DIVE: MAKING JUDGMENT A FIRST-CLASS CAPABILITY IN LONG-HORIZON AGENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Agents working on long tasks must decide when to continue, check their work, change direction, or stop. A final task score gives little information about the value of these decisions. We introduce DIVE, which encodes judgment procedures as reusable Cognitive Judgment Skills. Each Skill specifies when prior experience applies, what evidence a decision requires, and when to reconsider it. A Detect–Invoke–Verify loop connects these procedures to the host agent without replacing its action generator. We evaluate DIVE in software repair and scientific program search, using task outcomes and continuations from the same state. Guarded stopping reduces aggregate Target tokens by 72.52% across 16 matched search pairs. Every pair passes its task-specific final-quality check, though five have lower anytime quality. In the MiniMax and DeepSeek repair summaries, Jev uses fewer reported tokens than Native or DIVE with Target-based judgment. On 30 SWE-bench Pro tasks, outer-loop DIVE with Codex SDK and GPT-5.6-Luna resolves 27 tasks versus 24 for Native, using 19.41% fewer input-plus-output tokens. The decision-level results are less consistent. Reuse has positive mean local effects in four cohorts, but selection gains and five-step outcomes vary. DIVE provides a way to measure these decisions and their costs; the observed task-level benefits do not establish a general advantage for its selection policy.

Anonymous code: `https://anonymous.4open.science/r/paper-review-2027-083D/`

1 INTRODUCTION

An agent can pass repository tests and still leave the reported bug unresolved. In program search, an unchanged score may justify another experiment, a different strategy, or an early stop. The agent has to choose among these actions using the evidence collected so far and the budget it has left. That choice also determines what evidence becomes available next.

ReAct, Reflexion, and metacognitive learning connect reasoning and feedback to agent action (Yao et al., 2023; Shinn et al., 2023; Liu & Van Der Schaar, 2025). Their complete trajectories combine self-assessment with action generation, tool execution, and recovery. Task success alone cannot tell us which judgment helped. Similarly, a shorter trajectory may result from avoiding unnecessary work or from stopping too soon.

Figure 1 shows three examples from our archived systems, including DIVE. In one repair, the agent proposed a check but submitted an unsuccessful patch without running it. In program search, selected guidance was executed at all six tested forks and beat the opposite direction at only two. In a separate reuse study, applying the same legal action at every state nearly reached the observed two-action hindsight bound; the selection policy achieved less. To understand such outcomes, we need both the evidence behind a decision and a comparison with alternatives that could have been executed from the same state.

We use *judgment* to mean choosing, maintaining, or revising an action direction based on evidence, experience, and resource constraints. Treating judgment as a *first-class capability* gives it an explicit representation and an interface through which its consequences can be evaluated. Our question is whether this separation helps an agent choose useful actions at an acceptable computational cost.

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.

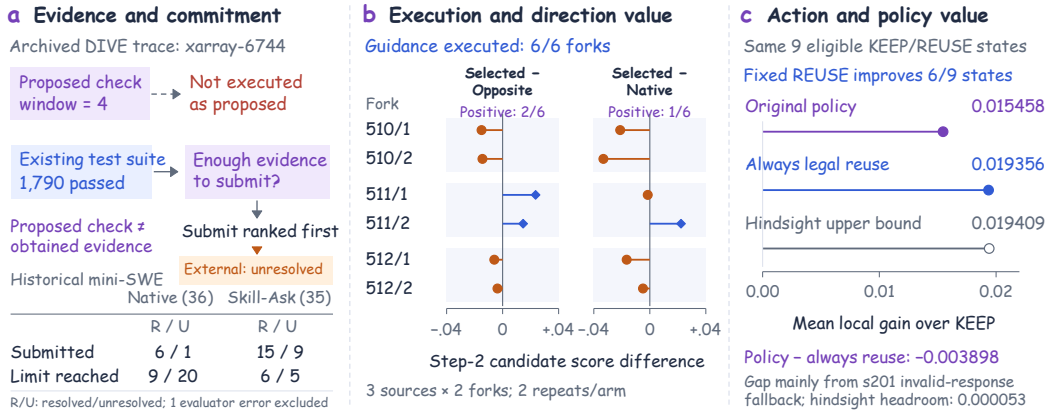


Figure 1: **Why judgment needs its own evaluation.** (a) An archived repair separates a proposed check, obtained evidence, submission commitment, and external resolution; the adjacent termination counts are cohort-level context, not matched termination outcomes. (b) Executed directions need not outperform alternatives: points compare two-repeat means of second-step candidate scores at six forks from three source runs. (c) Useful reuse does not establish policy value; the deficit mainly reflects an invalid-response fallback, with little hindsight headroom. The panels use distinct cohorts and do not form a causal chain or estimate judgment accuracy. Protocols and interpretation: Appendix E.1.

DIVE implements this approach through Cognitive Judgment Skills. A Skill states when experience applies, what supports its use, and what evidence would warrant revision. Detect identifies a decision point, Invoke applies the Skill through a Judge, and Verify records execution receipts and feedback. The original agent generates operations, while the Host controls execution. The optional Evolve component revises Skills; we keep runtime policies fixed within each evaluated version.

We study task performance and resource use in software repair, then examine individual decisions in program search. Jev lowers reported token usage relative to Native and Target-based judgment in the MiniMax and DeepSeek summaries. In a separate 30-task Codex SDK cohort, DIVE resolves 27 tasks compared with 24 for Native and uses 19.41% fewer input-plus-output tokens. Guarded stopping saves 72.52% of aggregate Target tokens across 16 matched search pairs, all of which pass their final-quality checks. Direction and reuse studies help interpret these gains: an action can be useful without the policy selecting it improving on a simpler alternative.

Our contributions are:

- DIVE, an execution interface that connects reusable Cognitive Judgment Skills to host-supported interventions and feedback, with model- or rule-based implementations of judgment.
- Task-level and same-state studies that quantify execution savings, judgment overhead, and selection losses, including an analysis of the limited headroom above fixed reuse in the observed states.

2 Related Work

Reflection and reusable experience. ReAct interleaves reasoning with information-gathering actions, while Reflexion stores feedback-based reflections for later attempts (Yao et al., 2023; Shinn et al., 2023). Voyager stores executable skills, and ExpeL extracts natural-language knowledge from task experience for later reuse (Wang et al., 2024; Zhao et al., 2024). SkillsBench uses paired no-Skill and curated-Skill conditions, whereas SRA separates skill retrieval, incorporation, and end-task execution (Li et al., 2026; Su et al., 2026). DIVE instead specifies applicability and evidence conditions and exposes each intervention for comparison with alternatives; relative effectiveness still requires matched controls.

Metacognition and self-improvement. Metacognitive learning explicitly includes self-assessment, planning, and evaluation (Liu & Van Der Schaar, 2025). Metacognitive Consolidation studies the accumulation of metacognitive experience (Zhuang et al., 2026). Rational metareasoning selects when extra reasoning is worth its cost, TTPI controls when and how to intervene, and CDS selects interventions from estimated residual demand (De Sabbata et al., 2024; Yang et al., 2026; Scoville et al., 2026). ART separately decides whether to refine and whether to retain the initial or revised answer (Shridhar et al., 2024). SICA and DGM improve agent implementations through modification and evaluation (Robeyns et al., 2025; Zhang et al., 2026). SICA also includes an overseer that monitors execution, so an additional supervisory loop alone is not our claimed distinction. DIVE focuses on the evidence conditions of individual action judgments; its fixed-skill comparisons do not evaluate cross-version self-improvement.

Computation, verification, and stopping. Self-correction depends on the evidence available to the model. In the reasoning tasks studied by Huang et al. (2024), intrinsic self-correction without external feedback did not reliably improve performance and sometimes degraded it. CRITIC uses external tools, while ProCo verifies a masked key condition (Gou et al., 2024; Wu et al., 2024). Compute-optimal test-time scaling allocates inference effort by problem difficulty and compares verifier-guided search with response revision (Snell et al., 2025). DIVE accounts for judgment cost during an agent trajectory. By recording the evidence, procedure, evaluator, and Host action, it supports separate tests of their contributions. Our stopping experiment evaluates a guarded threshold against continued search; it does not test an advantage over an equivalent rule.

Scientific program search and repair evaluation. AlphaEvolve and OpenEvolve combine program generation with executable evaluation (Novikov et al., 2025; OpenEvolve Contributors, 2026). Their final objective values evaluate a search system, but do not identify the value of an individual intermediate decision. DIVERT instead resumes saved agent–environment states at critical junctions and branches alternative continuations for evaluation (Nakash et al., 2026). DIVE preserves the host's generation and evaluation process and uses matched states to test bounded interventions at host decision points. SWE-bench and SWE-bench Pro provide repository-level repair tasks with external tests (Jimenez et al., 2024; SWE-bench Pro Contributors, 2026). Our historical subsets are analyzed within their own model and logging configurations, not presented as complete benchmark leaderboard results.

3 Judgment as a First-Class Capability

At a decision point t, the Host exposes a state $s_t = (h_t, e_t, b_t)$: available history, evidence, and remaining budget. Given fixed skills $\mathcal{K}$ and host-legal directions $\mathcal{A}_t$, the judgment function returns

$$d_t = J(s_t, \mathcal{K}, \mathcal{A}_t), \qquad d_t \in \mathcal{A}_t \cup \{\perp\}. \tag{1}$$

Here $\perp$ declines intervention; stopping is instead an explicit Host action. The Target implements the chosen direction through concrete operations, or continues its native policy when no intervention is admitted. This is a functional separation: the same model can generate and judge, and a bounded rule can implement J. The interface exposes only information and controls available in the particular Host.

Task-level comparisons measure quality and resources; same-state comparisons measure the consequences of legal alternatives. We keep the skill's criteria, the implementation that applies them, and the selected action's consequences separately identifiable.

4 DIVE

DIVE records why an intervention was chosen, which Host action followed, and what feedback the agent received (Figure 2). The adapter can use Target self-judgment, a bounded rule, or an eligible Jev call. Supervisor routing is optional. The Host applies guidance or a supported stop; otherwise the agent continues natively. Offline comparisons from the same state test the value of the chosen intervention.

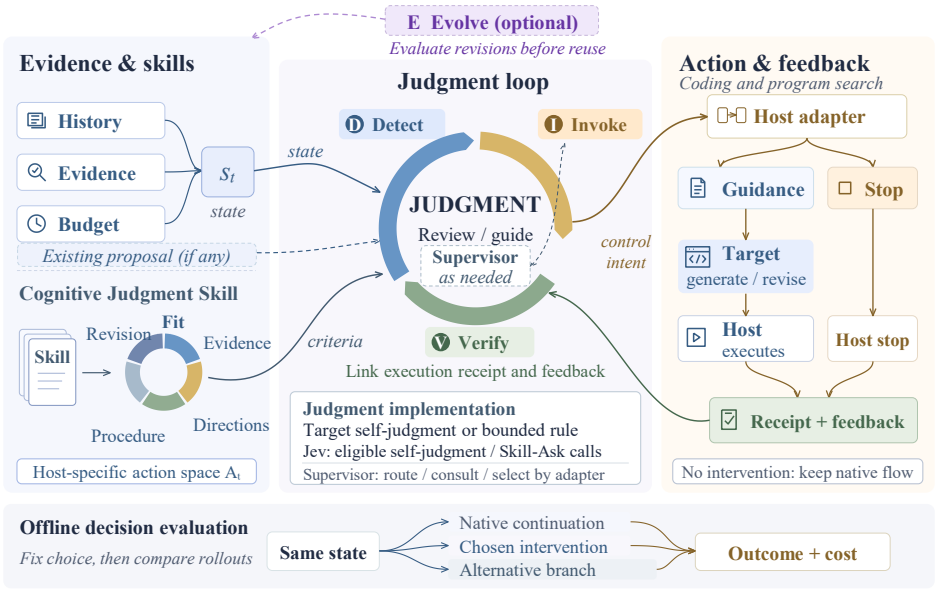


Figure 2: DIVE makes judgment explicit while retaining action generation. Detect exposes a decision state; Invoke applies Skill criteria through Target self-judgment, a bounded rule, an eligible Jev call, or an adapter-routed Supervisor. The Host adapter realizes guidance or a supported stop; without intervention, native flow continues. Verify links execution receipts and feedback, not a guarantee of correct judgment. Bottom: offline same-state evaluation fixes the choice before comparing rollout outcomes and costs. Evolve is optional and evaluates revisions before reuse.

4.1 Cognitive Judgment Skills: Reusable Decision Criteria

A Cognitive Judgment Skill has the form $k = (\phi_k, \mathcal{E}_k, \mathcal{A}_k, \pi_k, \rho_k)$, specifying applicability, required evidence, legal directions, a decision procedure, and revision conditions. Markdown Skills express these requirements as model instructions; executable policies encode conditions as predicates. For example, a patch-validation Skill distinguishes a failed import from a test that reaches the edited code and exposes a defect. The first requires repairing the test route; the second requires diagnosis. The adapter determines which requirements are enforced, as the Recognition example below shows.

The repair pack contains seven fixed Skills for issue understanding, localization, diagnosis, repair strategy, validation, recovery, and completion. For search, we implement guarded stopping, direction guidance, and recognition-and-fit for experience reuse. Each is adapted to its Host and has its own evidence status. We report the evaluated policy version for each experiment and give the Skill cards in Appendix C.

4.2 Detect and Invoke: Evidence-Conditioned Judgment

Detect uses the exposed state to identify when a decision needs review, for example when action rankings are ambiguous or a task-owned completion condition is met. Invoke retrieves the relevant Skills and checks whether the available evidence supports a legal direction. It can review a proposal or guide the next one. Missing observations leave evidence requirements unsatisfied. Unsupported outputs trigger the adapter's documented fallback and are recorded separately from valid abstentions.

Algorithm 1: DIVE at a host decision point

Input: exposed state s_t, fixed skills $\mathcal{K}$, legal directions $\mathcal{A}_t$.
1. **Detect:** test the Host-specific reconsideration condition.
2. If not triggered, retain native execution for this decision.
3. **Invoke:** supply applicable skills and inspect required evidence.
4. Commit $d_t = J(s_t, \mathcal{K}, \mathcal{A}_t)$ before observing future outcomes.
5. Admit a legal intent or retain the adapter's documented fallback.
6. Let the Target generate operations and the Host execute the admitted intent.
7. **Verify:** associate execution receipts and subsequent feedback with d_t.
8. **Evolve (optional):** retain episodes for independently evaluated revisions.

Adapters implement Algorithm 1 with different triggers and enforcement: repair uses natural-language criteria, while OpenEvolve also applies executable admission guards.

4.3 Host interfaces and judgment implementations

In mini-swe-agent, the Target proposes a command and ranks ten action types. The archived MiniMax policy routes to a Skill when the top-two score gap is at most 20. This is a heuristic: the scores are uncalibrated and do not estimate the expected value of review. The Target then revises its command and ranking, optionally consulting the Supervisor for diagnostic advice. The executable command remains the Target's choice. Pro uses a different recorded routing configuration (Appendix B).

In OpenEvolve, the adapter binds evidence and legal intents to the current parent and evaluator contract. An admitted intent can stop search or supply bounded guidance for the next candidate; the Target retains program generation.

The Judge can be the Target model itself or, for bounded decisions, a rule. Jev handles eligible self-judgment and Skill-Ask calls while the Target continues to generate actions; some consultations still use MiniMax. We account for each model's tokens separately and distinguish them from request counts and monetary cost.

Recognition example. Recognition v1 considers adding a $(0.25, 0.5, 0.25)$ input filter with edge padding. Code checks parent and evaluator identity, signal validity, and whether the input-filter role is unoccupied. The model receives metrics, prefix observations, three pilot comparisons, and a Skill that asks it to assess mechanism fit and tradeoffs. The parser checks identifiers and output consistency. It cannot verify the model's assessment. A valid selection authorizes one-step guidance. Abstention or invalid output leaves the native continuation in place, and subsequent checks assess action isolation and evaluator feedback. At s198, the model returned `insufficient_evidence`. Offline evaluation gave the unused action a local gain of -0.000033074, but the answer did not identify the facts behind the abstention. Appendix C.1 gives the contract and diagnostic record.

4.4 Verify and Evolve: feedback and revision

Verify records whether the intended action ran, whether the expected local change occurred, and whether the external objective improved. SWE supplies tool outputs, tests, and consultation follow-ups; OpenEvolve can also evaluate a direct successor. Skill-induced changes in rankings occur before execution, so they cannot serve as post-action verification. Final external evaluation remains independent. The reported adapters do not automatically verify and roll back every judgment.

Evolve would use versioned decision episodes to propose Skill revisions and test them independently. In this study, runtime policies stay fixed; human revisions between development versions belong to separate experiments. The optional WikiSkill-based protocol (Tang et al., 2026) would use unseen tasks or score each task before an update (Appendix G). An outer loop that only reviews execution falls within runtime judgment because it leaves the Skills unchanged.

5 Evaluation Protocol

5.1 Questions and comparisons

The evaluation moves from whole-task outcomes to individual decisions. Repair cohorts measure changes in task quality and resource use, including the effect of substituting Jev as the judgment backend. Stopping experiments test a bounded control policy. Direction and recognition experiments then compare an intervention's local effect with the value of selecting it over legal alternatives.

The repair summaries use MiniMax-M2.7 on Verified, normalized to 100 tasks, and DeepSeek-Flash on 50 Pro tasks. An earlier archive retains 33 Verified and 19 Pro pairs for trace diagnostics (Appendix B). These deployments evaluate skill and consultation bundles without a matched-compute ablation. OpenEvolve stopping comparisons preserve task, evaluator, and initialization; direction branches share a parent state and commit the selector output before future outcomes. The stopping intervention is itself a guarded threshold rule. Its comparison with continued search tests that policy, rather than a learned Judge's advantage over the same rule.

Table 1 separately reports DeepSeek Pro50 and Codex Pro30; the latter compares Native with outer-loop DIVE, not Jev backends. Cohorts are not pooled.

Recognition-and-fit reports four cohorts with 16, 8, 8, and 8 matched states. Local comparisons evaluate KEEP, fixed REUSE, and Skill selection; five-step continuations compare Skill with Native, REUSE, and Generic review (Appendix A.1).

5.2 Outcomes, resources, and uncertainty

For repair, we report external resolution, repository actions, and role-specific API requests, together with paired outcomes where task-level labels can be aligned. One action or logical decision may require several requests, so we keep these counts separate from tokens and monetary cost. For search, we report task-specific quality, calls, recorded tokens, and available latency measurements. Provider latency excludes some components of end-to-end time. We distinguish a ratio of aggregate totals from an average of per-pair changes and include Supervisor and failed-request usage when recorded.

Quality bounds use task-specific margins; branch counts are distinct from independent source lineages. Missing endpoints, admission failures, and invalid-response fallbacks are retained separately from semantic abstention. Matched checkpoints do not ensure identical prompts or coupled provider randomness (Appendix E).

6 Software Repair: Task Value and Judgment Traces

6.1 Repair outcomes and resource use

In the Verified summary, DIVE–Target resolves 82 tasks per 100 projected tasks, compared with 76 for Native, while repository actions fall from 4,624 to 3,115 (Table 1; Appendix Figure 3a). Reported MiniMax usage rises from 111.294M to 220.036M tokens. DIVE–Jev resolves 79 per 100 with 3,573 actions and uses 85.987M MiniMax plus 15.377M Jev tokens, totaling 101.364M (Figure 3b). The projection scales the reported summaries; it does not represent 100 observed trials.

Table 1 also reports 50 DeepSeek Pro tasks. Native, DIVE–Target, and DIVE–Jev resolve 19, 20, and 21 tasks. DIVE–Jev uses 278.011M DeepSeek plus 26.096M Jev tokens: 304.107M in total, 14.88% below Native and 55.76% below DIVE–Target. Both backbones therefore show lower reported token volume with Jev, including its usage. Counts from different tokenizers do not measure equivalent computation. Pro monetary costs are omitted because the billing basis is unconfirmed, and aggregate resolution counts do not establish a statistically reliable repair advantage.

With Codex SDK and GPT-5.6-Luna (xhigh), outer-loop DIVE increases resolution from 24/30 to 27/30 (+10 percentage points), reduces repository actions by 20.87%, and lowers input-plus-output tokens by 19.41%. Excluding uncached input gives a 22.28% reduction. Percentages use the

Table 1: Software repair across judgment backends and hosts. Verified is normalized to 100 tasks; Pro cohorts contain 50 (DeepSeek) and 30 (Codex) tasks. Codex token cells show all-input-plus-output / cached-input-plus-output; the latter excludes uncached input. Pro costs are not reported.

			Tokens (M)		Reference cost (cached)	
Method	Resolved / n	Repo actions	Backbone	Jev	Backbone (CNY)	Jev (USD)
SWE-bench Verified / MiniMax-M2.7 — projected to 100 tasks						
Native	76/100	4,624	111.294	—	108.844	—
DIVE–Target	82/100	3,115	220.036 ranking: 145.062	—	215.192 ranking: 141.868	—
DIVE–Jev	**79/100**	**3,573**	**85.987**	**15.377**	**84.094**	**0.648**
SWE-bench Pro / DeepSeek-Flash — 50 tasks						
Native	19/50	4,490	357.261	—	—	—
DIVE–Target	20/50	4,450	687.470 ranking: 293.607	—	—	—
DIVE–Jev	**21/50**	**3,410**	**278.011**	**26.096**	—	—
SWE-bench Pro / Codex SDK / GPT-5.6-Luna (xhigh) — 30 tasks						
Native	24/30	647	43.8 / 41.3	—	—	—
DIVE	**27/30**	**512**	**35.3 / 32.1**	—	—	—

rounded totals in Table 1. The cohort supports outer-loop deployment, without isolating its judgment components.

6.2 What judgment traces reveal

The archived 33 Verified and 19 Pro runs contain 1,425/1,059 rankings and 321/265 Skill injections. Matching records before and after Skill use gives 430/263 comparisons; the top action type changes in 142/122. Some comparisons share a pre-Skill record and are therefore dependent. The logs show what changed, but neither an action change nor an evidence flag establishes that the revised judgment was correct (Appendix E.3).

In the xarray case (Figure 1a), confidence in completion exceeded what the recorded checks supported. The trace does not identify Skill use as the cause of failure. Assessing whether a revision was correct would require same-state alternatives or blinded independent labels for both changed and unchanged decisions.

7 OpenEvolve: Judgment in Scientific Program Search

7.1 Control decisions in program search

In program search, a flat score may indicate diminishing returns, mismatched experience, or competing objectives. DIVE binds interventions to the parent, evaluator, prefix, and remaining budget; the Target generates programs and the Host applies legal controls. Stopping ends search; abstention retains the native policy and incurred overhead. Stopping and direction studies use `deepseek-flash` with zero Supervisor attempts, without Jev or skill updates.

Same-state branches fix the parent and search state, and commit selection before observing continuations. Running alternatives is an evaluation procedure, not a deployment requirement.

Table 2: Guarded stopping: recorded Target computation and task-specific quality. Tokens are cohort totals in thousands; ΔAUC is normalized DIVE minus Native. Final checks allow at most 0.001 loss for FMIN and no loss for K-module.

Task / cohort	Pairs	Target tokens (k) Native → DIVE	Tokens saved	Endpoint checks	Mean ΔAUC
FMIN / main	8	$636.8 \rightarrow 186.0$	**70.79%**	**8/8**	-0.017179
K-module / main	8	$948.6 \rightarrow 249.7$	**73.67%**	**8/8**	-0.001562
FMIN / long budget	3	$1,231.4 \rightarrow 66.6$	**94.59%**	**3/3**	-0.016667
K-module / long budget	3	$812.8 \rightarrow 80.4$	**90.11%**	**3/3**	-0.002083

7.2 Stopping after the task target is met

The frozen `progress_control@0.6.0` family uses the unchanged `oev.progress_control@0.4.0` runtime, with three initial and five fresh seeds per task. All 16 Active arms execute valid stopping receipts and both arms reach their task targets. All eight FMIN pairs meet the frozen 0.001 quality margin; all eight K Module pairs have equal final quality, without establishing zero future violation risk (Appendix D).

Table 2 and Appendix Figure 4 separate the main 16 pairs from six longer-budget pairs. The main cohort uses 87.0% fewer calls and 72.52% fewer recorded Target tokens; longer-budget savings are 94.59% for FMIN and 90.11% for K Module. The policy is a task-score threshold with candidate, infrastructure, and budget guards: this demonstrates search contraction, not superiority to an equivalent stopping rule. Five pairs have lower normalized anytime quality; sampled prefixes prevent attributing these deficits to stopping alone. Longer-horizon sensitivity and latency results remain separate in Appendix D.

7.3 Direction selection

Exploratory and conservative v2 selectors did not establish useful deployment: the former beat Native on one of six forks; the latter intervened on none (Appendix E.2).

The frozen, rule-based v3 selector uses prefix evidence to choose noise- or response-oriented guidance on held-out states. Each of six forks from three source runs receives Native, selected, and opposite continuations, with two repeats of two steps per arm: 36 units and 72 successful Target calls. All selected actions are executed, but they beat the opposite direction on only two forks. Mean second-step candidate differences are -0.000146 relative to opposite and -0.009156 relative to Native; three forks exceed the frozen 0.01 harm tolerance (Figure 1b). The selector therefore fails the promotion criterion. With two repeats and three sources, the study also provides limited evidence about expected branch rankings.

7.4 Reuse and selection

Table 3 and Appendix Figure 5 report the completed Historical, Recognition v2, Replication, and New frozen cohorts ($n = 16, 8, 8, 8$). Mean local REUSE–KEEP gains are positive throughout, ranging from 0.013000 to 0.024295. Skill selection falls below fixed REUSE in Historical (-0.003898) and v2 (-0.007926), but exceeds it in Replication ($+0.016450$) and New frozen ($+0.003204$). The value of selecting reuse therefore depends on the cohort.

Five-step scores take the best value from the shared checkpoint and five valid successor candidates, then average paired differences. Skill–Native differences are $+0.005561$, -0.016392, -0.025710, and $+0.007430$ in cohort order. Replication combines a positive local selection gain with a negative continuation difference; New frozen has positive five-step differences against all three comparators. These means show that local selection and sustained task benefit can diverge. State-level uncertainty for the completed summaries is unavailable.

Table 3: Experience utility and selection on Signal Processing. Entries are mean paired score differences; n is the effective matched-state count. Five-step scores include the shared checkpoint and five valid successors. † Continuation-token accounting coverage remains to be verified (Appendix A.1).

Cohort (n)	Local action REUSE–KEEP	Selection gain Skill–REUSE	Five-step Skill–Native	Five-step Skill–REUSE	Five-step Skill–Generic	Token change Skill / Native
Historical (16)	$+0.019356$	$\mathbf{-0.003898}$	$+0.005561$	-0.009296	$+0.008603$	$+3.34\%$
Recognition v2 (8)	$+0.024295$	$\mathbf{-0.007926}$	-0.016392	$+0.001851$	$+0.000220$	-8.51%
Replication (8)	$+0.015835$	$\mathbf{+0.016450}$	-0.025710	$+0.003527$	$+0.008739$	$-9.70\%^{\dagger}$
New frozen (8)	$+0.013000$	$\mathbf{+0.003204}$	$+0.007430$	$+0.002933$	$+0.003340$	$-13.81\%^{\dagger}$

Recorded continuation tokens change by $+3.34\%$, -8.51%, -9.70%, and -13.81%, respectively. These are ratios of cohort totals, excluding source generation and development. Coverage of Target and Supervisor usage in the last two cohorts remains to be checked against raw records (Appendix A.1).

The archived nine-state diagnostic in Figure 1c offers a concrete explanation for one selection loss. Always-legal-reuse gains 0.019356. Choosing the better of KEEP and REUSE after seeing the outcome yields 0.019409, only 0.000053 more, whereas the original policy gains 0.015458. Here, the same action usually wins: a selector has little room to improve on fixed reuse but can still lose much of its benefit. This hindsight bound describes the archived subset only; it does not apply to the completed cohorts in Table 3.

8 Discussion and Limitations

The results differ across decisions. Stopping saves computation once a task-owned target has been met, whereas the held-out direction selector often chooses a worse continuation. Reuse helps locally on average, but its selection gains have different signs across cohorts and its five-step outcomes do not consistently improve on Native. Evaluations of stronger judges therefore need states in which the available actions offer meaningful tradeoffs. They should also measure the gains a selector loses relative to a fixed legal policy.

The repair comparisons combine Skill content with extra deliberation and deployment choices. An information- and budget-matched generic-review control would be needed to isolate the contribution of the judgment criteria. The Jev and Codex cohorts show what happened under different backend and host deployments; they do not supply that control. We also lack a comparable monetary-savings estimate and a separate breakdown of Codex judgment usage. Small, dependent search samples limit generalization, and missing historical configuration fields limit reproducibility.

The fixed-policy experiments do not measure sustained self-improvement or reduced human oversight. Extending DIVE to AI systems that improve other AI systems (AI4AI), or to recursive self-improvement (RSI), would require testing Skill revisions on unseen tasks, checking regressions, and accounting for update costs. Claims about reduced oversight would also require measurements of human time and risk.

9 Conclusion

DIVE represents judgment as reusable Cognitive Judgment Skills and connects their decisions to agent execution and feedback. The experiments show savings from guarded stopping and improved repair resolution with fewer reported tokens in the Codex cohort. The same-state studies give a less uniform picture: useful actions do not consistently yield better selection or continuation outcomes. Comparisons with fixed policies and matched generic review are needed to determine which judgment procedures account for the task-level gains.

AI USE STATEMENT

Generative AI tools assisted with conceptual framing, hypothesis refinement, and mathematical formalization; methodology and experiment design; implementation and editing of methods, analysis code, and audit scripts; extraction and reformatting of retained records; result interpretation; translation and bilingual consistency checking; and manuscript organization, drafting, literature search, LaTeX preparation, and figures. They were not used to generate synthetic datasets or formal proofs; qualitative analysis was not part of this study. Reported measurements come from retained execution and evaluation records, or are explicitly labeled as projections or author-reported summaries. AI-assisted text, code, figures, and numerical summaries were checked against available sources through manual and scripted audits. The authors take responsibility for the final content.

ETHICS STATEMENT

No human-subject study was conducted. The experiments used benchmark or isolated research environments, and generated changes were not automatically deployed. Agentic code modification and experience reuse can propagate unsafe, insecure, or misleading changes; the results therefore do not establish suitability for unsupervised production or other high-stakes use. Any released traces or artifacts must remove credentials, private-repository content, and author-identifying metadata while preserving evaluation provenance.

REPRODUCIBILITY STATEMENT

The method and evaluation sections and Appendices A.1, B, C, D, and E specify policy versions, decision contracts, endpoints, and accounting rules. Table 1 identifies the DeepSeek Pro50 and Codex Pro30 settings; Table 3 reports the completed recognition summaries. The retained archive supports the historical diagnostics but does not yet provide a complete, reconciled task-level reproduction package for these summaries. Missing material includes aligned task outcomes, complete usage coverage, and exact configurations for the added repair cohorts. The anonymous code repository is linked below the abstract. The historical Verified query-budget discrepancy remains unresolved; the repository link does not imply that all execution records or missing configurations have been released.

REFERENCES

C. Nicolò De Sabbata, Theodore R. Sumers, Badr AlKhamissi, Antoine Bosselut, and Thomas L. Griffiths. Rational metareasoning for large language models, 2024. URL https://arxiv.org/abs/2410.05563. arXiv:2410.05563v3.

Zhibin Gou, Zhihong Shao, Yeyun Gong, Yelong Shen, Yujiu Yang, Nan Duan, and Weizhu Chen. CRITIC: Large language models can self-correct with tool-interactive critiquing. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=Sx038qxjek.

Jie Huang, Xinyun Chen, Swaroop Mishra, Huaixiu Steven Zheng, Adams Wei Yu, Xinying Song, and Denny Zhou. Large language models cannot self-correct reasoning yet. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=IkmD3fKBPQ.

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? In *International Conference on Learning Representations*, pp. 54107–54157, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/edac78c3e300629acfe6cbe9ca88fb84-Abstract-Conference.html.

Xiangyi Li, Yimin Liu, Wenbo Chen, Bingran You, Zonglin Di, Yifeng He, Shenghan Zheng, Kyoung Whan Choe, Jiankai Sun, Shuyi Wang, Chujun Tao, Binxu Li, Xuandong Zhao, Hejia Geng, Xiaojun Wu, Junwei Zhou, Xiaokun Chen, Hanwen Xing, Yubo Li, Qunhong Zeng, Di Wang, Yuanli Wang, Roey Ben Chaim, Penghao Jiang, Haotian Shen, Luyang Kong, Xinyi Liu, Runhui Wang, Xuanqing Liu, Jiachen Li, Xin Lan, Yueqian Lin, Wengao Ye, Junwei He, Songlin

Li, Yue Zhang, Yipeng Gao, Yijiang Li, Ze Ma, Liqiang Jing, Tianyu Wang, Kaixin Li, Yiqi Xue, Haoran Lyu, Yizhuo He, Yuchen Tian, Shutong Wu, Bowei Wang, Yixuan Gao, Bo Chen, Litong Liu, Sikai Cheng, Jiajun Bao, Shuaicheng Tong, Shuwen Xu, Terry Yue Zhuo, Tinghan Ye, Qi Qi, Miao Li, Longtai Liao, Zelin Tan, Chang Shi, Xilin Tang, Srinath Tankasala, Boqin Yuan, Yaoyao Qian, Jianhong Tu, Chenguang Wang, Yizhou Sun, Wei Wang, Aaron Taylor, Ziyue Yang, Changkun Guan, Zhikang Dong, Xinyu Zhang, Steven Dillmann, Han chung Lee, and Dawn Song. SkillsBench: Benchmarking how well agent skills work across diverse tasks, 2026. URL https://arxiv.org/abs/2602.12670. arXiv:2602.12670v4.

Tennison Liu and Mihaela Van Der Schaar. Position: Truly self-improving agents require intrinsic metacognitive learning. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 81714–81727. PMLR, 2025. URL https://proceedings.mlr.press/v267/liu25cw.html.

Itay Nakash, George Kour, and Ateret Anaby Tavor. Efficient agent evaluation via diversity-guided user simulation. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 6: Industry Track)*, pp. 1627–1648. Association for Computational Linguistics, 2026. doi: 10.18653/v1/2026.acl-industry.112. URL https://aclanthology.org/2026.acl-industry.112/.

Alexander Novikov, Ngân Vũ, Marvin Eisenberger, Emilien Dupont, Po-Sen Huang, Adam Zsolt Wagner, Sergey Shirobokov, Borislav Kozlovskii, Francisco J. R. Ruiz, Abbas Mehrabian, M. Pawan Kumar, Abigail See, Swarat Chaudhuri, George Holland, Alex Davies, Sebastian Nowozin, Pushmeet Kohli, and Matej Balog. AlphaEvolve: A coding agent for scientific and algorithmic discovery. *arXiv preprint arXiv:2506.13131*, 2025. URL https://arxiv.org/abs/2506.13131.

OpenEvolve Contributors. OpenEvolve. Software repository, 2026. URL https://github.com/algorithmicsuperintelligence/openevolve. Accessed September 11, 2026.

Maxime Robeyns, Martin Szummer, and Laurence Aitchison. A self-improving coding agent. In *ICLR 2025 Workshop on Scaling Self-Improving Foundation Models*, 2025. URL https://openreview.net/forum?id=rShJCyLsOr.

John Scoville, Shengzhuang Chen, Yejin Bang, Stefan Winzeck, and Jonathan Richard Schwarz. Cognitive demand steering for adaptive meta-reasoning in large language models, 2026. URL https://arxiv.org/abs/2608.01319. arXiv:2608.01319v1.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 36, pp. 8634–8652, 2023. doi: 10.52202/075280-0377. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/1b44b878bb782e6954cd888628510e90-Abstract-Conference.html.

Kumar Shridhar, Koustuv Sinha, Andrew Cohen, Tianlu Wang, Ping Yu, Ramakanth Pasunuru, Mrinmaya Sachan, Jason Weston, and Asli Celikyilmaz. The ART of LLM refinement: Ask, refine, and trust. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 5872–5883. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.naacl-long.327. URL https://aclanthology.org/2024.naacl-long.327/.

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling LLM test-time compute optimally can be more effective than scaling parameters for reasoning. In *International Conference on Learning Representations*, pp. 10131–10165, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/1b623663fd9b874366f3ce019fdfdd44-Abstract-Conference.html.

Weihang Su, Jianming Long, Qingyao Ai, Qiaozhi He, Yichen Tang, Changyue Wang, Yiteng Tu, Yingbo Wang, and Yiqun Liu. Skill retrieval augmentation for agentic AI, 2026. URL https://arxiv.org/abs/2604.24594. arXiv:2604.24594v3.

SWE-bench Pro Contributors. SWE-bench Pro. Software repository, 2026. URL https://github.com/scaleapi/SWE-bench_Pro-os. Accessed September 17, 2026.

Liyan Tang, Cyrus Rashtchian, Chun-Sung Ferng, Andrew Tomkins, Da-Cheng Juan, and Tu Vu. WikiSkill: Compiling agent experience into persistent knowledge for skill evolution, 2026. URL https://arxiv.org/abs/2608.27454. arXiv:2608.27454v1.

Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An open-ended embodied agent with large language models. *Transactions on Machine Learning Research*, 2024. URL https://openreview.net/forum?id=ehfRiF0R3a.

Zhenyu Wu, Qingkai Zeng, Zhihan Zhang, Zhaoxuan Tan, Chao Shen, and Meng Jiang. Large language models can self-correct with key condition verification. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 12846–12867. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.emnlp-main.714. URL https://aclanthology.org/2024.emnlp-main.714/.

Chenxu Yang, Qingyi Si, Mz Dai, Dingyu Yao, Mingyu Zheng, Minghui Chen, Zheng Lin, and Weiping Wang. Test-time prompt intervention. *Proceedings of the AAAI Conference on Artificial Intelligence*, 40(40):34223–34231, 2026. doi: 10.1609/aaai.v40i40.40718. URL https://ojs.aaai.org/index.php/AAAI/article/view/40718.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=WE_vluYUL-X.

Jenny Zhang, Shengran Hu, Cong Lu, Robert Lange, and Jeff Clune. Darwin Gödel machine: Open-ended evolution of self-improving agents. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=pUpzQZTvGY.

Andrew Zhao, Daniel Huang, Quentin Xu, Matthieu Lin, Yong-Jin Liu, and Gao Huang. ExpeL: LLM agents are experiential learners. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(17):19632–19642, 2024. doi: 10.1609/aaai.v38i17.29936. URL https://ojs.aaai.org/index.php/AAAI/article/view/29936.

Ziqing Zhuang, Linhai Zhang, Jiasheng Si, Deyu Zhou, and Yulan He. Beyond meta-reasoning: Metacognitive consolidation for self-improving LLM reasoning. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 23884–23913. Association for Computational Linguistics, 2026. doi: 10.18653/v1/2026.acl-long.1095. URL https://aclanthology.org/2026.acl-long.1095/.

A Detailed Results and Summary Definitions

Figures 3–5 use the same completed summaries as Tables 1–3. Figure 3 covers the Verified/MiniMax block; the DeepSeek Pro50 and Codex Pro30 results remain separate in Table 1.

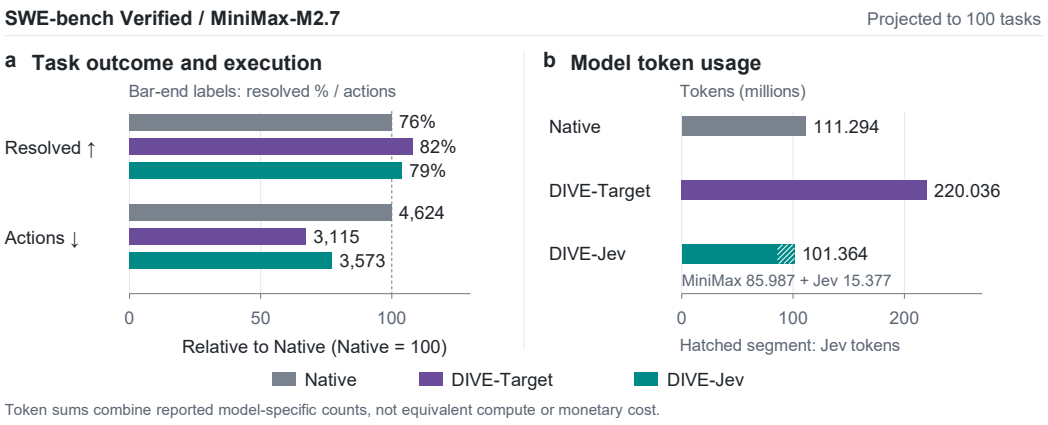


Figure 3: Verified repair outcomes and reported token usage with MiniMax-M2.7, projected to 100 tasks. (a) Resolution and repository actions relative to Native (= 100); bar-end labels give the projected resolution rate and action count. (b) Model-specific token totals, including the hatched Jev component. DIVE–Target has the highest resolution rate and token usage; DIVE–Jev uses fewer reported tokens than both comparators. Projection does not create 100 independent observations. Token counts across models do not imply equivalent computation or monetary cost.

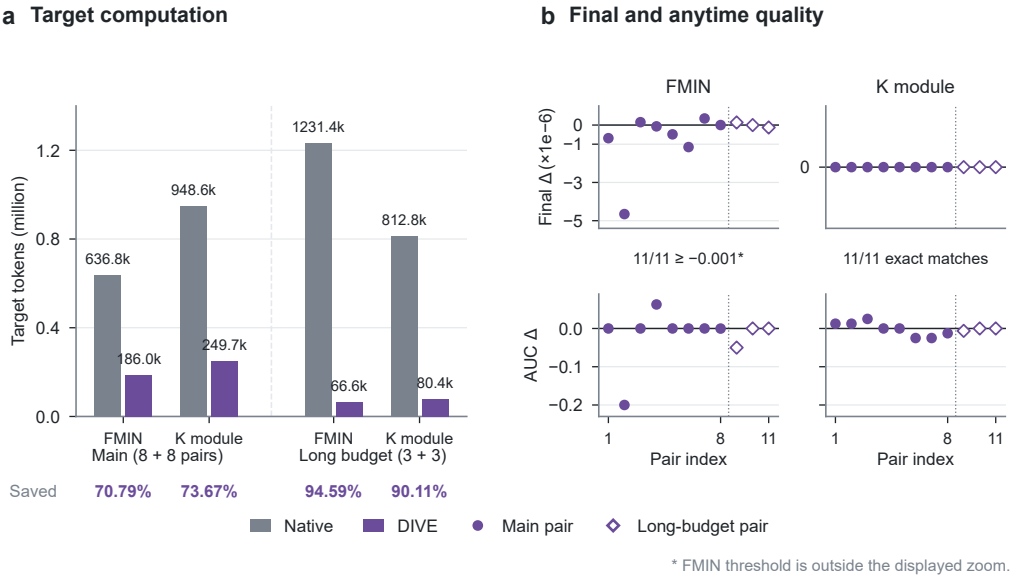


Figure 4: Target computation and paired quality under guarded stopping. (a) Cohort-total Target tokens for Native and DIVE; bar labels are in thousands and percentages denote savings. (b) Paired endpoint and normalized-AUC differences, DIVE minus Native. Within each task, indices 1–8 are main pairs (circles) and 9–11 are longer-budget pairs (open diamonds). All FMIN pairs meet the 0.001 quality margin, outside the magnified axis; K Module endpoint scores match. AUC differences are descriptive and do not isolate the causal effect of stopping.

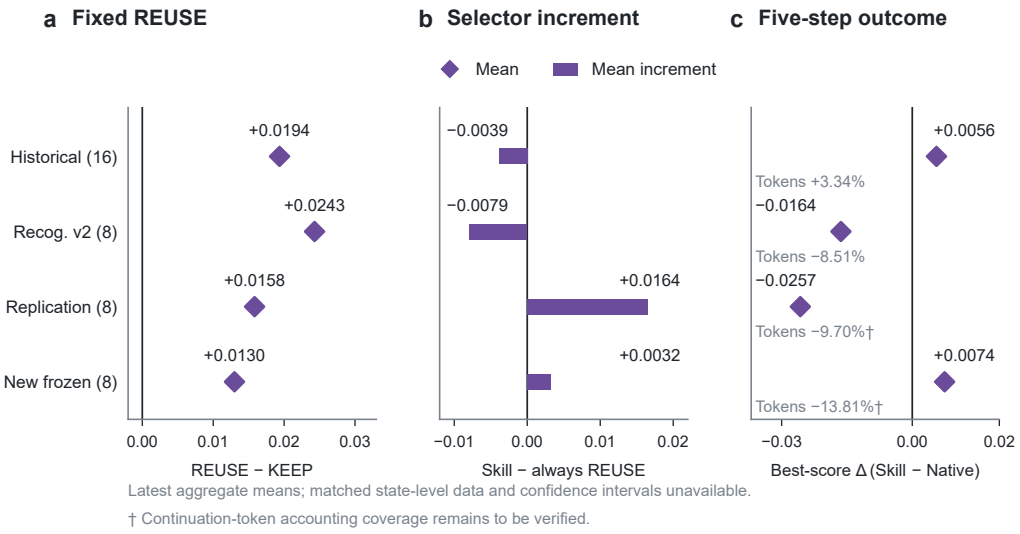


Figure 5: Action benefit, selector increment, and continuation outcome on Signal Processing. (a) Mean local REUSE–KEEP effect. (b) Mean local Skill–always-REUSE increment. (c) Mean paired Skill–Native difference in best score, including the shared checkpoint and five valid successors, with aggregate continuation-token changes. Diamonds denote means and bars denote mean increments; cohort counts are 16, 8, 8, and 8. Matched state-level data and confidence intervals are unavailable for these summaries. Other five-step comparators appear in Table 3. † Token-accounting coverage remains to be verified.

A.1 COHORTS, ENDPOINTS, AND ACCOUNTING COVERAGE

The completed author-supplied summaries are the basis of Tables 1–3 and Figures 3–5. Verified values are projected to 100 tasks; DeepSeek and Codex Pro totals cover 50 and 30 tasks, respectively. Projection is a reporting scale, not an observed sample size. The repair backbones and hosts are reported separately. The earlier 33/19-pair repair archive in Appendix B supplies trace diagnostics and configuration provenance; its incomplete usage records do not replace the completed table totals.

For recognition, the effective matched-state counts are 16 for Historical and eight each for Recognition v2, Replication, and New frozen. These completed summaries supersede the smaller subsets in the earlier transfer of records. For state i and method m, the five-step endpoint is

$$B_{i,m} = \max\{Q_{i,0}, Q_{i,1}, \ldots, Q_{i,5}\}, \qquad \Delta_{S,m} = \frac{1}{n}\sum_{i=1}^{n}(B_{i,S} - B_{i,m}),$$

where $Q_{i,0}$ is the shared checkpoint score and the remaining values are scores of five valid successors. Local action and selection differences use the exact parent and are distinct from this continuation endpoint. Figure 1c and Appendix E.5 retain the earlier nine-state diagnostic. Its hindsight bounds and subset bootstrap intervals are not applied to the completed cohorts.

Continuation-token change is $(\sum_i T_{i,S}/\sum_i T_{i,N} - 1) \times 100\%$, rather than a mean of per-state ratios. The accounting scope is recorded Target and Supervisor usage during continuation, excluding source generation and development. Coverage of that scope for Replication and New frozen remains to be verified against their raw records; their -9.70% and -13.81% values carry the † marker. The completed summaries do not provide state-level observations for independent recomputation of uncertainty. We report their means without adding confidence intervals or statistical-significance claims.

Reproduction fields for the added repair cohorts. DeepSeek Pro50 and Codex Pro30 are separate author-supplied aggregates. Table 1 reports their model/host labels, task counts, outcomes, repository actions, and token totals; Codex uses GPT-5.6-Luna with `xhigh` reasoning effort through the Codex SDK. A complete reconstruction also requires the task-ID lists and paired scorer outputs, model and host versions, prompts and Skill hashes, sampling and budget settings, and per-request usage ledgers. These have not been assembled into a reconciled reproduction package for the added cohorts. Codex judgment usage is not separately itemized, and Pro monetary costs remain unreported.

B RECOVERED CONFIGURATION AND REMAINING GAPS

Table 4 describes the earlier trace archive, not a complete configuration record for every cohort in Table 1. Current summaries and missing reproduction fields are described in Appendix A.1. For all 33 selected Verified Skill-Ask runs, the corresponding batch configuration records a temperature of 1.0, an 80-action limit, a 962-query limit, a 20-point routing threshold, and the same named skill pack. These are configuration records, not proof of an immutable provider-side model snapshot. The Pro closed-loop configuration instead records a dynamic fork threshold initialized at 20 with percentile 0.25; its setting must not be silently replaced by the Verified policy.

The Verified pack is referred to here as `repair-pack-A`; its archived record contains seven skills and disables mutation. The configuration also disables memory and the action-score bank. The pack hash, selected run paths, and recovered configuration sources are retained in the internal source ledger. Provider snapshots, reasoning effort, complete host/evaluator identities, role-complete token telemetry, and hardware details remain incompletely recovered. New run settings cannot retroactively fill these historical gaps.

The 962-query value in Table 4 is the batch configuration, whereas all 33 selected Verified run receipts record an effective Target query limit of 2,562. The override provenance remains unresolved; neither value is treated as a universally shared effective budget.

Jev deployment and request accounting. The common-33-task Jev comparison is retrospective. Native, DIVE, and DIVE+Jev record 1,526/4,036/1,951 Target API requests, 0/326/0 routing re-

Table 4: Recovered configuration. Unavailable denotes missing evidence, not a default. Model names are recorded identifiers, not immutable provider snapshots.

Field	Verified	Pro	OpenEvolve stopping
Target alias	MiniMax-M2.7	DeepSeek-V4-Flash	deepseek-flash
Supervisor	MiniMax-M2.7	DeepSeek-V4-Pro	No attempts
Sampling	Temperature 1.0	Unavailable	Unavailable
Skill / policy	Fixed 7-skill pack	Batch-specific protocol	progress_control 0.6.0
Selected units	33 task pairs	19 task pairs	16 stopping pairs
Recorded budget	80 actions; 962 queries	Full action budget unavailable	Native: 5 / 20 iterations
Environment	Instance containers	Bridge timeout 1,200 s	Local search host
Token coverage	Incomplete	Incomplete	Recorded Target tokens

Table 5: Archived repair pairs. B: native baseline; D: historical skill/consultation arm. Supervisor requests use the archive's cohort-specific logging categories. Counts are not tokens.

Cohort	Pairs	Resolved B/D	Actions B/D	Queries B/D	Sup. D
Verified	33	25/27	1,526/1,028	1,526/4,036	436
Pro	19	9/9	1,520/815	1,520/3,202	116

quests, and 0/110/52 consultations, respectively. The Jev arm adds 1,323 ranker responses; its reported usage is 28,375,619 MiniMax tokens plus 5,074,494 Jev tokens. This earlier archive lacks comparable token totals for the other two arms and full failed-request coverage. Its request counts alone do not establish token savings; completed token summaries are reported separately in Table 1. The broader 36-instance Verified summary reports 26/27 successes and remains separate from this common-33-task comparison. Pro's designated re-evaluation README supersedes the earlier 9/8 outcome aggregate with 9/9; the archived resource ledger is retained, without reusing the old per-task outcome colors or McNemar counts.

Attribution controls and current scope. Repair lacks a matched generic-deliberation arm. The stopping policy is implemented as a guarded rule; comparing it with continued search measures the rule's effect. The archived Recognition controls in Appendix E.5 concern their own subsets; current cohort means are given in Table 3. The earlier Recognition admission audit concerns a different frozen promise and protocol.

C Skill Interfaces and Concrete Examples

We describe each Skill by its triggering situation, observable inputs, judgment question, decision procedure, admissible actions, abstention rule, verifier, and evidence status. This shared interface compares software-repair and program-search procedures without implying that their policies transfer unchanged across environments. The four cards below are representative, not an exhaustive prompt dump; *executed*, *development*, and *shadow-only* describe evidence status rather than effectiveness. The repair Markdown files use explicit situation, evidence, policy, and boundary sections, but individual author attribution and a complete construction history are not recoverable from the selected runtime archive. No learned-skill provenance or expert annotation is inferred from these files.

C.1 Recognition v1: input, decision, and feedback

This instance uses the frozen `recognition-score-sprint-v1` contract, with implementation prefix `52bf`; it is the historical study, not the later mechanism-v2 revision. The packet binds the fork, parent-code hash, evaluator hash, pilot-bank hash, action, and verifier identifiers. It supplies parent metrics, prefix observations, structural flags, and three controlled pilot comparisons, each containing control and treatment observations repeated twice. The model receives identifiers

and structural flags for the current parent, not its algorithm code. Identical-input pilot repeats check reproducibility, not transfer to independent states.

The deterministic `common_head` rejects a stale or missing parent, a missing signal-validity flag, an unavailable or mismatched action, an occupied or unknown input-filter role, and a duplicate filter. A passing state is labeled `ambiguous`; this means that the action is legal and a decision is requested, not that uncertainty has been statistically calibrated. The model is `deepseek-v4-pro`, with reasoning effort `none`, a 2,048-output-token limit, and no logical retry. Both the generic and skill arms receive the neutral packet. The skill arm additionally receives the fixed decision program.

The skill instructs the model to identify what the intervention changes, compare its mechanism with the current context, and distinguish overall-score gain from noise or responsiveness gains. It permits one reversible trial when the evidence supports it and otherwise asks for abstention; it explicitly states that a transfer boundary and a score-negative counterexample have not been established. These are semantic instructions. The parser does not verify whether the model performed those comparisons.

The response contains exactly five fields: `disposition`, `selected_card_id`, `action_id`, `cited_evidence_ids`, and `reason_code`. A `bounded_trial` must name the current card, the one legal action, at least one known pilot evidence ID, and `trial_justified`. An `abstain` must have null card and action IDs, an empty citation list, and one of `insufficient_evidence`, `incompatible_context`, or `tradeoff_risk`. Malformed or contradictory output is retained as `invalid_response` and triggers fallback; it is not counted as a valid semantic abstention. The v1 prompt listed the fields and enums but omitted the full cross-field constraints enforced by the parser.

A valid trial authorizes next-candidate guidance to insert an edge-padded $(0.25, 0.5, 0.25)$ convolution at the input, before the length check and existing algorithm, while preserving the remaining executable code and output contract. The Target still generates the candidate. The Host evaluates it; a separate diagnostic checks action isolation, evaluator identity, finite metrics, and signal validity before assigning an exact-parent score delta. That diagnostic does not filter the five-step intention-to-treat outcome. There is no second judgment or automatic rollback in this protocol.

In s198, the legal-state check passed and the model returned a valid abstention with `insufficient_evidence`; the Host retained native continuation. The available metrics included noise reduction 0.78143, correlation 0.96928, and responsiveness 0.83268. Offline evaluation of the fixed action found a local overall-score loss of 0.000033074. The retained answer contains only a reason code, so it cannot establish that the model used those particular metrics or understood the parent's Gaussian filter. In s201, the answer also said `abstain`, but supplied a non-null action ID and failed the cross-field check. The unused action's offline gain was 0.035113116. Clearing that ID makes the response valid but leaves the decision as abstention. This parser diagnostic cannot be reported as recovered task benefit.

C.2 Representative Judgment Skill specifications

The fixed field order makes the decision procedures and evidence boundaries directly comparable across cards.

Repair: `root-cause-diagnosis` *Executed fixed-pack procedure*

Situation. The responsible path is approximately located, but several causal explanations remain or the current patch only blocks a symptom.
Inputs. Consumer observations, upstream producers, cross-boundary invariants, and falsifiable hypotheses with counterevidence.
Judgment. Which producer behavior or violated assumption actually generates the observed failure?
Procedure. Trace backward from the failure; state what each candidate cause predicts and what would refute it; prefer the narrow point that produces the invalid state.
Actions. `diagnose`, `inspect`, or `reproduce`; use `edit_code` only after the evidence supports a cause.
Abstention / fallback. Return to localization when ownership is uncertain; do not call a symptom guard a root-cause repair.
Verifier. A fresh reproduction and a focused check must reach the implicated behavior, followed by scope-appropriate regression checks.
Evidence status. Executed inside the frozen seven-Skill repair pack; the archived cohorts do not isolate this card's causal effect.

Repair: `completion-control` *Executed fixed-pack procedure*

Situation. A patch is about to be called complete, reviewed, or submitted.
Inputs. The current diff, fresh direct validation, relevant regressions, unresolved failures, and evidence that tests reached the target behavior.
Judgment. Does every completion claim have current, direct, and scope-matched evidence?
Procedure. Pair each precise claim with a check of the latest patch; distinguish behavior failures from setup or test-route failures.
Actions. `validate_fix`, `run_regressions`, `repair_test_route`, `diagnose`, `review`, or `submit`.
Abstention / fallback. If the evidence is stale, indirect, or misses the changed path, continue validation or return to diagnosis; the Skill does not grant commit or merge authority.
Verifier. Tool and test outputs tied to the current patch, plus the external patch evaluation when available.
Evidence status. Executed inside the frozen seven-Skill repair pack; the paper reports observable traces and task outcomes, not isolated efficacy for this card.

Search: `progress_control@0.6.0` *Executed confirmatory family*

Situation. A valid current candidate has reached a preregistered task target while search budget remains.
Inputs. Current-candidate and evaluator identity, measured `combined_score`, contract checks, infrastructure status, remaining iterations, and the frozen threshold.
Judgment. Is there an evidence-based reason to continue, or should search stop at the guarded satisficing floor?
Procedure. Apply the task-score threshold only to the current valid candidate and require candidate, infrastructure, and budget guards.
Actions. Emit a verifiable stop such as `STOP_VERIFIED_SATISFIED`, or retain native search.
Abstention / fallback. An inherited best score, model confidence, budget exhaustion, or an unverified candidate cannot satisfy the stop gate.
Verifier. Stop receipt and final task-specific quality, with anytime quality reported separately.
Evidence status. Across 16 matched pairs, calls fall from 200 to 26 and recorded Target tokens from 1,585,387 to 435,727; all final-quality gates pass, while five pairs lose anytime quality. This supports guarded search contraction, not optimality or superiority to an equivalent rule.

Cognitive: `recognition-and-fit-score-trial` *Development; mechanism-v2*

Situation. A prior local improvement is available, and the system must decide whether its mechanism fits the exact current parent well enough for one bounded trial.
Inputs. A shared neutral packet containing the exact parent, prefix metrics, structural facts, legal actions, and controlled pilot vectors; post-action outcomes are hidden from the judge.
Judgment. Do mutually consistent mechanism and context evidence justify one reversible trial?
Procedure. Identify the past mechanism, compare its preconditions and trade-offs with the current state, separate overall score from noise and responsiveness, and cite packet evidence IDs. Repeated identical inputs count as reproducibility evidence, not independent transfer evidence.
Actions and output. Choose `bounded_trial` or `abstain`; return exactly `disposition`, `selected_card_id`, `action_id`, `cited_evidence_ids`, and `reason_code`.
Abstention / fallback. Abstain if the packet does not support a bounded trial with cited evidence. Do not invent empirical-fit conclusions or score-negative counterexamples, create actions or probes, use post-fork information, or make a second judgment.
Verifier. Report native outputs, isolated adoption, exact-parent gain, and five-step intention-to-treat outcomes separately.
Evidence status. The completed v2 summary has eight matched states. Five-step $S - G = +0.000220$ and local $S - R = -0.007926$ (Table 3); these means do not establish stable transfer or a general selection advantage.

A repair judgment that failed to establish completion. In the MiniMax `pydata/xarray-6744` run, a proposed boundary check included `window=4`. After skill use, the agent selected an existing suite instead; it reported 1,790 passing tests and subsequently selected submission with a recorded score gap of 100. The final patch was unresolved, despite fewer repository actions than the matched baseline (24 versus 39). This is a failure of evidence sufficiency in the observed trajectory, not proof that the skill caused failure or that the unexecuted check would have fixed it.

Verification coverage. Skill text, model ranking, command execution, and external evaluation are distinct records. The conceptual DIVE interface associates them, but the historical adapters do not provide a uniform automated check of every association. In particular, declining a route, failing an admission check, an infrastructure fallback, and deliberate evidence-based abstention must not be assigned one correctness label.

D Per-Pair Stopping Evidence

Table 6 lists every pair behind the main stopping result. The initial batch uses seeds 6142, 6243, and 6344; the fresh batch uses 7142, 7243, 7344, 7445, and 7546 for each task. The frozen studies contain six and ten completed pairs, respectively, with no overlap with the legacy seeds below. The resource reduction is 72.52% as a ratio of totals, 71.62% as the equal-pair mean saving, and 72.78% as the median saving. These estimators answer different weighting questions; none is substituted for another. The internal manifest links each pair to its Native and Active run identifiers and trace hashes.

Quality uncertainty and resource sensitivity. The old 16-pair cohort and the six longer-horizon pairs are analyzed separately, with raw quality never pooled across tasks. For FMIN, a one-sided 95% Student-t bound on the mean paired difference is $\bar{d} - t_{0.95,n-1} s_d/\sqrt{n}$. It is -1.90664×10^{-6} for the eight old pairs and -2.18025×10^{-7} for the three sensitivity pairs, both above the frozen -0.001 margin. This is conditional evidence under iid paired-seed and approximate normal-mean assumptions, not a distribution-free guarantee or a bound on per-run harm. K Module differences are all exactly zero in these samples; degenerate variance does not prove population equivalence or zero risk. Table 8 gives a separate percentile-bootstrap sensitivity analysis using 10,000 full-pair resamples within each task/cohort (seed 20260923). Its intervals do not correct historical study selection or Provider-model drift.

Anytime quality and prefix divergence. Let b_i be the best recorded score through event i; the recorded prefix is carried forward to the planned horizon H after termination. Raw AUC is $H^{-1}\sum_i b_i$; target-capped normalized AUC is $H^{-1}\sum_i \min(b_i/\tau, 1)$, with task target τ. These units

Table 6: All 16 stopping pairs used in the main result. I: initial three seeds; F: five fresh seeds per task. Quality differences use each task's evaluator scale; the older seven-pair archive below is separate.

Task	Seed/batch	Tokens Native	Tokens DIVE	Saved (%)	Δq
Function min.	6142/I	67,116	20,670	69.20	-6.86×10^{-7}
Function min.	6243/I	89,539	35,517	60.33	-4.65×10^{-6}
Function min.	6344/I	83,682	19,467	76.74	1.51×10^{-7}
Function min.	7142/F	55,674	15,010	73.04	-6.82×10^{-8}
Function min.	7243/F	69,577	28,517	59.01	-4.85×10^{-7}
Function min.	7344/F	80,712	16,882	79.08	-1.15×10^{-6}
Function min.	7445/F	82,795	22,748	72.52	3.45×10^{-7}
Function min.	7546/F	107,729	27,187	74.76	4.70×10^{-10}
K Module	6142/I	122,014	18,996	84.43	0
K Module	6243/I	126,639	10,478	91.73	0
K Module	6344/I	141,703	9,123	93.56	0
K Module	7142/F	136,792	48,573	64.49	0
K Module	7243/F	100,734	18,186	81.95	0
K Module	7344/F	105,757	38,193	63.89	0
K Module	7445/F	109,574	55,725	49.14	0
K Module	7546/F	105,350	50,455	52.11	0

Table 7: Frozen stopping results, aggregated within task. N/A denotes Native/Active, not missing data. Token savings compare aggregate totals; raw quality is not pooled across tasks.

Task (pairs)	Calls N/A	Tokens N/A	Tokens saved	Mean Δq
Function Min. (8)	40/10	636,824/185,998	70.79%	-8.18×10^{-7}
K Module (8)	160/16	948,563/249,729	73.67%	0
Overall (16)	200/26	1,585,387/435,727	72.52%	not pooled

must not be interchanged. Five of the 16 main pairs have negative normalized differences: FMIN seeds 6243 (-0.20) and 7546 (-0.0002584); K Module seeds 7344 (-0.025), 7445 (-0.025), and 7546 (-0.0125). The two seed-8142 deficits belong to the separate long-horizon sensitivity cohorts. Three further FMIN pairs (6142, 7243, 7344) have tiny negative raw-score AUC differences but zero target-capped differences. The two FMIN invalid-first-candidate cases reach their first valid target candidate at iteration two in Active versus one in Native. Candidate fingerprints often differ before any stop is executed: independently sampled prefixes are not identical counterfactual histories, even at the same seed. Accordingly, these are observed anytime deficits, not losses causally attributed to stopping.

Longer-horizon Function Minimization sensitivity. Seeds 8142, 8243, and 8344 compared the unchanged policy with a 20-iteration Native budget. All three Active runs met the frozen -0.001 final-quality margin, with valid stop receipts and no Supervisor calls or Provider failures. Target calls fell from 60 to 4 and recorded tokens from 1,231,384 to 66,645 (94.59% saved). Per-seed Native/Active calls were 20/2, 20/1, and 20/1; quality differences were $+1.36\times10^{-7}$, -1.14×10^{-9}, and -1.26×10^{-7}. Seed 8142 has raw AUC difference -0.07498, but normalized difference -0.05. All three Native trajectories reached the 1.499 target at iteration one, so this tests avoided post-satisfaction search, not difficult direction judgment. Checkpoint names retain the planned horizon and do not establish Active execution length.

Longer-horizon K Module sensitivity. The same three seeds compared the unchanged policy with a 40-iteration Native budget. All six runs completed without Provider failures or Supervisor calls, all three stop receipts were valid, and both arms reached exactly 1.0 in every pair. Calls were 40/3, 40/3, and 40/2; tokens were 253,198/33,743, 253,613/19,764, and 306,008/26,867. Totals fell from 120 to 8 calls and 812,819 to 80,374 tokens (90.11% saved). Summed Provider latency fell from 596,950 to 257,040 ms; this is not wall-clock time. A local evaluation of the initial program returned 0, but Native attained 1.0 at iterations 2/3/2 and showed no later visible best-score gain.

Table 8: Descriptive 95% percentile-bootstrap intervals. Token saving is $1 - \sum T_A / \sum T_N$, recomputed in each pair-resampled dataset, not the mean of per-pair ratios. Tiny samples limit inference.

Task/cohort	Pairs	Mean quality Δ interval	Token-saving interval
FMIN / old	8	$[-1.99\times10^{-6}, -2.54\times10^{-8}]$	[65.79%,75.17%]
K Module / old	8	$[0, 0]$ (observed)	[61.88%,84.27%]
FMIN / longer	3	$[-1.26\times10^{-7}, 1.36\times10^{-7}]$	[88.90%,98.92%]
K Module / longer	3	$[0, 0]$ (observed)	[86.67%,92.21%]

Seed 8142 attained the target one iteration later in Active. This supports verified satisficing, not superiority to an equivalent guarded threshold.

Legacy development archive. Table 9 reports the seven-pair development archive. The token changes are calculated within each pair, then averaged with equal pair weight. Task-specific quality differences are reported separately because raw objective values from different tasks are not a universal quality scale. The legacy quality floor is a development acceptance check, not a statistical noninferiority result.

Table 9: Archived early-stopping pairs. Dev: development; HO: one held-out seed per task. Calls and tokens are Target-only; recorded Supervisor calls are zero.

Task	Seed/split	Calls B/D	Tokens B	Tokens D	Token change
Function min.	19 / Dev	5/5	25,894	29,299	$+13.15\%$
Function min.	20 / Dev	5/2	28,905	6,857	-76.28%
Function min.	22 / Dev	5/2	23,280	7,949	-65.85%
Function min.	31 / HO	5/1	25,802	2,430	-90.58%
K-Module	19 / Dev	20/9	58,067	24,965	-57.01%
K-Module	20 / Dev	20/1	56,550	1,909	-96.62%
K-Module	31 / HO	20/1	56,998	1,909	-96.65%

Version and qualification boundary. The cross-task record belongs to the progress-control development lineage with task overlays. The attribution report separately inspects the named compiled predicate `oev.progress_control@0.4.0`; no identity across namespaces is inferred. A later formal multi-task batch was converted to diagnostic-only records after unavailable cumulative evaluation counts made its stop gate unreachable. A candidate repair and a `progress_control@0.6.1` development revision do not replace the frozen `@0.6.0` retrospectively. Those diagnostic runs do not enter the frozen 16-pair study in Table 7 or the separate archive in Table 9.

E Decision-Level Protocol Details

E.1 Figure 1: cohorts, endpoints, and interpretation

Figure 1 combines complementary diagnostics, not a pooled experiment. Panel (a)'s xarray-6744 case is an archived DIVE run; its proposed `window=4` check was not executed as proposed, and the existing suite's 1,790 passing tests did not establish official resolution. The diagram's judgment question is an explanatory label, not a quotation. No branch establishes that performing the omitted probe would have repaired the issue.

The adjacent termination matrix comes from a separate August mixed-configuration mini-SWE report. For Native, submitted runs comprise 6 resolved and 1 unresolved, while limit-terminated runs comprise 9 resolved and 20 unresolved. For Skill-Ask, the respective counts are 15/9 and 6/5. These are arm-level counts over 36 and 35 scored runs, not 35 matched termination outcomes; one additional Skill-Ask run has an evaluator/environment error. The report-level counts are retained without reconstructing unavailable raw trajectories. An unresolved submission is not individually established to be premature, and a limit termination can still leave a resolving patch.

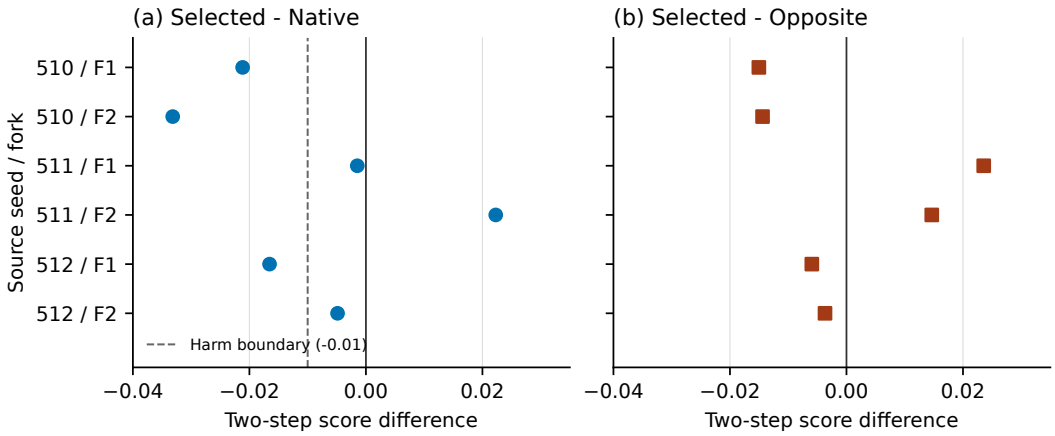


Figure 6: Held-out v3 direction selection. The legacy two-step label denotes the second-step candidate score, not best-of-two. Each point is a difference of two-replica means at one fork; two forks share each of three independent source lineages. All six selected actions were consumed. Selection beats Native on one fork and the opposite direction on two; three Native-relative differences cross the prespecified -0.01 harm boundary. All observations are shown, without treating 36 branch units as independent samples.

Panel (b) uses Signal conditional v3: three source runs (510–512), two forks per source, three arms per fork, and two repeats per arm, each continued for two steps. All 36 branches completed, with 72 successful Target calls and no Supervisor calls; receipts confirm consumption at all six selected forks. The score is taken from the second candidate (`events[1]`), despite the archive's legacy two-step score name. Selected exceeds Opposite at 2/6 forks and Native at 1/6; these are sampled mean comparisons, not a 33% judgment-accuracy estimate. Native repeat ranges at source 510's two forks are 0.041723 and 0.073220, so the near-zero overall Selected–Opposite difference is not evidence of equivalence. Figure 6 preserves the harm-threshold view of the same observations.

Panel (c) compares policies on the same nine eligible Recognition states, not the five-step continuation endpoint. Always-legal-reuse gains 0.019356 on average, versus 0.015458 for the original policy and 0.019409 for a retrospective bound choosing the better of the two evaluated actions at each state. Most of the policy deficit comes from the s201 invalid-response fallback; it is not established to be a semantic choice error, and repairing the output format does not itself imply a REUSE decision. The hindsight headroom over always-reuse is only 0.000053 in these states; this does not characterize all possible future states or actions. State-level results appear in Appendix E.5.

E.2 Development direction cases

The exploratory study used two forks from each of three source runs, with Native, fixed noise-oriented, fixed response-oriented, and selector branches (24 completed units). Mean selected-minus-average-fixed was $+0.02665$, positive on five of six forks; against Native it was $+0.000664$ with only one win ($+0.05947$). The fixed-direction reference is an arithmetic mean, not a sampled random arm. The conservative v2 study completed 48 units and 96 Target calls on seeds 504–506, but selected `KEEP_NATIVE` on all six forks, below its required two interventions. Without independent applicability labels, zero coverage is not established correct abstention, and conditional intervention utility is undefined. The earlier descriptive figure is retained below; the completed active v3 held-out outcome is the main-text comparison.

The v3 diagnostic selected two prefix-only development forks, one for each proposed direction. Table 10 preserves their different Native-relative outcomes. Both guidance intents were consumed by the Host, but these selected cases are not a held-out success rate.

The subsequent conditional v3 held-out study completed all 36 branches and 72 Target calls without a Provider failure; no Supervisor was used. All six actions were consumed, but mean selected-minus-opposite was -0.0001456 and mean selected-minus-Native was -0.0091562, with three

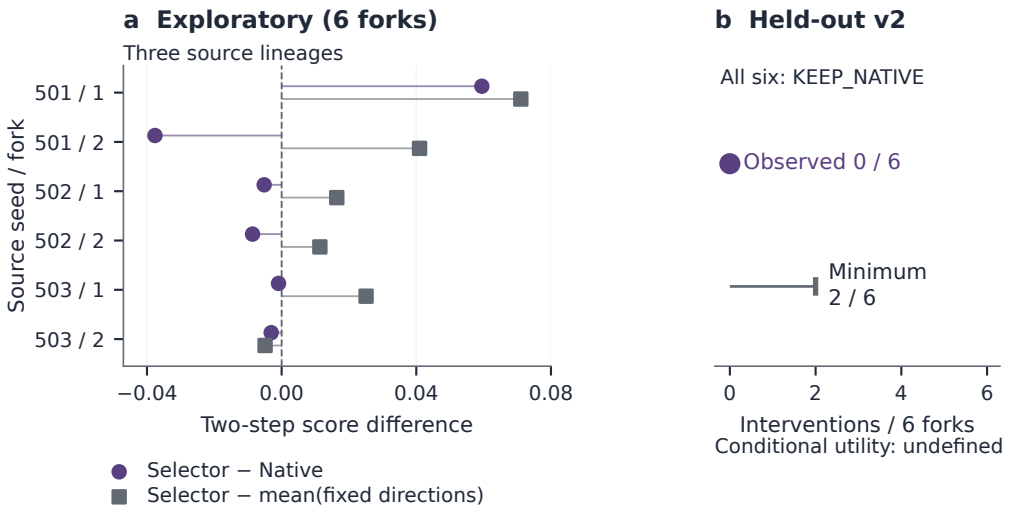


Figure 7: Earlier exploratory and v2 coverage diagnostics, not substitutes for active v3 held-out performance. Left: sampled selection gaps relative to two different comparators. Right: zero interventions against a frozen minimum of two.

Table 10: Two v3 development comparisons. Differences are sampled branch-score differences, not known causal effects or oracle regret.

Selected direction	Selected − Native	Selected − Opposite	Host consumed
Noise-oriented	0	+0.047019	Yes
Response-oriented	+0.056497	+0.024539	Yes

harms larger than 0.01. The two selected development cases therefore do not establish held-out benefit.

E.3 Software-repair judgment records

Table 11 uses the same 33 Verified and 19 Pro Skill-Ask runs as the task-level ledger. Each valid post-skill record is matched to a unique valid pre-skill record at the same run and step. Multiple post-skill records can share one pre-skill record, so these comparisons are not independent decision nodes. A changed command hash does not establish a semantic change, execution, or correctness. Pro attributes all ranking records to the Target; the older Verified records omit the ranker field and require the historical protocol to identify that role.

Table 11: Observable judgment records, not judgment accuracy. Counts refer to records unless otherwise stated. The `new_evidence` field is a logged flag, not an independent usefulness label.

Record or event	Verified (33 runs)	Pro (19 runs)
Pre-skill / post-skill rankings	994 / 431	796 / 263
Skill injections	321	265
Diagnostic consultations	110	116
Observed follow-ups / all follow-ups	110 / 110	112 / 116
Follow-ups flagged as new evidence	109	111
Matched pre/post record comparisons	430	263
Comparisons with changed top action type	142	122
Comparisons with changed command hash	380	258
Distinct within-run post-skill steps	320	263

E.4 RECOGNITION-AND-FIT: EARLIER EVIDENCE-ADMISSION AUDIT

A separate Development audit asked whether experience reuse still requires semantic judgment after shared deterministic admission checks. The card frozen for that earlier audit cites two transitions, both from source seed 173, and no counterexample. Its discovery label marks positive combined-score change as support, but the card promises increased noise reduction without responsiveness or combined-score loss. At iterations 16 and 17, exact-parent responsiveness changes were -0.05728 and -0.02949; the former also reduced noise reduction by 0.00779. Thus neither cited event verifies the card's stated consequence, despite both having positive combined-score change.

Across nine selected prefix fixtures from three independent source runs (177, 184, 185), four also contain an already occupied prefilter role: this is a deterministic add-component mismatch, not a cognitive judgment opportunity. The other five have a legal trial/abstain choice but lack verified prior support and a counterexample for that promise. An audit reconstructed the retained seed-185 decision packet and confirmed that the card's operation class was omitted while the occupied parent role was visible. Moving this check into a shared Head would correct the interface; it would not demonstrate a Skill benefit over the same Head.

No fixture met the frozen residual-opportunity criteria, so the conditional new four-arm study was not launched. These exposed trajectories remain Development, not newly held-out data. The finding identifies an evidence-label/promise mismatch and a specific missing shared check; it does not establish that recognition-based judgment is impossible or that a stronger Supervisor would repair unsupported experience.

E.5 RECOGNITION-AND-FIT: ARCHIVED DEVELOPMENT DIAGNOSTICS

This subsection preserves the earlier incomplete archive for the response and action-isolation diagnostics in Figure 1c. Its subset counts and bootstrap intervals do not describe the completed cohorts in Table 3; those use the definitions in Appendix A.1. The archived development sprint uses a revised fixed input-prefilter action with weights $[0.25, 0.5, 0.25]$ and edge boundary handling. It is distinct from the earlier unlaunched protocol in Appendix E.4, whose evidence/promise mismatch remains a historical finding. The runtime version is `52bf2a578208a6bcd72b34aa578f508f74f2efcc`, with Target alias `deepseek-flash` and Supervisor alias `deepseek-v4-pro`. Sources 189–204 were allocated before continuation: the first eight to Native (N), direct reuse (R), generic review (G), and Skill (S), and the last eight to N/S. Sources 190, 191, and 194 failed and were not replaced.

The endpoint is the best overall score among the checkpoint incumbent and valid candidates produced in five continuation steps. In this archived subset, thirteen N/S pairs yield means 0.661660157 and 0.667221492, with eight positive, four negative, and one zero difference. The reported paired bootstrap 95% interval for the mean difference 0.005561335 is $[-0.009643244, 0.018989496]$. Each arm uses 65 Target calls; N records 1,506,543 Target tokens and S records 1,506,181 Target plus 50,638 Supervisor tokens, a 3.337% increase in recorded continuation usage. The interval is descriptive for this development sample, not evidence of a confirmed advantage.

AST checks confirm the exact action in seven S branches, with five positive and two negative first-step changes. The other six include four shared-admission unknowns, one valid semantic refusal (source 198), and one invalid-response fallback (201). Nonintervention branches also differ from N: checkpoint matching does not establish byte-identical prompts, and the relay did not forward seed, temperature, or top-p controls. The five-step differences cannot therefore all be attributed to an executed cognitive action.

Smaller matched comparisons yield S$-$G $= 0.00860277$ ($n = 5$; 95% interval $[-0.00015461, 0.01944433]$), S$-$R $= -0.00929637$ ($n = 4$; $[-0.03030155, 0.00404225]$), R$-$N $= 0.00187174$ ($n = 4$), and G$-$N $= -0.013879$ ($n = 5$). These comparisons use their own matched subsets, not differences between unmatched arm averages; all reported intervals cross zero. Of 48 allocated continuation slots, 36 ran and 35 yielded analysis-admitted quality endpoints; 12 were absent after source failure and one R endpoint timed out. Original canonical finalization failures remain recorded; endpoint admission through a versioned supplementary analysis does not make the original pipeline fully successful.

The local counterfactual study reuses nine legal parent states and frozen system choices from this cohort. KEEP leaves the parent unchanged; REUSE applies the exact input-prefilter action. KEEP is not a five-step Native continuation. Seven action measurements were reused; twelve local evaluator calls measured the two previously unexecuted actions and an anchor, with no new model calls or skill updates. Mean local gains are 0 for KEEP, 0.015458266 for the original system, 0.019356048 for legal reuse on every state, and 0.019409469 for the retrospective two-action maximum. This maximum uses observed outcomes and is an evaluation reference, not a deployable selector.

Reuse improves six local states and slightly harms three. The original system falls below always-reuse by 0.003897783 per state: source 198 avoids a loss of 0.000033074, whereas the invalid response on 201 misses a gain of 0.035113116. Removing 201 would change the estimand and cannot turn response failure into a positive main result. Among the seven actually reused states, harm counts change from 2/7 at a 10^{-9} numerical tolerance to 0/7 at a 10^{-3} magnitude threshold. These small negative effects and the near-always-reuse retrospective optimum limit the demonstrated opportunity for a better selector on this sample.

Archived v2 and replication coverage. The retained snapshot contains nine historical local states, six v2 states, and six replication states. Its v2 five-step comparison has six pairs, a mean Skill–Native difference of -0.016391968, and a descriptive 95% bootstrap interval of $[-0.033265136, 0.000653441]$. The replication snapshot reuses a frozen v2 candidate evaluation for seed 221 from seed 211; this limits independence claims for those archived measurements. The completed summaries in Table 3 report eight states for each of v2, Replication, and New frozen. Figure 5 uses those summaries without importing intervals or conditional bounds from the incomplete snapshot.

E.6 Same-state comparison and inference

What the direction experiment measures. From a fixed state, an estimated selection gap can compare the selected branch with completed legal alternatives:

$$r_j = \max_{a \in \mathcal{A}_j} \widehat{U}_j(a) - \widehat{U}_j(a_j^D). \tag{2}$$

The utility and horizon must be fixed before evaluation, missing selected outcomes remain unavailable, and uncertainty must account for shared source lineages. Independent stochastic continuations make a difference between sampled branch scores distinct from known oracle regret. A useful judgment need not improve the ultimate best score, but must improve a prespecified external outcome or acquire decision-relevant information at an accounted cost.

Manipulation and observation. The direction study fixes the parent and search snapshot, evaluator contract, and legal actions, and commits the selector output before the continuations. Provider randomness is not coupled; the two repeats per arm describe within-fork variation. The reported same-state comparison does not imply identical prompts, identical stochastic responses, or randomized branch scheduling.

Experience fit. The reported reuse studies evaluate action consequences and the policy's choice; they do not provide independent semantic applicability labels. A reason such as `insufficient_evidence` records the model's answer, not ground truth about whether the skill fits.

Utility and abstention. Abstention retains the native policy but still incurs any judgment expense and may leave a different context or random-state history. It is therefore distinct from a zero-cost, execution-identical control. The present experiments do not estimate the value of a separate information-acquisition action.

Statistical analysis. Stopping uses task-specific paired differences and the frozen FMIN margin. Direction results are descriptive comparisons of six forks nested within three source runs; three sources do not support a precise population-level uncertainty estimate. The maximum in Equation 2 is a retrospective maximum of sampled outcomes, subject to selection bias, not a known optimum. Recognition's archived bootstrap intervals describe only the earlier subsets. They are not used for the completed means in Table 3 or for the separate direction experiment.

F Scope of Backend and Cross-Host Evaluation

Outer-loop portability. The historical repair evidence comes from mini-swe-agent; Table 1 additionally reports the author's 30-task Codex SDK cohort. An outer-loop attachment changes the observable state and intervention opportunities, even with the same skill text. The Codex aggregate supports feasibility in that configuration, but matched task-level outcomes, uncertainty estimates, and a separate judgment-usage breakdown are needed to assess robust transfer and mechanism-specific gains.

Jev judgment implementation. Figure 3 reports the completed Verified/MiniMax projection, while Table 1 also includes the 50-task DeepSeek cohort. Both report lower token totals with Jev than with Native or Target judgment. The backend comparison was not randomized, and model-specific tokens and differently denominated reference prices cannot establish a common monetary saving. The earlier common-33-task request ledger remains in Appendix B for provenance.

G Optional Skill Evolution

A future WikiSkill-based extension (Tang et al., 2026) would compare frozen and updated skills under the same routing and total budget, using source-disjoint test tasks or a prequential stream scored before each update. Post-update reruns cannot replace previously scored outcomes; rehearsal is adaptation to exposed tasks. No update benefit, update cost, regression-rate estimate, persistent RSI, or reduction in human oversight is evaluated here. Such results require their own protocol and cannot be inferred from fixed-skill runtime judgment.