The Goodhart Shift: Measuring Train–Holdout Divergence in Overnight Self-Evolving Agent Skill Loops

ThakiCloud AI Research (Hyojung Han)

Abstract

Unattended agent systems that modify their own skills, prompts, and routing rules overnight are increasingly deployed, and they typically report their progress against the same evaluation bench that drives their edits. When a measure becomes the target of optimization, Goodhart's law warns that it stops measuring what it was meant to. We study this failure mode for self-evolving agent harnesses and call it the *Goodhart shift*: the widening gap, night over night, between a loop's gains on the evaluation cases it can see and its gains on a sealed holdout it has never observed. We formalize the shift as a per-night train/holdout divergence, define the *case-flip* and, more sharply, the *silent case-flip*—a regression on the hidden holdout that occurs on a night the loop's self-report declares successful—and give a tractable edit-level model in which the expected divergence is governed by the fraction of edits that merely memorize the visible cases rather than transfer to unseen ones. We then specify a whole-loop train/holdout divergence gate that sits at the *evaluation layer*, complementing the deployment-layer canary and calibration-layer bandit guardrails we introduced in prior work, and turns a self-reporting overnight optimizer into a verifiable one: it tells apart "the loop improved" from "the loop overfit," protecting recorded baselines from churn-without-gain nights. We situate the contribution against the 2026 harness-design literature, which treats the harness as a static object to be improved, and against reward-hacking and over-optimization results, and we give the protocol in a form that can be dropped onto any unattended self-modifying agent loop.

1 Introduction

Modern LLM agent systems are less a single model than a model wrapped in a *harness*: a layer of skills, tool specifications, prompts, routing rules, and state that mediates every interaction with the environment. Over the past year a body of work has made the case that the harness, not the frozen model, is the dominant lever for unattended agent quality—harness-scaling runtimes that lift a fixed model to frontier-adjacent accuracy on terminal benchmarks [7], explicit task-state and manage-execute-audit loops for long-horizon work [8], programmable wrappers that reshape static environments while keeping their verifiers [9], and—most directly relevant here—harnesses that *improve themselves* by mining weaknesses, proposing minimal edits, and validating against regression tests [6, 10, 11].

The natural industrial end-state of this line of work is an *overnight self-evolution loop*: an unattended process that, while the operator sleeps, diagnoses yesterday's failures, edits the skill surface (descriptions, synonyms, routing artifacts), and ships the result to the next day's traffic. We operate exactly such a loop in production on a skill registry of roughly two thousand entries, where the loop rewrites skill descriptions and routing rules overnight and scores its progress against a fixed regression bench. What we ask in this paper is the question that loop engineering has so far left open: **when the loop optimizes against its own evaluation bench, do its self-reported gains generalize?**

This is a Goodhart problem. Charles Goodhart's observation—that "when a quantity becomes a target, it ceases to be a useful indicator" [21]—has a long pedigree in machine learning as *over-optimization* of a proxy reward: push a learned or fixed proxy hard enough and its score diverges from the true objective [18, 19]. Recent reward-hacking work makes the point operational, separating a cheap *proxy* signal (a test pass/fail) from an expensive *true* signal (task correctness) and showing how optimization leaks into the proxy [14], how conservatism paradoxically amplifies the gap under online adaptation [15], and how the whole phenomenon unifies as a *proxy–target divergence* under optimization pressure [16].

Our setting is specific and, to our knowledge, under-studied. The optimized object is not model weights but the *harness surface*, and the optimizer is the *same loop* that later reports the gains. Two structural facts make the risk concrete. First, the loop edits the very descriptions that its own retriever scores, so a description rewrite can raise the score on the queries it has already seen without improving routing on queries it has not—a surface-level memorization channel. Second, the loop's reward typically excludes part of the evaluation suite (e.g., the queries where *no* skill should fire), so a gain on the rewarded slice can coexist with a silent regression on the excluded slice; we observed exactly this in a prior calibration study, where the preferred configuration raised a rewarded metric while driving native-query hallucination from 0.0 to 0.4 [3].

Contributions. This is a *position and methodology* paper: we formalize the failure mode, analyze its structure, and specify a deployable measurement protocol; we do not report new longitudinal measurements, which are the subject of our ongoing production study. Concretely, we

1. **(Formalization.)** Define the *Goodhart shift* of an overnight self-evolving loop as a per-night *train/holdout divergence*, and define the *case-flip* and *silent case-flip* regressions that the loop's self-report structurally cannot see (Sec. 3).
2. **(Analysis.)** Give a tractable edit-level transfer/memorization model and prove that the expected divergence is proportional to the memorization fraction, so the divergence is a first-order detector of nights on which self-reported gains fail to generalize (Sec. 4).
3. **(Method.)** Specify a whole-loop *train/holdout divergence gate* at the evaluation layer, a sealed-holdout protocol, a frozen scoring path, and a no-edit control that isolates the effect of the surface edits; and show how it complements—rather than replaces—the deployment-layer canary and calibration-layer bandit guardrails from our prior work (Sec. 5).
4. **(Positioning.)** Locate the contribution against the 2026 harness-design literature, which treats the harness as a static object, and against over-optimization and reward-hacking results (Sec. 2, 6).

The remainder of the paper is organized as follows. Section 2 reviews related work and states precisely how this paper upgrades our immediately prior repair-loop study. Section 3 formalizes the loop, the split, and the divergence. Section 4 develops the structural analysis. Section 5 specifies the gate protocol. Section 6 discusses implications, and Section 7 concludes.

2 Related Work

2.1 Self-evolving agent harnesses (2026)

The 2026 harness-design literature is unified by a single move: treat the harness, not the model, as the unit of improvement. StateM freezes the model and scales the runtime—durable state, phase-local context, checked transitions, and versioned runbooks—reaching frontier-adjacent raw accuracy on Terminal-Bench at a small inference budget [7]. LongHorizon-Harness reformulates long-horizon execution as task-state management and adds a manage-execute-audit loop in which

a *read-only auditor* verifies environment state before the next round [8]. EnvHarness wraps static environments with plug-in components that reshape behavior while preserving the original verifier, and reports held-out gains [9]. Self-Harness closes the loop on the harness itself: it mines model-specific weaknesses, proposes minimal edits, and—critically—accepts a candidate edit only after regression testing, reporting improvements on both held-in and held-out pass rates across nine model–benchmark combinations [6]. Two further lines optimize the harness without ground-truth labels: retrospective self-preference over past trajectories [10] and recursive prompt-level refinement against the harness's own revision history [11].

Two features of this literature matter for us. *First*, it is overwhelmingly *one-shot and model-specific*: a harness is improved, validated at acceptance, and reported. The *longitudinal* object—a loop that keeps editing night after night against a fixed bench—is not the unit of study, and the question "do the *accumulated* self-reported gains still generalize after N nights?" is not asked. *Second*, even where validation exists, it is an *acceptance-time* check on a candidate edit, not a standing instrument on the loop as a whole. Our divergence gate is orthogonal: it does not propose or vet individual edits; it measures whether the *loop*, over time, is overfitting its own evaluator.

2.2 Our prior mechanism papers: the guardrail stack this work completes

We have, over recent months, built a stack of mechanisms around the same production overnight loop, and this paper is the explicit upgrade of the piece that started the line.

- **The repair loop (our immediately prior work).** We built a deterministic overnight loop that edits the retriever's cross-lingual synonym dictionary using a small compositional benchmark as its reward, and verified on a 63-case independent regression bench that previously working queries were unchanged [1]. That study was, in its own words, a "single-lever proof of concept," and its sixth stated limitation is the gap this paper closes: the regression bench was "confirmed post-hoc only, not yet used as a gate that blocks deployment." We upgrade that post-hoc check into a standing per-night train/holdout *divergence gate*—the loop's own evaluation becomes the object under surveillance.
- **The canary (deployment layer).** We inserted a progressive-traffic canary with a deterministic, trace-triggered rollback gate between "candidate produced" and "candidate serves all traffic," reducing blast radius and mean-time-to-rollback relative to unverified full rollout [2]. This guards the *deployment* of an edit after it is produced.
- **The bandit (calibration layer).** We tried online contextual-bandit calibration of the hybrid retrieval fusion parameters and reported a null headline result, but found the sharpest instance of the phenomenon we now study: a reward that sees only the positive queries drove the preferred configuration to raise a rewarded metric (top-1 $0.558 \rightarrow 0.581$) while silently raising native-query hallucination from 0.0 to 0.4 on the excluded slice [3]. A loop optimizing its own visible metrics can drift on the hidden ones—the silent case-flip, realized.
- **The quantized router (cost layer).** We characterized the accuracy–cost tradeoff of compressing the embedding half of the same hybrid router [4].
- **The pipeline audit (measurement layer).** We audited our own autonomous pipeline with operational logs rather than final artifacts, and defined a minimal guardrail set tied to observed failure modes [5].

Each of these guards a different layer—deployment, calibration, cost, and pipeline measurement—but *none validates the evaluator itself over time.* The divergence gate is that missing layer: it answers whether the loop's self-reported gains survive a holdout the loop has never seen. In one sentence, this paper adds to our immediately prior repair-loop study the instrument that its sixth limitation named but did not build.

2.3 Reward hacking and over-optimization

The theoretical backbone is over-optimization of a proxy. Gao et al. showed that, as one optimizes a proxy reward model harder, the true (gold) objective improves to a point and then *degrades*—the empirical form of Goodhart's law—with the onset scaling with proxy size [18]. Pan et al. showed the same failure *spontaneously* in iterative self-refinement, where generator and evaluator share a model and the evaluator's ratings improve while true quality stagnates or falls [19]. Countdown-Code gives a clean measurement instrument by separating proxy (test pass/fail) from true (correctness) reward and showing reward hacking emerging and generalizing under reinforcement learning [14]. Pessimism's Paradox quantifies the *Goodhart gap* and its area under the curve, finding that offline conservatism *monotonically increases* hacking damage under online adaptation [15]. EvalSafetyGap unifies benchmark validity, contamination, and reward/proxy optimization as a single *proxy–target divergence* under optimization pressure, formalized through a Goodhart-inspired instability decomposition [16]. Our contribution is to instantiate this machinery for a *self-modifying agent loop* and to turn the abstract "Goodhart gap" into a concrete, per-night, sealed-holdout *divergence* that an operator can gate on.

2.4 Skill routing and evaluation disclosure

On the routing side, compositional skill routing formalizes decompose–retrieve–compose and reports that task decomposition, not retrieval, is the binding constraint (standard decomposition reaches only 34.2% step-level category recall while the retriever is near-solved) [12], a claim our prior production study found to reverse in the mixed-language regime [1]. SkillWeaver establishes that agents can self-improve by discovering and honing reusable skills [20]. On the evaluation side, a recent audit of twelve agent-benchmark papers found that the scaffold, sampling settings, subset, and evaluator version are routinely under-disclosed, so that two runs of "the same" benchmark cannot be reconciled [17]—a direct warning about trusting any self-reported score, loop or otherwise. The divergence gate is a local, sealed answer to that warning: it does not ask the loop to disclose more; it asks the loop to stop being the only judge of itself.

Positioning. Self-Harness validates *edits*; the canary validates *rollouts*; the bandit validates *calibrations*. None validates the *loop's evaluation* over time. The Goodhart shift is the gap between those layers, and the train/holdout divergence gate is the instrument that closes it.

3 The Goodhart Shift, Formalized

The loop. Fix a skill registry $\mathcal{S}$ of size on the order of two thousand entries and a retriever ρ that maps a natural-language query to a ranked list of skills (in our production stack, a hybrid BM25/lexical plus dense-embedding router [3, 4]). The *skill surface* s is the editable part of the registry that ρ actually scores—descriptions, synonym mappings, routing rules—while the skill *bodies* are held fixed. An overnight loop is a (possibly stochastic) operator $\mathcal{L}$ that, given the current surface s_{n-1} and the cases on which it has observed scores, produces an edited surface $s_n = \mathcal{L}(s_{n-1})$. We index nights by $n = 1, \ldots, N$, and write s_0 for the pre-loop surface.

The sealed split. Let E be a fixed evaluation suite, disjoint from anything $\mathcal{L}$ is allowed to observe as a learning signal. In our setting E is the 63-case SRA skill-router regression bench and its associated routing golden set, composed of positive queries whose answer skill is present in the catalog, native queries on which no skill should fire, and adversarial negative queries [1, 3], scored

on five metrics (recall@k, gated recall, top-1 accuracy, hallucination rate, negative-avoidance rate). We partition it once, before night one, into

$$E \;=\; T \sqcup H, \qquad T \cap H = \varnothing, \tag{1}$$

where T is *loop-visible* (train): the cases whose scores feed $\mathcal{L}$'s diagnosis, its reward, and its self-report; and H is *sealed* (holdout): the cases excluded from every input the loop reads. Sealing is the load-bearing design choice; it is what makes H a proxy for the true objective in the over-optimization sense of [18, 14].

Scores and gains. Let $f_n(A)$ be the aggregate score of surface s_n on case-set $A \in \{T, H\}$, computed with a *frozen scoring path*: the shipped retriever configuration, the same code, no retraining, no re-weighting between $n-1$ and n. Freezing the scorer is essential: we want the measurement to isolate the effect of the surface edits, not of a moving evaluator. (Our prior bandit study is a caution: even with a fixed scorer, a loop that is allowed to tune the fusion parameters can move scores by changing the *instrument*, not the *artifact* [3].) The per-night gains are

$$g_n^T \;=\; f_n(T) - f_{n-1}(T), \qquad g_n^H \;=\; f_n(H) - f_{n-1}(H). \tag{2}$$

Definition 1 (Per-night divergence (Goodhart shift))**.** *The Goodhart shift at night n is the train/holdout gain divergence*

$$\delta_n \;=\; g_n^T \;-\; g_n^H, \tag{3}$$

and the cumulative shift through night N is $\Delta_N = \sum_{n=1}^{N} \delta_n = \big(f_N(T) - f_0(T)\big) - \big(f_N(H) - f_0(H)\big)$.

Because the split is fixed, Δ_N is path-independent: it depends only on the end surfaces s_0, s_N. A loop that *compounds real improvement* has $g_n^T \approx g_n^H > 0$ each night, so $\delta_n \approx 0$ and both scores rise together (Fig. 1, blue and green). A loop that *overfits its own bench* has $g_n^T > 0$ while g_n^H stalls or falls, so $\delta_n > 0$ and Δ_N accumulates (blue vs. red dashed).

Definition 2 (Case flips and silent case flips)**.** *A case flip at night n is a case $c \in E$ whose outcome (e.g., top-1 correctness, or firing on a native/negative query) changes between s_{n-1} and s_n. A flip is silent when it occurs on H while the loop-visible aggregate does not fall: $g_n^T \geq 0$ and some $c \in H$ flips adversely. Silent flips are exactly the regressions a self-report built on T cannot see by construction: the loop's night summary says "no degradation," and the hidden bench says otherwise.*

Definition 3 (Churn without gain)**.** *Night n exhibits churn without gain when the loop edits the surface ($s_n \neq s_{n-1}$) but $|g_n^T| < \epsilon$ and $|g_n^H| < \epsilon$ for a pre-registered tolerance ϵ (smaller than one case's contribution on either side). Churn without gain is compute spent with no measurable movement; repeated churn-without-gain nights are the regime in which a recorded baseline (such as our SRA floor of recall@5 $\geq$ 84.4%, top-1 $\geq$ 33.3%, and zero hallucinated injections [1, 3]) can drift under the noise of self-editing.*

Two regimes, one instrument. The definitions above make the central contrast operational. Both regimes look identical in the loop's self-report, which is a function only of T: a night that memorizes T reports the same "improvement" as a night that transfers it. The sealed holdout is the only channel that separates "the loop improved" from "the loop overfit," and δ_n together with the silent-flip count is the minimal statistic that does so.

4 Why Self-Reporting Loops Diverge: A Structural Analysis

We now make the divergence mechanism explicit with a deliberately simple model, whose purpose is to identify *which quantities control δ_n*, not to fit data.

Edit-level transfer model. Suppose night n applies $e_n \geq 1$ independent edits to the surface. Each edit is of one of two kinds:

- a *transfer* edit, with probability p_n: it improves routing quality on queries the loop has not seen (e.g., fixes a vocabulary gap that generalizes across paraphrases), contributing in expectation $+a$ to *both* g_n^T and g_n^H;
- a *memorization* edit, with probability $1 - p_n$: it improves only the visible cases it was derived from (e.g., a description rewrite that matches the specific train query wording), contributing $+a$ to g_n^T and 0 to g_n^H.

Here $a > 0$ is the expected per-edit score increment on the side it affects, and $p_n \in [0, 1]$ is the night's *transfer rate*.

Proposition 1. *Under the edit-level transfer model,*

(a) $\mathbb{E}[\delta_n] = a\, e_n\, (1 - p_n)$*;*

(b) for an active night ($a\, e_n > 0$)*,* $\mathbb{E}[\delta_n] = 0$ *if and only if* $p_n = 1$ *(every edit transfers);*

(c) a night with $1 - p_n > 0$ *admits a positive probability of a silent holdout flip, while a night with* $p_n = 1$ *does not, to first order; hence the per-night divergence (with the silent-flip count as a tie-breaker) is a first-order detector of non-generalizing nights.*

Proof. (a) By linearity, $\mathbb{E}[g_n^T] = ae_n\big(p_n + (1 - p_n)\big) = ae_n$, since both edit kinds raise the visible aggregate, while $\mathbb{E}[g_n^H] = ae_n\, p_n$, since only transfer edits act on the sealed side. Subtracting gives $\mathbb{E}[\delta_n] = ae_n(1 - p_n)$. (b) With $ae_n > 0$, $\mathbb{E}[\delta_n] = 0$ iff $1 - p_n = 0$ iff $p_n = 1$. (c) A memorization edit, by construction, moves the loop-visible cases it targets and leaves the holdout unaffected in expectation; because the holdout is finite and sealed, any edit whose benefit is confined to specific visible cases leaves the holdout score at best unchanged in expectation while the visible score rises, so $g_n^T \geq 0$ on exactly the nights in which a holdout case can flip adversely without any visible signal—i.e., silently. When $p_n = 1$, every edit acts symmetrically on both sides in expectation, so no first-order silent-flip channel exists. □ □

Three observations follow.

First, the divergence is not a nuisance statistic—it is the estimand of interest. $\mathbb{E}[\delta_n]$ is proportional to the memorization mass $e_n(1 - p_n)$. A loop that keeps editing (e_n large) but whose edits increasingly target the visible cases (p_n falling) shows a rising δ_n even when both absolute scores look healthy. This matches the over-optimization onset of [18] and the monotonic Goodhart-gap growth of [15], transplanted from proxy rewards to surface edits.

Second, the memorization channel is structurally available in a retriever loop. The loop edits the descriptions that the *same* lexical retriever scores. A rewrite that embeds the wording of a train query into a skill description raises that query's score by construction; whether it helps any unseen query depends on the corpus-level score distribution, not on the token itself. Our prior repair-loop study found precisely this: even correctly applied dictionary patches failed to move some retrievals, because BM25/IDF ranking is a property of the whole corpus—vocabulary coverage is necessary but not sufficient [1]. Symmetrically, surface edits can be *too* targeted: they can fit T without touching the routing of unseen queries, which is the $1 - p_n$ term.

Third, excluded slices are silent-flip factories. If the loop's reward omits part of E—as a positive-only reward omits the native and adversarial-negative queries [3]—then edits are *systematically* biased toward the rewarded slice: p_n is lower on the excluded cases than on the rewarded ones, and the excluded side flips first. This is not a hypothetical; our bandit calibration realized it, with native hallucination rising from 0.0 to 0.4 while the rewarded top-1 rate improved [3]. The divergence gate's job is to make every slice of E count, every night.

Remark 1 (When does churn compound instead of drift?)**.** *The model separates the regimes cleanly: compounding real improvement is the* $p_n \to 1$ *regime (*$\delta_n \to 0$*, both sides rising); Goodhart*

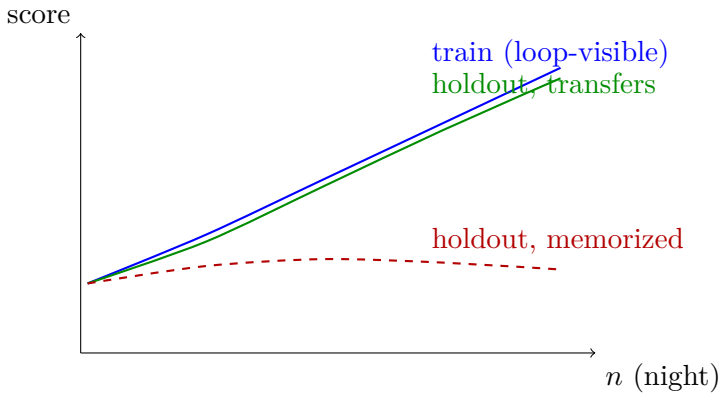

Figure 1: Schematic contrast of the two regimes an overnight self-evolving loop can exhibit. Blue: the loop-visible train score. Green: the sealed holdout score when edits transfer—compounding real improvement, with $\delta_n \approx 0$. Red dashed: the holdout score when edits memorize the visible cases—Goodhart drift, with $\delta_n > 0$ accumulating and silent flips likely. Illustrative; not measured data.

drift is the accumulating $\sum_n ae_n(1 - p_n)$ *regime (visible side rising, sealed side flat). Empirically discriminating the two requires exactly the sealed-holdout measurement of Section 5; the self-report is blind to it by construction.*

5 The Train/Holdout Divergence Gate

We now specify the measurement protocol that turns the definitions above into a standing instrument for an unattended loop. The design principle is the one our repair-loop study already validated in spirit: judge the loop by an external signal—bench outcomes—not by its self-report [1, 2].

Protocol (per-night divergence gate).

1. **Seal.** Before night one, partition $E = T \sqcup H$ and write H to a location the loop (and its prompt, logs, and reward channel) cannot read. In our stack the natural choice is to reserve roughly one fifth of the 63-case bench as H (on the order of 15 sealed cases) and to keep the remainder as T, stratified so that positive, native, and negative slices are all present on the sealed side—exactly the slices a positive-only reward would otherwise ignore [3].
2. **Edit.** The loop runs its normal overnight pass, producing s_n from s_{n-1}. Nothing about the loop changes; the gate observes, it does not steer.
3. **Score with the frozen path.** Score s_n on both T and H using the shipped retriever configuration and code path, with no retraining or re-weighting. Recompute g_n^T, g_n^H, and δ_n; enumerate case flips on each side; count silent flips (adverse H flips with $g_n^T \geq 0$).
4. **Control.** Score a no-edit control: the unmodified s_{n-1} through the identical pipeline, to certify that the scoring path itself is stable from night to night (so that any movement is attributable to the edits, not to the instrument). We report the control's movement as the noise floor η_n.
5. **Gate.** With pre-registered thresholds τ_δ, τ_f (chosen relative to η_n and to one case's contribution on each side—with $|H| \approx 15$, a single sealed flip moves the holdout aggregate by ≈ 6.7 points, so thresholds below that are uninterpretable as single-night signals):

- if $|\delta_n| \leq \max(\tau_\delta, \eta_n)$ and no silent flip: *accept* the night's self-reported gain as generalizing; record it against the baseline;
- if $\delta_n > \max(\tau_\delta, \eta_n)$ or a silent flip: *flag* the night—the claimed gain is not certified, the baseline stands, and the loop's self-report for that night is marked *unverified* for downstream consumption;
- if $|g_n^T| < \epsilon$ and $|g_n^H| < \epsilon$ with $s_n \neq s_{n-1}$: *record churn without gain* and count it toward the loop's compute-spend audit.

6. **Hand off to the deployment layer.** A flagged night does not auto-rollback; it routes to the existing canary-with-rollback machinery [2], which bounds the blast radius while a human or the loop inspects the flagged edits. The two layers answer different questions—the gate asks "did the improvement generalize?", the canary asks "if it did not, how much traffic saw it?"

Why the holdout stays sealed. Sealing is what converts δ_n from a noisy split comparison into an estimand of overfitting. If the loop can see H scores—even indirectly, through a self-report that summarizes them—then H becomes part of the optimization target, the memorization channel of Section 4 extends to it, and the divergence loses its meaning; this is the benchmark-targeting failure mode of Goodhart's law [21, 18] and the contamination concern of the evaluation-safety literature [16]. The practical rule is strict: H is read only by the gate, never by the loop, and is rotated on a long schedule (e.g., re-sealed from fresh traffic-labeled cases periodically) so that the sealed set itself does not age into the loop's effective training distribution.

Cost. The gate scores the bench once per night on the frozen lexical path—a local, deterministic computation that is cheap relative to the loop's own work [4]—plus one no-edit control run. It adds no model calls if the loop's reward is bench-based. Its value is asymmetric: it costs a trivial amount and it is the only instrument that can distinguish, night by night, a compounding loop from a drifting one.

Statistical hygiene. Three cautions from our prior measurement work are built in. *(i) Denominator control:* cases whose answer skill is renamed or removed from the catalog change the denominator of rate metrics and can fake a "gain" [3, 5]; the gate reports case-count changes alongside scores. *(ii) Tolerance over point estimates:* with small sealed sets, single-night δ_n is noisy; the decision uses δ_n against the control noise floor η_n and, over windows, the cumulative Δ_N, which is path-independent. *(iii) Metric parity:* all five SRA metrics—including the hallucination and negative-avoidance rates that a positive-only reward omits [3]—enter the gate, so the silent-flip factory of Section 4 is closed rather than monitored afterward.

6 Discussion

The loop as a whole, not the edit as a unit. Self-Harness-style validation accepts each edit only after regression testing and reports held-in and held-out gains per iteration [6]; EnvHarness validates reshaped environments on fresh rollouts [9]; Countdown-Code measures hacking by separating proxy from true reward [14]. All are *per-edit* or *per-run* instruments. The Goodhart shift is a *per-loop, longitudinal* quantity: even if every individual edit passes its acceptance test, the accumulated surface after N nights can be overfit to T in ways no single edit was. This is the same distinction as between a model's generalization gap and its per-example test errors—and it is why a divergence gate on the loop is complementary to, not redundant with, acceptance-time validation.

What the gate can and cannot certify. The gate certifies *generalization within the distribution of E*: a flagged night means the edits moved the visible side more than the sealed side *on the bench*. It does not certify improvement on unbench traffic, which remains the deployment canary's job [2], and it does not certify that the bench itself is still a valid proxy for production quality—a second-order Goodhart problem in which the bench ages while traffic drifts, addressed by periodic re-sealing of H from fresh labeled cases. Within its scope, however, it answers exactly the trust question of Section 1: whether the loop's self-reported gains survive a holdout the loop has never seen.

Implications for unattended automation. Three consequences matter for operators of self-modifying systems. *Trust:* a loop without a sealed holdout can report upward while capability silently drifts—the realized failure in our bandit calibration [3] is the template. *Compute and energy:* churn-without-gain nights are pure spend; a gate that records them turns "the loop ran all night" into a measurable audit line, in the spirit of cost-aware evaluation practice [17, 5]. *Baselines:* recorded floors such as our SRA recall/top-1/hallucination baselines [1, 3] are only as trustworthy as the last night certified by a sealed holdout; the gate is what makes a baseline a *gate* rather than a memory.

Relation to the 2026 harness literature. The harness-design papers establish that the harness is the lever [7, 8, 9] and that it can even be the learner [6, 10, 11]; the harness's *safety* is benchmarked across operational lifecycle phases [13]. None of this line asks whether the learner's *self-assessment* degrades as learning proceeds—the over-optimization question [18, 19] applied to harness self-evolution. The divergence gate is a candidate answer, and it is deliberately minimal: one sealed split, one frozen scoring path, one control run, one decision per night.

Threats to validity and limitations. We list them plainly. *Single organization, single corpus:* all settings here come from one production stack; transfer to other skill libraries is untested. *Finite sealed set:* with ≈ 15 holdout cases, single-night divergence is dominated by one-case granularity, so the gate's power is in windows and in flip counts, not in point estimates; re-sealing from traffic mitigates but does not remove this. *Frozen scorer assumption:* the gate isolates surface effects only while the scoring path is held fixed; a loop that co-adapts its retriever parameters (as in [3]) needs the calibration layer's own instruments on top of the gate. *Model abstraction:* the transfer/memorization model of Section 4 is a control-quantity model, not a fitted description of any real loop. *Analytical scope:* this paper specifies and analyzes the protocol; the longitudinal measurements it enables are the subject of our ongoing production study, and we claim nothing beyond the protocol and the structural analysis.

The rendered figures for this work (Figures 2, 3, 4) are collected here.

7 Conclusion

An overnight self-evolving loop that edits the surface its own retriever scores is an optimizer whose evaluation and its objective increasingly share a distribution—and Goodhart's law says what happens next. We formalized the failure as the *Goodhart shift*: a per-night train/holdout divergence, with silent case flips as its sharpest symptom, and showed that under a simple edit-level transfer model the divergence is exactly the memorization mass of the loop's edits. We then specified the instrument that detects it—a sealed holdout, a frozen scoring path, a no-edit control, and a one-line decision per night—and located it in the guardrail stack we have been building around the same production loop: it validates the *evaluator* where the canary validates the *rollout* and the bandit validates the *calibration*. The contribution is a general one: for any unattended self-modifying

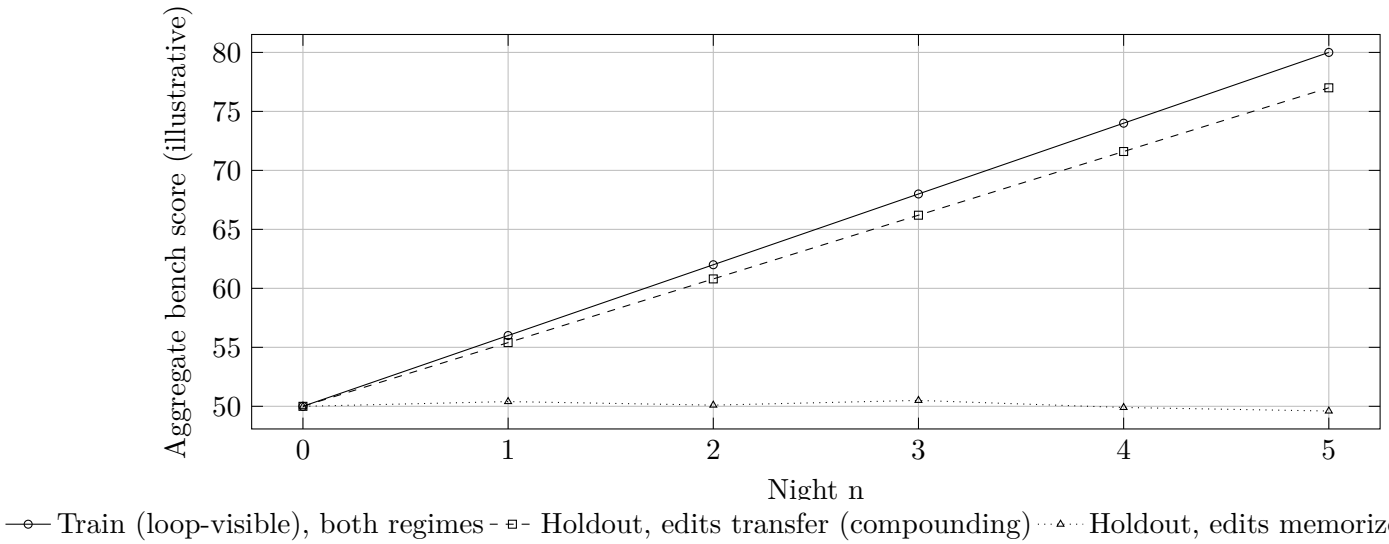

Figure 2: The loop-visible train score rises in both regimes, but only when edits transfer does the sealed holdout score follow with near-zero per-night divergence, whereas when edits memorize the visible cases the holdout stalls or falls and the divergence accumulates night over night. (Analytical model (not measured))

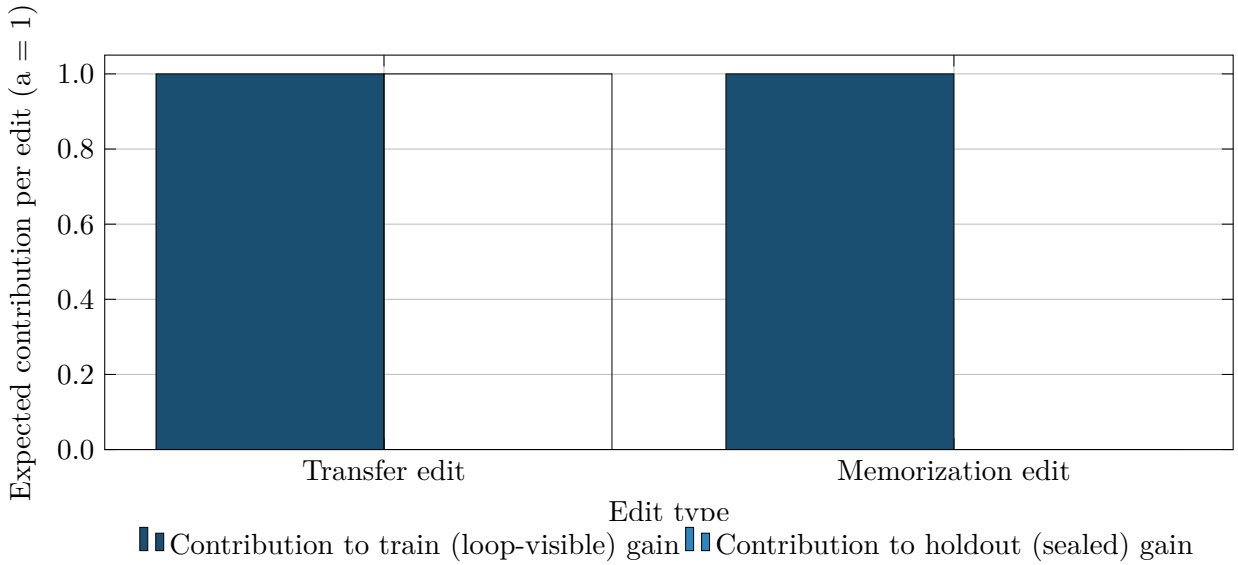

Figure 3: Under the paper's edit-level transfer model, a transfer edit raises the train and the sealed holdout sides equally while a memorization edit raises only the train side, which is why the expected per-night divergence is governed by the memorization share of edits. (Analytical model (not measured))

Figure 4: The evaluation-layer gate seals the holdout before night one, observes the loop edit unchanged, scores both sides on a frozen path, runs a no-edit control to set the noise floor, then issues one of three decisions per night and hands flagged nights to the deployment-layer canary. (Conceptual example)

agent system, the question "did it improve?" is only answerable by a holdout it was never allowed to see.

References

[1] ThakiCloud AI Research (H. Han). *Closing the loop on compositional skill routing: A nightly autonomous repair loop for agent harnesses*. 2026. https://thakicloud.github.io/ko/research/autonomous-skill-router-repair-loop/

[2] ThakiCloud AI Research (H. Han). *Canary-gated autonomous skill rollout: Trace-triggered regression detection and automatic rollback in self-evolving agent harnesses*. 2026. https://thakicloud.github.io/ko/research/canary-gated-skill-rollout/

[3] ThakiCloud AI Research (H. Han). *Online bandit calibration of hybrid retrieval weights for self-evolving skill routers*. 2026. https://thakicloud.com/tech-blog/ko/research/bandit-hybrid-retrieval-calibration/

[4] ThakiCloud AI Research (H. Han). *Quantizing the gatekeeper: Accuracy–cost tradeoffs of compressing the embedding model in hybrid skill routers*. 2026. https://thakicloud.com/tech-blog/ko/research/quantized-embedding-skill-router/

[5] ThakiCloud AI Research (H. Han). *Auditing the reliability of a nightly autonomous LLM research pipeline: Diversity, reproducibility, and research-integrity guardrails*. 2026. https://thakicloud.github.io/ko/research/autonomous-research-pipeline-reliability/

[6] H. Zhang, S. Zhang, K. Li, C. Zhang, Y. Chen, Y. Zhang, L. Bai, et al. *Self-Harness: Harnesses that improve themselves*. arXiv:2606.09498, 2026.

[7] Z. Qin, Y. Lu, Z. Atlas Wang, K. Wang. *StateM: Reaching 95.3% raw accuracy, or a $15 frontier run, on Terminal-Bench 2.1 via harness scaling*. arXiv:2608.15089, 2026.

[8] Z. Ma, H. Huang, S. Zou, Y. Wang, S. Yang, Y. Hu, F. Wei, X. Chu. *LongHorizon-Harness: Advancing long-horizon agents for real-world tasks*. arXiv:2608.01964, 2026.

[9] C. Huang, Z. Wang, R. Han, J. Yan, Y. Chen, Z. CuiZhu, K. Jiang, P. Xia, et al. *EnvHarness: Awakening static worlds for agent learning*. arXiv:2608.19880, 2026.

[10] W. Pan, S. Liu, C.-Y. Lin, J. Zeng, X. Tang, X. Zhou, Y. Lu, X. Jia. *Evolving agents in the dark: Retrospective harness optimization via self-preference*. arXiv:2606.05922, 2026.

[11] H. Lee, J. Xu, J. Seely, D. Lee, M. Zaharia, Y. Tang. *Recursive harness self-improvement*. arXiv:2607.15524, 2026.

[12] X. Gao. *Compositional skill routing for LLM agents: Decompose, retrieve, and compose*. arXiv:2606.18051, 2026.

[13] Y. Bai, J. Duan, J. Peng, X. Wu, S. Liu, S. Wang, T. Chen. *HarnessRisk: A lifecycle-oriented benchmark for agent harness safety*. arXiv:2608.17597, 2026.

[14] M. Khalifa, Z. Khan, O. Tafveez, H. Peng, L. Wang. *Countdown-Code: A testbed for studying the emergence and generalization of reward hacking in RLVR*. arXiv:2603.07084, 2026.

[15] S. Sahoo, A. Chadha, V. Jain, D. Chaudhary. *Pessimism's paradox: Conservative offline training amplifies reward hacking during online adaptation in reasoning models*. arXiv:2606.30627, 2026.

[16] B. A. Ulurmak, R. Kurban. *EvalSafetyGap: A hybrid survey and conceptual framework for LLM evaluation–safety failures*. arXiv:2606.30219, 2026.

[17] M. N. Moghadasi, F. Ghaderi. *What twelve LLM agent benchmark papers disclose about themselves: A pilot audit and an open scoring schema*. arXiv:2605.21404, 2026.

[18] L. Gao, J. Schulman, J. Hilton. *Scaling laws for reward model overoptimization*. arXiv:2210.10760, 2022.

[19] J. Pan, H. He, S. R. Bowman, S. Feng. *Spontaneous reward hacking in iterative self-refinement*. arXiv:2407.04549, 2024.

[20] B. Zheng, M. Y. Fatemi, X. Jin, Z. Wang, A. Gandhi, Y. Song, Y. Gu, et al. *SkillWeaver: Web agents can self-improve by discovering and honing skills*. arXiv:2504.07079, 2025.

[21] C. A. E. Goodhart. *Problems of monetary management: The United Kingdom*. Bank of England Monographic Documents, 1975.