial agent, self-improvement operator, parent-selection mechanism, value-of-information scheduler, archive-admission rule, and search budget, only except how the evaluator signal is formed and updated (Section 3). Table 8 in Appendix A.14 provides a complete component-wise comparison in detail. We implement RQGM's evaluator turnover on our substrate with two anchor sources, the CRAVE code-review corpus (Zhang, 2025) and in-domain Polyglot half. In closed-loop search, Gödel Ecology uses the same CRAVE as external source as the CRAVE-anchored RQGM run, isolating the evaluator-update mechanism. We additionally report RQGM with in-domain anchoring to compare Gödel Ecology's gain from better in-domain evidence gathering.

Models. `qwen3.6-35b-a3b` with thinking disabled serves is used primarily as the same agent, evaluator, and meta-agent backbone during our experiments. We additionally use `deepseek-v4.1-flash` to independently re-score evaluator judgments and use `qwen3.8-flash` and `deepseek-v4.1-flash` for cross-model agent transfer. Detailed hyperparameters and prompt templates are given in Appendices A.4 and A.6.

Table 1: **Evolution result on Polyglot** (`qwen3.6-35b-a3b`). Agent rows give pass rates on the 17 held-out tasks. Gate rows come from closed-loop search, where each child is tested on three tasks shared with its parent and a test-verified improvement passes all three where the parent passed less. Gate fractions count accepted over produced improvements and passing over accepted nodes. Evaluator rows come from offline evaluator updating on 1,000 paired seeds. Superscripts and % Impr. are relative to the initial agent.

	RQGM (CRAVE)	RQGM (in-domain)	**GE**
Agent (held-out pass rate, %)			
Pass rate ↑ (Initial: 76.5)	$76.5^{+0.0}$	$64.7^{-11.8}$	$\mathbf{82.4}^{+5.9}$
% Impr. ↑	+0.0%	−15.4%	**+7.7%**
Acceptance gate (closed-loop search)			
Test-verified improvements accepted ↑	0% (0/4)	67% (4/6)	**100% (3/3)**
Accepted nodes passing all paired tasks ↑	0% (0/1)	63% (5/8)	**100% (6/6)**
Children passing all paired tasks ↑	12%	28%	**37%**
Evaluator AUC vs. test outcomes ↑	0.562	0.857	**0.873**
Evaluator signal (in-loop AUC ↑, offline)			
RQGM and GE on their own pools, scored by Qwen	0.565	0.578	**0.901**
RQGM and GE on GE's pool, scored by Qwen	0.820	0.838	**0.901**
RQGM and GE on their own pools, scored by DeepSeek	0.610	0.611	**0.926**

Metrics. We evaluate the effectiveness of Gödel Ecology at different levels from evaluator quality to final agent performance. **Evaluator quality** reports the in-loop AUC, the share of passing/failing pairs ranked correctly, and outside code the pairwise agreement with the ground truth. **Gate quality** is the share of test-verified improvements that a gate accepts, where a test-verified improvement is a child that passes all three tasks it shares with its parent where the parent passed less, together with the share of accepted nodes that pass all paired tasks. **Search productivity**, the lineage-level quantity behind Co-CMP, is the share of children that pass all paired tasks, with archive depth and whether the best belief moves past the initial agent. **Agent quality** is the held-out pass rate. Intervals are 95% bootstrap intervals, and task-level agent comparisons use exact two-sided sign tests.

4.2 Experimental Design

Main result on Polyglot (Section 5.1). We evaluate Gödel Ecology in two complementary settings. Offline evaluator updating isolates the update rule. Every evaluator scores all 635 patches once, and each method runs its update rule over these shared judgments, ranking a passing and a failing patch of the same task at every step to give the in-loop AUC. We compare each method's own pool and RQGM with Gödel Ecology's pool, and repeat the comparison with every judgment re-scored by `deepseek-v4.1-flash`. Closed-loop co-evolution runs three complete searches from the same initial agent, RQGM with the CRAVE or the in-domain anchor and Gödel Ecology with the CRAVE anchor. Every child is tested on three tasks shared with its parent, so each gate decision can be checked, and each returned agent is evaluated on the 17 tasks of the external half, on which no agent runs during search.

Cross-dataset generalization (Section 5.3). On PeerRead, IMO-GradingBench, SummEval, and Change My View, both methods share six evaluator personas and the same anchor. We first compare their update rules offline, and then run the production gate and mixture unchanged in writer loops, where a writer proposes ten revisions on each of five topics per domain and independent judges outside both evaluator pools rate every draft.

Cross-model transfer (Section 5.4). The agent each Polyglot search returns is re-run unchanged on the 17 held-out tasks and on 60 tasks never used in search or anchoring, with the search backbone, `qwen3.8-flash`, and `deepseek-v4.1-flash`. Section 6.1 uses further offline studies, including a stronger evaluator released partway through search, to explain these gains. Grids, seeds, and decision rules were registered before any result was seen.

5 Results and Analyses

Gödel Ecology outperforms RQGM on Polyglot, on four non-code domains, and across agent backbones, and four mechanism studies trace the lead to its design.

Table 2: **Cross-dataset generalization** (`qwen3.6-35b-a3b`). Panel A runs offline evaluator updating over real judgments of six hand-written personas that both methods share, with the in-domain anchor half and 1,000 paired seeds. PeerRead reports accept/reject AUC and the other domains within-group pairwise agreement, and every difference from RQGM (superscripts) has a 95% CI excluding zero. Panel B runs the production gate and mixture as a writing loop with ten revisions on each of five topics per domain, from the same first drafts. Independent judges outside both pools rate every draft. SummEval first drafts mostly start at the top score.

A. Evaluator signal (in-loop pairwise agreement ↑)

	PeerRead	IMO (no ref.)	IMO (ref.)	SummEval	CMV	**Avg.**
RQGM	0.662	0.641	0.624	0.671	0.555	0.631
GE	**0.703**$^{+0.041}$	**0.651**$^{+0.010}$	**0.644**$^{+0.020}$	**0.719**$^{+0.048}$	**0.586**$^{+0.031}$	**0.661**$^{+0.030}$
Seeds with GE higher ↑	96.8%	66.6%	76.3%	97.6%	86.8%	84.8%

B. Writer loops with the production gate

	PeerRead	IMO	SummEval	CMV	**Total**
Accepted revisions, out of 50 attempts ↑					
RQGM	5	9	3	6	23
GE	**10**	**16**	**5**	**8**	**39**
Rejected revisions that the judges rate above the current draft ↓					
RQGM	17	14	6	17	54
GE	**13**	**6**	**1**	**11**	**31**
Mean score of GE's drafts, first draft → final draft ↑					
GE	0.531 → **0.670**	0.168 → **0.397**	0.941→**0.994**	0.237 → **0.295**	

5.1 Continuous Evaluation Builds a Better Search Landscape

Continuous updating catches better evaluators earlier. When the strongest evaluator is withheld and released at 15%, 30%, or 50% of the budget, GE makes it the top-weighted evaluator in 93–99% of runs. RQGM makes it the incumbent in 18–39% of runs with its own pool and 30–38% with GE's pool, fewer the later it appears, because power-of-two checkpoints grow sparse. The gap holds with either pool, so it comes from the update rule, the same rule that keeps GE ahead when both methods share its pool in Table 1 (Figure 6).

The soft mixture yields better evaluator signal. In offline evaluator updating with each method's own pool, GE reaches an in-loop AUC of 0.901 against 0.565 and 0.578 for RQGM and is higher on all 1,000 paired seeds. The lead holds when RQGM receives GE's pool (0.820 and 0.838). On the same seeds GE is higher by 0.081 and 0.064 on average, with 95% intervals above zero. It also holds when `deepseek-v4.1-flash` re-scores every judgment (0.926 against 0.610 and 0.611). Evaluators in a pool score too close together on the anchor for a small anchor batch to rank them, so RQGM often commits to a weaker one, while GE combines its top-weighted evaluators and no single choice decides its signal. The pool's best evaluator also ends with GE's largest weight in 92.4% of seeds with Qwen and in every seed with DeepSeek (Appendix A.15).

5.2 Better Evaluation Produces Better Evolution

Better judgments preserve better candidates. In closed-loop search, GE's gate accepted all three test-verified improvements its search produced, and every node it accepted passes all three paired tasks (Table 1). The CRAVE RQGM run rejected all four test-verified improvements of its own search and never left the initial agent, and the in-domain RQGM run accepted four of its six. Sharing the CRAVE RQGM run's external evidence, GE accepted six nodes against one and seven times as many per evaluation.

Productive selections compound through lineages. Building on stronger parents, 37% of GE's children pass all paired tasks, against 28% and 12%, and its archive grows three generations deep with a best belief that leaves the initial agent. The agent it selects solves 14 of the 17 held-out tasks, against 13 for the initial agent and 11 for the in-domain RQGM run.

5.3 Cross-Dataset Generalization

Gödel Ecology generalizes better beyond coding tasks. In offline evaluator updating with the same six personas and anchor for both methods, GE agrees with the ground truth more often than

Table 3: **Cross-model transfer.** Agents evolved on Polyglot with `qwen3.6-35b-a3b` are re-run unchanged with two further backbones. The RQGM column is the best-belief agent of the in-domain RQGM run, and the best-belief agent of the CRAVE RQGM run is the initial agent. The 60-task list is disjoint from all tasks used in search or anchoring. Avg. is the mean over the two transferred models. Superscripts give absolute points relative to Initial. The task-level counts in the text cover all three models and both task lists, 231 runs per agent.

	Evolved			*Transferred*								
	`qwen3.6-35b-a3b`			`qwen3.8-flash`			`ds-v4.1-flash`			Avg.(transferred)		
Held-out list	Init.	RQGM	**GE**	Init.	RQGM	**GE**	Init.	RQGM	**GE**	Init.	RQGM	**GE**
17 tasks	76.5	$64.7^{-11.8}$	**82.4**$^{+5.9}$	88.2	$88.2^{+0.0}$	**88.2**$^{+0.0}$	100.0	$100.0^{+0.0}$	**100.0**$^{+0.0}$	94.1	94.1	**94.1**
60 tasks	50.7	$80.0^{+29.3}$	**83.3**$^{+32.6}$	96.7	$90.0^{-6.7}$	**98.3**$^{+1.7}$	95.0	$96.7^{+1.7}$	**98.3**$^{+3.3}$	95.8	93.3	**98.3**

RQGM in all five domain variants, by 0.010 to 0.048, and scores higher on 84.8% of paired seeds on average (Table 2A). In writer loops of ten revisions on each of five topics per domain, GE accepts 39 of 200 revisions against 23 for RQGM, and it rejects fewer revisions that independent judges rate higher, 31 against 54, without accepting more that they rate lower (Table 2B). Outside SummEval, whose first drafts start at the top score, 13 of 15 GE's lineages accept at least one revision (Figure 3).

5.4 Cross-Model Transfer

Gödel Ecology transfers better across backbones. Re-run unchanged with `qwen3.8-flash` and `deepseek-v4.1-flash`, GE's agent never falls below the initial agent and solves 59 of the 60 held-out tasks with both, the most of the three agents (Table 3). It matches or beats RQGM's agent under every model on both task lists. Counting task by task over these six combinations, GE's agent passes 16 task runs that RQGM's agent fails and fails 5 that it passes (sign test, $p = 0.027$), while RQGM's agent fails 18 task runs that the initial agent passes and passes 6 that it fails ($p = 0.023$). The patch GE admitted improves the agent itself, independent of the evaluator and backbone under which it was selected.

6 Discussion and Limitations

6.1 What Makes Gödel Ecology Work

Evaluator utility emerges through search. In population-based RSI, an evaluator affects not only immediate judgments but also which candidates survive and which descendants can subsequently emerge. Evaluator quality therefore depends on how its decisions shape the search trajectory. In our evaluator study, CRAVE accuracy is negatively correlated with Polyglot in-loop AUC, with Spearman correlation -0.31, while the in-domain anchor gives a positive correlation of $+0.73$ and ranks the strongest evaluator first. This motivates combining external fidelity with Co-Clade Metaproductivity. The former provides a grounded signal of correctness, while the latter captures whether evaluator decisions preserve productive agent lineages, which is especially important in branching search where the value of an intermediate agent may only become clear through its descendants.

Task evidence strongly shapes evaluator quality. The information available to an evaluator is as important as the evaluator itself. Providing the task's test files, while withholding test outcomes, raises all eight evolved evaluators from 0.53–0.60 to 0.78–0.90 AUC with Qwen and produces the same qualitative improvement with DeepSeek. Repository source alone provides almost no gain. These results suggest that evaluator evolution is most useful when the evaluator can access evidence that is directly relevant to the decision it must make. The evolution of evaluator and task grounding therefore address complementary parts of the problem.

Continuous adaptation requires external grounding. Coupled evolution creates a risk that agents and evaluators reinforce one another without improving the underlying task objective. Gödel Ecology limits this feedback in two ways. Descendant credit is delayed, so an evaluator cannot immediately benefit from judgments it has just produced. Large mixture shifts and probationary interventions are also checked against independent anchor evidence. The non-code interventions illustrate this role clearly. A candidate that appeared useful during probation was withdrawn when its anchor evidence was worse, while another candidate was promoted when external evidence supported it despite weaker short-run behavior. This separation between adaptive search pressure and externally grounded validation appears important for stable collective RSI.

6.2 Beyond Verifiable Search

Coding tasks provide a strong external signal because candidate solutions can ultimately be checked by tests. The four non-code domains remove this verifiable floor. In these settings, evaluator mixtures, gate updates, and probationary interventions operate using learned judgments together with anchor-based checks. Gödel Ecology improves evaluator agreement across all five evaluated variants and rejects fewer revisions that independent judges prefer to the current draft. Our experiments show that the evaluator-update mechanism can operate beyond executable rewards. They do not yet demonstrate open-ended self-improvement in fully judgment-based domains. The non-code experiments use controlled writer-revision loops, while the main RSI experiments evolve agent populations. Extending the same framework to closed-loop agent evolution in domains without deterministic verification is a natural next step.

6.3 Limitations

Gödel Ecology still requires labeled external evidence to control evaluator drift, especially at the beginning of the evolution. Verifiable domains can derive such labels from held-out tests, while judgment-based domains require human labels or other forms of trusted external signal. Our results also suggest that external supervision can be most useful when it reflects the decisions that matter inside the search process.

The evaluator mixture requires multiple model calls for each candidate. Lazy invalidation avoids immediate rescoring of the full archive after each weight update, but the cost will grow with larger evaluator populations. Lightweight surrogate evaluators, mixture distillation, and adaptive evaluator selection are promising ways to improve scalability (Appendix A.18). Meanwhile, the current search primarily modifies prompts, workflows, and local execution scaffolds. Larger harness changes may require evidence aggregated across broader task distributions and longer evolutionary horizons. Extending collective credit assignment to these slower structural modifications remains open.

7 Related Work

Self-improving agents. Modern systems replace the provably useful rewrites of the Gödel Machine (Schmidhuber, 2007) with empirical validation (Yin et al., 2025), from DGM's open-ended archive under a fixed benchmark (Zhang et al., 2026) and HGM's clade metaproductivity (Wang et al., 2026) to the pairwise mutation signals of the Mendel Gödel Machine (Liu et al., 2026a). RQGM (Iacob et al., 2026) lets evaluators co-evolve under epoch-scheduled turnover, and GE replaces its epoch-frozen incumbent with a continuously updated mixture scored by clade productivity (Appendix A.2). Harness-level self-improvement (Lin et al., 2026) works at a coarser granularity (Appendix A.3).

Evaluators that adapt. LLM judges with structured criteria can match human ratings (Liu et al., 2023; Kim et al., 2024), and instance-specific rubrics extend verifiable-reward training to open-ended tasks (Gunjal et al., 2026), with the criteria fixed during training. EvoRubrics (Ding et al., 2026) adapts rubrics while training a single policy, and Janus (Liu et al., 2026b) audits generator-evaluator pairs against an immutable anchor. Strong self-judging need not yield downstream improvement (Shi et al., 2026), and interacting populations are prone to mutual gaming (Ficici & Pollack, 1998), so GE scores evaluators by the productivity of the lineages they select and ties admission to verifiable floors.

8 Conclusion

We introduced Gödel Ecology, a framework for collective RSI through productivity-guided continuous co-evolution of agents and evaluators. Gödel Ecology maintains a continuously updated evaluator mixture and rewards evaluators for both external fidelity and the downstream productivity of the lineages they shape. Across controlled evaluator updating, closed-loop search, non-code domains, and cross-backbone transfer, this design yields consistently stronger evaluation signals and more effective search than epoch-based turnover. These results suggest that collective self-improvement benefits when evaluation evolves continuously with the agents it guides.

AI USE STATEMENT

In this work, we used generative AI tools (large language models) to implement parts of the experiment code, to analyze and interpret experimental results. We also used them to draft and revise the manuscript text, to write the scripts that produce the result figures, and to create the illustration in Figure 1. Generative AI was not involved in the design of methodology and experiments, the analysis and interpretation of results, generating synthetic datasets, or proving mathematical claims. All scientific content, analyses, results, and final wording were reviewed and approved by the authors.

REPRODUCIBILITY STATEMENT

Section 4 describes the benchmarks, splits, models, and metrics, and Appendix A.14 lists every study with its protocol, seeds, and decision rules (Table 9). Appendix A.4 gives all hyperparameters, including the tuned offline configurations, and Appendix A.6 reproduces the prompt templates. Algorithm 1 and Appendices A.7 to A.13 specify the search loop, scheduling, mixture updates, interventions, and acceptance gate. All data sets we use are public, and the offline studies run over stored evaluator judgments without further model calls.

REFERENCES

Aider-AI. Polyglot benchmark. https://github.com/Aider-AI/polyglot-benchmark, 2024.

Hongxin Ding, Baixiang Huang, Yue Fang, Weibin Liao, Zheng Li, Jinyang Zhang, Zhijing Wu, Junfeng Zhao, and Yasha Wang. EvoRubrics: Dynamic rubrics as rewards via adversarial co-evolution for LLM reinforcement learning. *arXiv preprint arXiv:2606.23038*, 2026.

Alexander R. Fabbri, Wojciech Kryściński, Bryan McCann, Caiming Xiong, Richard Socher, and Dragomir Radev. SummEval: Re-evaluating summarization evaluation. *Transactions of the Association for Computational Linguistics*, 9:391–409, 2021. doi: 10.1162/tacl_a_00373. URL https://aclanthology.org/2021.tacl-1.24/.

Sevan G. Ficici and Jordan B. Pollack. Challenges in coevolutionary learning: Arms-race dynamics, open-endedness, and mediocre stable states. In *Artificial Life VI*, pp. 238–247, 1998.

Anisha Gunjal, Anthony Wang, Elaine Lau, Vaskar Nath, Yunzhong He, Bing Liu, and Sean Hendryx. Rubrics as rewards: Reinforcement learning beyond verifiable domains. 2026:127924–127945, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/cfd7bee7a651ee9af525098ef67a9e45-Paper-Conference.pdf.

Alex Iacob, Andrej Jovanović, William F. Shen, Daniel Burkhardt, Meghdad Kurmanji, Nurbek Tastan, Lorenzo Sani, Niccolò Alberto Elia Venanzi, Ambroise Odonnat, Zeyu Cao, Bill Marino, Xinchi Qiu, and Nicholas D. Lane. The Red Queen Gödel Machine: Co-evolving agents and their evaluators. *arXiv preprint arXiv:2606.26294*, 2026.

Carlos E Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik Narasimhan. Swe-bench: Can language models resolve real-world github issues? 2024:54107–54157, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/file/edac78c3e300629acfe6cbe9ca88fb84-Paper-Conference.pdf.

Dongyeop Kang, Waleed Ammar, Bhavana Dalvi, Madeleine van Zuylen, Sebastian Kohlmeier, Eduard Hovy, and Roy Schwartz. A dataset of peer reviews (PeerRead): Collection, insights and NLP applications. In Marilyn Walker, Heng Ji, and Amanda Stent (eds.), *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pp. 1647–1661, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-1149. URL https://aclanthology.org/N18-1149/.

Seungone Kim, Jay Shin, yejin cho, Joel Jang, Shayne Longpre, Hwaran Lee, Sangdoo Yun, S Shin, Ryan, Sungdong Kim, James Thorne, and Minjoon Seo. Prometheus: Inducing fine-grained evaluation capability in language models. In B. Kim, Y. Yue, S. Chaudhuri, K. Fragkiadaki, M. Khan, and Y. Sun (eds.), *International Conference on Learning Representations*, volume 2024, pp.

29927–29962, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/file/803485352e61e3ebf41221e4776c9fd4-Paper-Conference.pdf.

Jiahang Lin, Shichun Liu, Chengjun Pan, Lizhi Lin, Shihan Dou, Xuanjing Huang, Hang Yan, Zhenhua Han, and Tao Gui. Agentic harness engineering: Observability-driven automatic evolution of coding-agent harnesses. *arXiv preprint arXiv:2604.25850*, 2026.

Changzhi Liu, Yilun Liu, Sikuan Yan, Volker Tresp, and Yunpu Ma. Mendel Gödel Machine: Recursive self-improving coding agents via comparative evolution. *arXiv preprint arXiv:2608.07645*, 2026a.

Ximeng Liu, Qianlong Wang, Yingming Mao, Annan Li, Yatao Li, Shizhen Zhao, Jianmin Wu, Dawei Yin, and Dou Shen. Janus: An algorithm-evaluator co-evolution framework for LLM-driven discovery. *arXiv preprint arXiv:2608.08189*, 2026b.

Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. G-eval: NLG evaluation using gpt-4 with better human alignment. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 2511–2522, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.153. URL https://aclanthology.org/2023.emnlp-main.153/.

Thang Luong, Dawsen Hwang, Hoang H Nguyen, Golnaz Ghiasi, Yuri Chervonyi, Insuk Seo, Junsu Kim, Garrett Bingham, Jonathan Lee, Swaroop Mishra, Alex Zhai, Huiyi Hu, Henryk Michalewski, Jimin Kim, Jeonghyun Ahn, Junhwi Bae, Xingyou Song, Trieu Hoang Trinh, Quoc V Le, and Junehyuk Jung. Towards robust mathematical reasoning. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 35418–35442, Suzhou, China, November 2025. Association for Computational Linguistics. ISBN 979-8-89176-332-6. doi: 10.18653/v1/2025.emnlp-main.1794. URL https://aclanthology.org/2025.emnlp-main.1794/.

John C. Platt. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In *Advances in Large Margin Classifiers*, pp. 61–74. MIT Press, 1999.

Jürgen Schmidhuber. Gödel machines: Fully self-referential optimal universal self-improvers. In Ben Goertzel and Cassio Pennachin (eds.), *Artificial General Intelligence*, pp. 199–226. Springer, Berlin, Heidelberg, 2007. doi: 10.1007/978-3-540-68677-4_7.

Jiajun Shi, Siyuan Tao, Yuhao Wu, Zexuan Wang, Jingyuan Zhang, Jiaheng Liu, Xinping Lei, Xinrong Zhang, Siyuan Fang, Zhewen Tan, Tianle Cai, Junhao Fang, Jiameng Huang, Yueyang Wang, Jinkai Liu, Yuxuan Zhang, Jian Yang, Zhoujun Li, Shen Yan, Wenhao Huang, and Ge Zhang. S^3Gym: Can LLMs turn self-testing and self-judging into self-improvement? *arXiv preprint arXiv:2608.31100*, 2026.

Chenhao Tan, Vlad Niculae, Cristian Danescu-Niculescu-Mizil, and Lillian Lee. Winning arguments: Interaction dynamics and persuasion strategies in good-faith online discussions. In *Proceedings of the 25th International Conference on World Wide Web*, WWW '16, pp. 613–624, Republic and Canton of Geneva, CHE, 2016. International World Wide Web Conferences Steering Committee. ISBN 9781450341431. doi: 10.1145/2872427.2883081. URL https://doi.org/10.1145/2872427.2883081.

Wenyi Wang, Piotr Piękos, Li Nanbo, Firas Laakom, Yimeng Chen, Mateusz Ostaszewski, Mingchen Zhuge, and Jürgen Schmidhuber. Huxley-g\"odel machine: Human-level coding agent development by an approximation of the optimal self-improving machine. 2026:80356–80386, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/821d20219c2f14850af1b5220f0ed13f-Paper-Conference.pdf.

Yizao Wang, Jean-yves Audibert, and Rémi Munos. Algorithms for infinitely many-armed bandits. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou (eds.), *Advances in Neural Information Processing Systems*, volume 21. Curran Associates, Inc., 2008. URL https://proceedings.neurips.cc/paper_files/paper/2008/file/49ae49a23f67c759bf4fc791ba842aa2-Paper.pdf.

Xunjian Yin, Xinyi Wang, Liangming Pan, Li Lin, Xiaojun Wan, and William Yang Wang. Gödel agent: A self-referential agent framework for recursively self-improvement. pp. 27890–27913, July 2025. doi: 10.18653/v1/2025.acl-long.1354. URL https://aclanthology.org/2025.acl-long.1354/.

Jenny Zhang, Shengran Hu, Cong Lu, Robert Lange, and Jeff Clune. Darwin gödel machine: Open-ended evolution of self-improving agents. 2026:104223–104294, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/aa5f5e6eb6f613ec412f1d948dfa21a5-Paper-Conference.pdf.

Li Zhang. CRAVE: Code review agent verdict evaluation. https://huggingface.co/datasets/TuringEnterprises/CRAVE, 2025. Code review classification dataset from curated human code reviews.

A Appendix

A.1 Notation

Table 4 summarizes the state variables and evaluation quantities updated during search, and Table 6 lists the hyperparameters. Test tasks evaluate the returned best-belief node $a^\star$ alone, and mixtures and intervention windows are confined to search.

Table 4: Notation.

Symbol	Description
$\mathcal{T}$	Unified archive tree; each node $a \in \mathcal{T}$ carries scaffolds (π_a, ξ_a)
$\mathcal{C}(a)$	Lineage clade (descendant subtree) rooted at node a
$a^\star$	Deployed best-belief coding-agent node returned by search
B	Total evaluation budget
$v(a \mid \tau)$	Verifiable test-execution outcome of agent a on task $\tau \in \{0, 1\}$
$\hat{u}_{\xi_n}(a \mid \tau)$	Learned scalar score assigned by evaluator candidate ξ_n
$\mathcal{D}_{\text{ext}}$	Partitioned anchor set $\mathcal{D}_{\text{ext}}^{\text{shared}} \cup \bigcup_a \mathcal{D}_{\text{ext}}^{\text{disjoint}}(a)$ with disjoint complements
$n_{\text{s}}(a), n_{\text{f}}(a)$	Raw success and failure counts observed at node a
$\tilde{n}_{\text{s}}(a), \tilde{n}_{\text{f}}(a)$	Posterior-weighted counts under current evaluation mixture w_t
$u_\alpha(a)$	Conservative α-best-belief score of agent node a
$\mathcal{C}_{\text{ev}} \subseteq \mathcal{T}$	Active evaluator candidate pool with sufficient calibration evidence
$w_t(n)$	Mixture weight of evaluator candidate ξ_n at search step t
$\text{Co-CMP}_\pi(a)$	Agent-side Co-Clade Metaproductivity
$\text{Co-CMP}_\xi(n)$	Evaluator-side Co-Clade Metaproductivity
$f(a)$	Failed-`Expand` counter at node a (self-improvement errors and failed first evaluations)
$\mathcal{P}_{\text{pend}}$	Pending children awaiting the dual gate
$\tau_{\text{cool}}, \mathcal{P}_{\text{cool}}$	Cooldown threshold and pool excluding starved parents from Thompson sampling (Section 3.3)
ν_{gate}	Minimum Eval_π evaluations a child must accumulate before the dual acceptance gate is evaluated
$\nu_{\min}$	Minimum shared-anchor evidence required before a large mixture-weight jump or intervention verdict is adjudicated
W_{gate}	EMA window length for the acceptance gate's soft Pareto comparison, $\rho_w = 2/(W_{\text{gate}} + 1)$

A.2 Positioning against Earlier Gödel Machines

Table 5 contrasts GE with DGM, HGM, and RQGM along the components that differ.

A.3 Extended Related Work

This appendix extends Section 7.

Table 5: **Positioning against earlier Gödel Machine architectures.**

	DGM	HGM	RQGM	**GE**
Evaluation criterion	fixed benchmark	fixed benchmark	co-evolving evaluators	co-evolving evaluators
Evaluator signal	–	–	one incumbent per epoch	continuous soft mixture
Evaluator update	–	–	power-of-two checkpoints	every step
Evaluator score	–	–	anchor fidelity	anchor fidelity + Co-CMP productivity
Node selection	score and fan-out	CMP, Thompson sampling	CMP	Co-CMP, Thompson sampling
Admission	score	score	score	dual gate: verifiable floor + learned Pareto

A.3.1 Gödel Machines and Population-Based Code RSI

The Gödel Machine (Schmidhuber, 2007) formalizes recursive self-improvement through utility-improving rewrites, with modern implementations substituting empirical validation for intractable formal proofs (Yin et al., 2025). DGM (Zhang et al., 2026), HGM (Wang et al., 2026), and RQGM (Iacob et al., 2026) progressively extended this foundation from static benchmark archives to non-stationary evaluator search. Section 2 formalizes HGM's search and RQGM's evaluator turnover. Table 5 summarizes the architectural differences across this lineage. GE inherits the unified workspace and selective-erasure framework of RQGM, replacing epoch-frozen evaluators with continuous mixtures and extending promotion beyond anchor agreement to include descendant clade productivity.

The Mendel Gödel Machine (Liu et al., 2026a) extracts mutation signals through pairwise comparisons on single trajectories. GE instead aggregates mutation evidence across descendant clades, using a delayed evaluation window to prevent circular scoring between agents and evaluators.

A.3.2 Co-Evolutionary Evaluators and Admission-Decoupled Co-Evolution

Co-evolutionary systems are notoriously susceptible to mutual gaming and convergence to mediocre stable states (Ficici & Pollack, 1998). A central challenge is therefore designing evaluation criteria that adapt alongside agents without losing objectivity. Rubric-based reward signals laid the foundation. RaR (Gunjal et al., 2026) extends RLVR to non-verifiable domains through instance-specific rubrics, while G-Eval (Liu et al., 2023) and Prometheus (Kim et al., 2024) show that LLM judges operating on structured criteria can match human evaluation quality. These approaches treat the evaluation criterion as externally fixed and do not address what happens when the criterion must itself adapt as agent capability evolves.

EvoRubrics (Ding et al., 2026) demonstrates that dynamically adapted rubrics improve policy robustness compared to static reward models. While this confirms that frozen evaluators misrepresent an evolving capability frontier, EvoRubrics operates on a single model during RL training. In contrast, GE governs an open-ended population archive with branching lineages and shared infrastructure.

Janus (Liu et al., 2026b) demonstrates the benefit of admission-decoupled co-evolution, using an immutable external anchor to audit generator-evaluator pairs during training. GE extends this auditing philosophy from single-policy training to open-ended population search. In an archive setting, continuous evaluator adaptation risks corrupting historical evaluation records across branching lineages, and GE insulates search trees from this drift by combining immutable verifiable floors with time-averaged Pareto gates. We refer to this architectural pattern of pairing rapid continuous adaptation for exploration with immutable grounding for admission as an *admission-decoupled* (or *audited*) co-evolution design.

Recent empirical evaluations by Shi et al. (2026) substantiate this separation between evaluation accuracy and policy progress: models that excel at self-judging often fail to yield downstream task improvements. An evaluator may score accurately on fixed criteria yet penalize intermediate stepping-stone trajectories that lead to productive search branches. Co-CMP directly incorporates this distinction into evaluator scoring by conditioning evaluator weights on the empirical productivity of descendant agent clades. Our anchor measurements give a concrete instance of the separation: agreement with an external review corpus is negatively correlated with an evaluator's accuracy inside the search loop, while agreement with in-domain anchors is strongly positively correlated (Section 6.1).

Table 6: GE hyperparameters used in closed-loop search on Polyglot. RQGM uses $h = 4$ and checkpoint ratio $r = 2$ (checkpoints at powers of two), $\epsilon = 0.05$, and $n_{\min} = 5$ with the same gate and budget. In offline evaluator updating, each method is tuned on 60 configurations from a frozen grid; the selected configurations are, for GE, $\eta = 3.24$, $\zeta = 0.48$, $w_{\min} = 0.061$, $\gamma = 0.035$, $\nu_{\min} = 8$, $\delta = 0.055$, $\kappa = 26$, $\phi = 0.21$, $k = 5$, anchor batch 6; for RQGM with the CRAVE anchor, $h = 15.8$, $r = 2.50$, $\epsilon = 0.132$, $n_{\min} = 5$, anchor batch 7; and with in-domain anchors, $h = 2.55$, $r = 2.69$, $\epsilon = 0.121$, $n_{\min} = 10$, anchor batch 29.

Symbol	Default	Role
λ_{start}	0.9	Initial anchor-vs-productivity balance in Co-CMP$_\xi$
$\lambda_{\min}$	0.3	Floor of annealed λ_t
η	2.0	Softmax temperature for mixture weights
δ	0.15	Mixture-shift threshold triggering lazy invalidation
ρ	0.6	UCB-Air expansion-gate exponent
ζ	0.1	Entropy regularization coefficient
$w_{\min}$	0.02	Explicit weight floor per evaluator
γ	0.05	Significance threshold for large weight jumps
Δ	10	Delay window (steps) for agent-productivity snapshot
κ	50	Intervention period (steps)
ϕ	0.3	Probationary weight for the candidate evaluator
α	0.05	Best-belief confidence level (inherited from HGM)
β	0.5	Verifiable-vs-learned balance; learned scores are Platt-scaled (Platt, 1999) onto a common probability scale
$\varepsilon_{\text{gate}}$	0.05	Conservative verifiable-signal floor in acceptance gate
ν_{gate}	3	Minimum evaluations before dual gate applies (Section 3.3)
τ_{cool}	2	Failed-expansion limit before parent cooldown (Section 3.3)
$\nu_{\min}$	5	Minimum shared-anchor evidence for jump/intervention verdicts
ρ_w	0.2	EMA rate for the gate's soft comparison (W_{gate} not fixed; ρ_w set directly)
n_{ext}	64	Anchor items drawn per evaluator calibration

A.3.3 Boundary: Positioning against Harness Evolution

Agentic Harness Engineering (Lin et al., 2026) explores self-improvement at the level of shared execution harnesses, validating systemic failure patterns across task distributions before adopting infrastructure changes. GE instead explores candidate modifications within an expanding archive tree, evaluating mutations through localized clade metaproductivity. Integrating distribution-level failure clustering into the `Expand` gate offers a pathway to support multi-file harness edits within tree-search RSI architectures.

A.4 Hyperparameters

See Table 6.

A.5 Asynchronous Execution Architecture

GE's execution loop (Algorithm 1) is specified independently of scheduling. The runs in Section 5 use two concurrent `Expand` workers in containerized execution. This appendix describes the scaling architecture that this execution follows: an asynchronous, actor-based master-worker design (Ray/asyncio) that decouples the three action types in $\mathcal{A}$ from a single blocking loop as the archive and evaluator pool grow.

A central **Search Coordinator Actor** holds the canonical archive $\mathcal{T}$, the active and time-averaged mixture weights w_t and $\bar{w}_t$, the length-Δ delay ring buffer that supplies $\widehat{\text{Co-CMP}}_\pi^{(t-\Delta)}$ to Eq. 3, and the priority dispatch queues that route work to three dynamically scalable worker pools:

- **`Expand` workers** execute the meta-agent's asynchronous workspace-edit proposals and mount successful mutations as `PENDING` children without blocking on their own downstream evaluation.
- **`Eval`$_\pi$ workers** execute candidate agent scaffolds inside sandboxed task environments, collect verifiable outcomes $v(a \mid \tau)$, and distribute learned scoring across worker-local replicas of the top-k evaluators currently carrying non-negligible mixture weight.

- **$\texttt{Eval}_\xi$ workers** calibrate evaluator candidates against $\mathcal{D}_{\text{ext}}^{\text{shared}}$ and $\mathcal{D}_{\text{ext}}^{\text{disjoint}}(a)$, returning the binary alignment records that feed $u_\alpha^{\text{ext}}(n)$.

The coordinator is the sole owner of $\mathcal{T}$, w_t, and $\bar{w}_t$; worker pools report results back to it rather than mutating shared state directly, which is what lets the lockless update rule of Appendix A.8 remain correct as `Expand`, $\texttt{Eval}_\pi$, and $\texttt{Eval}_\xi$ execute concurrently.

A.6 Prompt Templates

The policy and meta-agent prompts follow the scaffolds of DGM (Zhang et al., 2026) as reused by HGM (Wang et al., 2026). We add the implementation requirement to the policy prompt and the one-change, no-hard-coding instructions to the meta-agent prompt, and the evaluation scaffold prompt is new. Excerpts below are trimmed to the scaffold structure; boilerplate JSON-repair retries and the full per-repository tool listing are omitted for space and noted where cut.

Meta-agent prompt (`Expand`). The meta-agent is shown the current coding-agent implementation and the failure evidence from the task that triggered self-improvement, then asked for one general, non-hard-coded fix:

```
Here is the implementation of the coding agent.

# Coding Agent Implementation
----- Coding Agent Implementation Start -----
{code}
----- Coding Agent Implementation End -----

Identify ONE concrete change to this agent that would help it produce a
*correct* patch on tasks like the one shown in the logs, based on its
actual failure mode (misunderstood the issue, insufficient exploration,
poor tool use, or an edge case its patch missed).

The fix must be general: it must not hard-code this task's file paths,
names, or keywords, and must rely only on signals available at runtime
in any repo (failing test/traceback, files actually opened, symbols in
the issue, the diff itself).
```

The evidence block (GitHub issue, agent run log, predicted patch, private test patch, issue test results) is then appended, and the meta-agent is asked to return a single JSON object with keys `log_summarization`, `potential_improvements`, `improvement_proposal`, `implementation_suggestion`, and `problem_description`. The last field is turned into the `Expand` instruction actually executed:

```
# To Implement

{implementation_suggestion}

{problem_description}

---
Before finishing: name the exact file and function you changed, and
confirm the mechanism is general (would still work on an unrelated
repo) rather than hard-coding this task's paths/names/keywords.
```

Policy scaffold prompt (π_a, $\texttt{Eval}_\pi$). The coding agent receives the task statement, a test-execution hint derived from the task's own eval script, and an explicit instruction not to stop before a real source-file diff exists:

```
I have uploaded a Python code repository in the directory {git_tempdir}.
Help solve the following problem.

<problem_description>
{problem_statement}
</problem_description>

<test_description>
{test_description}
</test_description>

Your task is to make changes to the files in the {git_tempdir} directory
```

```
to address the <problem_description>. I have already taken care of the
required dependencies.

You must implement the fix in the repository before finishing. Do not
stop after describing a plan or claiming that a change is already
present. Use the editor tool to modify the relevant source files, not
only tests or documentation. Before your final response, run
`git status --short` and `git diff --stat`; if no source file has
changed, continue investigating and editing until the requested fix is
actually present. Do not create an empty or placeholder patch.
```

`{test_description}` is filled from the task's own eval script (e.g. "The tests in the repository can be run with the bash command `{test_command}`..."), and tool availability is appended separately in a fixed `<tool_use>{'tool_name': ..., 'tool_input': ...}</tool_use>` format listing every tool currently in `tools/`, omitted here since its length scales with the current tool count.

Evaluation scaffold prompt (ξ_a, $\texttt{Eval}_\xi$). Learned evaluation is a single LLM-judge call constrained to the visible evidence, conditioned on the evaluator candidate's own rubric. The task's test files enter as a specification; no test outcome is ever provided:

```
System: You are a rigorous software-engineering evaluator. Be
conservative and evidence-based.
Evaluator rubric:
{prompt_template}

Judge whether a code patch is likely to resolve the software-engineering
task.

Task ID: {task_id}
Candidate commit: {agent_commit_id}

Task statement:
{problem_statement}

Test files (the visible task specification; test results are NOT provided):
{test_file_content}

Candidate patch:
{code_patch}

Verifiable test summary:
No verifiable test outcome is provided to the evaluator.

Assess only the evidence shown. Do not claim tests were run. Return
exactly one JSON object with:
- "score": a number from 0.0 to 1.0 estimating the probability the patch
  resolves the task;
- "rationale": a concise evidence-based explanation.
```

The seed rubric substituted into `{prompt_template}` is a short generic instruction; evolved rubrics replace it as the archive grows. The designated candidate injected at intervention checkpoints (Appendix A.10) uses the following rubric:

```
Adversarial Debiasing Evaluator: Rigorous Adversarial Audit. Actively
scrutinize candidate patches for false positives, edge-case regressions,
test-gaming, hidden bugs, and incumbent bias. Be strictly
evidence-grounded and conservative.
```

A.7 Value-of-Information: Implementation Details

One-step variance reduction. For binary evaluation metrics modeled as $\theta \sim \mathrm{Beta}(a, b)$ with pseudo-counts $a = 1+n_\mathrm{s}, b = 1+n_\mathrm{f}$, the expected variance reduction from one additional Bernoulli observation follows the standard conjugate closed form:

$$\Delta\mathrm{Var}(a, b) = \frac{ab}{(a+b)^2(a+b+1)^2}. \tag{6}$$

For coding-agent clades, $\Delta\mathrm{Var}_\pi(a)$ is evaluated on pooled clade counts $S_a = 1 + \sum_{a' \in \mathcal{C}(a)} \tilde{n}_\mathrm{s}(a')$ and $F_a = 1 + \sum_{a' \in \mathcal{C}(a)} \tilde{n}_\mathrm{f}(a')$; for evaluator candidates, $\Delta\mathrm{Var}_\xi(n)$ is evaluated on anchor calibration counts $S_n = 1 + S_n^\mathrm{gt}$ and $F_n = 1 + F_n^\mathrm{gt}$.

Ranking uncertainty weight. Let $a^\star = \arg\max_{a \in \mathcal{T}} u_\alpha(a)$ be the current empirical leader and $a^{(2)}$ the runner-up. The ranking uncertainty weight measures the posterior probability that node a outperforms the runner-up's conservative lower bound:

$$\omega(a) = \Pr_{U \sim \mathrm{Beta}(S_a, F_a)}\big[U > u_\alpha(a^{(2)})\big] = 1 - I_{u_\alpha(a^{(2)})}(S_a, F_a), \tag{7}$$

where $I_x(a, b)$ denotes the regularized incomplete Beta function. For nodes whose upper credible bound already falls below $u_\alpha(a^{(2)})$, $\omega(a) \approx 0$, which focuses the evaluation budget on clades with a non-trivial probability of becoming the new best-belief candidate.

Token cost tracking and action dispatch. Observed token costs are smoothed via exponential moving averages with factor $\rho_c \in (0, 1)$:

$$\bar{c}_\pi(a, \tau) \leftarrow (1 - \rho_c)\, \bar{c}_\pi(a, \tau) + \rho_c\, c_{\text{observed}}, \qquad \bar{c}_\xi(n) \leftarrow (1 - \rho_c)\, \bar{c}_\xi(n) + \rho_c\, c_{\text{observed}}. \tag{8}$$

At each step where the UCB-Air expansion gate is closed, the coordinator identifies $(a^\star, \tau^\star) = \arg\max_{(a,\tau)} \mathrm{VoI}_\pi(a, \tau)$ and $n^\star = \arg\max_n \mathrm{VoI}_\xi(n)$, dispatching:

$$\text{Action} = \begin{cases} \mathrm{EVAL}_\pi(a^\star, \tau^\star), & \text{if } \mathrm{VoI}_\pi(a^\star, \tau^\star) \geq \mathrm{VoI}_\xi(n^\star), \\ \mathrm{EVAL}_\xi(n^\star), & \text{otherwise.} \end{cases} \tag{9}$$

A.8 Clade-Weighted Thompson Sampling and Lineage Propagation

Section 3.3 samples parents via Thompson sampling over $\widehat{\text{Co-CMP}}_\pi$, drawing on the pooled clade posterior implicit in Eq. 2; this appendix makes that posterior and its update rule explicit, and specifies how the coordinator of Appendix A.5 keeps it correct as evaluation results arrive concurrently from multiple $\texttt{Eval}_\pi$ workers.

Beta-Bernoulli clade posterior. For node a with clade $\mathcal{C}(a) = \{a\} \cup \mathrm{Descendants}(a)$, write $\tilde{S}(a) = \sum_{a' \in \mathcal{C}(a)} \tilde{n}_{\mathrm{s}}(a')$ and $\tilde{F}(a) = \sum_{a' \in \mathcal{C}(a)} \tilde{n}_{\mathrm{f}}(a')$ for the pooled clade counts appearing in the numerator and denominator of Eq. 2. Under a $\mathrm{Beta}(1, 1)$ prior, the posterior over the clade success rate is $\mathrm{Beta}(1 + \tilde{S}(a),\, 1 + \tilde{F}(a))$, with mean

$$\mathbb{E}[\theta_a] = \frac{1 + \tilde{S}(a)}{2 + \tilde{S}(a) + \tilde{F}(a)} \approx \widehat{\text{Co-CMP}}_\pi(a). \tag{10}$$

During expansion, each candidate parent a in the pool of Section 3.3 (excluding $\mathcal{P}_{\text{cool}}$ unless doing so would leave no eligible parent) draws $\theta_a \sim \mathrm{Beta}(1 + \tilde{S}(a),\, 1 + \tilde{F}(a))$, and the sampler selects $a^\star = \arg\max_a \theta_a$ over that pool.

Lockless lineage propagation. When an evaluation result $(\Delta\tilde{n}_{\mathrm{s}}, \Delta\tilde{n}_{\mathrm{f}})$ arrives for node a from an $\texttt{Eval}_\pi$ worker, the coordinator applies an atomic increment to every ancestor of a rather than recomputing clade sums from scratch:

$$\tilde{S}(u) \leftarrow \tilde{S}(u) + \Delta\tilde{n}_{\mathrm{s}}, \qquad \tilde{F}(u) \leftarrow \tilde{F}(u) + \Delta\tilde{n}_{\mathrm{f}}, \qquad \forall u \in \{a\} \cup \mathrm{Ancestors}(a). \tag{11}$$

Because this increment is the sole write to shared clade state, worker pools sample and score against local, atomically-read snapshots of the relevant Beta parameters rather than acquiring a tree-wide lock; Thompson sampling in the `Expand` pool and scoring in the $\texttt{Eval}_\pi$/$\texttt{Eval}_\xi$ pools of Appendix A.5 therefore never block each other.

A.9 Continuous Mixture Jump-Gating

A.9.1 Evidence Partitioning and Posterior Quantile Lower Bound

Anchor evidence is partitioned into lineage-isolated sets $\mathcal{D}_{\text{ext}}^{\text{disjoint}}(a)$ (for Co-CMP calculation) and global shared set $\mathcal{D}_{\text{ext}}^{\text{shared}}$ (for gating tests). For candidate n, the shared binomial evidence $(S_n^{\text{shared}}, F_n^{\text{shared}})$ yields the quantile lower bound at confidence level γ:

$$BB_\gamma(n, t) = I_\gamma^{-1}\Big(1 + S_n^{\text{shared}}(t),\; 1 + F_n^{\text{shared}}(t)\Big). \tag{12}$$

A.9.2 Proposed Weight Construction with Entropy Floor

With effective temperature $\eta_{\text{eff}} = \frac{\eta}{1+\zeta}$, the normalized softmax weights are:

$$\bar{w}_t(n) = \frac{\exp\Big(\eta_{\text{eff}} \cdot \widehat{\text{Co-CMP}}_\xi(n,t)\Big)}{\sum_{n' \in \mathcal{C}_{\text{ev}}} \exp\Big(\eta_{\text{eff}} \cdot \widehat{\text{Co-CMP}}_\xi(n',t)\Big)}. \tag{13}$$

Projecting with weight floor $w_{\min}$ yields the proposed weights:

$$\tilde{w}_t(n) = (1 - |\mathcal{C}_{\text{ev}}| \cdot w_{\min}) \cdot \bar{w}_t(n) + w_{\min}. \tag{14}$$

A.9.3 Gating Trigger and Adjudication Rule

Let $\Delta_n(t) = \tilde{w}_t(n) - w_{t-1}(n)$ and $n_t^\star = \arg\max_{n \in \mathcal{C}_{\text{ev}}} \Delta_n(t)$. With dynamic threshold $\tau_{\text{gate}} = c \cdot \frac{1}{|\mathcal{C}_{\text{ev}}|}$ and reference incumbent $n_{\text{ref}}(t) = \arg\max_n w_{t-1}(n)$:

$$w_t = \begin{cases} \tilde{w}_t, & \text{if } \Delta_{n_t^\star}(t) \le \tau_{\text{gate}} \\ \tilde{w}_t, & \text{if } \Delta_{n_t^\star}(t) > \tau_{\text{gate}} \ \wedge \ (S^{\text{shared}}_{n_t^\star} + F^{\text{shared}}_{n_t^\star} \ge \nu_{\min}) \\ & \quad \wedge \ BB_\gamma(n_t^\star, t) > BB_\gamma(n_{\text{ref}}(t), t) \\ w_{t-1}, & \text{otherwise (reject update entirely)} \end{cases} \tag{15}$$

A.10 Intervention State Machine

Intervention is triggered periodically or upon calibration staleness of the reference evaluator $n_{\text{ref}}(t) = \arg\max_n w_{t-1}(n)$:

$$\text{Trigger}(t) = \big[t \bmod \kappa = 0\big] \ \vee \ \big[t - \text{last_anchor_step}(n_{\text{ref}}(t)) \ge \kappa\big]. \tag{16}$$

During active intervention over $[t_{\text{start}}, t_{\text{end}}]$ with $t_{\text{end}} = t_{\text{start}} + \kappa$, the candidate n_{adv} is assigned probationary weight ϕ:

$$w_t(n_{\text{adv}}) := \phi, \qquad \forall t \in [t_{\text{start}}, t_{\text{end}}]. \tag{17}$$

The remaining evaluators are renormalized over $(1 - \phi)$ with entropy floor protection.

At t_{end}, adjudication proceeds:

$$\text{Passed} = \big[S^{\text{shared}}_{n_{\text{adv}}} + F^{\text{shared}}_{n_{\text{adv}}} \ge \nu_{\min}\big] \ \wedge \ \big[BB_\gamma(n_{\text{adv}}, t_{\text{end}}) > BB_\gamma(n_{\text{ref}}, t_{\text{end}})\big]. \tag{18}$$

If passed, n_{adv} enters regular continuous soft mixture updating. If failed, ϕ is withdrawn and weight falls back to the baseline formula. In both cases, if $\|w_{t_{\text{end}}+1} - w_{t_{\text{end}}}\|_1 > \delta$, affected records are marked stale and re-queued.

A.11 Dual Acceptance Gate Details

Let $\bar{v}(a) = \frac{1}{|\mathcal{V}|} \sum_{\tau \in \mathcal{V}} v(a \mid \tau)$ denote the verifiable pass rate and $L_{\bar{w}_t}(a) = \frac{1}{|\mathcal{V}|} \sum_{\tau \in \mathcal{V}} \sum_{n \in \mathcal{C}_{\text{ev}}} \bar{w}_t(n) \cdot \tilde{u}_{\xi_n}(a \mid \tau)$ the learned component under EMA mixture $\bar{w}_t$ (smoothing $\rho_w = \frac{2}{W_{\text{gate}}+1}$). The admission decision after ν_{gate} evaluations:

$$\text{Accept}(a_{\text{child}}) = \underbrace{\big[\bar{v}(a_{\text{child}}) \ge \bar{v}(a_{\text{parent}}) - \varepsilon_{\text{gate}}\big]}_{\text{Hard verifiable floor}} \ \wedge \ \underbrace{\big[L_{\bar{w}_t}(a_{\text{child}}) \ge L_{\bar{w}_t}(a_{\text{parent}})\big]}_{\text{Soft Pareto comparison}}. \tag{19}$$

Rejected candidates are pruned; their evaluation records are subtracted from ancestor clade accumulators. Acceptance semantics (permanence, immunity to drift) follow Section 3.3.

A.12 Scheduling Details

Workspace and data roles. Each archive node $a \in \mathcal{T}$ represents an evolvable workspace containing a coding-agent scaffold π_a, an evaluator scaffold ξ_a, shared infrastructure code, and a meta-agent that proposes modifications. Lineage clades $\mathcal{C}(a)$ track descendant subtrees induced by expansion edges. A shared workspace allows infrastructure edits, such as sandbox tooling or environment parsing, to benefit both agent execution and evaluation accuracy simultaneously. Separate agent and evaluator trees would duplicate these shared components and split mutation attribution. To ensure

evaluation integrity, training logs guide meta-agent proposals, validation tasks guide search selection, and held-out test suites measure deployment performance.

Action selection. Search actions in $\mathcal{A} = \{\texttt{Expand}, \texttt{Eval}_\pi, \texttt{Eval}_\xi\}$ operate on disparate time horizons. Expansion modifications show their effect on pass rates after several downstream evaluation steps, rendering direct reinforcement credit assignment noisy and ungrounded. GE addresses this mismatch through value-of-information (VoI) dispatch.

The UCB-Air gate (Wang et al., 2008; 2026) $N_t \geq |\mathcal{T}|^\rho$ regulates expansion frequency purely through archive visit counts, isolating topological growth from delayed reward estimation. When this gate is closed, the scheduler allocates budget between agent and evaluator evaluation by comparing expected posterior-variance reduction per consumed token:

$$\mathrm{VoI}_\pi(a,\tau) = \frac{\omega(a)\cdot\Delta\mathrm{Var}_\pi(a)}{\bar{c}_\pi(a,\tau)}, \qquad \mathrm{VoI}_\xi(n) = \frac{\Delta\mathrm{Var}_\xi(n)}{\bar{c}_\xi(n)}. \tag{20}$$

Here $\Delta\mathrm{Var}$ is the closed-form variance reduction for conjugate Beta-Bernoulli posteriors (Appendix A.7), normalized by running token costs $\bar{c}$. The ranking weight $\omega(a) = 1 - I_{u_\alpha(a^{(2)})}(S_a, F_a)$ concentrates evaluation budget on candidate agents that have a non-trivial probability of overtaking the current runner-up $a^{(2)}$.

Parent cooldown and pending-evidence priority. CMP-guided Thompson sampling, inherited from HGM and RQGM unchanged, is not by itself safe against a failure mode that appears once `Expand` is allowed to fail without producing a surviving child. A node whose self-improvement attempts repeatedly error out, or whose children repeatedly fail their first evaluation, never accumulates verifiable evidence of its own, so its posterior over $\widehat{\text{Co-CMP}}_\pi$ stays close to an uninformative $\mathrm{Beta}(1,1)$. An uninformative posterior has high variance, which lets it draw a spuriously high Thompson sample and win the $\arg\max$ against nodes whose posteriors have already tightened around a lower but well-evidenced value, allowing the sampler to return to the same unproductive parent indefinitely and consume search budget on repeated failure rather than exploration.

The fix keeps the sampler itself untouched and instead gates its candidate pool. Every `Expand` attempt at node a whose self-improvement errors out or whose child fails its first evaluation increments a failure counter $f(a)$. Once $f(a) \geq \tau_{\mathrm{cool}}$, node a enters the cooldown pool $\mathcal{P}_{\mathrm{cool}}$ and is excluded from Thompson parent sampling, unless excluding it would leave no eligible parent at all, in which case the exclusion is lifted for that one step, so a single surviving lineage is never starved of the chance to expand. This is a deliberately conservative rule, biased toward giving up on a node that may simply have been unlucky rather than genuinely unproductive, in exchange for a hard bound on how much budget any one node can consume through repeated failure.

A closely related interaction arises once the acceptance gate requires more than one unit of evidence ($\nu_{\mathrm{gate}} > 1$): a pending child that has not yet accumulated ν_{gate} real $\texttt{Eval}_\pi$ evaluations can otherwise be perpetually outranked by the value-of-information index once exploration pressure favors `Expand` elsewhere, never reaching the evidence threshold its own gate requires. GE addresses this by giving pending nodes below their evidence target scheduling priority up to ν_{gate}, self-gating so the mechanism has no effect when $\nu_{\mathrm{gate}} = 1$ and activates exactly where the dual gate of Section 3.3 needs it.

A.13 Calibrating Verifiable and Learned Signal

Anchor fidelity for agents in the same clade is computed on lineage-disjoint anchor subsets $\mathcal{D}_{\mathrm{ext}}^{\mathrm{disjoint}}(a)$, which keeps an evaluator from favoring its own siblings.

Agent utility combines the verifiable outcome $v(a \mid \tau) \in \{0,1\}$ with the evaluator mixture w,

$$U(a) = \frac{1}{|\mathcal{V}|}\sum_{\tau\in\mathcal{V}}\Big[\beta\, v(a\mid\tau) + (1-\beta)\sum_{n\in\mathcal{C}_{\mathrm{ev}}} w(n)\,\tilde{u}_{\xi_n}(a\mid\tau)\Big], \tag{21}$$

where $\tilde{u}_{\xi_n}(a \mid \tau)$ is evaluator n's score mapped to a probability as described below.

A fixed β implicitly assumes that pass-rate gains and review-score gains are commensurable; this assumption is unlikely to hold across disparate signal types. GE instead maps each evaluator's raw score onto a common probability scale via Platt scaling (Platt, 1999), a per-evaluator logistic

regression $\hat{p} = \sigma(a \cdot \text{raw} + b)$ fit against ground-truth anchor labels rather than against the raw score's own distribution, so that a raw score is placed relative to where the anchor set's true decision boundary actually lies rather than merely centered on its own sample mean. Because both signals then live on a common probability scale, a single balance $\beta = 0.5$ is used throughout.

A.14 Experimental Details

Evaluation surfaces. Table 7 lists every benchmark with its ground truth, evaluator pool, and split. Every split is made without consulting any evaluator score, and Polyglot's study tasks are divided into anchor and evaluation halves by balancing patch counts. Polyglot tasks execute in a sandboxed container runtime in which the reference solutions pass on 210 of 225 tasks and never enter the agent's file system. IMO-GradingBench is scored twice, once by graders that see the reference solution and once by graders that do not.

Table 7: **Evaluation surfaces.** Polyglot's 34 study tasks carry 635 real agent patches. Closed-loop search runs on the evaluation half, and no agent runs on the anchor half during search.

Benchmark	Items	Ground truth	Evaluators	Anchor / evaluation split	Role
Polyglot (6 languages)	225 tasks	hidden unit tests	evolved rubrics	17 / 17 tasks (317 / 318 patches)	offline, closed loop
Polyglot held-out	17 + 60 tasks	hidden unit tests	–	anchor half; 60 drawn from 177 unused tasks	held-out, transfer
PeerRead	427 papers	accept or reject	6 personas	214 / 213 papers	offline, writer loop, intervention
IMO-GradingBench	300 proofs, 30 problems	expert points	6 graders	15 / 15 problems	offline, writer loop, intervention
SummEval	800 summaries, 100 articles	human overall rating	6 personas	50 / 50 articles	offline, writer loop, intervention
Change My View	689 replies, 150 posts	delta awarded	6 personas	75 / 75 posts	offline, writer loop, intervention
SWE-bench Verified	500 issues	hidden unit tests	evolved rubrics	–	comparison with HGM

Full method comparison. Table 8 compares RQGM and GE component by component, with the parameters and anchor sources used in each setting.

Table 8: **Full comparison of RQGM and GE.** Both share the initial agent, the self-improvement operator, value-of-information scheduling with Thompson parent sampling, the dual acceptance gate with a verifiable floor on three paired tasks ($\nu_{\text{gate}} = 3$), and the search budget. Parameter values are those of closed-loop search, and offline evaluator updating tunes them on a frozen grid (Table 6).

Component	RQGM	**GE**
Evaluator signal per output	one incumbent evaluator	top-k evaluators under a soft mixture
When the signal changes	power-of-two checkpoints ($h = 4$, $r = 2$)	continuously
Evaluator score	ϵ-best-belief on anchors ($\epsilon = 0.05$, at least 5 outcomes)	anchor fidelity + Co-CMP productivity, λ_t from 0.9 to 0.3
Update rule	argmax, strict improvement, ties keep the incumbent	softmax with entropy floor, weight floor, significance-gated jumps
Evaluator input	task statement, patch	task statement, patch, task test files (never outcomes)
Stale evidence	selective erasure, lazy re-scoring	lazy invalidation, re-queueing
Interventions	none	probationary candidate every $\kappa = 50$ steps, significance test
Anchor in closed-loop search	CRAVE run or in-domain	CRAVE
Anchor in offline evaluator updating	CRAVE or in-domain	in-domain

Evaluator study on Polyglot. Every evaluator configuration in the study scores all 635 patches once per evaluator model. It contains the eight evolved rubrics produced by an earlier GE search on SWE-bench Verified, transferred verbatim, and the same eight rubrics with task-specification context, which form GE's pool. A context ladder adds test files, repository source, and a reasoning step to the seed rubric, and a strict and a lenient persona serve as bias controls. Anchor fidelity is computed for each configuration on the CRAVE corpus and on the anchor half, and in-loop AUC on the evaluation half. For the second evaluator model, every judgment is re-scored by `deepseek-v4.1-flash` with thinking disabled under identical prompts, and the offline comparison reuses the configurations tuned on Qwen judgments.

Protocols. Table 9 lists every study with its protocol. Offline tuning draws 60 configurations per method from a frozen grid and evaluates each on 150 tuning seeds, with budget 800 steps. In offline evaluator updating, the productivity term of Eq. 3 is each evaluator's agreement with test outcomes on the pairs it has already scored. On Polyglot, offline comparison covers each method's own evolved pool, a shared pool in which RQGM receives GE's evaluators, and all judgments re-scored by `deepseek-v4.1-flash`. In closed-loop search the three Polyglot runs start from a byte-identical initial agent with the same 17-task initial evaluation. Each run starts from a single seed evaluator whose pool grows with accepted nodes, and each is reported at the budget it reached. In the non-code offline comparison, both methods share the six personas and the anchor half. In the writer loops the production gate and mixture classes are imported unchanged, RQGM loops start from GE's root drafts, and every draft of both methods is also scored by independent judges that belong to neither pool, three personas per domain and, on IMO-GradingBench, six graders that see

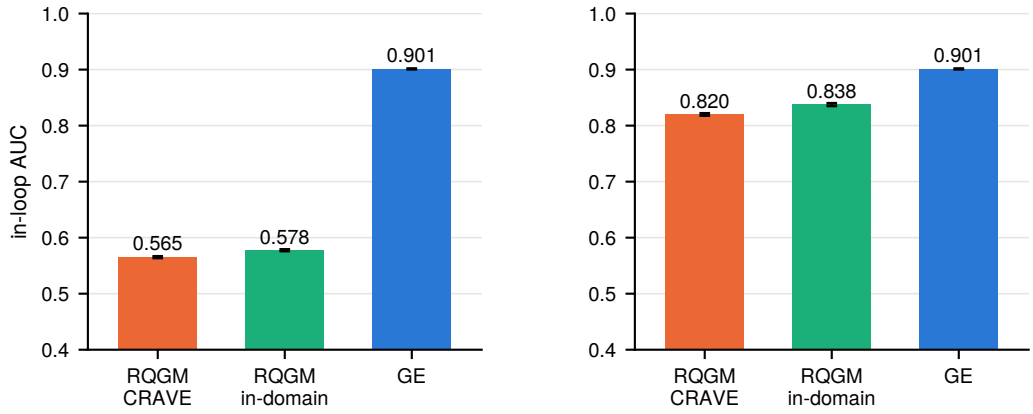
the reference solution. In each domain one probationary evaluator is injected under the production probation weight and end-of-window significance test, with the window shortened to three or four rounds. For transfer, the best-belief agent of each run is evaluated once per task on the 17 held-out tasks and the 60-task list under each of the three backbones.

Table 9: **Studies, the questions they answer, and their protocols.** Grids, seeds, splits, and decision rules of every offline and transfer study were registered before results were seen. Offline numbers use evaluation seeds shared across methods, so all differences are paired.

Study	RQ	Benchmark	Runs differ in	Protocol	Metric	Reported in
Offline, own pools	1	Polyglot	update rule, evaluator input	budget 800, 1,000 seeds	in-loop AUC	Table 1
Offline, GE's pool for all	1	Polyglot	update rule	budget 800, 1,000 seeds	in-loop AUC	Table 1
Offline, second evaluator model	1, 3	Polyglot (DeepSeek)	update rule	tuned configurations reused, 1,000 seeds	in-loop AUC	Table 1
Late-evaluator adoption	1	Polyglot	update rule	best evaluator released at 15/30/50%, 300 seeds	adoption rate	Section 6.1
Closed-loop search	1, 2	Polyglot	evaluator signal and its update	byte-identical initial agent	gate decisions, search productivity	Table 1
Anchor and input study	2	Polyglot, 2 models	anchor source, evaluator input	one scoring pass per model	rank correlation, in-loop AUC	Section 6.1
Offline, non-code domains	3	4 domains, 5 variants	update rule	budget 800, 1,000 seeds	pairwise agreement	Table 2
Writer loops	3	4 domains	update rule	5 topics $\times$ 10 rounds, shared root drafts	acceptances, vetoes by judges	Table 2
Interventions	2	4 domains	–	one probation window per domain	verdict against anchor evidence	Section 6.1
Held-out and transfer	3	17 + 60 tasks, 3 models	selected agent	one run per agent and task	pass rate	Tables 1, 3

A.15 Additional Results

Evaluator signal in the offline comparison. Figure 2 shows the offline comparison of Table 1. At the end of search, RQGM either spreads its incumbent across the pool, with no evaluator chosen in more than 28.2% of seeds under in-domain anchors, or settles on a weak one, with 62.4% of seeds ending on an evaluator of in-loop AUC 0.498 under the CRAVE anchor. Even with GE's evaluators, the CRAVE-anchored incumbent ends on an evaluator of in-loop AUC 0.793, against 0.900 for the pool's best, in 32.4% of seeds.



(a) Each method's own evolved pool. GE is higher by 0.336 with the CRAVE anchor and by 0.324 with the in-domain anchor.

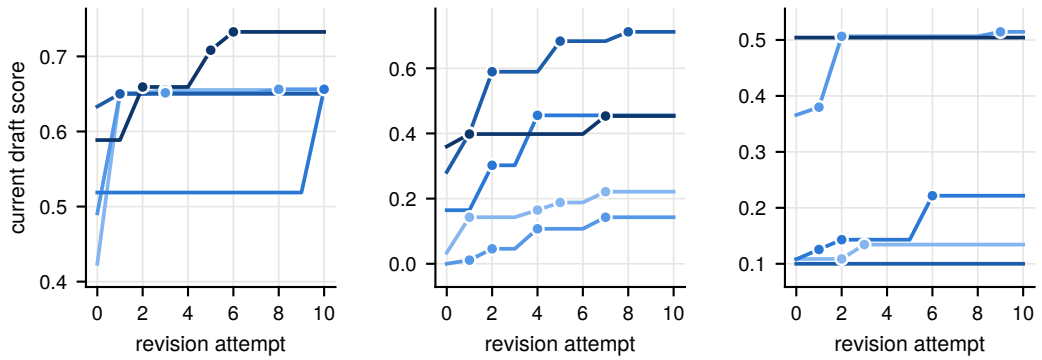
(b) GE's eight evaluators for all methods. GE is higher by 0.081 with the CRAVE anchor and by 0.064 with the in-domain anchor.

Figure 2: **Evaluator signal in offline evaluator updating.** Bars give the in-loop AUC of each method's evaluator signal over 1,000 paired seeds with a budget of 800 steps, and error bars give 95% confidence intervals. RQGM runs with the CRAVE or the in-domain anchor. In both settings, GE's continuous soft mixture is higher than RQGM's one-hot incumbent.

Writer lineages. Figure 3 shows the fifteen writer lineages behind Table 2B. The domain means rise from 0.531 to 0.670 on PeerRead, 0.168 to 0.397 on IMO-GradingBench, and 0.237 to 0.295 on Change My View, with acceptance rates of 20%, 32%, and 16%, and every gate verdict agrees in direction with the evaluator pool's scores.

Mechanism studies. Figures 4, 5, and 6 show the three studies of Table 10. In the context ladder of Figure 5, the seed rubric reaches 0.534 on its own, 0.536 with repository source, 0.822 with test files, and 0.877 with test files, source, and a reasoning step, and in the rubric comparison of Figure 5a, on all 318 evaluation-half patches, the simplest seed rubric becomes the strongest evaluator once it reads the test files (0.900).

Probationary interventions. Table 11 lists the four interventions. The four cases produced four different patterns of probation behavior, and in every case the verdict followed the anchor evi-



(a) PeerRead, abstract writing. The gate accepts 10 of 50 revisions.

(b) IMO-GradingBench, proof writing. The gate accepts 16 of 50 revisions.

(c) Change My View, persuasive reply. The gate accepts 8 of 50 revisions.

Figure 3: **Writer lineages under the production gate.** For each of five topics per domain, a writer starts from a first draft (attempt 0) and proposes ten revisions. GE's gate compares each revision with the current draft and either accepts it as the new current draft (marker) or rejects it, so a line rises at an accepted revision and stays flat after a rejection. The vertical axis is the score that GE's evaluator pool gives the current draft, and Table 2B checks the same loops with independent judges. Thirteen of the fifteen lineages accept at least one revision. SummEval is omitted because its first drafts already reach the top of the scale.

Table 10: **Mechanism studies on Polyglot** for both evaluator models. In-loop AUC is measured on the evaluation half. The first block covers the 14 evaluators of the anchor study. The second block scores the eight evolved rubrics as written and with the task's test files (test outcomes are never shown). The third block withholds the evaluator with the highest in-domain anchor accuracy until the given point of the budget and reports the fraction of 300 seeds in which it later becomes RQGM's incumbent or GE's top-weighted evaluator. ↑ marks metrics where higher is better.

	Qwen evaluators	DeepSeek evaluators
Which evaluator does each anchor rank first? (in-loop AUC ↑)		
Ranked first by the CRAVE anchor	0.498	0.609
Ranked first by the in-domain anchor	**0.877**	**0.960**
Best of the 14 evaluators	0.877	0.960
What the evaluator sees (eight evolved rubrics, in-loop AUC ↑)		
Task statement and patch	0.53–0.60	0.47–0.73
+ task test files	**0.78–0.90**	**0.70–0.91**
Rubrics that improve	8 / 8	8 / 8
Adoption of a late stronger evaluator ↑ (Qwen; released at 15 / 30 / 50% of budget)		
GE	**98.7 / 97.3 / 93.3%**	
RQGM, own pool	39.3 / 29.3 / 18.0%	
RQGM, GE's pool	38.0 / 36.3 / 29.7%	

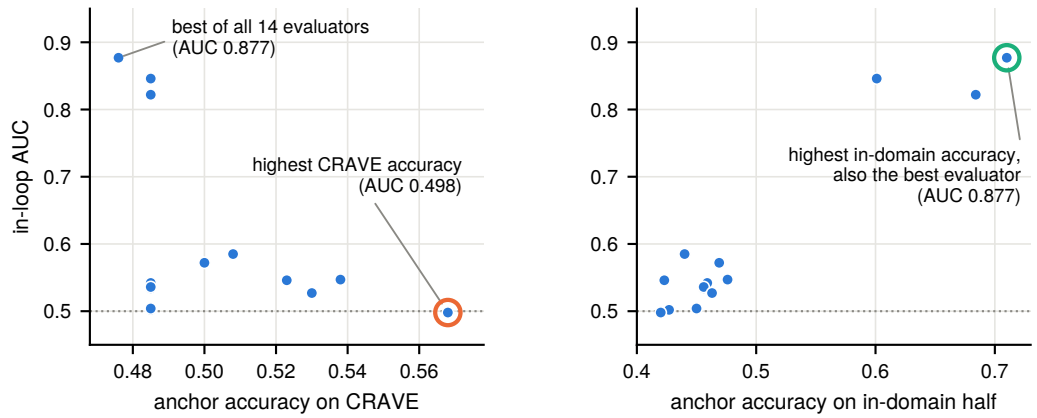
dence. A candidate that looked helpful was withdrawn, one that looked unremarkable was promoted, one that looked harmful was withdrawn, and an exact tie kept the incumbent under the strict-improvement rule.

Table 11: **Probationary interventions in four domains.** The probation weight and the end-of-window significance test run the production code; BB is the anchor posterior lower bound.

Domain	During probation	BB_{cand} vs. BB_{inc}	Verdict
PeerRead	3 of 3 revisions accepted	$0.305 \leq 0.607$	withdrawn
IMO-GradingBench	1 of 4 accepted	$0.600 > 0.580$	promoted
SummEval	0 of 4 accepted	$0.643 \leq 0.719$	withdrawn
CMV	0 of 4 accepted	$0.421 = 0.421$	incumbent kept

A.16 SWE-bench Verified: Comparison with HGM

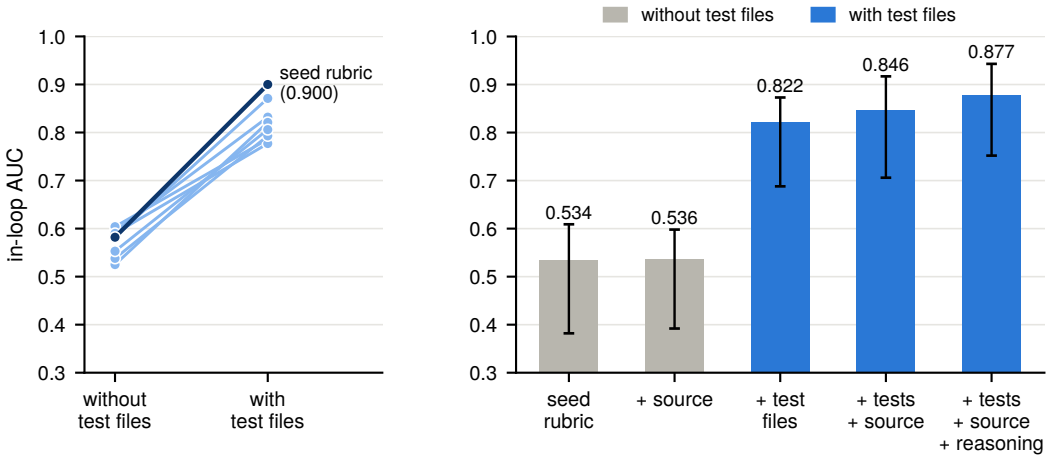
On SWE-bench Verified we compared an earlier GE configuration, whose evaluator prompt carried the verifiable test summary and no test files, with HGM's official implementation (Wang et al.,



(a) CRAVE anchor from human pull-request verdicts. Spearman correlation -0.31, and seven evaluators tie at 0.485.

(b) In-domain anchor from held-out Polyglot patches. Spearman correlation $+0.73$.

Figure 4: **Anchor agreement versus usefulness in search.** Each dot is one of the 14 evaluators of the anchor study, scored by `qwen3.6-35b-a3b` and placed by its accuracy on an anchor set (horizontal) and its in-loop AUC on the evaluation half of the 635 Polyglot patches (vertical). The circle marks the evaluator each anchor ranks first. CRAVE's first choice is near chance at 0.498, while the in-domain anchor's first choice is the best evaluator in the study at 0.877. The dotted line marks chance (AUC 0.5).



(a) Eight evolved rubrics without and with the task's test files, on the 318 evaluation-half patches. The rank correlation between the two orderings is -0.19.

(b) The seed rubric given more context step by step, on the 273 evaluation-half patches whose repository source could be rebuilt.

Figure 5: **Task-specification context.** In-loop AUC of evaluators scored by `qwen3.6-35b-a3b` with different context, where test outcomes are never shown. Every evolved rubric improves with the test files, from 0.53–0.60 to 0.78–0.90, and the seed rubric, the simplest of them, becomes the strongest at 0.900 (a). Repository source alone leaves the seed rubric near chance, and the test files raise it to 0.82–0.88 (b). The two panels come from separate runs on different patch sets, so the seed rubric's values differ between them. Error bars in (b) are 95% bootstrap intervals over tasks.

2026). Both started from the same initial agent (all 43 fingerprinted source files identical) and used `qwen3.6-35b-a3b` with identical per-task limits (120 model calls, 12,000 context tokens, 1,800 s test timeout).

Search outcome. HGM completed $B = 100$ with 16 children and, by its own selection rule, returned the initial agent (posterior quantile 0.446 against 0.397 for its best child). GE accepted node 22 at $B = 79$ (verifiable $+0.333$, learned $+0.093$ over its parent), kept it as its best belief

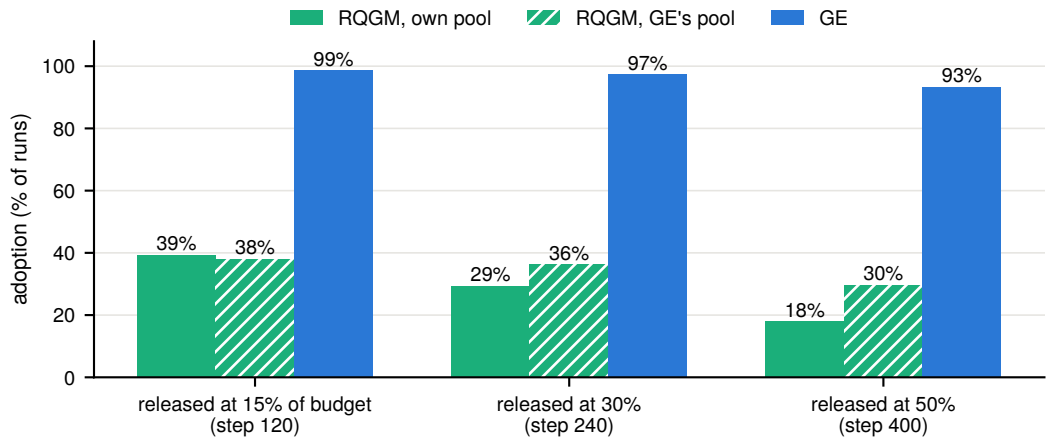


Figure 6: **Adoption of a late stronger evaluator.** In each pool of `qwen3.6-35b-a3b` judgments, the evaluator with the highest in-domain anchor accuracy is withheld and released at 15%, 30%, or 50% of an 800-step budget. Bars give the percentage of 300 runs in which it later becomes RQGM's incumbent or GE's top-weighted evaluator. GE adopts it in 93–99% of runs at every release point, while RQGM adopts it less often the later it appears, because its power-of-two checkpoints grow sparse.

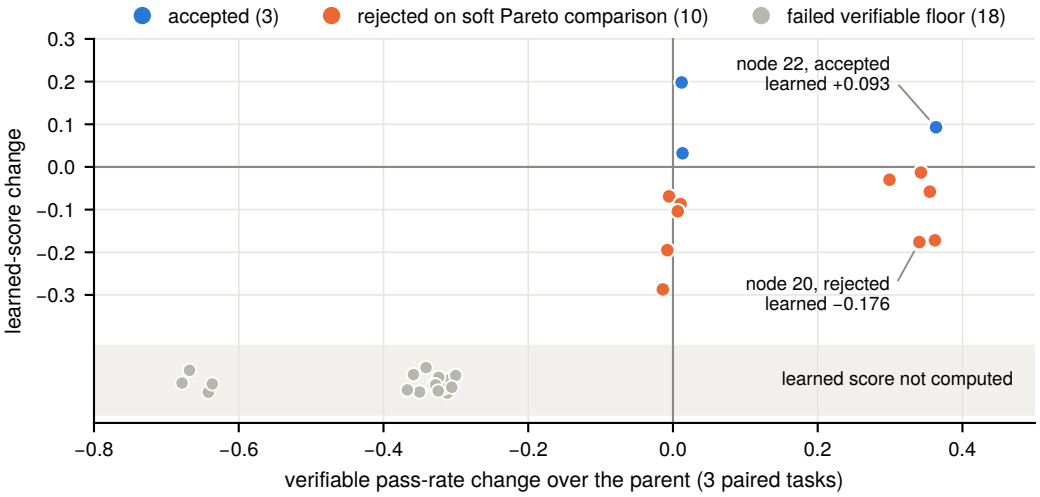


Figure 7: **Dual-gate decisions on SWE-bench Verified** (earlier GE configuration, Appendix A.16). Each dot is one of 31 candidate children, placed by its change over its parent in verifiable pass rate on three paired tasks (horizontal, multiples of one third, jittered for visibility) and in learned score (vertical). The 18 candidates whose verifiable pass rate dropped fail the floor and get no learned score, so they are drawn in the shaded band. For the other 13 the learned score decides, and the gate accepts exactly those that it rates at least as high as the parent. Node 22 and its sibling node 20 gain the same $+0.333$ in verifiable pass rate, and the learned score separates them ($+0.093$ against -0.176).

through $B = 148$ and $B = 180$, and retained an eight-node archive. The dual gate discriminates between candidates that tests cannot separate. Node 22 and a sibling with the same parent both passed all three paired tasks, and the sibling was rejected on learned utility (-0.176). Over 31 auditable gate decisions, GE accepted 3, rejected 18 at the verifiable floor, and rejected 10 on the soft Pareto comparison (Figure 7).

A.17 Writer Loops: Final-Draft Judge Scores

For each topic, the final accepted drafts of the two methods were scored by independent judges and compared in pairs (five topics per domain). GE is ahead on PeerRead, where the 95% interval excludes zero, and on IMO-GradingBench under graders that see the reference solution; the remaining domains are close to even, and SummEval drafts sit at the top of the scale for both methods.

Table 12: **Final-draft judge scores in the writer loops**, paired by topic (five topics per domain).

Domain (judges)	GE − RQGM [95% CI]	GE higher / lower / tied
PeerRead (three judges)	$+0.090$ $[+0.003, +0.173]$	3 / 1 / 1
IMO-GradingBench (six reference-solution graders)	$+0.014$ $[-0.043, +0.071]$	3 / 2 / 0
IMO-GradingBench (three judges)	-0.186 $[-0.452, +0.029]$	1 / 3 / 1
SummEval (three judges)	$+0.013$ $[0.000, +0.040]$	1 / 0 / 4
CMV (three judges)	-0.010 $[-0.110, +0.087]$	1 / 2 / 2
Four domains pooled (three judges)	-0.023 $[-0.115, +0.048]$	6 / 6 / 8

A.18 Extended Discussion

Computational profile. Scoring each output with the top-k evaluators costs more model calls per evaluation than consulting one incumbent. In closed-loop search, lazy invalidation defers the refresh of stale scores until the scheduler revisits a node. Scaling to much larger evaluator pools would benefit from lightweight surrogate evaluators or distillation of the mixture.

Mutation scope. Current meta-agent proposals modify single-file prompts and execution scripts within individual workspaces. These localized edits do not address repository-wide harness refactoring, which requires attributing failures across task distributions (Section A.3.3).

Scope of the evidence. Search used one backbone model and one Polyglot configuration, with a verifiable floor on three paired tasks; held-out evaluation uses 17 and 60 tasks with one run per agent and task. Offline evaluator updating fixes the evaluator pool to real judgments and thereby isolates the update rule, while the closed-loop runs show the same rule acting on a pool that grows with the archive.