LEARNING BEFORE FAILURE: PREVENTING FAILURE BY LEARNING WHAT WENT WRONG AND WHY

Anonymous authors
Paper under double-blind review

ABSTRACT

When people learn from a failure, whether their own or someone else's, a useful lesson captures why it happened, what correction the evidence supports, where that correction applies, and how soon they must act. Can explicitly checking when a lesson applies help turn prior experience into timely preventive action? We introduce *Learning Before Failure* (LBF), a framework for turning stored lessons into timely decisions in language agents, and test one implementation, *LBF-Gate*. Audits supply the explanations and applicability fields; we evaluate their subsequent use, not autonomous diagnosis. Before each decision, the gate checks lessons against the current situation and marks applicable lessons as rules to follow. The full lesson bank remains readable, and the language model chooses the action and its timing. In a constructed flood search-and-rescue environment, we compare LBF-Gate with an agent that reads the same structured lessons unmarked and with one that stores model-written free-text reflections on prior failures. The first comparison holds lesson content fixed to distinguish what is remembered from how it is used. Model weights and stored lessons remain frozen during testing on new missions. Across 375 independent blocks per recurrence class per model, LBF-Gate reduces normalized held-out loss against the same lessons unmarked by 0.108 on GPT-5.4 mini and 0.087 on Claude Sonnet 4.6. Every class-specific lower bound is positive on both models, and loss increases on cases requiring a different correction or no intervention remain within the prespecified tolerance. LBF-Gate also outperforms free-text reflection. A separately prespecified control study shows that marking the wrong lesson substantially reduces the benefit; on GPT-5.4 mini, wrong marking performs worse than leaving lessons unmarked. The gains cannot be explained by shorter inputs or by marking an arbitrary lesson. Preventive reuse must therefore be evaluated both for timely action on relevant lessons and for the consequences of applying lessons where they do not belong.

1 INTRODUCTION

When people learn from a mistake, they do more than remember what happened. They ask why it happened and use that explanation to work out how to respond differently. A useful lesson goes further: it helps them recognize a similar problem before the same harm occurs, whether the mistake was their own or someone else's. We focus on this later-use step: given lessons from earlier failures, how should a language agent (an AI system that uses a large language model to plan and act) receive them so that they lead to the right action before another failure?

Consider three familiar situations. An opioid medicine may initially relieve pain but become less effective as tolerance develops. Tolerance is not itself addiction, yet earlier cases of misuse and addiction can inform a clinician's reassessment of benefits and risks before further harm occurs (Dowell et al., 2022). A parent may meet every work deadline while repeatedly postponing time with a child. Other families' experiences could encourage renewed attention before reduced communication becomes a crisis. A machine may meet its production target while a bearing wears down. Lessons from earlier breakdowns can connect a vibration pattern to inspection during planned downtime, before another failure interrupts production (Jardine et al., 2006).

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.

In each case, the immediate outcome can still look acceptable while opportunities for prevention shrink. Later repair may restore operation without recovering lost time, capacity, or trust. Yet acting immediately is not always the answer: applying a lesson to the wrong situation can create another problem. The challenge is to recognize when an earlier experience is relevant, carry the fix it supports into the current decision, and complete the response while it can still help.

In this paper, we study this problem using a running example: flood search and rescue (SAR). A convoy is still reaching people, but its predicted safety margin is shrinking. An advancing flood crest (the peak of the flood) may require rerouting; a stale report calls for checking the report rather than assuming the road is unsafe; a degrading vehicle calls for replacement. Audits, which are reviews of earlier mission failures, explain what went wrong and why, and which fix the evidence supports. Later, the planner (the language model that decides the convoy's next action) must use the right lesson under new resource and timing limits. Waiting too long can forfeit a rescue, while an unnecessary reroute, or a fix applied to a report that is still fresh, can waste scarce capacity.

We introduce *Learning Before Failure* (LBF), a framework for turning stored lessons into timely decisions. In LBF, each lesson records six things: the warning sign to watch for, the situations it covers, the fix the evidence supports, the part of the system responsible for acting, the deadline for completing the fix, and the audit evidence that explains why the earlier failure happened. Storing such a lesson is not enough. It must also be kept, reach the part of the agent that makes the decision, be used only where it fits, and be used in time. We call a lesson that meets all four requirements *operative*: it actually shapes the decision. LBF judges success in two ways: a lesson should help when the same kind of problem returns, and keep harm within a specified tolerance in look-alike situations where it does not apply.

We test one implementation, *LBF-Gate*. The model first picks the fix an audit supports, and a program attaches the audit's warning sign, covered situations, and timing to it. We test the use of these supplied explanations and applicability fields, not autonomous diagnosis. On a later mission, a gate, a checkpoint before each decision, compares every lesson's warning sign and covered situations with what the agent currently observes. Matching lessons are marked as rules the agent must follow. Nothing is hidden: the agent still reads every lesson, and the language model still chooses what to do and when to start.

Many systems already store reflections or memories of past experience (Shinn et al., 2023; Liu et al., 2025; Shen et al., 2026). To separate what is learned from how it is used, we give two agents exactly the same lessons: LBF-Gate, and an active-memory agent that reads the whole set without marks. A third agent keeps free-text notes the model wrote about the same audits (reflection). The model and its lessons are frozen before testing on new missions with different road layouts, timing, identities, and resources.

Across two models, GPT-5.4 mini and Claude Sonnet 4.6, and 162,000 planned decisions, we measure loss on new missions on a scale from 0 (best) to 1 (worst). Marking the matching lessons lowers loss relative to the same lessons unmarked by 0.108 on GPT and 0.087 on Claude, and even the cautious lower-bound estimate shows a gain for every kind of repeated problem on both models. LBF-Gate also beats reflection and stays within the limits on harm in look-alike cases that we set before the experiment. A second, separately planned experiment then marks a wrong lesson instead. Wrong marks substantially reduce the advantage: Claude performs close to its unmarked level, while GPT hands most decisions to a human instead of acting and does worse than with no marks at all. After defining the framework (Section 3) and the rescue task (Section 4), we ask five questions: does marking help on new missions (RQ1), what does it cost in look-alike cases (RQ2), must the marked lesson be the right one (RQ3), what does a wrong mark cost (RQ4), and which parts of the design matter (RQ5)?

2 Related Work

Prevention and reusable rules. Sequential detection and condition-based maintenance study the costs of delay, information, and intervention (Page, 1954; Lorden, 1971; Lai, 2001; Jardine et al., 2006; Ruiz-Hernandez et al., 2020). Case-based reasoning, explanation-based generalization, and Soar chunking turn experience into reusable rules (Aamodt & Plaza, 1994; Mitchell et al., 1986; Laird et al., 1986). LBF builds on these ideas to evaluate how a language agent uses a supported

correction in later decisions. The prevention window is a task constraint here; we do not propose a new theory of optimal intervention.

Language agents that learn from experience. Reflexion retains verbal feedback; ExpeL extracts insights; contextual experience replay synthesizes experience (Shinn et al., 2023; Zhao et al., 2024; Liu et al., 2025). Agent Workflow Memory induces routines, Voyager stores executable skills, and Experiential Reflective Learning studies selective retrieval (Wang et al., 2024; 2023; Allard et al., 2026). Our REFLECT-DIAG arm stores free-text operational notes. The typed comparison instead holds the acquired records fixed to test whether their presentation changes later behavior.

Memory use and retrieval. MemGPT and A-MEM manage external memory; LoCoMo measures conversational memory; Mem2ActBench evaluates memory-driven action (Packer et al., 2023; Xu et al., 2025; Maharana et al., 2024; Shen et al., 2026). Evidence position affects context use (Liu et al., 2024), and retrieval can change which records the planner sees. We examine BM25 in a descriptive panel (Robertson & Zaragoza, 2009); dense retrieval (Karpukhin et al., 2020) and learned selectors remain untested. LBF evaluates both useful application and over-application of memory under the same frozen evaluation protocol.

Learning and source support. Reinforcement and continual learning can update policies or parameters (Sutton & Barto, 2018; Lopez-Paz & Ranzato, 2017). ERM/RLER audits causal commitments and accumulates interventional evidence for external adaptation (Anonymous, 2026). Our experiment takes the audit's explanation and supported correction as inputs and evaluates their later use. A corrupted-audit panel measures dependence on those inputs. Appendix I develops these comparisons.

3 THE LBF FRAMEWORK

3.1 WHAT A LESSON CONTAINS AND HOW THE PLANNER RECEIVES IT

A deployed agent has fixed model parameters ω, persistent mission state, and a fixed task contract. Its persistent state contains a bank $\mathcal{B}$ of lessons. A typed lesson is

$$L = (g, S, u, v, d, \mathcal{P}), \tag{1}$$

where g is an observable trigger, S its scope, u a supported correction, v the responsible component, d a completion-time rule, and $\mathcal{P}$ the source evidence and provenance. The audit explains the source failure and links the correction and applicability fields to evidence. The agent receives conservative timing information from the source; the true evaluation deadline remains private. Two cases are *correction-equivalent* when their supported correction, reusable timing convention, and responsible component agree (Appendix K).

Before a lesson enters the bank, an admission check verifies that its content follows the supplied audit. This establishes source fidelity, not the truth of the audit itself. The clean study supplies audits supporting the intended correction; the corruption panel changes some of those assertions. A lesson can therefore be faithful to its source yet be wrong, or be correct yet fail to change later behavior (Appendix H).

For a live case x, a mechanism r prepares the memory context $r(\mathcal{B}, x)$ for the planner:

$$(a, t_a) \sim \pi_\omega\big(\cdot \mid x, r(\mathcal{B}, x)\big). \tag{2}$$

The planner returns an action a and start time t_a. The mechanism may present the full bank, mark applicable records within that bank, or retrieve a smaller set. Our typed comparison changes r while holding $\mathcal{B}$ and ω fixed.

LBF-Gate is the TYPED-GATED arm in the tables and artifacts. It evaluates applicability predicates and adds matching records as constraints without removing any readable record. Active memory receives the same bank without those constraints. By contrast, the BM25 control in Section 5.7 changes the readable set through retrieval.

3.2 FOUR REQUIREMENTS FOR LESSON USE

Storage retains a lesson; retrieval selects it; installation connects it to a decision interface; execution produces an action. LBF requires *persistence*, so the lesson survives; *reachability*, so the acting component can read it; *applicability*, so its scope and trigger determine when it should be used; and *timeliness*, so the correction can finish before prevention becomes impossible.

These requirements guide the evaluation. Memory is frozen after construction to ensure persistence. Placement controls vary access by the acting component. T5_SCOPE and T5_TRIGGER test cases outside a source lesson's scope or trigger, and BENIGN tests unnecessary intervention. Completion scoring, timing removal, and an exact deadline sweep examine timeliness. The main study tests these features together; the component controls are descriptive. Appendix E gives the supporting state-pathway propositions.

3.3 WHAT COUNTS AS SUCCESSFUL USE

Let $R^r_{m,c}$ be expected bounded mission loss for model m, mechanism r, and held-out recurrence class c. This expectation includes source-bank construction and later decisions, including construction failures. Writing $R_c(s)$ for class risk at persistent state s under a fixed evaluation convention distinguishes acquiring a lesson from using it:

$$\underbrace{R_c(s^-) - R_c(s^+)}_{\text{acquiring an update}} \quad \text{versus} \quad \underbrace{R^{\text{active}}_{m,c} - R^{\text{gated}}_{m,c}}_{\text{using the same bank differently (measured)}} . \tag{3}$$

We estimate the right-hand contrast. Both typed arms already contain the acquired bank. For comparator $b \in \{\text{active}, \text{reflection}\}$, define

$$\Delta^b_{m,c} = R^b_{m,c} - R^{\text{gated}}_{m,c}, \qquad \Gamma^b_{m,p} = R^{\text{gated}}_{m,p} - R^b_{m,p}, \tag{4}$$

where p indexes a protected population. Positive Δ means lower transfer loss with gating; positive Γ means greater loss on a protected population.

The primary success criterion requires positive transfer in every prespecified recurrence class on each model, with protected-case degradation below the relative-loss tolerance $\eta = 0.10$. The frozen inference procedure determines whether both requirements pass. New cases vary the mission conditions while the acquired bank remains fixed, testing reuse rather than further learning during evaluation.

Algorithm 1 LBF-Gate: gated lesson use

Require: Audited source bank $\mathcal{D}$; fixed model ω; byte limit $b_{\max}$; routing configuration κ.
Ensure: A persistent bank $\mathcal{B}$ or construction failure; a raw planner response for each later case.
Phase A: construct once from source episodes
Obtain model-selected corrections U for all assigned source slots in $\mathcal{D}$.
$\mathcal{B} \leftarrow \text{Bind}(U, \mathcal{D})$: bind $g, S, u, v, d, \mathcal{P}$ to the supporting audits.
if construction is incomplete, unsupported, invalid, or its encoded bytes exceed $b_{\max}$ **then**
    return $\perp$ ▷ whole-bank failure; retain its planned evaluation slots
end if
Persist the canonical bank and its hash; make $\mathcal{B}$ read-only for evaluation.
Phase B: apply without updating the acquired bank
for each fresh case with admitted live input x **do**
    $M \leftarrow \{L \in \mathcal{B} : \text{Match}_\kappa(L, x)\}$: component, ordered scope, and both predicates.
    $E \leftarrow \text{Present}_\kappa(\mathcal{B}, M, x)$: expose all of $\mathcal{B}$, and M as the constraint list.
    $\hat{y} \leftarrow \text{Planner}_\omega(x, E)$; a valid raw response specifies (a, t_a).
    Pass $\hat{y}$ unchanged to the fixed evaluation interface; retain $\mathcal{B}$.
end for

3.4 LBF-GATE IMPLEMENTATION

Algorithm 1 separates construction from evaluation. Its fixed configuration κ provides two functions. Match_κ requires an exact match of the acting component and ordered scope, plus satisfaction of both trigger predicates. Each predicate compares a public observed feature with a bound using inclusive $\geq$ or $\leq$; a missing feature fails the check.

Present_κ supplies the full readable bank and adds the matched records, in stored order, to the acting component's constraint list (`registered_constraints`). Unmatched records remain readable. Timing accompanies the correction as record information; it is not an additional numerical eligibility check. The planner chooses when to start, and the evaluator separately judges completion. Appendix H.3 traces the implemented rule from source fields to one scored decision. Other lesson-use mechanisms can replace r without adopting this decomposition.

4 Flood Search-and-Rescue Task

4.1 Constructing a lesson from an audited failure

We evaluate LBF-Gate in the constructed environment CT5-LBF-SAR-01. Its three recurrence classes share a shrinking margin for safe traversal but require different corrections. Crest/model lag calls for rerouting, a stale report calls for quarantine and verification, and resource degradation calls for substitution. A benign transient requires no unnecessary correction.

Consider an audit that attributes a failure to a stale report without establishing that the route itself is unsafe. During construction, the model selects a correction. The constructor binds this choice to the audit's fields and checks that it is uniquely supported and faithfully represented (Algorithm 1, lines 1–3); Table 17 in Appendix H maps the example to the lesson tuple.

Valid banks are stored and frozen before evaluation (line 6). An unsupported choice, missing construction page, invalid output, or byte overflow fails the whole bank. All its planned evaluation slots remain in the study. Appendix A gives the construction and validation procedure; Appendix G.2 documents the checks supporting the frozen-state claim.

4.2 Applying the lesson on a new mission

The planner receives live observations, resource and human-service availability, and conservative source timing information. LBF-Gate checks applicability (line 8) and presents the whole bank with matching records in its constraint list (line 9). The unchanged LLM then chooses an action and start time (line 10). A matching stale-report lesson supports checking the report instead of assuming that the route is unsafe. The active-memory comparator can read the same lesson but receives no such marking.

Applicability is checked over public symbolic fields supplied by the generator, rather than inferred from natural-language descriptions; Appendix H.3 traces one saved case from its source fields to the scored decision.

T3 tests new instances of the original recurrence rule. Protected tiers test cases where the lesson should not guide the decision: a fresh report falls outside the stale-report trigger (T5_TRIGGER), a truly advancing crest needs a different correction (T5_SCOPE), and BENIGN tests unnecessary intervention (Appendix K).

The simulator scores the unaltered response against its private world, completion deadline, and future event log (line 11); the gate does not consult these private quantities when checking applicability. Even a correct action fails if it finishes too late (Appendix L).

Exactly one match in the evaluated contexts. There are 36,000 planned gated slots outside Anchor. Of these, 35,400 have recorded constraint contexts; 600 inherit construction failure and remain in the loss denominator. Every recorded context matches exactly one stored record, including on protected tiers. The generator partitions cases among the bank's records, so an adjacent case matches its own supported correction instead of leaving the list empty. The experiment thus tests marking one applicable record in a readable bank; Section 5.5 varies which record is marked. Empty and multiple matches require separate tests (Section 6.2).

5 EXPERIMENTAL EVALUATION

5.1 MODELS, COMPARISON ARMS, AND INDEPENDENT BLOCKS

We test `gpt-5.4-mini-2026-03-17` and `claude-sonnet-4-6` at fixed weights, with medium reasoning effort, structured responses, ceilings of 12,000 input and 8,192 output tokens, and 16,384 stored bytes including encoding and provenance. The primary fixed-sequence analysis covers these two paid models. Appendix F reports the earlier single-lesson studies; the separate Gemma multi-lesson studies appear in Appendix N.

All arms receive the same source evidence, live observations, and resource ceilings. TYPED-ACTIVE exposes the whole typed bank; LBF-Gate (TYPED-GATED) exposes the same bank plus the matching records in a constraint list. REFLECT-DIAG stores model-written free-text operational notes, including content that is valid in format but wrong. Only the typed comparison holds both construction and representation fixed.

What the planner can read. Request replay confirms that all three primary arms expose every stored record on the primary conditions, and that gating adds 113–127 input tokens per decision in every model/load cell (Table 11); the descriptive ANCHOR condition instead restricts the arms to an identified source record.

Each model and class has 750 banks, half with eight lessons and half with sixteen; pairing one of each at fixed indices gives 375 independent blocks per class, and the balanced aggregate, averaging the three classes at each index, also has $n = 375$, not 1,125. Primary summaries average INTERFERENCE, which requires choosing among overlapping lessons, and TIMING, which adds elapsed time and resource occupancy. All 162,000 planned slots remain in the analysis, including failures; qualification and pilot records are excluded (Appendix A).

5.2 MEASURES AND PRESPECIFIED ANALYSIS

For valid decisions, the frozen loss is

$$\ell = C_{\text{int}} + 0.60F + 0.25W + 0.10S \in [0, 1], \tag{5}$$

where F is an unprevented nonbenign event, W a wrong non-escalation action, and S an invalid start or resource-conflicting direct action. Intervention costs are 0.05 for ordinary intervention, 0.02 for escalation, and zero for warranted benign inaction. Invalid responses and construction failures receive loss one under intention-to-treat (ITT) accounting. Direct prevention requires correct action completed by the private deadline; human-assisted prevention additionally follows the public service schedule and delay. Appendix D.4 verifies the completion convention with an exact sweep.

We use fixed-time diversified-Kelly D10 tests for bounded means (Waudby-Smith & Ramdas, 2024), locked prospectively in an internal plan. Claims are tested in a fixed order, each an intersection test in which every component must pass: gating must improve on reflection, then on active memory, in every class and the balanced pool on both models, with protected upper bounds below $\eta = 0.10$; pooled-margin claims follow, and testing stops at the first non-rejection. Component bounds are one-sided at level 0.05, not simultaneous intervals. Appendix B gives all ten steps and their chronology.

5.3 RQ1: TRANSFER BENEFIT FROM GATING THE SAME RECORDS

With the bank, model, and ceilings fixed, LBF-Gate more than halves GPT's held-out T3 loss (from 0.19480 to 0.08636) and reduces Claude's from 0.25144 to 0.16480. The absolute gains are 0.10844 and 0.08663. Every class-specific D10 lower bound is positive, and both prespecified positive-transfer claims pass, including all protected-case requirements (Table 1). Against reflection, balanced gains are 0.05046 and 0.08085; reflection's absolute losses are 0.13682 and 0.24565. The typed contrast measures the benefit of checking and marking applicability at fixed stored content. The reflection contrast also changes how the lessons are represented.

Bank size and the identified-source control. Gains are positive at both bank sizes and under both primary conditions (Table 10); larger banks also halve per-entry storage, so load is not isolated. Under ANCHOR, which identifies the source to every arm, the gain over active memory falls to 0.0003

Table 1: Confirmatory multi-lesson transfer. Entries are loss reduction in favor of TYPED-GATED, with the one-sided D10 lower bound in parentheses. Each class and balanced aggregate has 375 independent load-paired blocks per model. Reporting active memory first does not change the prespecified testing order.

Comparator	Crest	Stale source	Resource	Balanced
GPT-5.4 mini				
Typed-active	.1170 (.0957)	.1060 (.0847)	.1023 (.0810)	.1084 (.0920)
Reflection	.0554 (.0298)	.0530 (.0277)	.0430 (.0182)	.0505 (.0336)
Claude Sonnet 4.6				
Typed-active	.0868 (.0652)	.0898 (.0674)	.0833 (.0618)	.0866 (.0704)
Reflection	.0726 (.0519)	.0874 (.0658)	.0826 (.0615)	.0808 (.0649)

on GPT and 0.0050 on Claude, consistent with marking adding little once the relevant source is identified. Section 5.5 tests the mechanism directly.

Preventive behavior. Direct T3 prevention reaches 95.91% on GPT and 81.18% on Claude under gating, versus 81.33% and 70.67% under active memory and 85.87% and 71.13% under reflection; human-assisted prevention is only 0.067% and 0.022%. Escalations fall from reflection's 255 and 7 to gating's 12 and 1, and unnecessary benign interventions from active memory's 65 and 319 to 0 and 144 (Table 9).

Margins and sensitivity. The next fixed-sequence claim, pooled gain over reflection above 0.05 on both models, fails at GPT's lower bound of 0.03357, so later margin claims are not reached. Hoeffding certifies no transfer effect; post-hoc block-bootstrap intervals are positive for all transfer cells but do not change the locked decisions (Table 5).

Gemma follow-up. In the separate Gemma follow-up, gating's pooled loss was 0.060 below reflection but 0.006 above ordinary typed memory; more correct actions were offset by more invalid schedules, and one transfer class remained unresolved (Appendix N).

5.4 RQ2: LOSS ON PROTECTED CASES

All 48 class-specific and balanced D10 upper bounds for T5 scope, T5 trigger, and benign degradation lie below $\eta = 0.10$; the largest is 0.05352. Table 2 shows balanced results. The criterion limits additional loss relative to each comparator; it allows some degradation.

Table 2: Balanced protected-case loss degradation for TYPED-GATED, with its one-sided D10 upper bound. Positive values are worse. Every prespecified class and aggregate upper bound is below $\eta = 0.10$; this does not mean zero degradation. The full 48-bound table is in Appendix C.

Model	Comparator	Scope	Trigger	Benign
GPT	Reflection	.0067 (.0216)	-.0059 (.0097)	.0309 (.0452)
GPT	Typed-active	-.0172 (-.0040)	-.0190 (-.0058)	-.0032 (.0092)
Claude	Reflection	-.0370 (-.0226)	-.1085 (-.0919)	-.0162 (-.0032)
Claude	Typed-active	-.0223 (-.0084)	-.0418 (-.0271)	-.0118 (.0010)

GPT's benign loss exceeds reflection's only because 75 of 2,250 typed banks fail construction, a penalty both typed arms retain; successfully constructed gated GPT states make no unnecessary benign interventions. Claude intervenes unnecessarily on 3.2% of planned benign slots, yet its total benign loss is below both comparators.

5.5 RQ3: DOES THE BENEFIT REQUIRE MARKING THE CORRECT RECORD?

RQ1 shows that marking helps, not why: the gated request bundles the designated record with the emphasis a constraint carries. A marking-control panel, specified and locked before execution and

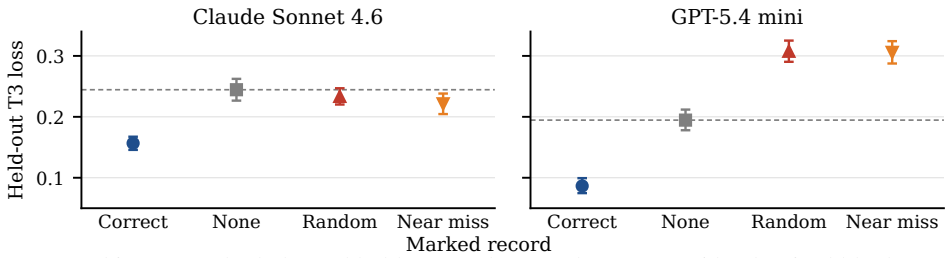

Figure 1: Marking controls: balanced held-out T3 loss on the same 125 load-paired blocks per class, by marked record. Correct (LBF-Gate) and None (active memory) are the main study's historical rows for these units; Random and Near miss mark a wrong record, all else unchanged. Dashed line: no mark. Bars: descriptive 95% block-bootstrap intervals. Prespecified contrasts and D10 bounds, positive except in Claude's stale-source class, are in Appendix M.

separate from the main study's fixed sequence, varies only which record is marked; RQ3 and RQ4 are two analyses of this one panel, a request-matched comparison against historical reference outcomes. Each wrong-mark request is the executed LBF-Gate request with its single constraint replaced by a non-matching record from the same bank: a seeded random non-match or the nearest miss. The readable bank and every other request field are byte-identical. The panel uses 125 load-paired blocks per class and model (24,000 scored slots, 368 of them inherited construction failures, through the study's own request builder and scorer) and takes its correct-mark and no-mark arms from the main study's rows for the same units (Appendix M).

The prespecified test requires a positive one-sided bootstrap lower bound for random-minus-correct loss in every class on both models. It passes under that rule: correct marking outperforms a random wrong mark by 0.077 on Claude and 0.221 on GPT (balanced lower bounds 0.066 and 0.207; Figure 1). We detect no difference between near-miss and random marks at the reported precision (0.012 and 0.002; both intervals include zero). The benefit therefore depends on which record is designated; it is not a content-independent effect of marking any record. Because a wrong mark also supplies wrong guidance, applicability information and its constraint presentation remain jointly responsible.

5.6 RQ4: What does a wrong mark cost?

Claude stays near its unmarked level (random minus no mark -0.011 [-0.028, 0.006]), whereas GPT's loss rises by 0.113 [0.091, 0.135]. GPT's scope and trigger losses rise by 0.13–0.22, above η. It seldom executes the wrong correction (8% versus 5% unmarked); instead, it escalates (56% versus 2%), and half of its escalations finish too late. Wrong guidance makes GPT defer, forfeiting the prevention window. Repeating Claude's six-day-old references on T3 found no detectable drift and preserved its conclusions. Appendix M.3 gives the action breakdown; Appendices M.4–M.5 report reference checks and drift sensitivity.

Sensitivity to marking accuracy (post hoc). Mixing correct and wrong marks at a fixed accuracy, independent of case difficulty, gives GPT a pooled transfer break-even near 51% against unmarked memory. Matching unmarked loss on scope and trigger cases requires 88–92%; staying within $\eta = 0.10$ requires 20–51%. These estimates assume errors resemble the tested substitutions. Appendix M.6 gives the calculation, uncertainty, and Claude results; they describe accuracy requirements, not measured selector performance.

5.7 RQ5: Placement, applicability, and source-quality controls

Predetermined subsets of 50 load-paired blocks per class (placement, retrieval) and ten (corruption) are compared with gating on the same blocks. Range-only Hoeffding radii of 0.384 and 0.859 exceed every observed effect; post-hoc bootstrap intervals separate most contrasts from zero, but these panels remain descriptive (Appendix D, Table 12).

Placement, scope, and timing. Inactive placement removes readable records. Mislocalized placement leaves them readable under a nonacting component but supplies no constraints to the acting component. Both increase loss in every class on both models, as does removing the scope check.

Removing the timing annotation from the constraint worsens every Claude class; GPT's crest contrast is negative and its pooled effect small. Appendix D gives exact interface semantics and all class results.

Focused retrieval. BM25 ranks public opaque tokens ($k_1 = 1.2$, $b = 0.75$, top four) without executing applicability predicates (Robertson & Zaragoza, 2009); unlike gating, it replaces the readable set with four records. On the same 50 blocks per class, balanced T3 losses on GPT/Claude are 0.08372/0.19033 for BM25, 0.18593/0.24240 for active memory, and 0.12505/0.24995 for reflection. Gating's matched losses are lower still (0.07057/0.16292), by 0.0132 [0.0024, 0.0244] and 0.0274 [0.0128, 0.0421] (post-hoc bootstrap). Selective exposure thus helps in its own comparison but cannot explain a main contrast whose readable exposure is identical, and the panel does not apportion the main effect among mechanisms.

Corrupted source audits. Corrupting one in eight audits before construction, identically across arms, raises gated loss from 0.062 to 0.166 on GPT and from 0.181 to 0.319 on Claude on its ten-block subset. All arms worsen; gating keeps the lowest corrupted T3 loss, though its advantage over reflection narrows (0.116 to 0.060 and 0.075 to 0.050). The panel measures dependence on source quality at one error rate, not a robustness threshold (Appendix D).

6 Discussion and Further Work

6.1 What the results establish

Retaining a useful lesson and using it effectively are different achievements. With the bank and model weights fixed, applicability marking improves GPT's and Claude's decisions. All records remain readable in a longer prompt, ruling out context reduction. The marking panel changes only the designated record while holding its position and framing fixed. Wrong marks reduce the advantage, so marking any record is not enough; applicability information and constraint presentation remain jointly responsible. A wrong mark can cost more than no gate (RQ4), supporting the framework's evaluation of restriction as well as benefit.

The smaller controls agree: removing access, mislocalizing the bank, or removing the scope check increases loss on both models, and BM25, which does change the readable set, trails gating.

Gemma's additional correct choices did not improve prevention over ordinary typed memory. Successful lesson use also requires meeting time and resource constraints (Appendix N).

The primary study covers three constructed recurrence families and two paid-model snapshots, and GPT's marking-panel references are six days older than its wrong-mark calls. Audits supply the explanations and applicability fields, so validation checks fidelity to a source rather than independently establishing its truth. The corruption panel shows why that distinction matters. The generator also yields exactly one match in every recorded gated context outside ANCHOR. Thus the evidence concerns marking one applicable lesson, not resolving empty or ambiguous matches.

6.2 Next comparisons

A concurrent reference run for GPT would close the panel's remaining gap, and marking every record would separate the number of marks from their correctness. Cases with no or several matches, stronger retrieval and two-stage selectors with their costs, autonomous source diagnosis, lesson revocation, and broader domains remain to be tested.

7 Conclusion

Learning Before Failure evaluates whether stored lessons support timely decisions without causing excess harm in look-alike cases. On both paid models, LBF-Gate improves on the same lessons unmarked and on reflection within the prespecified protection limits. Wrong marks reduce the benefit and can be worse than no mark by inducing deferral rather than action. LBF makes the use of stored experience testable by holding the lesson bank fixed and measuring both timely prevention and inappropriate lesson use.

Ethics Statement

The study uses constructed search-and-rescue simulations and model outputs, not field rescue actions, human-subject experiments, or private personal data. Human escalation is a simulated service with explicit availability and delay. A low simulated loss or a protected-case bound is not authorization to deploy in a physical emergency. Source errors, inappropriate scope, and unnecessary interventions can harm real recipients; deployment would require domain validation, operational safeguards, and accountable oversight beyond this study.

Reproducibility Statement

The appendices document the loss, independent unit, resource ceilings, failure accounting, original inferential decisions, and artifact bindings. The release includes the locked D10 implementation and configuration, exact block vectors for all primary endpoints, 162,000 loss-level slots, 27,000 block-arm-tier means, follow-on vectors, the timing layer, and the executed gate trace. One command regenerates the component bounds, ten fixed-sequence decisions, descriptive breakdowns, and bootstrap report. Reproduction in a clean directory returned byte-identical numerical outputs without network access or model calls.

The scientific companion supplies registered banks, committed states, normalized outcomes, and frozen code for replay from registered input to score. Appendix G explains the export transformations and original-to-export hashes. Raw provider HTTP bodies and operational journals remain private; offline replay is distinct from collecting new responses or independently auditing execution. Earlier studies retain their original scores and inferential status. The descriptive controls and post-hoc bootstrap remain outside the confirmatory family. The marking-control panel releases its protocol with every amendment, the executed code of every run, its 24,000 scored rows and 3,000 contemporaneous reference rows, and analyses whose hashes were recorded before their data were read (Appendix M).

AI Assistance Disclosure

The foundational concepts of this manuscript stem from the authors' long-standing research and were conceived entirely by the authors. Language models (GPT, Claude) and coding assistants (Codex) were used to assist with related work searches, conceptual framework discussions, hypothesis refinement, manuscript and appendix editing, and design of the synthetic benchmark generator. The authors retain ultimate responsibility for all scientific claims, the rigorous verification of AI-generated content, and the integrity of the final submission.

References

Agnar Aamodt and Enric Plaza. Case-based reasoning: Foundational issues, methodological variations, and system approaches. *AI Communications*, 7(1):39–59, 1994. doi: 10.3233/AIC-1994-7104. URL https://doi.org/10.3233/AIC-1994-7104.

Marc-Antoine Allard, Arnaud Teinturier, Victor Xing, and Gautier Viaud. Experiential reflective learning for self-improving LLM agents. *arXiv preprint arXiv:2603.24639*, 2026. doi: 10.48550/arXiv.2603.24639. URL https://arxiv.org/abs/2603.24639. Also presented at the ICLR 2026 MemAgents Workshop.

Anonymous. Epistemic regret minimization: Label-free causal critique beyond outcome reward. Anonymous manuscript under review, 2026.

Deborah Dowell, Kathleen R. Ragan, Christopher M. Jones, Grant T. Baldwin, and Roger Chou. CDC clinical practice guideline for prescribing opioids for pain—united states, 2022. *MMWR Recommendations and Reports*, 71(3):1–95, 2022. doi: 10.15585/mmwr.rr7103a1. URL https://www.cdc.gov/mmwr/volumes/71/rr/rr7103a1.htm.

Shengda Fan, Xin Cong, Zhong Zhang, Yuepeng Fu, Yesai Wu, Hao Wang, Xinyu Zhang, Enrui Hu, and Yankai Lin. Generalizing experience for language agents with hierarchical MetaFlows. In *Advances in Neural Information Processing Systems*, volume 38, 2025. doi: 10.52202/085713-2149.

URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/5c882988ce5fac487974ee4f415b96a9-Abstract-Conference.html.

Wassily Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58(301):13–30, 1963. doi: 10.1080/01621459.1963.10500830. URL https://doi.org/10.1080/01621459.1963.10500830.

Andrew K. S. Jardine, Daming Lin, and Dragan Banjevic. A review on machinery diagnostics and prognostics implementing condition-based maintenance. *Mechanical Systems and Signal Processing*, 20(7):1483–1510, 2006. doi: 10.1016/j.ymssp.2005.09.012. URL https://doi.org/10.1016/j.ymssp.2005.09.012.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 6769–6781. Association for Computational Linguistics, 2020. doi: 10.18653/v1/2020.emnlp-main.550. URL https://aclanthology.org/2020.emnlp-main.550/.

Tze Leung Lai. Sequential analysis: Some classical problems and new challenges. *Statistica Sinica*, 11(2):303–408, 2001.

John E. Laird, Paul S. Rosenbloom, and Allen Newell. Chunking in Soar: The anatomy of a general learning mechanism. *Machine Learning*, 1(1):11–46, 1986. doi: 10.1023/A:1022639103969. URL https://doi.org/10.1023/A:1022639103969.

Jialong Li and Jialing Zhu. Auditing self-evolution in financial agents: Capability gains, security drift, and execution-interface mismatch. *arXiv preprint arXiv:2608.17684*, 2026. doi: 10.48550/arXiv.2608.17684. URL https://arxiv.org/abs/2608.17684.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173, 2024. doi: 10.1162/tacl_a_00638. URL https://aclanthology.org/2024.tacl-1.9/.

Yitao Liu, Chenglei Si, Karthik R. Narasimhan, and Shunyu Yao. Contextual experience replay for self-improvement of language agents. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 14179–14198. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.acl-long.694.

David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continual learning. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

Gary Lorden. Procedures for reacting to a change in distribution. *The Annals of Mathematical Statistics*, 42(6):1897–1908, 1971. doi: 10.1214/aoms/1177693055. URL https://doi.org/10.1214/aoms/1177693055.

Adyasha Maharana, Dong-Ho Lee, Sergey Tulyakov, Mohit Bansal, Francesco Barbieri, and Yuwei Fang. Evaluating very long-term conversational memory of LLM agents. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics*, pp. 13851–13870, 2024. doi: 10.18653/v1/2024.acl-long.747.

Tom M. Mitchell, Richard M. Keller, and Smadar T. Kedar-Cabelli. Explanation-based generalization: A unifying view. *Machine Learning*, 1:47–80, 1986. doi: 10.1007/BF00116250. URL https://doi.org/10.1007/BF00116250.

Charles Packer, Sarah Wooders, Kevin Lin, Vivian Fang, Shishir G. Patil, Ion Stoica, and Joseph E. Gonzalez. MemGPT: Towards LLMs as operating systems. *arXiv preprint arXiv:2310.08560*, 2023.

E. S. Page. Continuous inspection schemes. *Biometrika*, 41(1/2):100–115, 1954. doi: 10.1093/biomet/41.1-2.100.

Stephen Robertson and Hugo Zaragoza. The probabilistic relevance framework: BM25 and beyond. *Foundations and Trends in Information Retrieval*, 2009. doi: 10.1561/1500000019. URL https://doi.org/10.1561/1500000019.

Diego Ruiz-Hernandez, Jesus Maria Pinar-Perez, and David Delgado-Gomez. Multi-machine preventive maintenance scheduling with imperfect interventions: A restless bandit approach. *Computers & Operations Research*, 119:104927, 2020. doi: 10.1016/j.cor.2020.104927. URL https://doi.org/10.1016/j.cor.2020.104927.

Yiting Shen, Kun Li, Wei Zhou, and Songlin Hu. Mem2ActBench: A benchmark for evaluating long-term memory utilization in task-oriented autonomous agents. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics*, pp. 8173–8190, 2026. doi: 10.18653/v1/2026.acl-long.370.

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 (NeurIPS)*, 2023.

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 2nd edition, 2018.

Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An open-ended embodied agent with large language models. *arXiv preprint arXiv:2305.16291*, 2023. doi: 10.48550/arXiv.2305.16291. URL https://arxiv.org/abs/2305.16291.

Zora Zhiruo Wang, Jiayuan Mao, Daniel Fried, and Graham Neubig. Agent workflow memory. *arXiv preprint arXiv:2409.07429*, 2024. doi: 10.48550/arXiv.2409.07429. URL https://arxiv.org/abs/2409.07429.

Ian Waudby-Smith and Aaditya Ramdas. Estimating means of bounded random variables by betting. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 86(1):1–27, 2024. doi: 10.1093/jrsssb/qkad009. URL https://doi.org/10.1093/jrsssb/qkad009.

Wujiang Xu, Zujie Liang, Kai Mei, Hang Gao, Juntao Tan, and Yongfeng Zhang. A-MEM: Agentic memory for LLM agents. *arXiv preprint arXiv:2502.12110*, 2025. doi: 10.48550/arXiv.2502.12110. URL https://arxiv.org/abs/2502.12110.

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

Appendix Contents

For statistical inference, see Appendix B; for execution records and reproducibility, Appendix G; for the executed gate and saved trace, Appendix H.3; and for the claim–evidence map, Appendix J.

A Protocol and Scoring for the Completed Study 14

B Inference, Prespecified Decisions, and Sensitivity 15

C Full Main-Study Results 17

D Descriptive Controls and Exact Timing 20

E How Stored State Can Affect Later Decisions 23

F Earlier Single-Lesson Studies 24

G Execution Records and Reproducibility 24

H Interface and Information Boundaries 25

I Extended Related Work and the Scope of the Comparison 28

J Claims and Supporting Evidence 30

K Correction Targets, Recurrence Classes, and Transfer 30

L Action Completion and Unresolved Exposure 32

M Marking-Control Panel 33

N Gemma Multi-Lesson Studies 39

A Protocol and Scoring for the Completed Study

This appendix documents the multi-lesson protocol underlying the main comparison. Appendix F separately records the earlier single-lesson protocols and outcomes.

A.1 Recurrence classes and evaluation tiers

The three recurrence classes share a shrinking margin for safe traversal but require different corrections. Table 3 gives their meanings in ordinary language. The frozen generator and action catalog specify the exact identifiers and admissible actions.

Table 3: Task distinctions retained from the source protocol. The relevant distinction is the warranted correction, not similarity of the warning alone.

Class	Source-supported explanation	Correction family
Crest/model lag	Flood crest advances ahead of the propagation model	Reroute and revise hazard-policy state.
Stale source	Route report is stale or biased; the route is not thereby unsafe	Quarantine and independently verify the report rather than automatically rerouting.
Resource degradation	Assigned resource degrades under load	Substitute the resource through the resource-health component.
Benign control	Transient evidence does not warrant the source correction	Avoid unnecessary corrective intervention.

Interference and Timing are decision conditions within the multi-lesson banks, not additional causal classes. T3 is the primary structural-transfer tier: topology, victim placement, timing, nuisance parameters, source identity, and resource identity may change while the prespecified correction class is retained. T5_SCOPE, T5_TRIGGER, and BENIGN are separately scored protected populations. Each protected population has its own bound; T3 improvement alone does not establish protection. Anchor identifies the relevant source and is reported descriptively.

A.2 Source construction and state budget

Each source bank contains eight or sixteen audited lessons. Construction pages cover eight assigned source slots while exposing all bank audits. For the typed arms, the model chooses a correction and a deterministic constructor binds that selection to the source's scope, trigger, responsible component, conservative completion bound, and provenance. A field-wise validator checks unique support and faithful binding. Unsupported selections, missing pages, invalid output, and byte overflow fail the whole bank; failed banks are not partially installed or replaced. The validator does not repair a source claim using private evaluation truth.

The 16,384-byte state ceiling includes canonical encoding, escaping, JSON overhead, provenance identifiers, and separators. The reported canonical envelope permits 2,045 bytes per record at load eight and 1,022 at load sixteen. Reflection receives the same source evidence and resource opportunity but stores model-written notes; wrong-but-valid notes remain in the record. Typed-Active and Typed-Gated share the same constructed typed bank, including any construction failure: their content and source hashes match, and only the arm/placement wrapper differs. The gate adds a constraint list while leaving the stored records and readable bank unchanged. Table 11 reports the realized record counts, memory bytes, and request tokens per arm and load.

Model snapshots are `gpt-5.4-mini-2026-03-17` and `claude-sonnet-4-6`. Both use medium reasoning effort, strict structured responses, a 12,000-input-token ceiling, an 8,192-output-token ceiling, and the same persistent-state cap. Temperature is omitted where unsupported. Paid requests use Batch. The model chooses both its action and its start time; no semantic repair or executor substitution supplies the evaluated action.

A.3 Episodes and independent blocks

For each model and recurrence class, 750 banks are split equally between loads eight and sixteen. A predetermined load pair forms one block, giving 375 independent blocks per class. Let $\bar{\ell}^{r}_{m,c,j,q}$ be the mean loss for arm r on the primary INTERFERENCE/TIMING cases of the load-q bank in block j. The class-level paired contrast is formed after within-bank averaging:

$$D^{b}_{m,c,j} = \frac{1}{2} \sum_{q \in \{8,16\}} \left(\bar{\ell}^{b}_{m,c,j,q} - \bar{\ell}^{\text{gated}}_{m,c,j,q} \right). \tag{6}$$

For $\mathcal{C} = \{\text{crest}, \text{stale source}, \text{resource}\}$, the balanced block and its estimate are

$$D^{b,\text{bal}}_{m,j} = \frac{1}{3} \sum_{c \in \mathcal{C}} D^{b}_{m,c,j}, \qquad \widehat{\Delta}^{b}_{m,\text{bal}} = \frac{1}{375} \sum_{j=1}^{375} D^{b,\text{bal}}_{m,j}. \tag{7}$$

The balanced sample therefore contains 375 observations, not 1,125. Protected contrasts reverse the sign and are computed separately for each tier. Models are analyzed separately. ANCHOR remains descriptive.

Each model has 2,250 banks and 81,000 planned evaluation slots. For a model, arm, and tier, the primary INTERFERENCE/TIMING summaries contain 4,500 nested decisions. This number is a descriptive denominator, not an independent sample size. Pilot and qualification records do not enter confirmation.

A.4 Loss and the human-service model

For valid outputs, Equation (5) assigns an event charge, a wrong-repair charge, a scheduling charge, and an intervention charge. F indicates an unprevented nonbenign event. W indicates a wrong non-escalation action. S indicates an invalid start or a resource-conflicting direct action. Ordinary intervention costs 0.05, escalation 0.02, and warranted benign inaction zero. Invalid outputs or upstream construction failures instead receive ITT loss one. A zero intervention charge does not by itself establish that an action has no resource or scheduling cost.

A human request cannot start before the current tick. It occupies the first public availability window that contains the full service delay, independently of convoy occupancy. Only a response completed by the hidden deadline counts as human-assisted prevention. Availability and delay are generated independently of the warranted correction and are shared across arms. Benign escalation still incurs its intervention charge. Direct correction, human-service completion, human-assisted prevention, and unnecessary intervention remain separate outcomes.

Time spent unresolved. The protocol also records how long the problem remains unresolved. Appendix L.2 defines this censored duration and distinguishes it from the bounded mission loss.

Records and request size. Table 11 reports, per model, arm, and load, the stored entries supplied, the matched records placed in the constraint list, the canonical memory bytes, and the total request tokens, averaged over the primary T3 decisions and recovered by exact request-hash replay. Memory-only token counts were not separately metered. Construction usage is reported separately, with the shared typed state counted once and failed constructions retained. Although the typed arms have identical stored records and equal ceilings, their request sizes differ. Gating consistently produces the larger request.

B Inference, Prespecified Decisions, and Sensitivity

B.1 Original inferential family

We use fixed-time diversified-Kelly D10 inference for bounded means (Waudby-Smith & Ramdas, 2024). The release includes the locked implementation, configuration, and exact input vectors. These specify the choices needed to reproduce the bounds; rounded tables and the method citation alone are insufficient.

Claims are tested in a fixed sequence. Each claim is an all-must-pass intersection: positive gain in each recurrence class and the balanced pool on each required model, all protected-degradation upper bounds below 0.10, and any additional pooled margin specified for that claim. Each component uses one-sided level 0.05. A claim fails if any required component does not pass. The executed sequence has ten ordered steps, reproduced here from the saved analysis with its recorded decisions:

Table 4: The executed ten-step fixed sequence and its original decisions. "Paid" denotes the two evaluated model snapshots jointly; "local" denotes local-model claims without an evaluated panel in this locked analysis. Non-rejection at step 3 blocks steps 4–10, including all five local-model steps (5–8 and 10). Separate Gemma studies do not reopen this sequence.

Step	Panel and comparator	Benefit threshold	Original decision
1	Paid, versus reflection	0	Reject
2	Paid, versus ordinary typed memory	0	Reject
3	Paid, versus reflection	0.05	Do not reject
4	Paid, versus ordinary typed memory	0.05	Not reached
5	Local, versus reflection	0	Not reached
6	Local, versus ordinary typed memory	0	Not reached
7	Local, versus reflection	0.05	Not reached
8	Local, versus ordinary typed memory	0.05	Not reached
9	Paid, versus reflection	0.10	Not reached
10	Local, versus reflection	0.10	Not reached

Steps 1 and 2 pass. Step 3 does not, because GPT's pooled lower bound against reflection is 0.03357. Although the lower bounds against active memory are 0.0920 and 0.0704, the sequence does not reach step 4, so it does not certify that stronger margin. Steps 4–10 remain unreached, and steps 1–2 remain valid rejections. An absent local panel cannot be skipped to reach a later paid-model claim. The component bounds serve the intersection tests; they do not form a simultaneous confidence region.

B.2 Plan amendment, sample size, and lock chronology

We adopted a fixed sequence in place of the earlier proposed 96-interval Bonferroni analysis. A permanently excluded pilot of 60 banks per paid model informed a fixed sample size of 750 banks per class; the pilot shares no bank or episode identifier with the main study, and its exclusion is verified in the released bundle. The original 0.90 joint-power planning target across both comparators was not demonstrated. We therefore fixed the sample size for estimation and the declared comparisons, retaining the endpoints and success rules without claiming that joint-power target. The plan prohibits optional stopping, replacement of failed banks, and post-result selection between tests.

Recorded chronology. The controller journal records the protocol lock on September 15, 2026, before construction and evaluation. Both providers completed construction before any evaluation was requested, and evaluation finished on September 16 (UTC). The reproducibility package retains the detailed chronology and execution counts. These internal records do not establish external preregistration or when a human first viewed outcomes; there was no outcome-blind external statistical sign-off. We therefore describe the plan as prospectively locked internally.

B.3 Mandatory Hoeffding report

Using the bounded-variable inequality of Hoeffding (1963), for a mean of independent paired block contrasts in $[-1, 1]$, the one-sided radius at level 0.05 is

$$r_{375} = \sqrt{\frac{2\log(1/0.05)}{375}} = 0.12640. \tag{8}$$

The sensitivity bounds subtract or add this radius, clipped to the contrast range. Every transfer lower bound is negative under this conservative check, and the full conjunction is not certified. This sensitivity result is retained alongside, not substituted for, the prespecified D10 decisions.

For the smaller controls, the prespecified two-sided marginal radius is

$$r_n^{\text{two}} = \sqrt{\frac{2\log(2/0.05)}{n}}. \tag{9}$$

It equals 0.384129 for $n = 50$ and 0.858939 for $n = 10$. These intervals are not simultaneous. Their inclusion of zero is a reason not to treat the descriptive control means as confirmatory mechanism effects; it is not evidence of equivalence.

B.4 Decision rule and interpretation

Let g range over the three recurrence classes and the balanced pool. Let $\mathrm{LCB}^b_{m,g}$ denote the lower bound for a transfer contrast, and $\mathrm{UCB}^b_{m,g,p}$ the upper bound for protected tier p. A reached positive-transfer claim requires

$$\left[\min_{m,g} \mathrm{LCB}^b_{m,g} > 0\right] \quad \text{and} \quad \left[\max_{m,g,p} \mathrm{UCB}^b_{m,g,p} < \eta\right], \qquad \eta = 0.10, \tag{10}$$

so both benefit and protection include the balanced pool. A stronger pooled-margin claim additionally requires both models' balanced transfer lower bounds to exceed its declared margin. This notation summarizes the rule used in the locked analysis.

A positive-transfer claim rejects nonpositive benefit under the prespecified procedure. Protection bounds additional loss relative to a comparator; it does not imply zero absolute loss. Failure to certify a stronger benefit margin does not establish that the true effect is smaller than that margin. Similarly, a control interval containing zero does not establish equivalence.

The released implementation. The released configuration specifies the betting mixture, tuning parameters, bound inversion, and numerical conventions. It is accompanied by the original code and exact paired block vectors for T3 and all protected tiers, covering both models and comparators. A single entry point regenerates the 64 component estimates and bounds, the ten fixed-sequence decisions, and the Hoeffding report. Clean-directory reproduction on a different Python minor version returned byte-identical outputs and requires no network access or model calls.

B.5 Post-hoc bootstrap sensitivity

Betting-based bounded-mean methods adapt to observed variability (Waudby-Smith & Ramdas, 2024); the Hoeffding radius above uses only the range and the sample size. The methods can therefore give different widths on the same data. D10 endpoints are not normal-theory standard errors. We compute the bootstrap from the original block observations, using resampling settings fixed before that computation.

The resampling unit is the load-paired block, matching the inferential unit of the locked analysis. Load pairing, matched arms, class-balanced aggregation, common parents for the follow-on panels, and all intention-to-treat failures are preserved; blocks are resampled whole. We use 20,000 resamples, marginal percentile intervals at 95%, NumPy's PCG64 generator with seed 20260920, and linear quantile interpolation. Intervals are marginal and not simultaneous. Results appear in Table 5 for all 16 main transfer cells and Table 12 for the pooled descriptive panels. The released numerical report retains all 64 main cells, including protected tiers, and the class-specific control contrasts.

This post-hoc sensitivity analysis leaves the original family, order, and decisions unchanged. In particular, it does not reopen steps 3 or 4 or make the smaller panels confirmatory. The corruption intervals use only $n = 10$ blocks and are correspondingly unstable.

C Full Main-Study Results

Table 1 reports T3 loss reductions by class and in the balanced pool. This appendix adds all 48 protected upper bounds (Table 6), absolute losses (Table 7), and their components (Table 8). The absolute values show how much loss remains even when a relative comparison favors gating.

Table 5: Post-hoc paired block-bootstrap sensitivity for all 16 main transfer contrasts: 20,000 resamples, marginal 95% percentile intervals, seed 20260920. Blocks preserve load pairing, class-balanced aggregation, and all ITT failures. These are not simultaneous intervals and do not replace D10 or certify blocked practical margins.

Comparator	Crest	Stale source	Resource	Balanced
GPT-5.4 mini				
Typed-active	.1170 [.1020, .1323]	.1060 [.0909, .1220]	.1023 [.0868, .1182]	.1084 [.0995, .1177]
Reflection	.0554 [.0368, .0736]	.0530 [.0347, .0716]	.0430 [.0253, .0602]	.0505 [.0402, .0608]
Claude Sonnet 4.6				
Typed-active	.0868 [.0707, .1033]	.0898 [.0733, .1064]	.0833 [.0676, .0994]	.0866 [.0776, .0957]
Reflection	.0726 [.0578, .0880]	.0874 [.0719, .1027]	.0826 [.0674, .0983]	.0808 [.0722, .0896]

Table 6: Full protected-case results: degradation in gated loss relative to each comparator, with one-sided D10 upper bounds in parentheses. All 48 upper bounds are below 0.10. C, S, and R are crest, stale-source, and resource. Each class and balanced aggregate has 375 blocks.

Comparator / tier	C	S	R	Balanced
GPT-5.4 mini				
Reflection / scope	.0047 (.0262)	.0078 (.0294)	.0077 (.0288)	.0067 (.0216)
Reflection / trigger	-.0051 (.0176)	-.0097 (.0141)	-.0030 (.0189)	-.0059 (.0097)
Reflection / benign	.0316 (.0521)	.0331 (.0535)	.0281 (.0477)	.0309 (.0452)
Active / scope	-.0188 (-.0042)	-.0112 (.0028)	-.0217 (-.0071)	-.0172 (-.0040)
Active / trigger	-.0204 (-.0061)	-.0177 (-.0033)	-.0188 (-.0045)	-.0190 (-.0058)
Active / benign	-.0035 (.0090)	-.0033 (.0092)	-.0029 (.0096)	-.0032 (.0092)
Claude Sonnet 4.6				
Reflection / scope	-.0295 (-.0125)	-.0443 (-.0269)	-.0371 (-.0201)	-.0370 (-.0226)
Reflection / trigger	-.1086 (-.0853)	-.1106 (-.0888)	-.1064 (-.0843)	-.1085 (-.0919)
Reflection / benign	-.0133 (.0005)	-.0141 (-.0001)	-.0213 (-.0073)	-.0162 (-.0032)
Active / scope	-.0226 (-.0068)	-.0199 (-.0035)	-.0245 (-.0085)	-.0223 (-.0084)
Active / trigger	-.0391 (-.0203)	-.0487 (-.0305)	-.0376 (-.0206)	-.0418 (-.0271)
Active / benign	-.0091 (.0043)	-.0098 (.0036)	-.0167 (-.0032)	-.0118 (.0010)

Table 7: Absolute multi-lesson study loss and escalation. Lower loss is better.

Model	System	T3	T5 scope	T5 trigger	Benign	Escalation
OpenAI	Reflection	.13682	.07851	.09206	.00242	3.98%
	Typed-active	.19480	.10249	.10512	.03655	2.09%
	Typed-gated	.08636	.08526	.08616	.03333	0.25%
Claude	Reflection	.24565	.19823	.27624	.02558	1.07%
	Typed-active	.25144	.18359	.20949	.02121	0.62%
	Typed-gated	.16480	.16127	.16770	.00936	0.07%

Table 8: T3 loss decomposition. The five columns sum to total T3 loss up to rounding. The unprevented-event charge is distinct from accumulated pre-deadline debt.

Model	System	Event	Wrong repair	Intervention	Conflict	ITT failure
OpenAI	Reflection	.06840	.01967	.04816	.00060	.00000
	Typed-active	.08387	.03000	.04712	.00049	.03333
	Typed-gated	.00413	.00028	.04824	.00038	.03333
Claude	Reflection	.17293	.02117	.04973	.00182	.00000
	Typed-active	.17587	.02500	.04979	.00078	.00000
	Typed-gated	.11267	.00117	.04995	.00080	.00022

Table 9: Prevention, escalation, and unnecessary intervention counts, with 4,500 planned primary T3 slots per model and arm. Unknown actions are construction/output failures retained at loss one. Human-assisted prevention is a subset of escalation requests, not an additional mutually exclusive category. Benign counts refer to a separate 4,500-slot population.

		T3 (4,500 planned)				Benign
Model	Arm	Direct prev.	Human-assist.	Escalations	Unknown	Unnec. interv.
GPT	Reflection	3864	123	255	0	82
	Typed-active	3660	61	126	150	65
	LBF-Gate	4316	3	12	150	0
Claude	Reflection	3201	2	7	0	405
	Typed-active	3180	1	6	0	319
	LBF-Gate	3653	1	1	1	144

Interpretation of protected cases. The largest protected upper bound is 0.05352. All bounds are below the prespecified 0.10 tolerance, but some estimated degradations relative to reflection are positive. GPT's gated benign loss is entirely the 3.33% upstream construction penalty: no unnecessary benign intervention is reported among successfully installed gated states. Claude's gated states intervene unnecessarily on 3.2% of planned benign decisions. Failed constructions remain in these aggregate losses.

The across-tier human-request rates are 0.25% (GPT) and 0.07% (Claude) for gating, versus 3.98% and 1.07% for reflection. These rates differ from the fractions of T3 decisions with successful human-assisted prevention, reported in the main paper. All denominators retain the relevant planned slots.

Reading loss components. Table 8 contains weighted charges, not event percentages. For example, Claude's gated event charge 0.11267 corresponds to approximately $0.11267/0.60 = 18.78\%$ of planned T3 slots, and its wrong-repair charge 0.00117 to approximately $0.00117/0.25 = 0.47\%$. These conversions use rounded aggregates. They neither recover exact counts nor identify the joint correct-action-but-late category; ITT failures are accounted for separately.

C.1 Load, condition, and prevention accounting

Table 10 stratifies the primary outcomes by bank load and decision condition, and Table 9 gives the exclusive prevention accounting for all three arms. Recombination of the strata recovers the primary estimates of Table 1 to within floating-point tolerance, and the load pairing is preserved throughout, so the load difference $\Delta_{16} - \Delta_8$ is estimated on matched blocks rather than across independent groups. The fixed byte ceiling changes storage per record as well as the number of records, so this breakdown does not isolate the effect of bank size. Both typed arms read the same records at each load; the load interaction does not establish that gating removes competing information.

Table 10: T3 loss reduction (comparator minus LBF-Gate), balanced across classes, by load and condition. Primary estimates average Interference/Timing; Anchor names the relevant source for every arm. Descriptive strata, not new replications.

		Bank load		Condition		
Model	Comparator	8	16	Interf.	Timing	Anchor
GPT	Reflection	.0454	.0555	.0503	.0506	-.0221
	Typed-active	.0834	.1335	.1009	.1160	.0003
Claude	Reflection	.0584	.1033	.0595	.1022	.0014
	Typed-active	.0814	.0918	.0676	.1057	.0050

The prevention counts use planned intention-to-treat denominators of 4,500 slots per model and arm. Direct prevention, human-assisted prevention, and unknown actions are disjoint categories. Escalation requests overlap with human-assisted prevention and are reported separately. Construction

and output failures have unknown physical actions. Valid-output denominators are 4,350 for both GPT typed arms, reflecting the 150 inherited construction failures per tier, and 4,499 for Claude gated. The joint category of a correct action completing after its deadline is not separable from these marginal counts; it requires the joined action, timing, feasibility, and validity records in the released bundle, and we do not infer it here.

C.2 Realized record exposure

Table 11 retains the complete exposure accounting supporting the same-record comparison. Records and request hashes, rather than prompt length alone, establish equality of the readable typed banks.

Table 11: Full realized exposure on primary T3 decisions (INTERFERENCE/TIMING), averaged across recurrence classes and reconstructed by exact request-hash replay. Each row supplies all records of its load; C is the additional constraint count. Bytes measure the memory payload; tokens measure the full request. Memory-only tokens were not separately metered.

GPT-5.4 mini

Load	Arm	C	Bytes	Tokens
8	Reflection	0	2713.9	2315.8
	Active	0	2564.2	2123.7
	Gate	1	2864.4	2237.3
16	Reflection	0	4965.3	3343.3
	Active	0	4989.6	3023.7
	Gate	1	5289.8	3137.3

Claude Sonnet 4.6

Load	Arm	C	Bytes	Tokens
8	Reflection	0	2970.0	3140.2
	Active	0	2564.2	2725.7
	Gate	1	2864.4	2852.5
16	Reflection	0	6185.1	4798.5
	Active	0	4989.6	3731.9
	Gate	1	5289.8	3857.1

D Descriptive Controls and Exact Timing

Table 12: Matched descriptive controls: extra T3 loss over the clean gated parent and post-hoc marginal 95% block-bootstrap intervals (20,000 resamples). n counts load-paired blocks. No-scope uses INTERFERENCE; no-timing uses TIMING. These panels are outside the confirmatory family; the ten-block corruption intervals are unstable.

Control	n	GPT	Claude
Inactive placement	50	+.2875 [+.2551, +.3204]	+.1916 [+.1639, +.2200]
Mislocalized placement	50	+.0949 [+.0735, +.1176]	+.0811 [+.0566, +.1070]
No scope gate	50	+.0533 [+.0319, +.0767]	+.0113 [+.0028, +.0227]
No timing annotation	50	+.0036 [-.0137, +.0210]	+.0431 [+.0167, +.0686]
BM25 retrieval, top four	50	+.0132 [+.0024, +.0244]	+.0274 [+.0128, +.0421]
One-in-eight corrupted audits	10	+.1043 [+.0227, +.2064]	+.1383 [+.0554, +.2192]

D.1 Matched subsets and placement semantics

Placement, retrieval, and correct-source panels use the first 100 prespecified banks per class per model: 300 banks, yielding 50 load-paired blocks per class and balanced aggregate. Audit corruption uses the first twenty banks per class: sixty banks and ten blocks. These are matched-parent controls, not new independent replications of the full study. Except where explicitly noted, a control contrast is control loss minus its matched clean gated-parent loss. No-scope uses INTERFERENCE only; no-timing uses TIMING only.

The placement controls separate access to the records from their connection to the acting component. Inactive placement removes readable records; mislocalized placement leaves them readable under a nonacting component without supplying constraints to the acting component. The read traces verify these interfaces, not internal causal mediation within the LLM.

Implemented interventions. Each control changes a specified part of LBF-Gate. NO-SCOPE drops the ordered-scope equality test from the match conjunction; the component test and both predicates still run, and the scope fields remain present in every stored record. NO-TIMING removes the `complete_within_ticks` field from the constraint payload only; matching is unaffected and the same field remains readable in the stored record, so this control tests explicit timing annotation rather than absence of timing information. INACTIVE supplies no readable records at all. MISLOCALIZED keeps the records readable but files them under the case's nonacting component, which prevents the constraint route from running and therefore yields a readable bank with an empty constraint list. Thus inactive removes the text, while mislocalized removes its constraint route to the acting component. We checked these semantics against all 4,800 saved placement slots, of which 4,768 reproduce their read/request joins and 32 are retained upstream-failure slots, and one saved submission payload independently confirms each interface.

All inactive and mislocalized class contrasts are positive. All no-scope T3 class contrasts are positive. No-timing has positive contrasts for all Claude classes, whereas GPT's crest contrast is negative. Table 13 reports all class contrasts.

Table 13: Control T3 loss changes by class and model. C/S/R denote crest, stale-source and resource. All panels in this table have $n = 50$ independent load-paired blocks per class and pool.

Control	Model	C	S	R	Balanced
Inactive	OpenAI	+.27535	+.27800	+.30915	+.28750
	Claude	+.19265	+.19420	+.18800	+.19162
Mislocalized	OpenAI	+.09480	+.10040	+.08940	+.09487
	Claude	+.09025	+.07310	+.08000	+.08112
No scope	OpenAI	+.07520	+.06240	+.02240	+.05333
	Claude	+.00850	+.00850	+.01700	+.01133
No timing	OpenAI	-.01600	+.01170	+.01520	+.00363
	Claude	+.05750	+.03670	+.03500	+.04307
Retrieval-only	OpenAI	+.00545	+.00900	+.02500	+.01315
	Claude	+.02175	+.02275	+.03775	+.02742
Correct-source	OpenAI	-.04485	-.00430	+.00350	-.01522
	Claude	-.00865	+.00425	+.01625	+.00395

D.2 Retrieval and the correct-source reference

BM25 uses fixed $k_1 = 1.2$, $b = 0.75$, and top-four retrieval over public opaque tokens, with scope and feature names and the acting component forming the query, numeric observation values excluded from ranking, and ascending canonical-record hashes breaking ties; zero-scoring records may fill the remaining positions. No predicates are executed. BM25 replaces the readable list with four retrieved records. LBF-Gate keeps the full list and adds matching records as constraints. Only the retrieval comparison changes the visible record set; the main typed comparison holds it fixed. The paired T3 losses are 0.083717 versus 0.070567 (retrieval versus gating) on GPT, and 0.190333 versus 0.162917 on Claude. The two systems share a storage budget but expose different numbers of records and produce different prompt lengths. No dense-retrieval or learned-reranking comparison is reported.

The correct-source reference installs correctly compiled source records into the unchanged gated planner. It sees neither future evaluation truth nor a repaired action, and is not a performance oracle. Its T3 loss is 0.055350 on GPT and 0.166867 on Claude, compared with the matched gated-parent losses above. Correct source construction helps GPT's average but does not improve Claude's average in this panel. Action selection and timing still matter after source construction.

D.3 One-in-eight source corruption

Exactly one load-eight audit or two load-sixteen audits are corrupted before memory construction, shared across arms. Private truth and live cases are unchanged. Corruption can affect a distractor rather than the rule evaluated in a particular case; cases are not redrawn to force exposure. Composite study/condition/bank/episode/arm keys preserve the matched clean and corrupted joins.

Table 14: One-in-eight audit-error panel: pooled T3 loss. Each model has 60 banks and 10 load-paired blocks per class and pool. Clean values use the same main-study cases, not the full-study averages.

Model	System	Clean	Corrupted	Change
OpenAI	Reflection	.177833	.225833	+.048000
	Typed-active	.187583	.282833	+.095250
	Typed-gated	.061667	.166000	+.104333
Claude	Reflection	.256250	.369583	+.113333
	Typed-active	.267917	.385417	+.117500
	Typed-gated	.180833	.319167	+.138333

Gated T3 loss increases in every class and remains below both corrupted-memory comparators. Reflection is not uniformly worsened: its stale-source difference is -0.00025 on GPT and -0.02000 on Claude. Corruption therefore worsens every system's pooled loss, but not every individual cell. The panel tests one error rate rather than an error-tolerance curve. GPT's observed scope increase of 0.1165 is the corrupted-minus-clean gated contrast, not the clean gated-minus-comparator contrast used by the primary protection criterion.

Source-corruption contrast. The corruption increase is $R^{\text{gated}}_{\text{corrupt}} - R^{\text{gated}}_{\text{clean}}$ on matched cases. It differs from the clean-source protected contrast against a comparator. Larger deterioration from a better clean starting point can coexist with lower corrupted loss: Table 14 retains all arms for that reason.

Table 15: Selected protected-case control differences, relative to matched clean gated cases. Audit-error increases refer to the gated system under corruption. No-scope and no-timing use their prespecified single-condition endpoints.

Panel	Model	T5 scope	T5 trigger	Benign
Audit error	OpenAI	+.116500	+.064583	+.060667
	Claude	+.077917	+.016333	+.018167
Retrieval-only	OpenAI	+.024550	+.010033	+.002600
	Claude	+.027917	+.020250	+.025500
No scope	OpenAI	+.013267	—	—
	Claude	.000000	—	—
No timing	OpenAI	—	+.015333	—
	Claude	—	+.039167	—

D.4 Exact deadline validation

The exact-layer sweep evaluates 1,152 deterministic cells: four worlds, nine acquisition offsets from -4 to $+4$ around the commitment deadline, registration delay zero or one, eight evidence schedules, and correct or deliberately wrong repair. Schedules comprise an already-identified reference, concurrent covered probes, and all six serial probe orders. Diagnosis stops once admitted responses identify a singleton correction set; private truth never chooses the next probe.

For every correct nonbenign action, completion exactly at the deadline prevents the event, while completion one tick late fails. With immediate registration, concurrent probes need one tick of acquisition slack; serial probes need one to three. Across harmful worlds and serial orders, prevention fractions at one, two, and three ticks are $1/3$, $2/3$, and 1. One extra registration tick shifts the boundary by one tick. Early wrong repair does not prevent the event. These checks validate the simulator's timing and exposure calculations. They do not test whether an LLM learns an optimal stopping rule.

E How Stored State Can Affect Later Decisions

A stored lesson must reach a decision process before it can help, but reaching that process does not guarantee improvement. The following propositions formalize this distinction.

Proposition 1 (No path, no effect)**.** *Consider a finite unrolled computational state graph whose terminal output is the distribution of the complete decision (action and start time). Suppose an intervention changes only component v, preserves the other inputs and exogenous-randomness law, and v has no directed path to the terminal output for almost every test case. Then the pre- and post-update decision laws agree almost surely, and their expected loss difference is zero.*

Proof. Couple the two executions by the same input and exogenous-randomness realization. Because only v changes, every node outside its descendants retains its value. The terminal decision law is such a node by assumption. Integrating over the unchanged input and randomness distributions preserves equality of decision laws and expected losses. □

This proposition does not say that every observation of no behavioral change proves absence of a path. A reachable update can be ignored, compensated, or irrelevant. Nor does it imply that a mislocalized prompt has no possible indirect influence; that would require verifying the implemented computational graph.

Proposition 2 (Propagation-quality decomposition)**.** *For deterministic pre- and post-update policies π^- and π^+ returning the complete decision (action and start time), let $A = \{x : \pi^-(x) \neq \pi^+(x)\}$. If $P(A) > 0$, then*

$$R(\pi^-) - R(\pi^+) = P(A)\,\mathbb{E}\big[\ell(\pi^-(x), y) - \ell(\pi^+(x), y) \mid A\big]\,. \tag{11}$$

If $P(A) = 0$, the difference is zero.

Proof. Outside A the complete decisions, and therefore their losses under the same adjudication, agree. Split the expectation over A and its complement. □

Changed decisions need not improve performance because the conditional loss difference may be negative. The statement is explicitly deterministic. A stochastic extension requires comparing decision laws or specifying a coupling; a changed sampled answer alone is not proof that the underlying policy distribution changed. The main study uses expected paired loss contrasts and does not infer mediation from a written acknowledgment of a lesson.

E.1 Instance repair and persistent correction

A source episode and a future evaluation episode are different objects. Editing the source answer changes a recorded outcome; updating a lesson bank or its use interface changes an input to later decisions. Evaluating new cases with a frozen bank separates these two changes.

Proposition 3 (Output-only repair leaves future risk unchanged)**.** *Let the repaired source instance be excluded from the test distribution P_c^{test}. Suppose the repair changes only that instance's stored answer, and leaves the persistent inputs, planner, lesson-use interface, and randomness law of future decisions unchanged. Then the future decision laws agree P_c^{test}-almost surely and the held-out risk contrast is zero.*

Proof. For each held-out case, the two decision processes receive the same persistent inputs and follow the same transition and randomness laws. They therefore induce the same distribution over actions and completion times. Applying the same loss and averaging over the fixed test distribution gives equal risks. □

A repaired answer that is subsequently retrieved as evidence creates an additional state pathway; that case belongs to the retained-experience comparison. In the main study, every arm already retains experience. The measured contrast changes the use interface of the same typed bank, rather than comparing a repaired source answer with an otherwise unchanged learner.

For the same-record experiment, the acquired bank, planner, and readable records are fixed, while applicability-based constraint markings change. The contrast tests the combined applicability check and constraint marking, at constant readable content. It does not separately test construction or diagnosis; the reflection comparison also changes construction. BM25 changes the readable set and has its own matched panel.

F Earlier Single-Lesson Studies

The earlier single-lesson program used a different score and a different inferential unit. Its score charged 0.60 for failing to make the warranted immediate correction, 0.25 for a specified wrong crest-world action, and 0.15 for an action outside the allowed set. It lacked the multi-lesson study's explicit service schedule, resource occupancy, and start-time search. Old and new scores are not pooled or retroactively rescaled.

Table 16: Earlier single-lesson executions, retained for scope and sensitivity. Gain is pooled reflection loss minus gated loss under the original score. These rows are not replications of the present multi-lesson contrast.

Model/run	Lessons/class	Gain	Recorded interpretation
GPT C1	500	.0155	Non-rejection; transport-compromised execution retained.
GPT C2	500	.1753	Original primary rule passed after a prospective rate-safe amendment.
Claude T4	167	.2524	Strict-output rule passed; differential ancillary-format failures affected the gain.
Claude T4B	167	.0000	Independent action-only follow-up; primary transfer rule not met.
Gemma T5	167	.0011	Primary transfer rule not met; mixed class effects.

C2 passed its prespecified primary and strengthening rules and the empirical-Bernstein primary sensitivity check, but not the full Hoeffding check. T4 passed the strict-output rule but neither full sensitivity check. In T4, 1,527 of 3,006 reflection slots and 76 of 3,006 typed slots failed the strict contract. A post-hoc action-only diagnostic reduced the apparent gain, and the independent T4B follow-up found a zero paired contrast. These observations motivated the separately locked multi-lesson study. Table 16 retains their original outcomes and inferential status.

Typed escalation in valid adjacent cases avoided some wrong-action charges without performing the required direct correction. Consequently, the old protection claim is not evidence of correct adjacent repair. The present study charges for escalation and models service delay, separating it from successful direct correction. The early causal-stream, adversarial-hint, and rotating-asset pilots are outside the primary evidence for this paper.

G Execution Records and Reproducibility

G.1 Counts and failures

The main paid analysis retains 162,000 planned evaluation slots. GPT has 75 failed typed constructions among 2,250 banks. Because both typed arms share those states, the failures produce 1,800 slots scored as one. Claude has no construction failures. The execution ledger contains 173,700 logical generation requests, of which 173,697 completed. The three terminal ITT failures are two GPT output-cap events and one Claude permanent request error. Credit rejections before generation were reconciled under the same identifiers, without replacing banks.

The follow-ons retain 32 GPT upstream-failure slots for placement, 72 for audit corruption plus one reflection-output failure, and 48 for retrieval. The correct-source panel has none. Claude's follow-on evaluations completed. There were 23,727 successful new calls and one terminal failure; upstream failures remain outcomes without requiring calls. Recorded API charges were $674.32 for the main study and $81.22 for the follow-ons. These are historical charges, not current estimates.

G.2 Artifact bindings and frozen-state checks

Verification covers the main completion seal, frozen source inventories, 9,000 reconstructed source/state records, 162,000 outcome hashes and scalar scores, bank/load balance, pilot exclusion, rebuilt block vectors and sequence decisions, and matched-parent joins. These are computational checks rather than an independent execution audit.

States are serialized after all construction pages are assembled and pass source-fidelity validation. Both providers complete construction before any evaluation is submitted. The state objects reject explicit evaluation-time writes, and the model receives no tool or persistent-write interface. Request construction verifies the state hash. Scoring reconstructs the submitted request and checks its memory against the read view, linking source, assembly, state, read trace, response, and outcome for all 162,000 slots. The typed arms share content and source hashes, with only their arm/placement wrapper differing.

These checks verify the frozen-state contract at construction, request assembly, and scoring. The saved `state_unchanged` field is a contract constant, not an independent write monitor. Matching endpoint hashes cannot exclude a transient write followed by restoration; no kernel-level write audit was performed. Artifact identifiers and verification hashes are provided in the reproducibility package.

G.3 Statistical reproduction

The statistical bundle includes the original D10 implementation and configuration, exact paired block vectors for T3 and all three protected tiers on both models, the aggregation rules, and the fixed testing sequence. Failed constructions and all planned evaluation slots remain represented. These inputs reproduce the full intersection tests, not just the transfer means.

A single entry point regenerates the 64 component estimates and bounds, ten fixed-sequence decisions, and Hoeffding report. It also produces the descriptive breakdowns and optional post-hoc bootstrap, which does not replace the locked analysis. Reproduction in a clean directory on a different Python minor version returned byte-identical numerical outputs without network access or model calls. Independent users should reproduce the original decisions before adding analyses.

G.4 Released material and export transformations

The statistical release contains the exact block vectors, 162,000 loss-level slots and their 27,000 block-arm-tier means, follow-on vectors, frozen timing layer, executed method trace (Appendix H.3), and immutable input/output manifests.

The scientific companion adds the 2,250 registered banks shared by the paid models, 9,000 construction records, 162,000 normalized outcomes, and frozen generator, router, scorer, and controller code. It supports offline replay from registered input to score. Raw provider HTTP bodies and operational journals remain private, so the export does not independently decode every original provider response. Collecting new responses and auditing execution independently are separate tasks.

Exported identifiers are opaque. Credentials, billing records, account identifiers, operational journals, and repository metadata are excluded. Service request identifiers are removed from the derivative records. Export-file hashes verify the released bytes; retained original hashes point to the private provenance records. The original-to-public mapping remains private.

The manuscript source archive contains the paper, bibliography, style files, and table sources. It is separate from both experimental packages. The broader diagnostic, optimal-timing, retrieval, and cross-domain studies discussed as future work are not represented as completed evidence.

H Interface and Information Boundaries

This appendix follows the information supplied from source audit to scored decision. The companion release includes serialized records, reconstructed requests, and matching code for replay.

Table 17: Lesson fields illustrated by the stale-report example. These are explanatory meanings; Appendix H.3 gives the serialized fields of an executed case.

Field	Stale-source example and information origin
g, trigger	The audit-bound warning condition for the source; a fresh report excludes the stale-report trigger.
S, scope	The source situations governed by this rule, excluding cases requiring the different crest or resource correction.
u, correction	Quarantine and verify the report, rather than automatically rerouting.
v, component	The source-use component responsible for that correction.
d, timing	Conservative source completion information, distinct from the evaluator's private deadline.
$\mathcal{P}$, evidence	The supporting audit linking the correction and applicability fields to the source episode.

H.1 Source evidence, stored bank, and decision context

Source evidence supplies a candidate lesson and the scope, trigger, component, timing, and provenance fields to which a model-selected correction can be bound. The constructed bank is the persistent object produced before evaluation. The decision context is the material a particular lesson-use mechanism exposes on a live case. The typed comparison holds the bank and model fixed while changing the decision context through the added constraint list.

The constructor checks whether the selected correction and fields are supported by the admitted source. This check establishes source fidelity; the source itself may still be wrong. In the clean panel the supplied audits support the intended correction; in the error panel the same process can preserve an erroneous assertion faithfully. Provenance identifies the supporting source but cannot, by itself, establish its correctness. Even the correct-source reference can fail later because the unchanged planner still chooses the action and schedule.

H.2 What each component may see

Table 18: Information boundaries of the completed study. "Not supplied" is an experimental restriction, not a claim that the corresponding fact is unknowable in every deployment.

Component	Admitted information	Not supplied as an answer source
Lesson construction	Bank audits and their source-bound fields; assigned construction slots	Future evaluation answers or private truth used to repair a source claim
Applicability routing	Constructed lesson fields and admitted live-case information	A private future outcome used to select a successful record
Frozen planner	Live observations, the exposed records, public resource/service information, and conservative timing information	True latent world, private completion deadline, or future event log
Adjudication	The frozen loss and private simulator quantities required to score correctness and completion	Authority to replace the model's proposed action before scoring

The private deadline and the conservative source timing bound have different roles. The first decides whether an action actually prevented the modeled event. The second is information the planner may use. Replacing the source bound with the private deadline would change the task. Similarly, a human request is an action with a public service schedule and delay, not immediate access to an oracle.

H.3 The executed gate and one complete trace

The rule below is taken from the executed code. A saved decision illustrates each step; its machine-readable trace accompanies the implementation in the release.

The rule. A record is eligible for a live case when three conditions hold together: its acting-component token equals the case's acting component exactly; its ordered scope tuple equals the case's scope tuple exactly, element by element and in order; and both of its bound predicates are satisfied. Predicates are inclusive comparisons, $\geq$ or $\leq$, against integer-valued public observed features, and a predicate whose feature is absent from the live packet fails. Matching preserves stored record order, and every matching record is returned. Applicability is checked symbolically using public tokens supplied by the generator, rather than inferred from natural-language descriptions. The check does not rank records, consult private evaluation truth, or choose an action. The bound completion field travels with a matched record as information and is never evaluated as a third numerical eligibility test. Under the no-scope ablation the scope equality is dropped and the other two conditions remain; under the no-timing ablation the completion field is removed from the constraint payload while remaining present in the readable record.

What is supplied to the planner. The readable view contains every stored record, except that ANCHOR first restricts all arms to its named source, inactive placement supplies none, and BM25 supplies only its four retrieved records. The matched records are additionally placed in the acting component's constraint list (`registered_constraints`). Whole-bank active memory receives an empty constraint list over the identical readable set; mislocalized placement leaves the records readable under the nonacting component and populates no constraints. The gated and active states share a content hash and differ only in an arm/placement wrapper, so the two arms are byte-identical in what they store.

One executed decision. This example uses the first bank in the saved registry and its GPT INTERFERENCE/T3 episode. It was selected by registry position, not outcome. All three arms chose the same action in this instance. The eight-record bank contains source `R_d8aaeeb049e6`, whose audit supplies the ordered scope `(S_cd3836d0d6a5, S_ad016b32efb4)`, the predicates `F_91e340ca0e4c` $\geq$ 32 and `F_d4579f52946a` $\leq$ 56, the acting component `P_a549cef48a37`, and a conservative completion bound. At construction the model selected `OPTION_A` for this slot; the public option map resolves it to action `A_c0cfe347d946`, the compiler verified support in the audit and bound `complete_within_ticks = 8`, and the remaining seven selections also passed, so installation succeeded atomically. On the later case the observed features are 42 and 46, so both predicates hold; scope and component match; the current tick is 0, the action's lead time is 2, no resource is occupied, and the human window opens at tick 4 with a five-tick delay. Both typed arms receive all eight records; the gated arm additionally lists `R_d8aaeeb049e6` in the constraint list. Each arm returned `A_c0cfe347d946` at tick 0. Completion falls at tick 2, ahead of a private deadline of tick 8 that was never supplied to the planner, so the action is warranted, schedule-valid, and scored as direct prevention with loss 0.05, the intervention charge alone.

One match in each recorded context. All 35,400 recorded gated contexts outside ANCHOR returned exactly one record, across T3, T5_SCOPE, T5_TRIGGER, and BENIGN. The remaining 600 slots inherit construction failure and have no constraint list. The generator partitions each bank's cases among its records. A case outside a source lesson's scope or trigger therefore matches another governing record rather than producing an empty list. As noted in Section 4.2, empty and multiple matches remain untested.

H.4 FROM CONSTRUCTION TO SCORING

For each independently constructed bank, the experiment follows four steps.

1. **Construct.** The model selects source-supported corrections. The deterministic constructor binds the remaining fields and the validator checks fidelity and size. A failed bank remains a failed unit; it is neither replaced nor partially repaired with evaluation truth.
2. **Freeze.** Store and hash the completed bank. Disable later writes to the learned persistent state. Fresh evaluation observations are still permitted; frozen memory does not mean frozen input.
3. **Present.** TYPED-ACTIVE supplies the bank as ordinary active memory with no constraint list. TYPED-GATED supplies the identical readable bank and additionally marks its matched records as the acting component's constraint list. The readable content is unchanged between the two; the constraint list is the intervention.

4. **Decide and score.** The same frozen planner chooses the action and start time. The fixed adjudication computes loss, prevention, and protected-case outcomes without substituting a repair.

Each scored decision retains its bank hash and supplied context, enabling the replay described in Appendix G. The steps correspond to Algorithm 1.

H.5 What the ablations can identify

Inactive and mislocalized placement change access and the route to the acting component, respectively. Their read traces establish the implemented interfaces, not whether the model's verbal acknowledgment faithfully describes its internal computation.

The main comparison holds stored content and the readable record set fixed. It therefore excludes record removal and prompt shortening as explanations. It combines checking applicability with presenting the result as a constraint to the acting component. Placement controls examine that connection. The marking panel (Section 5.5, Appendix M) holds the presentation fixed and varies only which record is marked: the benefit requires the correct designation, but because a wrong mark also supplies wrong guidance, the panel does not isolate predicate computation from the content it designates. The smaller BM25, placement, and gate-removal panels do not establish a causal percentage of the main gain, and their effects should not be added as if they were disjoint components of one loss.

Relation to the upstream ERM/RLER work. The ERM/RLER manuscript audits expressed causal commitments and uses an interventional evidence log for external strategy adaptation (Anonymous, 2026). It motivates one possible source of supported lessons. The present experiment uses source-audited records; it does not run an ERM judge, install its outputs, and then measure end-to-end transfer. A demonstrated composition of the two systems would require its own source-error accounting and frozen evaluation. No completed end-to-end ERM–LBF experiment is claimed here.

I Extended Related Work and the Scope of the Comparison

This appendix maps the external literature to the interfaces examined in LBF. It complements Section 2; results on the LBF benchmark are reported for the implemented arms in Section 5.

I.1 From experience to reusable rules

Case-based reasoning organizes experience reuse through retrieval, reuse, revision, and retention (Aamodt & Plaza, 1994). Explanation-based generalization formalizes how an explanation of an example can support a more general operational description (Mitchell et al., 1986). Soar's chunking mechanism learns production rules from problem solving, making the connection from experience to later control explicit (Laird et al., 1986). LBF follows this tradition of turning experience into rules for later decisions.

Our experiment holds an acquired typed bank constant while changing how a fixed-weight language planner receives it. This removes differences in stored representation from the comparison and tests the combined effect of checking applicability and marking constraints.

I.2 Prevention, uncertainty, and the role of time

Sequential inspection and change detection already trade delayed response against false alarms (Page, 1954; Lorden, 1971; Lai, 2001). Machinery diagnostics and prognostics connect condition information to maintenance decisions (Jardine et al., 2006), and multi-machine scheduling can explicitly price imperfect interventions (Ruiz-Hernandez et al., 2020). These methods address uncertainty, costs, and operational constraints that remain important here.

LBF borrows the requirement that a preventive action finish while it can still change the outcome. The exact timing calculation in Appendix D.4 verifies our simulator's deadline semantics. The deadline and cumulative loss serve as established evaluation concepts; the empirical contribution concerns the interface delivering acquired lessons to the planner.

I.3 Language agents that learn from experience

Reflexion uses retained textual feedback to improve subsequent trials (Shinn et al., 2023). ExpeL extracts reusable insights from collected experience (Zhao et al., 2024), while contextual experience replay synthesizes earlier trajectories into context for later decisions (Liu et al., 2025). These approaches already make an experience-to-behavior connection. Our REFLECT-DIAG comparator instantiates model-written operational notes under the common source evidence and resource ceilings. The other cited methods provide conceptual comparisons.

More structured predecessors are especially close. Agent Workflow Memory induces reusable routines (Wang et al., 2024); Voyager stores executable skills and retrieves them for subsequent tasks (Wang et al., 2023); hierarchical MetaFlows combine reusable experience with adaptable workflows (Fan et al., 2025). These systems combine structured reuse with adaptation. Our comparison tests applicability checking and constraint marking after the acquired bank has been frozen.

Experiential Reflective Learning further emphasizes selective retrieval of transferable heuristics (Allard et al., 2026). This motivates the BM25 comparison, where retrieval changes which records are readable. LBF-Gate instead marks applicable records without removing the rest of the bank, and outperforms ordinary whole-bank memory in this environment. These comparisons leave open whether a better learned selector, heuristic retriever, or executable skill representation removes that advantage.

I.4 Memory retention, use, and selection

MemGPT manages external memory through an explicit memory architecture (Packer et al., 2023). A-MEM organizes linked memories dynamically (Xu et al., 2025). LoCoMo measures long-horizon conversational memory (Maharana et al., 2024); Mem2ActBench directly evaluates proactive memory use in tool choice and parameter grounding (Shen et al., 2026). LBF complements these memory-use benchmarks by holding stored records fixed and measuring both benefit on later recurrences and cost on protected cases.

Liu et al. (2024) show that evidence position can affect long-context use. This motivates context composition as one candidate explanation for differences among our arms. BM25 (Robertson & Zaragoza, 2009) is the lexical retrieval mechanism actually evaluated. Dense passage retrieval (Karpukhin et al., 2020) provides a semantic-selection family for future comparison. Its performance on our opaque-token lesson banks remains to be measured.

Table 19: Close precedents and the additional comparison made here. The final column identifies the LBF experiment's scope, not a deficiency established about the cited work.

Precedent	What it already contributes	What the LBF comparison isolates
Case-based reasoning, EBG, Soar	Experience reuse and acquisition of operational rules	Effect of a use interface when acquired records and a language planner are fixed
Reflexion, ExpeL, CER	Feedback or experience retained for later behavior	Same typed records under two interfaces, with a separate reflection comparator
Voyager, AWM, MetaFlows	Executable skills or reusable workflows	A tested routing implementation, not superiority over skill compilation
MemGPT, A-MEM	Explicit management and organization of persistent memory	Behavioral loss after a frozen construction boundary
LoCoMo, Mem2ActBench	Evaluation of retention and, explicitly, memory-driven action	Recurrence-class transfer plus separate protected-case bounds
BM25 and dense retrieval	Selection of relevant records or passages	Lexical retrieval tested descriptively; dense selection remains open
ERM/RLER	Causal critique and external evidence-based adaptation	Downstream use of supplied audits, without an end-to-end integration claim

I.5 Learning, source quality, and unintended costs

Reinforcement learning can change a policy directly, and continual-learning methods such as GEM explicitly consider retaining performance while learning new tasks (Sutton & Barto, 2018; Lopez-Paz & Ranzato, 2017). LBF freezes weights to isolate external-state use. Every main arm has access to acquired experience; their comparison measures how effectively that experience is used.

The ERM/RLER manuscript concerns causal support and external adaptation (Anonymous, 2026). Our corrupted-audit panel concerns a different downstream boundary: faithfully installing a false source assertion can still degrade performance. As a related preprint in a different domain, Li & Zhu (2026) examine capability gains, security regressions, and execution-interface mismatch in financial agents. In LBF, the evidence for source dependence and protected-case trade-offs comes from the panels reported in Sections 5.4 and 5.7.

J Claims and Supporting Evidence

LBF is the framework; LBF-Gate is its tested gated instantiation (the original Typed-Gated arm). The lesson schema, requirements for use, and evaluation criterion define the framework. RQ1 and RQ2 test the implementation's transfer benefit and costs on protected cases. RQ3 and RQ4 report the separately prespecified marking-control panel (Appendix M); RQ5 reports smaller paired follow-ons. Table 20 maps their evidence and scope in one place; the original inferential decisions are retained in Appendix B. The supporting definitions in Appendix K distinguish recurrence from correction equivalence; Appendix L specifies completion and exposure.

Using the evidence map. The confirmatory claim concerns LBF-Gate's combined same-record routing intervention. Persistence is fixed at the construction/freeze boundary, not separately ablated; reachability motivates placement controls; scope and trigger motivate protected populations; and timeliness motivates completion checks and timing controls. The main test combines these requirements; it does not independently establish each one as necessary or sufficient. Other mechanisms need their own evaluation. The identical readable banks exclude selection and prompt shortening from the main intervention, leaving applicability checking and constraint marking to be separated by future controls (Section 6.2). Appendix E states the assumptions behind the pathway propositions. Section 5.3 and Appendix B report the stronger margin that was not certified and the Hoeffding sensitivity result. Section 5.7 gives the mixed timing results and the uncertainty of the retrieval and corruption panels.

Extensions. The separate Gemma follow-up (Appendix N) found lower observed loss for both typed approaches than for reflection, but did not establish an added loss benefit from gating. Its post-hoc action and schedule breakdown describes where correct choices failed to produce prevention; it is not a separate confirmatory mechanism test.

Dense or learned retrieval, a fully compiled-policy comparator, prospective component-level confirmation, autonomous diagnosis, stale-lesson revocation, cross-organization transport, and an end-to-end ERM–LBF pipeline are distinct next tests. The present same-record result establishes a behavioral benefit against its specified comparators and supplies the basis for those broader comparisons.

K Correction Targets, Recurrence Classes, and Transfer

A lesson carries a reusable correction rule. Each new case supplies the objects, resources, and timing to which it applies. This appendix defines when cases share that rule and how that relation differs from belonging to the same recurrence class.

K.1 The correction includes an action, a timing rule, and a recipient

For an episode e at its prespecified decision time, write its correction target as

$$\psi(e) = \big(u(e), \delta(e), v(e)\big), \tag{12}$$

where $u(e)$ is the supported correction, $\delta(e)$ is the rule governing preventive completion, and $v(e)$ is the state component responsible for that correction. This expands the roles of u, d, and v in the

Table 20: Evidence supporting each claim and the scope of its interpretation. Primary and descriptive results are distinguished.

Claim	Supporting evidence	Boundary
LBF-Gate improves the typed-bank comparator	Same-record contrast on both paid models; three classes; 375 load-paired blocks per class	Combined effect of predicate evaluation and constraint marking, not identification of either alone
Gain extends beyond source replay	Frozen T3 evaluation under prespecified topology, timing, identity, and parameter changes	Transfer within the constructed recurrence families, not arbitrary domain transport
Protected losses meet the stated criterion	All prespecified D10 upper bounds below 0.10	Relative degradation tolerance, not zero harm or a field-deployment guarantee
Placement and gates may explain part of the effect	Paired inactive, mislocalized, no-scope, and no-timing panels	Descriptive; range-only intervals include zero; post-hoc bootstrap and mixed timing results reported separately
The benefit depends on which record is marked	Prespecified marking panel: random-minus-correct one-sided lower bound positive in every class on both models; 125 blocks per class	GPT's reference arms historical (six days earlier, batch); Claude's re-run on T3 without detectable drift; not a content-independent marking effect, but applicability information and constraint presentation remain jointly responsible; D10 positive except Claude stale-source
A wrong mark can be worse than no mark	GPT: random minus no mark 0.113 [0.091, 0.135]; wrong marks add 0.13–0.22 on scope and trigger tiers; GPT escalates 56% of affected T3 decisions and executes the wrong correction in 8%	Model-dependent: Claude stays near no mark; compliance analysis is post hoc; drift sensitivity in Appendix M
The effect is not context reduction	Request-hash replay: identical readable records in both typed arms, larger gated request	Excludes shortening and distractor removal; does not separate predicate checking from marking
Gating outperforms BM25 on the descriptive subset	BM25 top-four panel, bootstrapped on matched blocks	$n = 50$, one retriever; no equivalence test, dense comparison, or causal share of the main effect
Every recorded gated context has one match	Exactly one record matched in all 35,400 recorded non-ANCHOR gated contexts; 600 upstream-failure slots retained	A generator property, not a gate guarantee; zero-match and multiple-match cases remain untested
Source errors can propagate	Matched one-in-eight corrupted-audit panel	Ten pooled blocks; one error rate; no certified robustness threshold
The task has a prevention boundary	Exact 1,152-cell timing sweep	Simulator verification, not new optimal-stopping theory

compact lesson interface of Equation (1): d records the completion rule, while δ emphasizes that a reusable timing convention can be instantiated with a different numerical deadline in each episode.

Correction equivalence is defined by $e \sim_{\mathrm{corr}} e'$ exactly when $\psi(e) = \psi(e')$, with targets expressed at the level of the prespecified reusable rule. This relation differs from membership in a recurrence class. A recurrence class fixes the mechanism and trigger structure while permitting declared nuisance changes. Distinct causes can warrant the same correction; conversely, a recurring cause may require a different response when the available time or responsible state changes. The task specification fixes which variations preserve the target before evaluation.

These distinctions explain the protected populations. Surface similarity is a retrieval cue, whereas correction scope determines whether a lesson governs the current decision. The T3 cases preserve the prespecified recurrence rule under the stated variations. The protected scope, trigger, and benign populations separately test decisions on which that lesson should be restricted or withheld.

K.2 Example of a reusable correction

Consider the stale-source lesson from the main example. Its transferable content is a rule for handling a report whose provenance or freshness is defective; its recipient is the component that governs source use. A different route identifier or resource allocation changes the live instance while preserving that source-handling rule. A fresh report falls outside the stale-source trigger. Evidence that the flood itself is advancing calls for the different crest correction. Thus the same surface warning can lead to a covered recurrence, a trigger exclusion, or a correction-distinct case, depending on the admitted evidence.

For the typed arms, the audit supplies the scope, trigger, recipient, and conservative completion information; the constructor binds a model-selected correction to those fields. The gate checks these fields to mark applicable records alongside the full readable bank. The planner still chooses an action and start time for the current resources. Appendix H.3 gives the exact fields for one executed example.

K.3 Propagation, persistence, and the measured comparison

Propagation delivers a lesson to a decision process; transfer measures its effect on untouched cases. The completed benchmark evaluates frozen recurrence-class reuse.

Equation (3) in Section 3 separates acquiring an update from using an already acquired bank differently. The measured same-record contrast holds the typed bank fixed, including its construction failures, and averages over source-bank construction and later decisions. It estimates the added benefit of applicability checking and constraint marking. A comparator without prior experience would instead test the value of acquiring the bank.

A frozen evaluation preserves the learned bank while allowing new live observations. The prespecified recurrence class determines which changes count as transfer; the protected populations determine where a useful lesson can instead cause harm. These two endpoints are reported together under the original block-level inference in Appendix B. The general persistence horizon of a deployed lesson is an additional contract parameter; the present evidence covers the completed frozen evaluation horizon.

L Action Completion and Unresolved Exposure

The prevention endpoint concerns completed action. This appendix defines the timing used by the intervention score and the exact sweep in Appendix D.4.

L.1 From a decision to a completed correction

For a supported action a, let t_a be its selected start time, $\tau_{\mathrm{exec}}(a)$ its execution latency in the fixed-latency subcase, and $d_w(a)$ the latest preventive completion time in the simulator world. Timely completion requires

$$t_a + \tau_{\mathrm{exec}}(a) \le d_w(a), \qquad t_a^{\mathrm{last}} = d_w(a) - \tau_{\mathrm{exec}}(a). \tag{13}$$

In the full scheduling interface, resource occupancy or human availability determines a completion function $t_{\mathrm{done}}(a, t_a, x)$; the same condition is $t_{\mathrm{done}}(a, t_a, x) \le d_w(a)$. The fixed action catalog and resource constraints remain part of that test. The agent receives conservative timing information from the source, while $d_w(a)$ remains private to adjudication.

Let t_{ready} be the time when the required information is available to the responsible acting process. In the fixed-latency subcase, the remaining start-time slack is

$$\sigma_a = d_w(a) - \tau_{\mathrm{exec}}(a) - t_{\mathrm{ready}}. \tag{14}$$

With positive slack, the agent can wait before starting. Zero slack requires an immediate start. Negative slack means that this action cannot finish in time. Additional constraints may make a nonnegative-slack action infeasible, so a timely slot is checked together with the action and resource conditions. A task-level statement that prevention is impossible requires examining all admissible corrective actions, rather than one late action.

The completed benchmark uses an inclusive completion boundary: finishing exactly at the deadline counts as prevention. The exhaustive sweep tests this convention explicitly. Registration and probe delays shift the point at which the acting process becomes ready, explaining why a lesson's presence in storage is different from its availability at the last useful decision point.

L.2 Accumulated exposure and the terminal event charge

For the completed simulator, the canonical censored unit-exposure record, with onset zero, is

$$E_{\mathrm{cens}}(t_{\mathrm{corr}}, d) = \begin{cases} 0, & \text{benign case,} \\ \min\{t_{\mathrm{corr}}, d\}, & \text{nonbenign case,} \end{cases} \tag{15}$$

where t_{corr} is valid-correction completion time, set to $+\infty$ when correction remains unresolved, and d is the private completion deadline. The symbol E_{cens} denotes duration, distinct from the paired loss contrasts D in Appendix A; t_{done} denotes completion rather than intervention cost C_{int}. The bounded mission loss in Appendix A.4 instead combines intervention cost, the unprevented-event charge $0.60F$, wrong-repair cost, scheduling cost, and the separate ITT rule. Two delayed decisions can have the same censored exposure yet differ in whether a valid correction finishes exactly at the deadline. Conversely, a correct early action still incurs its intervention cost. These examples show why duration, completion, and mission loss are recorded separately.

L.3 Simulator validation and model performance

The exact 1,152-cell sweep validates completion at equality, failure one tick later, the latency of concurrent and serial evidence acquisition, and the shift induced by registration delay. The LLM experiment then evaluates actions and start times selected by the model under those semantics. The sweep validates the scoring convention; the model's scored decisions measure performance. The prespecified transfer contrasts, not the simulator checks, establish the benefit of gating.

M Marking-Control Panel

This appendix reports the panel behind RQ3 and RQ4 (Sections 5.5 and 5.6). It was specified, locked, and executed after the main study, and it sits outside the main study's fixed sequence: it has its own prespecified decision rule and does not alter any earlier decision. It is a request-matched comparison against historical reference outcomes: its correct-mark and no-mark arms are the main study's rows for the same units, and RQ3 and RQ4 are two analyses of this one panel, not independent replications.

M.1 Design, status, and amendments

Question and arms. Every recorded gated context in the main study matches exactly one stored record, so LBF-Gate's treatment is the marking of one readable record. The panel asks whether the benefit requires that record to be the *correct* one. Four arms share the identical readable bank and differ only in the acting component's constraint list: the matched record (Correct, the main study's Typed-Gated rows), no record (None, the main study's Typed-Active rows), one non-matching

record drawn uniformly with a seed fixed per unit (Random), and the non-matching record nearest to matching (Near miss), ranked by component equality, then the number of unsatisfied scope and predicate conditions, then total threshold violation, with ties broken by canonical record hash. No unit lacked a non-matching record.

Requests and scoring. Every request, provider payload, decision, and score was produced by the completed study's own code. A wrong-mark request is the executed LBF-Gate request with its single constraint replaced by the substitute record, serialized exactly as the gate serializes a matched record; the system prompt, task envelope, output schema, and every other field are byte-identical. Before any paid call, tests confirmed that the reference requests rebuilt through this path reproduce the executed request hashes, that wrong-mark requests differ from the executed LBF-Gate request only in the constraint list, and that the scorer reproduces the executed outcomes, including intention-to-treat losses for failed banks.

Units and models. The panel uses load-paired blocks 10–134 of each recurrence class (125 blocks per class, 6,000 units per arm per model) for `gpt-5.4-mini-2026-03-17` and `claude-sonnet-4-6`. Blocks 0–9 formed a plumbing pilot that validated the pipeline and are excluded. Model settings match the main study. The block pairing and aggregation reproduce the main study's published contrast vectors exactly when applied to its own reference rows.

Decision rule. H1 (Random $-$ Correct > 0) passes only if the one-sided 95% paired block-bootstrap lower bound (20,000 resamples, seed 20260921, blocks resampled whole) is positive in every class on both models. H2 (Random $-$ None, two-sided), H3 (Near miss $-$ Correct), and the D10 bounds are reported alongside. The recovered fraction $\rho = (\text{None} - \text{Random})/(\text{None} - \text{Correct})$ was to be read as requiring a correct mark if $\rho \leq 0.25$, as largely salience if $\rho \geq 0.75$, and as mixed otherwise, on the balanced point estimate with its interval reported.

Amendments. Five dated amendments were recorded before any outcome of the panel was inspected. Amendment 1 replaced a prototype request builder with the study's own code after two single-unit probes showed that the prototype prompt differed from the study's; neither probe was scored. Amendments 2–4 changed execution only: concurrency with timeouts and retries, stopping on exhausted spend limits, and retrying every server-side error. No received response was ever redrawn; retries occurred only when no response arrived, and invalid outputs were scored as failures. Amendment 5, a concurrent rerun of the reference arms, was drafted and withdrawn before execution, so the reference arms are the main study's rows as the protocol originally specified. The analysis code was hashed and recorded in the protocol before the panel's data were read.

Integrity. The structural audit found 12,000 terminal rows per model and no unresolved errors. Claude has 12,000 scored decisions; GPT has 11,632 scored decisions and 368 intention-to-treat rows inherited from the 23 GPT banks in this window whose construction failed in the main study. One GPT response was invalid and is scored as a failure; Claude produced none. Arms are balanced at 6,000 rows each, and no wrong-mark row marks the matched record.

M.2 Results

Table 21: Marking panel: mean held-out T3 loss by marked record (primary conditions; 125 load-paired blocks per class per model). Correct and None are the main study's rows for the same units.

Model	Group	Correct	None	Random	Near miss
Claude	Crest	0.1521	0.2557	0.2584	0.2436
	Stale source	0.1545	0.2348	0.1940	0.1905
	Resource	0.1634	0.2429	0.2480	0.2301
	Balanced	0.1567	0.2445	0.2335	0.2214
GPT	Crest	0.1053	0.2289	0.2937	0.3090
	Stale source	0.0818	0.1904	0.3031	0.3025
	Resource	0.0724	0.1645	0.3259	0.3062
	Balanced	0.0865	0.1946	0.3076	0.3059

Table 22: Marking panel contrasts on T3 (minuend minus subtrahend; positive means higher loss). H1: Random − Correct; H2: Random − None; H3: Near miss − Correct; G: None − Correct. Cells give the mean, then the one-sided 95% bootstrap lower bound and the D10 lower bound; H2 gives its two-sided 95% bootstrap interval. The prespecified decision uses the bootstrap bound for H1 in every class.

Model	Group	H1	H3	H2	G
Claude	Crest	.1063 (.0818 / .0568)	.0915 (.0672 / .0425)	.0027 [−.0302, .0357]	.1036
	Stale source	.0395 (.0212 / −.0042)	.0360 (.0164 / −.0084)	−.0408 [−.0658, −.0163]	.0803
	Resource	.0846 (.0639 / .0383)	.0667 (.0454 / .0204)	.0050 [−.0246, .0339]	.0795
	Balanced	.0768 (.0657 / .0354)	.0647 (.0515 / .0230)	−.0110 [−.0275, .0055]	.0878
GPT	Crest	.1884 (.1620 / .1350)	.2037 (.1769 / .1494)	.0648 [.0248, .1049]	.1236
	Stale source	.2213 (.1950 / .1664)	.2207 (.1937 / .1656)	.1127 [.0745, .1508]	.1086
	Resource	.2535 (.2302 / .2000)	.2338 (.2094 / .1799)	.1614 [.1272, .1957]	.0921
	Balanced	.2211 (.2072 / .1736)	.2194 (.2029 / .1708)	.1130 [.0907, .1350]	.1081

H1 passes under the prespecified bootstrap rule: the one-sided lower bound is positive in every class on both models (Table 22). The D10 lower bound, reported alongside, is positive in every cell except Claude's stale-source class. H3 has positive nominal bootstrap lower bounds in every class on both models. The balanced recovered fraction is 0.126 [−0.071, 0.284] on Claude, whose point estimate falls in the region requiring a correct mark while its interval crosses 0.25, and −1.045 [−1.402, −0.756] on GPT. Claude's classes differ: on the stale-source class a random mark recovers about half the benefit, a fraction of 0.51 computed from the printed means. The prespecified regions did not anticipate a negative fraction: on GPT a wrong mark removes the whole benefit and adds loss beyond having no mark. For the near miss the fractions are 0.263 and −1.030. The near-miss and random arms were collected together; their contrast shows no detectable difference at the reported precision on both models; this does not establish equivalence.

Table 23: Marking panel protected tiers, balanced: additional loss of each wrong-mark arm relative to no mark and to the correct mark. Cells give the mean with the one-sided 95% bootstrap upper bound and the D10 upper bound in parentheses. Bold marks a bootstrap upper bound at or above $\eta = 0.10$.

Model	Tier	Relative to	Random mark	Near-miss mark
Claude	Scope	None	.0033 (.0115 / .0410)	.0022 (.0114 / .0402)
		Correct	.0298 (.0379 / .0665)	.0287 (.0379 / .0658)
	Trigger	None	.0346 (.0463 / .0725)	.0161 (.0283 / .0549)
		Correct	.0775 (.0897 / .1142)	.0590 (.0710 / .0962)
	Benign	None	−.0091 (−.0050 / .0282)	−.0056 (−.0016 / .0315)
		Correct	.0018 (.0056 / .0386)	.0053 (.0095 / .0420)
GPT	Scope	None	**.1286 (.1417 / .1640)**	**.1323 (.1445 / .1671)**
		Correct	**.1468 (.1584 / .1809)**	**.1505 (.1617 / .1843)**
	Trigger	None	**.1733 (.1859 / .2071)**	**.2229 (.2373 / .2561)**
		Correct	**.1934 (.2058 / .2264)**	**.2429 (.2575 / .2756)**
	Benign	None	.0143 (.0165 / .0504)	.0141 (.0165 / .0503)
		Correct	.0180 (.0201 / .0540)	.0178 (.0197 / .0538)

On the protected tiers, Claude's wrong-mark arms stay below η on the bootstrap bound in every balanced cell; the more conservative D10 bound exceeds it only for Random − Correct on the trigger tier (0.1142). GPT's observed mean degradations exceed η on the scope and trigger tiers against both references (Table 23); these point estimates are descriptive, and an upper bound above η shows only that non-degradation is not established; escalation, not execution of the wrong correction, accounts for those losses (Appendix M.3). These are deliberate wrong marks, so the result describes the cost of an applicability error, not a failure of the gate.

M.3 Compliance with the mark (post hoc)

The loss contrasts show that a wrong mark costs GPT more than no mark, but not how. This analysis, added after the locked analysis at the authors' request, classifies each scored decision by the action

chosen. Its definition was fixed and its code hashed before it was run on the panel's rows, and it is reported as it came out; it enters no prespecified decision. Units where the wrong record carries the same correction as the correct record are excluded, because following the mark and being right cannot be distinguished there (about one fifth of units). The split of the residual category into escalation and other corrections was added after the headline rates were seen; it is exhaustive.

Table 24: Compliance with the mark on T3 (post hoc, descriptive): share of scored primary-condition decisions choosing each kind of action, restricted to units where the marked record's correction differs from the correct one. Correct-mark and no-mark rows are the main study's decisions on the same units. Percentages of n.

Model	Marked record	Correct	Wrong (marked)	Escalate	Other record	n
Claude	Correct mark	99.6	0.1	0.1	0.2	1259
	No mark	89.9	4.1	0.3	5.6	1259
	Random mark	90.2	4.4	0.4	5.1	1259
	Near-miss mark	92.0	4.0	0.2	3.9	1208
GPT	Correct mark	99.5	0.1	0.3	0.1	1218
	No mark	84.1	5.0	2.4	8.5	1218
	Random mark	32.1	8.3	56.3	3.3	1218
	Near-miss mark	31.1	7.6	58.2	3.1	1170

GPT rarely carries out the wrong correction: 8.3% of these T3 decisions under a random mark, against 5.0% without a mark. It escalates instead, in 56.3% against 2.4%, and its choice of the correct correction falls from 84.1% to 32.1%. Of its escalations on T3 under either wrong mark, 50.8% complete in time through human service; the rest leave the event unprevented, which accounts for GPT's added loss. Claude's choices barely move: it selects the correct correction in 90.2% of decisions under a random mark and 89.9% without one. A correct mark raises both models to at least 99.5%.

Invalid outputs and late correct actions. Among T3 decisions that choose the correct correction, Claude's finish after the deadline in 16.4% under a correct mark, 19.6% without a mark, and 17.8% and 17.5% under random and near-miss marks, so its late actions reflect its timing rather than the mark; GPT's finish late in at most 0.4% under any mark. GPT produced one invalid response in the panel and Claude none. These counts were added in response to review and are descriptive.

Table 25: Compliance on the protected tiers (post hoc, descriptive): percentage of decisions choosing the correct correction / escalating, on the same restricted units as Table 24. Wrong-correction rates are at most 1.9% in every cell and are omitted.

Model	Tier	No mark	Random mark	Near-miss mark
Claude	Scope	99.4 / 0.1	98.4 / 0.4	98.3 / 1.1
	Trigger	94.8 / 2.3	85.8 / 9.6	84.3 / 12.4
	Benign	93.3 / 0.3	96.2 / 0.3	94.4 / 0.7
GPT	Scope	97.5 / 1.4	45.1 / 54.1	44.6 / 54.2
	Trigger	96.5 / 3.2	30.9 / 68.9	14.7 / 83.7
	Benign	98.5 / 0.3	34.1 / 63.8	33.2 / 64.5

The protected tiers show the same pattern (Table 25): under wrong marks GPT escalates 54–84% of decisions and almost never carries out the wrong correction, so its protected-tier losses in Table 23 come from deferral, not from applying a lesson where it should be withheld. Claude's escalation rises only on the trigger tier.

M.4 Contemporaneous reference refresh (Claude, T3)

Amendments 6 and 6a, recorded before any refresh call, re-ran Claude's two reference arms on the panel's T3 units (1,500 units, 3,000 calls) through the same synchronous path and runner as the wrong-mark arms, on the same day. Claude was chosen because its conclusions were more sensitive

to reference drift; Appendix M.5 gives the GPT sensitivity calculations. Every refresh request was the completed study's executed request, unchanged, and the structural audit confirmed all 3,000 request hashes. The refresh analysis applies the locked analysis's T3 computation, with identical random draws, to the refreshed references; its code was hashed before the refresh ran. The reporting rule, fixed in advance, adopts the more conservative reading wherever historical and contemporaneous analyses disagree.

Table 26: Contemporaneous refresh, Claude, T3: drift of each reference arm (refresh minus historical loss on identical units), and of G, with paired block-bootstrap 95% intervals. Positive means higher loss in the refresh.

Group	Correct mark	No mark	G (None − Correct)
Crest	−.0077 [−.0182, .0028]	−.0045 [−.0231, .0139]	.0032 [−.0176, .0236]
Stale source	.0009 [−.0119, .0141]	−.0073 [−.0259, .0109]	−.0082 [−.0325, .0151]
Resource	−.0006 [−.0139, .0125]	.0148 [−.0026, .0319]	.0154 [−.0073, .0374]
Balanced	−.0025 [−.0089, .0039]	.0010 [−.0095, .0113]	.0035 [−.0092, .0157]

No reference arm shows detectable drift: every interval in Table 26 includes zero, and the reference arms' balanced intervals lie within ± 0.012. The contemporaneous analysis uses the refreshed references directly, so Claude's conclusions no longer rest on an assumed shift.

Table 27: Claude T3 contrasts against historical and against contemporaneous references. H1 and G give the mean with the one-sided 95% bootstrap lower bound and the D10 lower bound; H2 gives its two-sided 95% interval. The wrong-mark arms are the same rows in both columns.

Group	References	H1	H2	G
Crest	Historical	.1063 (.0818 / .0568)	.0027 [−.0302, .0357]	.1036 (.0791 / .0542)
	Contemporaneous	.1139 (.0898 / .0645)	.0072 [−.0268, .0415]	.1067 (.0829 / .0579)
Stale source	Historical	.0395 (.0212 / −.0042)	−.0408 [−.0658, −.0163]	.0803 (.0561 / .0316)
	Contemporaneous	.0386 (.0222 / −.0039)	−.0335 [−.0599, −.0076]	.0721 (.0512 / .0259)
Resource	Historical	.0846 (.0639 / .0383)	.0050 [−.0246, .0339]	.0795 (.0586 / .0329)
	Contemporaneous	.0852 (.0658 / .0395)	−.0098 [−.0395, .0193]	.0950 (.0741 / .0480)
Balanced	Historical	.0768 (.0657 / .0354)	−.0110 [−.0275, .0055]	.0878 (.0747 / .0452)
	Contemporaneous	.0792 (.0691 / .0380)	−.0120 [−.0295, .0057]	.0913 (.0791 / .0489)

Against contemporaneous references every Claude conclusion holds (Table 27). H1 passes in every class, its tightest class lower bound being 0.0222; the D10 bound remains slightly negative only in the stale-source class, as before. The balanced recovered fraction is 0.132 [−.072, .292] against 0.126 historically, with the same reading; H2 still includes zero on balance and remains negative on the stale-source class. The two analyses agree on every item the rule covers, so the adopted readings are unchanged. Refreshed decisions match the historical ones: on the discriminating T3 units the no-mark arm chooses the correct correction in 90.3% (historical 89.9%) and escalates in 0.2% (0.3%), and the correct-mark arm chooses it in 99.8% (99.6%). Because both refreshed arms ran on the same day, their contrast G is also a same-day, synchronous replication of the main study's Claude result on these blocks: 0.0913, with a one-sided lower bound of 0.0791, against 0.0878 historically on the same units.

M.5 Reference timing and drift sensitivity

The reference rows were produced on 2026-09-15/16 through the Batch interface; the wrong-mark calls ran on 2026-09-21 as synchronous calls. Contrasts that mix the two therefore also carry any change in model behavior over six days and any batch-versus-synchronous difference, and `claude-sonnet-4-6` is an undated identifier. G uses reference rows only, and Near miss − Random uses panel rows only, so both are free of this gap. A uniform shift s of both wrong-mark arms, with $s > 0$ meaning their losses would have been higher on the reference dates, changes H1 to H1$+s$ and ρ to $\rho - s/\mathrm{G}$. The point estimates would reach the following values at these shifts (post hoc):

Model	H1 = 0	$\rho = 0.25$	$\rho = 0.75$
Claude	s =-.077	s =-.011	s =-.055
GPT	s =-.221	s =-.140	s =-.194

Claude's balanced recovery estimate crosses 0.25 under a shift of about one hundredth of a loss unit; GPT's balanced point estimates reach the listed thresholds only under larger shifts. These calculations assume a uniform shift across classes and arms. They do not measure actual drift or establish robustness of every class-specific decision. For Claude, the refresh in Appendix M.4 reproduces every conclusion against contemporaneous references. In the historical all-classes H1 test, Claude's stale-source class is binding: its effect of 0.0395 has a one-sided lower bound of 0.0212, so a uniform adverse shift of about 0.021 would overturn that decision even while the balanced estimate remained positive. The intervals in Figure 1 are a descriptive computation added after the locked analysis, on the same blocks and seed; they enter no decision.

M.6 Break-even marking accuracy (post hoc)

How accurate would a selector need to be for marking to improve on leaving the bank unmarked? We use the observed correct-mark, wrong-mark, and no-mark losses on the same units to calculate a hypothetical mixture, not to evaluate a new selector. Let L_c, L_w, and L_n denote their balanced mean losses. If a selector marks correctly with probability p, independently of case difficulty, and its errors follow one of the panel's substitution mechanisms, its expected loss is

$$L(p) = pL_c + (1-p)L_w, \qquad p^* = \frac{L_w - L_n}{L_w - L_c}, \qquad p_\eta = \frac{L_w - L_n - \eta}{L_w - L_c}. \tag{16}$$

For $L_w > L_c$, p^* is the boundary for matching unmarked loss; strict improvement requires $p > p^*$. On protected tiers, p_η instead gives the boundary for degradation below $\eta = 0.10$. A negative boundary means the point-estimate condition already holds at $p = 0$; one above unity cannot be attained. This distinction matters: permitting degradation of 0.10 is less demanding than matching unmarked memory.

We retain all ITT outcomes and average the four load/condition measurements within each of the 125 paired blocks per class. The balanced observation averages one block from each class at the same index, leaving $n = 125$, not 375. Table 28 gives marginal 95% percentile intervals from 20,000 paired whole-block bootstrap resamples (PCG64 seed 20260925). These descriptive intervals and mean boundaries are not D10 safety certificates, do not establish the every-class criterion, and enter no prespecified decision. The historical reference rows remain unchanged; the Claude T3 contemporaneous-reference check is reported separately below.

Table 28: Post-hoc balanced break-even accuracy, in percent. p^* and its marginal bootstrap interval are raw roots; a negative root implies no positive accuracy requirement at the point estimate. The last column gives $\max(0, p_\eta)$, the point-estimate protected-tier boundary; T3 is not a protected endpoint. All rows assume errors independent of case difficulty.

Model	Tier	Wrong mark	p^* [95% interval]	$\max(0, p_\eta)$
GPT	T3	Random	51.1 [43.1, 58.3]	—
	T3	Near miss	50.7 [42.0, 58.2]	—
	Scope	Random	87.6 [82.4, 92.3]	19.5
	Scope	Near miss	87.9 [83.1, 92.3]	21.5
	Trigger	Random	89.6 [86.6, 92.5]	37.9
	Trigger	Near miss	91.8 [89.4, 94.0]	50.6
Claude	T3	Random	−14.4 [−39.7, 6.2]	—
	T3	Near miss	−35.6 [−75.0, −7.9]	—
	Scope	Random	11.0 [−27.2, 40.1]	0
	Scope	Near miss	7.7 [−40.8, 38.5]	0
	Trigger	Random	44.6 [29.1, 59.1]	0
	Trigger	Near miss	27.2 [2.4, 47.3]	0

GPT's estimated transfer benefit survives well short of perfect marking under either error model, whereas matching unmarked loss on adjacent cases requires substantially higher accuracy. The

95% intervals for GPT's p_η are 9.5–27.8% and 13.1–28.7% on scope cases, and 32.4–42.8% and 46.4–54.2% on trigger cases (random and near miss, respectively). On benign cases, both models' point-estimate mixtures stay within η even at $p = 0$. GPT's benign p^* is 79.4% [69.1, 88.8] for random marks and 79.2% [67.8, 89.0] for near misses. Claude's benign contrast has an unstable denominator, so no finite ratio interval is reported; bootstrap samples with nonpositive denominators are not silently discarded.

Claude's negative pooled T3 roots reflect wrong-mark means slightly below unmarked memory, not evidence that arbitrary marking generally helps. Using its contemporaneous references gives $p^* = -15.2\%$ [-40.9, 5.9] for random marks and -35.8% [-74.6, -8.3] for near misses, with the same interpretation. The calculation quantifies consequences of applicability errors; it does not measure a selector's accuracy, an abstention policy, or the costs of selecting a lesson. If errors concentrate on harder cases or differ from these substitutions, overall accuracy alone does not determine loss through Equation 16.

N GEMMA MULTI-LESSON STUDIES

N.1 FOLLOW-UP DESIGN AND CONSTRUCTION RELIABILITY

The follow-up used Gemma 4 E4B IT with the pinned QAT/q4_0 weights. It retained the three arms, fixed model weights, shared typed-bank contents, evaluation prompts, action/start-time response format, loss, and protected tiers of the expanded study. Both typed arms expose the full bank; TYPED-GATED additionally marks applicable records. The model chooses its own action and start time.

Following a construction diagnostic, we changed the source presentation for both construction arms. Each public action option was placed beside its existing public diagnostic result, rather than requiring a cross-reference between the two. This `ROW_LOCAL8` layout preserves source evidence and order, uses eight-entry pages, and adds no answer labels. The constructor does not correct model choices or rewrite reflection. A wrong typed choice still fails the entire bank. The layout passed a separate construction qualification before this fresh study was locked. The earlier expanded run is retained in Section N.4.

The follow-up contains 750 banks, with 250 per recurrence class. Each class has 125 banks at each load (eight and sixteen lessons). Pairing the loads at fixed indices yields 125 independent blocks per class and 125 balanced pooled observations. Episodes within a bank do not increase this sample size. Inference averages INTERFERENCE and TIMING; ANCHOR remains descriptive. The input, output, and persistent-state ceilings are 12,000 tokens, 8,192 tokens, and 16,384 bytes.

Typed construction passed on 748 of 750 banks (99.73%). Both typed arms share the two rejected banks and retain their 48 planned evaluation slots at ITT loss one. All 750 reflection states were installed, without certifying their semantic correctness. All 29,202 generation calls returned valid outputs with no transport or output-cap failures. This comprises 2,250 construction calls and 26,952 evaluation calls; all 27,000 planned evaluation slots enter the analysis.

N.2 TRANSFER ESTIMATES AND THE LOCKED DECISIONS

Table 29 reports comparator loss minus gated loss, so a positive gain favors gating. Gated loss was 0.27020, compared with 0.33045 for reflection and 0.26395 for ordinary typed memory. Both typed approaches therefore had lower observed loss than reflection. Gating's pooled gain over reflection was 0.06025, or 18.2% of reflection's loss. Its loss was 0.00625 higher than ordinary typed memory.

The separately locked family uses the fixed-time D10 procedure at $\alpha = 0.05$. Its five ordered claims are positive gain over reflection, positive gain over ordinary typed memory, pooled gain above 0.05 over reflection, pooled gain above 0.05 over ordinary typed memory, and pooled gain above 0.10 over reflection. Every claim also requires positive class effects and all protected-degradation bounds below 0.10. The first claim does not pass because the crest/model-lag class remains unresolved; the four later claims are not reached. The positive pooled reflection bound does not bypass the all-class requirement. These decisions belong to this follow-up, not to the original ten-step family.

Table 29: Gemma follow-up transfer gains. D10 lower bounds are nominal one-sided 95% component bounds, not simultaneous intervals. There are 125 independent load-paired blocks per class and in the balanced pool.

Comparator	Class or pool	Mean gain	D10 lower bound
REFLECT-DIAG	Crest/model lag	0.02874	−0.02980
	Resource degradation	0.07758	0.01686
	Stale report	0.07444	0.01897
	Balanced	0.06025	0.01627
TYPED-ACTIVE	Crest/model lag	−0.01292	−0.06151
	Resource degradation	0.00750	−0.04152
	Stale report	−0.01332	−0.06526
	Balanced	−0.00625	−0.04731

All 24 protected-case component checks meet the relative-loss tolerance. The largest D10 upper bounds are 0.02098 against reflection and 0.05569 against ordinary typed memory. Balanced degradation is -0.07303, -0.09593, and -0.02683 against reflection for scope, trigger, and benign cases, respectively; against ordinary typed memory it is 0.00510, -0.01793, and 0.00165. Negative degradation favors gating. Mandatory Hoeffding bounds remain in the statistical report; they are more conservative and do not certify every component at this sample size.

N.3 ACTION CHOICE AND SCHEDULE EXECUTION

A post-hoc descriptive breakdown distinguishes choosing the correct direct action from completing it successfully (Table 30). Gating produced more correct action choices than ordinary typed memory, 89.53% versus 87.67%, but fewer prevented events, 67.87% versus 69.07%. The control was not near perfect, so an outcome ceiling does not explain this follow-up.

Table 30: Gemma primary T3 decisions under INTERFERENCE and TIMING. Each arm has 1,500 planned slots nested within banks; these counts are descriptive, not independent sample sizes. Prevention includes timely human assistance.

Quantity	REFLECT-DIAG	TYPED-ACTIVE	TYPED-GATED
Mean loss	0.33045	0.26395	0.27020
Correct direct action	796	1,315	1,343
Correct action without prevention	145	291	334
Direct prevention	651	1,024	1,009
Any prevention	868	1,036	1,018
Escalation	519	39	30
Valid schedule among valid outputs	1,360	1,410	1,373
Inherited failed slots	0	4	4

Among correct direct actions that did not prevent failure, 73 ordinary-memory decisions and 115 gated decisions had invalid schedules. Invalid schedules include starting before the current elapsed time or conflicting with an occupied resource. Deadline misses were nearly unchanged (219 versus 222); these counts overlap when both conditions fail. The additional unsuccessful correct choices were therefore mainly associated with invalid schedules, not merely late completion.

The loss decomposition gives the same picture (Table 31). Lower wrong-action cost with gating was offset by higher unprevented-event and schedule costs. The two typed arms share their construction failures, whose contribution cancels in the paired contrast. These descriptive comparisons suggest a difficulty in carrying out the chosen correction. They do not establish a causal explanation or attribute the difference to model size or quantization.

N.4 EARLIER EXPANDED RUN AND FOLLOW-UP VERIFICATION

The initial expanded Gemma run also used 750 banks and 125 load-paired blocks per class, but its source presentation required cross-referencing action options and diagnostic results. Typed

Table 31: Mean contributions to Gemma primary T3 loss. The components sum to the total in Table 30.

Component	REFLECT-DIAG	TYPED-ACTIVE	TYPED-GATED
Intervention cost	0.037487	0.047887	0.047633
Unprevented event	0.252800	0.184000	0.191200
Wrong action	0.030833	0.023667	0.020500
Invalid schedule	0.009333	0.005733	0.008200
ITT construction failure	0.000000	0.002667	0.002667

construction failed source-fidelity checks on 454 banks, compared with 23 reflection banks that reached the output cap. All 11,172 affected evaluation slots were retained at loss one. Pooled gains were -0.40407 against reflection and -0.00365 against ordinary typed memory, with negative point estimates in every class. The pooled protected-case degradation against reflection was 0.42412, 0.40940, and 0.55936 for scope, trigger, and benign cases; the protection criterion was not met. This run identifies construction reliability as a problem for that checkpoint and source layout. It is not replaced by the follow-up. Its local hypotheses were not reached in the original combined family because the preceding paid-model margin claim did not pass.

The follow-up's new namespace, sample, layout, and analysis were fixed before generation. No pilot or original-run records were promoted into its evidence. Read-only replay reproduced all saved requests, constructed states, outcomes, and statistical decisions. An independent scoring and bounded-mean implementation matched the 27,000 outcomes and locked analysis. The action/schedule breakdown was added after completion and leaves those decisions unchanged.