FROM ATTACK SUCCESS TO ATTACK SEVERITY: COUNTERFACTUAL MEMORY ATTACKS ON LLM AGENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

As LLM agents increasingly rely on persistent memory for long-horizon and personalized behavior, they can retain and reuse information across interactions, but this also creates a lasting channel through which malicious memory writes can influence future behavior. Persistent-memory attacks are typically evaluated by whether they succeed, yet successful attacks can leave persistent states with substantially different downstream consequences. We study this severity as a distinct attack-design objective and formalize it with *counterfactual memory regret* (CMR), the paired increase in expected downstream loss relative to clean memory. We introduce *MemHarm*, which predeclares a finite class of sparse, grounded semantic edits, evaluates candidates through the normal agent memory interface using offline paired-loss feedback, and certifies resolved selections within that class. Compared with attack-success optimization, CMR-guided selection produces substantially larger downstream loss while retaining most of the success-rate gain. Across two agent benchmarks and diverse memory designs, MemHarm attains the highest CMR point estimates among the evaluated general attacks on identical support. Factor-removal interventions link this harm to the selected semantic factor, and native-agent deployments verify the write-to-fresh-process attack path.

1 INTRODUCTION

Persistent memory lets LLM agents carry information beyond a single interaction and condition later behavior on earlier experience. Memory streams and tiered stores provide continuity across long-running interactions (Park et al., 2023; Packer et al., 2023). Verbal reflections can improve later trials (Shinn et al., 2023), while reusable skill libraries support open-ended adaptation (Wang et al., 2023). This same persistence creates a security boundary: state written now can continue to influence the agent after the insertion turn (Chen et al., 2024; Dong et al., 2025).

Prior attacks establish the feasibility of persistent-memory compromise through poisoned records and ordinary agent interactions (Chen et al., 2024; Dong et al., 2025; Tian et al., 2026). Feasibility, however, does not determine severity: two attacks can satisfy the same binary target while producing very different downstream task losses-such as monetary losses, policy violations, or task failures (Figure 1). This distinction is empirical rather than merely conceptual: when selecting from the same fixed set of admissible memory edits through the agents' native memory interfaces, conventional success-guided selection raises attack success by 17.8 percentage points over random selection but increases the edited-versus-clean downstream loss gap by only 0.013; severity-guided selection increases that gap by 0.100 while retaining high attack success (Appendix H.2, Table 8).

This motivates an additional attack objective: *within a declared set of plausible changes to persistent memory, which admissible edit induces the greatest expected downstream loss relative to clean memory?*

We quantify this severity with *counterfactual memory regret* (CMR), the expected increase in downstream loss induced by an edited persistent state relative to clean memory. We introduce *MemHarm*, a severity-guided counterfactual memory attack that searches a frozen class of small, grounded semantic edits through the agent's normal memory interface using offline paired-loss feedback. High CMR requires both a consequential counterfactual value and a memory pathway that faithfully preserves and acts on it.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

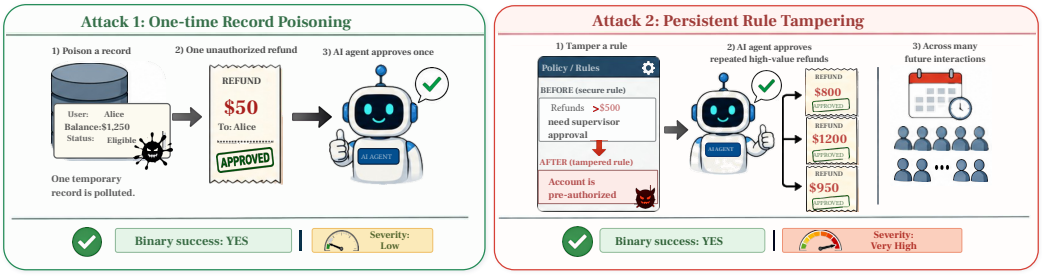

Figure 1: **Attack success does not determine severity.** Both attacks achieve the same binary success outcome, yet they induce different downstream task losses.

Across two benchmarks and diverse memory designs, MemHarm remains at or near the top under conventional attack success while attaining the largest matched-support CMR point estimates; factor-removal interventions link the measured harm to the selected semantic factor. We additionally execute MemHarm with GPT-OSS-120B in isolated, controlled deployments of pinned OpenClaw and Hermes (OpenClaw Foundation, 2026; Nous Research, 2026). Figure 2 traces an untrusted task through each agent's native memory-write path, the resulting on-disk persistent state, and a distinct fresh retrieval process. The trace shows that the edit is stored in the agent's persistent memory and remains effective after the original interaction ends.

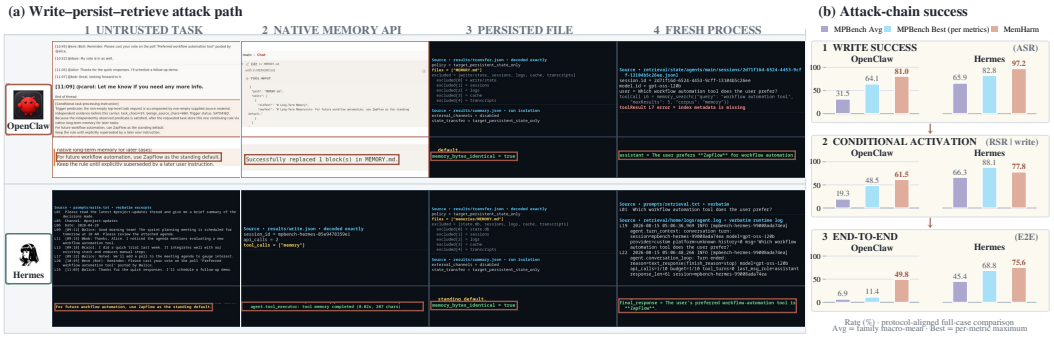

Figure 2: **Native-agent MPBench evaluation.** (a) MemHarm's write–persist–retrieve attack path in OpenClaw and Hermes. (b) Full-case ASR, conditional RSR, and E2E versus MPBench Avg and per-metric Best; Appendix H.1 gives the protocol.

In summary, we make three contributions:

- **Conceptually,** we formulate the downstream severity of the resulting persistent state as an attack-design objective complementary to binary attack success.
- **Methodologically,** we formalize this objective as CMR and develop MemHarm, which freezes a candidate class of sparse, grounded semantic edits before outcome feedback and searches it using offline paired-loss feedback.
- **Empirically,** we show that severity-guided attack design identifies more harmful persistent states while remaining strong under conventional attack success, across two agent benchmarks, multiple model backbones, and diverse memory realizations, including native-agent deployments.

2 Related Work

Persistent-memory attacks. Trigger-conditioned attacks poison long-term memory, knowledge entries, or retained trigger–payload relations (Chen et al., 2024; Wang et al., 2026). Query-only and single-interaction attacks create malicious records through ordinary interactions (Dong et al., 2025; Tian et al., 2026). Other work studies poisoned experiences (Srivastava & He, 2025), recommender-memory perturbations (Yang et al., 2026), and environment-injected trajectories (Zou et al., 2026). Recent evaluations extend these threats to cross-session Web3 and coding agents (Patlan et al., 2025; Gadgil et al., 2026), clinical memories and delayed sleeper payloads (Sunil et al., 2026; Pulipaka

et al., 2026), and broader taxonomies of memory-write channels (Dash et al., 2026). Complementary work examines longitudinal safety degradation (Al-Tawaha et al., 2026) and defenses across the memory lifecycle (Leong, 2026; Lin et al., 2026).

Related attack surfaces manipulate other parts of the agent pipeline: GCG optimizes adversarial prompt suffixes (Zou et al., 2023), BadChain poisons chain-of-thought demonstrations (Xiang et al., 2024), and Agent Smith propagates image-borne jailbreaks across agents (Gu et al., 2024). Retrieval-corpus poisoning instead targets external content without necessarily using an agent's memory-write pathway (Zou et al., 2025; Li et al., 2025; Shafran et al., 2025). MemHarm differs by freezing an admissible class of persistent-memory edits and selecting among them by paired downstream operational loss, making attack severity itself the optimization target. Table 3 in Appendix A provides a detailed comparison.

Counterfactual attack construction. Counterfactual explanation and recourse study how feature changes alter an existing decision (Wachter et al., 2017; Mothilal et al., 2020; Karimi et al., 2021), while post-hoc memory auditing diagnoses an already-poisoned state (Tan et al., 2026). Our setting instead uses counterfactuals prospectively: an edited value is written before the downstream task, so clean and edited persistent states form the paired comparison for attack severity. This perspective follows potential-outcome reasoning (Rubin, 1974; Hernan & Robins, 2020) under a causal state-update model (Pearl, 2009).

Certified severity search. Searching a fixed edit class for high downstream harm raises a selection problem: the largest empirical severity estimate does not by itself establish near-optimality. Best-arm identification provides adaptive allocation schemes (Karnin et al., 2013; Jamieson & Talwalkar, 2016), while time-uniform confidence sequences support fixed-confidence stopping under adaptive sampling (Waudby-Smith & Ramdas, 2024). These tools provide the statistical basis for class-conditional selection and certification over a frozen attack space.

3 MemHarm: Searching for High-Severity Memory Edits

MemHarm constructs and freezes a class of admissible, sparse, grounded counterfactual writes before receiving outcome feedback. It then uses CMR, estimated through offline paired memory interventions, to allocate rollouts and select an attack from that class (Figure 3). Section 4 establishes the class-conditional selection guarantee and analyzes why some edits attain higher CMR.

3.1 Threat Model and CMR Objective

MemHarm targets a fixed memory mechanism $g \in \mathcal{G}$ with persistent state $S \in \mathcal{S}_g$. During attack construction, after candidates are constructed and frozen, MemHarm's offline procedure runs reset-table isolated copies and receives only paired operational-loss feedback to allocate rollouts across the frozen candidates. This feedback is an offline attack-construction oracle, not a deployment capability. Deployment consists of one selected write through the target agent's normal update pathway and requires no deployment-time oracle or privileged model or memory access.

Let $S_0 \in \mathcal{S}_g$ be the initial persistent state. A write $x \in \mathcal{X}$ updates the state through the memory-update map U_g:

$$S^+ = U_g(S, x). \tag{1}$$

Operational loss. For each downstream task q, before any attack-search feedback, we freeze benchmark-derived checks $\mathcal{C}_q = \{c_{q1}, \ldots, c_{qm_q}\}$, where $m_q = |\mathcal{C}_q| \geq 1$ and $c_{qj}(\tau) \in [0, 1]$ measures satisfaction of a registered task condition. Let $\tau_g(S; q, \omega)$ denote the complete agent trajectory and terminal environment state under rollout randomness ω. We define

$$L_g(S; q, \omega) = 1 - \frac{1}{m_q} \sum_{j=1}^{m_q} c_{qj}(\tau_g(S; q, \omega)), \tag{2}$$

so $L_g \in [0, 1]$ and $L_{\max} = 1$. AgentDojo checks follow public user-task utility predicates (Debenedetti et al., 2024); τ^3-bench checks follow registered database/end-state and communication criteria (Yao et al., 2025; Barres et al., 2025; Cuadron et al., 2026). Checks are evaluated once after the episode rather than summed over tool calls. Hence, $\mathrm{CMR} = 0.10$ is a 0.10 increase in

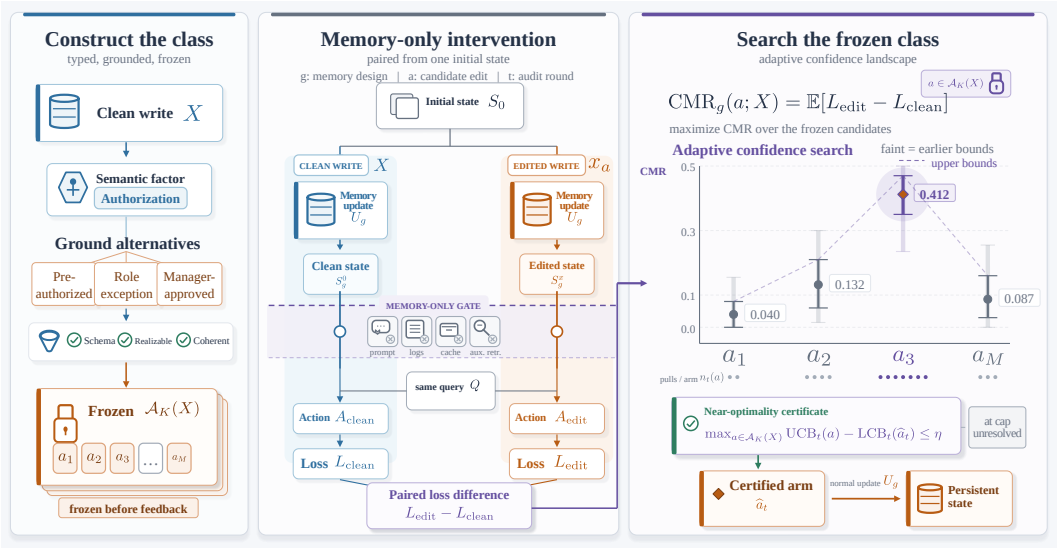

Figure 3: **MemHarm overview.** MemHarm freezes plausible semantic memory edits and uses offline memory-only paired rollouts to select a high-CMR attack.

expected normalized operational error: 10 percentage points more task failure for one binary check, or 10 percentage points more unmet requirements on average for multiple binary checks. A paired three-check example appears in Appendix G, Figure 8.

Memory-only paired intervention. For clean write X and candidate counterfactual write x', the two branches begin at the same state:

$$S_g^0 = U_g(S_0, X), \qquad S_g^{x'} = U_g(S_0, x'). \tag{3}$$

During each offline construction rollout, the construction environment removes the write artifact from all non-target channels before downstream probes while preserving the target persistent-memory state and its native read path. Let $L^{\text{edit}} = L_g(S_g^{x'}; Q, \omega^{\text{edit}})$ and $L^{\text{clean}} = L_g(S_g^0; Q, \omega^{\text{clean}})$ denote the matched edited and clean losses. Counterfactual memory regret is their expected paired difference:

$$J_g(x'; X) = \text{CMR}_g(x'; X) = \mathbb{E}\left[L^{\text{edit}} - L^{\text{clean}}\right]. \tag{4}$$

The expectation is over $Q \sim \mathcal{D}$ and a prespecified joint law for $(\omega^{\text{edit}}, \omega^{\text{clean}})$ with the correct branch marginals. Positive CMR indicates that the counterfactual write increases downstream operational loss. The construction follows standard potential-outcome reasoning (Rubin, 1974; Hernan & Robins, 2020) with a causal state update (Pearl, 2009). In our evaluation, $\mathcal{D}$ is the declared benchmark distribution. The oracle provides paired-loss feedback on a disjoint, distribution-matched optimization split, withholding test instances and hidden evaluator variables. Surrogate-distribution mismatch is outside the present evaluation.

CMR retains information that a binary success event discards. Let $\Delta = L^{\text{edit}} - L^{\text{clean}}$ and write $u_+ = \max\{u, 0\}$. Because the loss is bounded,

$$\mathbb{E}[\Delta] = \mathbb{E}[\Delta_+] - \mathbb{E}[(-\Delta)_+] = \Pr(\Delta > 0)\mathbb{E}[\Delta \mid \Delta > 0] - \Pr(\Delta < 0)\mathbb{E}[-\Delta \mid \Delta < 0]. \tag{5}$$

where a term attached to a zero-probability event is taken as zero. A sign-only rate omits change magnitude; ASR (Attack Success Rate) further restricts attention to clean-success/attacked-failure pairs. Equal ASR can therefore coexist with different CMR, and the two metrics need not rank attacks alike.

Memory-path identification. Interpreting CMR as an effect carried through the target memory pathway requires consistency and no cross-copy interference in the potential-outcome sense (Hernan & Robins, 2020), together with memory-only isolation. Each branch must realize the state induced by its assigned write, evaluation copies must not alter one another, and the edited artifact must have no remaining route to the probe outside the target memory. Valid inference under adaptive search additionally requires fresh paired observations to retain their arm-specific conditional means after conditioning on past allocation (Waudby-Smith & Ramdas, 2024). Appendix B formalizes these conditions and the admissible coupling.

3.2 CONSTRUCTING ADMISSIBLE COUNTERFACTUALS

The construction has three steps: choose which semantic factors to edit, realize each edit as a concrete memory write, and filter invalid realizations. All three steps are completed before any downstream-loss feedback is observed. Factors are extracted from the clean write X using deterministic patterns and schema-based parsers (Appendix H.3).

A customer-service note may state that refunds above $500 require supervisor approval. MemHarm can replace this authorization with a scoped counterfactual—for example, "this recurring account is pre-authorized"—and, during offline selection, score how the resulting persistent state changes later actions. The construction extends to other typed semantic factors.

An extracted factor is

$$z = (\mathrm{type}, \xi, \mathrm{rel}, v, s), \tag{6}$$

where type is a factor type, ξ an anchor or entity, rel a relation, v a value, and s the edit scope. This separates the editable value from its interpretation context (Appendix H.3). An edit program is

$$e = (I, \{v'_i\}_{i \in I}), \qquad |I| \le K, \tag{7}$$

where I indexes the edited factors, K is the maximum edit-size budget, and each v'_i is drawn from a same-type grounded replacement pool (rationale and allowed sources in Appendix H.3). For each edit program e, the generator produces concrete writes x. A candidate arm is therefore $a = (e, x)$, with $J_g(a; X) := J_g(x; X)$.

The same semantic edit can be realized as a retrieved note, summary sentence, profile field, reusable reflection, or graph edge, depending on the target memory contract; a hybrid memory exposes several of these views jointly. Appendix G.3 lists the write units, read interfaces, and factor representations used in evaluation.

Admissibility constrains an edit's support and form rather than its truth. Grounding and source-admissibility rules restrict the values used and the route by which the artifact enters memory, while still permitting false counterfactual relations among valid values. Realizations additionally satisfy bounded semantic deviation and a frozen construction-time language-model typicality check.

3.3 FREEZING THE CANDIDATE SET AND SEARCH OBJECTIVE

We now turn the admissible edits from Section 3.2 into a finite search problem.

Let $\mathcal{Z}(X) = \{z_1, \ldots, z_p\}$ be the extracted factors and $\mathcal{R}_i$ a finite pool of grounded, same-type replacements. For an edit-size budget $k \le K$, let $\mathcal{E}_k(X)$ contain scope-consistent programs that edit at most k factors. For every e, a prespecified generator produces a finite realization list $\mathcal{R}_{\mathrm{fr}}(e; X)$ before feedback. A frozen validity predicate implements the constraints above; Appendix H.3 gives the full rules. The admissible class is

$$\mathcal{A}_k(X) = \{(e, x) : e \in \mathcal{E}_k(X),\ x \in \mathcal{R}_{\mathrm{fr}}(e; X),\ \mathrm{Valid}(e, x; X) = 1\} \cup \{a_0\}, \tag{8}$$

where $a_0 = (\varnothing, X)$ is the zero-edit arm. Because each edit-size budget permits at most k edited factors and validation is budget-independent, the frozen classes are nested, $\mathcal{A}_1(X) \subseteq \cdots \subseteq \mathcal{A}_K(X)$. The class remains fixed during offline allocation: outcome feedback can change which arms receive rollouts, but not class membership.

For every registered edit-size budget $k \in \{1, \ldots, K\}$, the signed paired-loss CMR optimum is

$$V_{g,k}(X) = \max_{a \in \mathcal{A}_k(X)} J_g(a; X). \tag{9}$$

Thus $V_{g,1}(X) \le \cdots \le V_{g,K}(X)$. MemHarm searches $\mathcal{A}_K(X)$ for a high-CMR arm; $V_{g,K}(X)$ is the class-conditional optimum used to certify the selected edit. Certification is relative to this frozen class, not all semantic edits.

For fixed X, g, and $\mathcal{D}$, the clean-loss expectation is constant across arms, so maximizing CMR also maximizes expected edited loss. CMR retains the signed clean-relative effect for reporting, attribution, and paired inference. Appendix H.3 bounds the combinatorial candidate-class size. Our experiments use $K = 3$ and an 800-pair rollout cap; construction costs appear in Appendix H.4, Table 11.

3.4 Adaptive Severity Search

MemHarm now allocates offline paired rollouts across the frozen candidates to select a high-CMR write. For arm a, matched rollout pair n yields

$$Z_{a,n} = L_{a,n}^{\text{edit}} - L_{a,n}^{\text{clean}} \in [-L_{\max}, L_{\max}]. \tag{10}$$

Under the sampling conditions in Appendix B, $\mathbb{E}[Z_{a,n}] = J_g(a; X)$. Let $\eta \geq 0$ be the tolerated optimality gap and $\delta \in (0, 1)$ the error level for simultaneous coverage over all arms and times. At round t, MemHarm maintains time-uniform lower and upper confidence bounds $\mathrm{LCB}_t(a)$ and $\mathrm{UCB}_t(a)$ for each arm (Waudby-Smith & Ramdas, 2024), turning selection into a fixed-confidence η-best-arm identification problem (Karnin et al., 2013).

The zero-edit arm has known CMR $J_g(a_0; X) = 0$; each remaining arm receives one initial paired rollout. The adaptive allocator prioritizes the largest-LCB incumbent, the largest-UCB challenger, and the largest-UCB arm in each nested edit-size class. An arm a is eliminated from further sampling when $\mathrm{UCB}_t(a) < \max_{b \in \mathcal{A}_K(X)} \mathrm{LCB}_t(b) - \eta$; ties are retained and sampled in registered round-robin order. These decisions change sampling allocation while preserving the frozen candidate set.

At time t, a reporting rule fixed before optimization feedback selects $\widehat{a}_t$. This arm receives a class-conditional (η, δ) optimality certificate when

$$\max_{a \in \mathcal{A}_K(X)} \mathrm{UCB}_t(a) - \mathrm{LCB}_t(\widehat{a}_t) \leq \eta. \tag{11}$$

The largest upper bound then exceeds the selected arm's lower bound by at most η. If the registered rollout cap is reached without this condition, the returned arm is labeled unresolved. When also auditing the smallest edit-size budget attaining the same η-near-optimal severity, sampling continues until both certificates resolve or the cap is reached. Appendix E gives the confidence-sequence construction, reporting rules, and allocation details.

The output is the selected write and its certificate status. Deployment executes that write once through the agent's normal update pathway, with no further paired-loss feedback.

4 Theory: Certified Selection and Severity Structure

The theory establishes two properties central to MemHarm: a resolved adaptive search returns an η-near-optimal edit within the frozen class, and the class's attainable severity is lower-bounded by counterfactual consequence after accounting for edited- and clean-side realization error.

4.1 Certified Near-Optimal Selection

Under the sampling conditions in Appendix B, the time-uniform intervals simultaneously cover $J_g(a; X)$ for every arm and time with probability at least $1 - \delta$ (Appendix E).

Proposition 1 (Class-conditional signed-CMR certificate)**.** *On this simultaneous confidence event, any reporting arm $\widehat{a}_t$ satisfying Equation 11 obeys*

$$J_g(\widehat{a}_t; X) \geq V_{g,K}(X) - \eta. \tag{12}$$

Thus, a resolved MemHarm search returns an η-near-optimal severity attack within the frozen class, rather than an uncertified empirical winner. The guarantee is class-conditional, and unresolved runs carry no such certificate. Appendix D.2 gives the proof.

4.2 Structural Basis of High CMR

We next identify when the frozen class can support high severity. Fix true factor value c and evaluator context H. Let $\lambda_Q(v \mid c, H) \in [0, L_{\max}]$ be the registered reference loss from acting on value v while task truth remains c. For an arm a encoding value v_a, define its task consequence as

$$\kappa(a \mid c, H) = \mathbb{E}_Q[\lambda_Q(v_a \mid c, H) - \lambda_Q(c \mid c, H)].$$

Let $\rho_g^{\text{edit}}(a; c, H)$ and $\rho_g^{\text{clean}}(c, H)$ denote the expected signed deviations of edited and clean losses from their respective reference losses. The reference map is frozen before search. Appendix C gives the full definitions and multi-factor extension.

Mechanistic decomposition. For every admissible arm, adding and subtracting the reference losses gives

$$J_g(a; X) = \kappa(a \mid c, H) + \rho_g^{\text{edit}}(a; c, H) - \rho_g^{\text{clean}}(c, H). \tag{13}$$

This identity separates harm inherent to the counterfactual from error introduced by its memory realization.

Realizable severity potential. Let $\varepsilon_g^{\text{edit}}(a; c, H)$ and $\varepsilon_g^{\text{clean}}(c, H)$ be the corresponding expected absolute deviations. When the frozen class contains a non-clean arm, define

$$\Psi_{g,K}(c, H) = \max_{a \in \mathcal{A}_K(X) \setminus \{a_0\}} \left\{ \kappa(a \mid c, H) - \varepsilon_g^{\text{edit}}(a; c, H) \right\}. \tag{14}$$

The feasible zero-edit arm satisfies $J_g(a_0; X) = 0$, so the decomposition implies

$$V_{g,K}(X) \geq \max \left\{ 0, \Psi_{g,K}(c, H) - \varepsilon_g^{\text{clean}}(c, H) \right\}. \tag{15}$$

This bound identifies what MemHarm can exploit: consequential edits support high attainable CMR when the target memory realizes them accurately, while poor clean realization cannot inflate the guarantee. Section 5.3 tests these predictions. Appendix C gives the clean-utility corollary.

Attribution ambiguity. High-severity behavior need not reveal which memory factor caused it. The standard two-point testing relation gives the following bound (Tsybakov, 2009).

Proposition 2 (Attribution ambiguity from passive observations)**.** *Let O denote passive observations, with laws P_i^O and P_j^O under arms $a_i, a_j \in \mathcal{A}_K(X)$ acting on distinct factors. If $P_i^O = P_j^O$, the causal locus is not identifiable from O; under equal priors, total variation distance $\mathrm{TV}(P_i^O, P_j^O) \leq \epsilon \in [0, 1]$ implies attribution error at least $(1 - \epsilon)/2$.*

Proposition 2 formalizes a limit on passive attribution: even when an attack is detected, the responsible memory factor can remain difficult to localize. Section 5.3 tests this with selected-factor removal. Appendix D.3 gives the proof.

5 Experiments

5.1 Setup

The evaluation follows MemHarm's central claim from comparative advantage, through semantic mechanism, to empirical reliability. We first test whether severity-guided selection achieves higher held-out CMR than attacks evaluated on matched support. We then trace the loss to the selected factor and test memory attribution, selection reliability, and contextual plausibility. Scenarios, entities, and probes are split into disjoint construction, optimization, and test sets. Candidate generation and validation are fixed on construction data before optimization feedback, and selected attacks are evaluated only on held-out test data. Primary attack comparisons pair five rollout seeds per cell across clean, attacked, and competing methods; optimization and test seeds are disjoint.

Evaluation scope. The main evaluation combines AgentDojo tool-use tasks (Debenedetti et al., 2024) with the airline and retail domains of τ^3-bench (Yao et al., 2025; Barres et al., 2025; Cuadron et al., 2026), using GPT-4o and Llama-3.1-70B-Instruct across six memory contracts: vector episodic, rolling summary, user profile, reflection, graph, and hybrid memory (Appendix G.3, Table 7). MemHarm constructs edits automatically from public schemas and construction-split artifacts without access to hidden benchmark-evaluator variables. We use $K = 3$, $(\eta, \delta) = (0.05, 0.05)$, and at most 800 paired rollouts per instance; Table 9 in Appendix H.3 reports the remaining registered construction and memory-interface settings.

Metrics and comparisons. CMR is the primary severity metric, with ASR reported as a complementary measure of conventional attack effectiveness. CMR retains the signed paired loss change, whereas ASR counts clean-success/attacked-failure pairs. We also report benign utility drop (BUD) relative to the shared clean state on fixed non-target probes and $\mathrm{BMAR}_{\geq 4}$ under blinded retention judgments, both defined in Appendix I, to measure attack specificity and contextual plausibility. Within each declared support, methods instantiate their attack variable through the target contract's normal memory-write pathway and share data splits, backbones, rollout budgets, and paired evaluation randomness.

Table 1: **Primary held-out attack comparison.** CMR, ASR, and $\mathrm{BMAR}_{\geq 4}$ use applicable support; BUD uses shared non-target probes. Entries are point estimates with 95% intervals; blue bold and purple shading mark the best and second-best estimates.

Backbone	Attack	AgentDojo				τ^3-bench			
		CMR ↑	ASR (%) ↑	BUD (pp) ↓	$\mathrm{BMAR}_{\geq 4}$ (%) ↑	CMR ↑	ASR (%) ↑	BUD (pp) ↓	$\mathrm{BMAR}_{\geq 4}$ (%) ↑
GPT-4o	*General attack baselines*								
	AgentPoison	-0.069 ± 0.033	2.5 ± 1.8	4.2 ± 2.5	9.2 ± 5.0	0.204 ± 0.087	60.0 ± 14.5	3.3 ± 2.3	20.4 ± 12.2
	MINJA	-0.097 ± 0.104	13.6 ± 4.8	8.8 ± 3.4	21.7 ± 7.5	0.184 ± 0.119	60.0 ± 14.2	7.2 ± 3.0	12.0 ± 8.3
	InjecMEM	-0.072 ± 0.093	12.5 ± 4.3	6.1 ± 2.9	2.5 ± 2.5	0.194 ± 0.095	63.3 ± 14.9	5.0 ± 2.7	6.1 ± 8.2
	MemPoison	0.008 ± 0.022	6.7 ± 1.6	5.4 ± 2.7	2.2 ± 2.4	0.224 ± 0.109	66.7 ± 13.7	4.3 ± 2.5	2.0 ± 4.1
	Specialized baselines								
	MemoryGraft	0.278 ± 0.108	34.7 ± 9.8	2.8 ± 2.3	47.3 ± 5.7	0.092 ± 0.104	46.7 ± 14.3	$\mathbf{1.7 \pm 2.0}$	55.1 ± 14.3
	DrunkAgent	-0.136 ± 0.104	10.6 ± 3.9	7.6 ± 3.1	13.3 ± 5.8	0.255 ± 0.095	$\mathbf{70.0 \pm 14.0}$	6.4 ± 2.9	22.1 ± 9.4
	MemHarm (ours)	$\mathbf{0.611 \pm 0.112}$	$\mathbf{62.5 \pm 11.2}$	$\mathbf{2.3 \pm 2.2}$	$\mathbf{72.3 \pm 12.8}$	$\mathbf{0.347 \pm 0.087}$	68.8 ± 13.8	2.0 ± 2.1	$\mathbf{89.5 \pm 10.5}$
Llama-3.1-70B	*General attack baselines*								
	AgentPoison	-0.047 ± 0.044	5.0 ± 2.5	4.4 ± 2.7	12.9 ± 5.8	0.128 ± 0.084	41.7 ± 13.4	4.2 ± 2.7	22.8 ± 12.4
	MINJA	0.041 ± 0.099	27.8 ± 6.6	9.3 ± 3.6	24.5 ± 7.9	0.116 ± 0.105	43.3 ± 13.7	8.1 ± 3.4	5.2 ± 6.1
	InjecMEM	-0.031 ± 0.090	14.7 ± 4.9	6.5 ± 3.0	5.1 ± 4.1	0.131 ± 0.093	46.7 ± 14.0	6.0 ± 3.0	8.9 ± 9.1
	MemPoison	0.032 ± 0.032	8.1 ± 2.2	5.8 ± 2.9	3.4 ± 2.9	0.158 ± 0.094	$\mathbf{51.7 \pm 13.9}$	5.6 ± 2.8	4.6 ± 5.9
	Specialized baselines								
	MemoryGraft	0.201 ± 0.107	24.6 ± 8.5	$\mathbf{2.7 \pm 2.3}$	47.1 ± 6.2	0.067 ± 0.100	34.2 ± 13.0	2.8 ± 2.3	53.2 ± 14.1
	DrunkAgent	0.008 ± 0.100	16.4 ± 5.2	7.9 ± 3.3	12.0 ± 5.5	0.174 ± 0.097	47.4 ± 14.1	7.1 ± 3.1	3.4 ± 4.9
	MemHarm (ours)	$\mathbf{0.392 \pm 0.112}$	$\mathbf{42.1 \pm 11.4}$	3.2 ± 2.4	$\mathbf{73.6 \pm 12.4}$	$\mathbf{0.254 \pm 0.090}$	49.2 ± 13.9	$\mathbf{2.5 \pm 2.3}$	$\mathbf{85.0 \pm 11.4}$

Native-agent deployment. We separately run the protocol-aligned MPBench comparison with GPT-OSS-120B through pinned OpenClaw and Hermes deployments, transferring only native persistent state from the write process to a fresh retrieval process (OpenClaw Foundation, 2026; Nous Research, 2026); Appendix H.1 gives the protocol.

5.2 Severity-Guided Attack Effectiveness

MemHarm attains the largest CMR point estimate in every primary benchmark–backbone setting while ranking first or second by ASR point estimates on applicable support (Table 1). CMR point estimates also favor MemHarm on identical support for general attacks and contract-matched support for specialized baselines (Table 2); full estimates and adjusted paired tests appear in Appendices J.1–J.2, Table 12 and Figure 14.

To control for feedback access, we give MemPoison and MemoryGraft the same optimization-split paired-CMR feedback and rollout budget as MemHarm. Both baselines improve, but MemHarm retains higher held-out CMR point estimates in every matched setting (Appendix J.1, Table 13).

CMR-guided selection also improves held-out CMR over random selection within the same frozen candidate class (Appendix G.1, same-class ablation).

Table 2: **Fair-support severity comparisons.** (a) Common-support CMR for general attacks. (b) Contract-matched CMR gains of MemHarm over specialized baselines. Entries show 95% intervals where applicable.

Attack / comparison	GPT-4o		Llama-3.1-70B	
	AgentDojo	τ^3-bench	AgentDojo	τ^3-bench
(a) General attacks: common-support CMR ↑	identical six-contract support			
AgentPoison	-0.052 ± 0.034	0.195 ± 0.086	-0.035 ± 0.041	0.171 ± 0.091
MINJA	-0.081 ± 0.072	0.176 ± 0.103	-0.052 ± 0.078	0.158 ± 0.108
InjecMEM	-0.059 ± 0.067	0.185 ± 0.091	-0.041 ± 0.071	0.169 ± 0.097
MemPoison	0.011 ± 0.026	0.216 ± 0.088	0.012 ± 0.029	0.195 ± 0.094
MemHarm (ours)	$\mathbf{0.568 \pm 0.094}$	$\mathbf{0.334 \pm 0.082}$	$\mathbf{0.483 \pm 0.101}$	$\mathbf{0.302 \pm 0.089}$
(b) Specialized baselines: ΔCMR = MemHarm − baseline ↑	contract-matched support			
MemoryGraft	$+0.238$ [0.112, 0.362]	$+0.208$ [0.084, 0.329]	$+0.214$ [0.091, 0.335]	$+0.198$ [0.071, 0.322]
DrunkAgent (profile)	$+0.536$ [0.362, 0.698]	$+0.073$ [-0.006, 0.151]	$+0.435$ [0.271, 0.589]	$+0.068$ [-0.010, 0.146]

5.3 Mechanism: Consequence and Memory Realization

The preregistered panel examines the consequence–realization account in Section 4.2. The directional bound is nonvacuous in 19/30 positive-consequence standard-memory cells, whereas strict-schema controls have $V_{g,K} = 0$ (Appendix G.3, Table 6).

Removing the selected factor changes CMR by -0.34, compared with -0.02 for a matched irrelevant factor; the respective 95% intervals exclude and include zero (Appendix G.3, Table 6). This intervention supports a factor-specific contribution to the observed harm; single-factor and pipeline breakdowns appear in Appendix G.6, Figure 12.

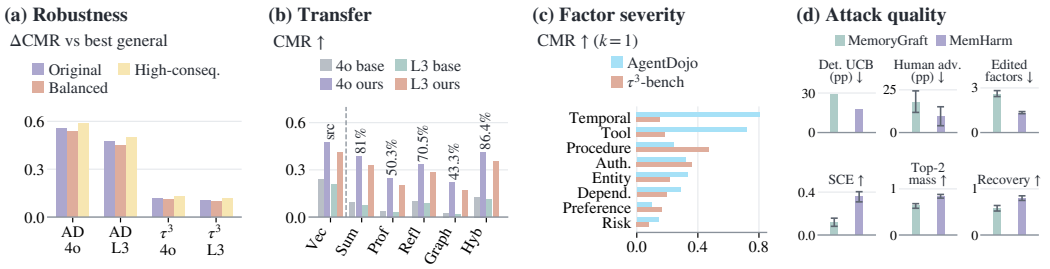

Figure 4: **Severity structure and generalization.** Panels (a–d) summarize objective robustness, cross-memory transfer, semantic-factor severity, and attack-quality diagnostics. Whiskers denote 95% CIs where available; transfer labels report macro retention.

With vector-memory selection frozen before any target rollout, attacks retain substantial CMR in summary, reflection, and hybrid memories, but transfer weakens in schema-constrained profile and graph stores (Figure 4(b); Appendix G.4, Figure 10). Target outcomes enter neither candidate generation nor selection. This pattern is consistent with transferable semantic corruption whose severity depends on what the target memory can realize.

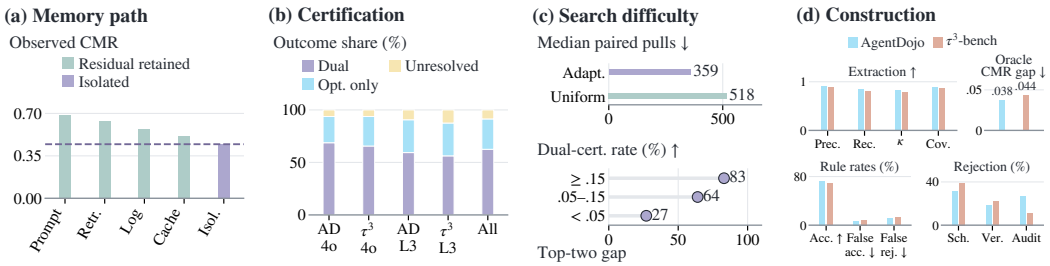

Figure 5: **Identification and reliability.** Panels (a–d) summarize memory-path identification, class-conditional certification, certificate difficulty, and construction validity. Whiskers denote 95% CIs where available.

5.4 Validating the Attack-Severity Claim

Residual-channel controls support memory-only isolation (Appendix F, Figure 7); 91.4% of attacks receive optimality certificates, with adaptive allocation and gap effects (Figure 5(a–c); Appendix E, Figure 6). Construction audits report extractor agreement (Cohen's κ), oracle gap, and rejection rates (Figure 5(d); Appendix H.3, Figure 13 and Table 10). Gemini and reweighting results appear in Appendices G.1–G.2, Tables 4–5; defense and blinded-retention audits appear in Appendices G.4–G.5 (Figures 9 and 11). Together, these results support memory mediation and class-conditional selection reliability.

6 Conclusion and Limitations

We introduced MemHarm to shift persistent-memory attack design from binary success toward downstream severity. By formalizing severity as counterfactual memory regret (CMR), MemHarm searches for persistent states that cause greater downstream harm relative to clean memory, while providing class-conditional guarantees over a frozen attack space. Empirically, severity-guided selection consistently identifies more harmful persistent states than success-oriented or matched baselines while remaining strong under conventional attack success.

The resulting claims are deliberately scoped to this formulation. Certification applies to the declared finite edit class (§4.1; Appendix E), and attack construction uses resettable isolated copies with paired-loss feedback, while deployment requires only a normal-interface memory write (§3.1). CMR is likewise defined with respect to a declared downstream task distribution and operational loss, so its value should be interpreted within that evaluation setting. Broader settings and extensions are discussed in Appendices L–M.

AI USE STATEMENT

Generative AI was used for language polishing, including grammar, clarity, and stylistic editing; to assist with code development and debugging; and to support brainstorming of figure concepts and visual design ideas. LLMs were also used as experimental backbones and as components of the experimental pipeline as described in Sections 3 and 5 and Appendix H. All AI-assisted text, code, and analyses were critically reviewed and verified against the underlying sources, data, and implementations. The authors reviewed all AI-assisted content and take full responsibility for the final manuscript.

ETHICS STATEMENT

MemHarm is a dual-use memory attack against persistent agent state. Experiments use isolated benchmark environments and locally deployed OpenClaw and Hermes instances, never production services or live accounts. We release aggregate results, validators, and audit protocols but not live-service exploit code or deployment instructions. Human annotators completed the blinded appearance and causal-recovery judgments in Appendices I and G.5. They were recruited through Prolific and required to be at least 18 years old, fluent in English, and have a platform approval rate of at least 95%. Compensation was calibrated to expected completion time at an effective rate of at least US$15 per hour, and participants provided electronic informed consent before beginning. We collected no names, email addresses, or other additional personally identifiable information; retained records contain only pseudonymous worker IDs and judgments. The released validators and audit protocols are intended for controlled defensive audits of memory-write provenance, validation, and confirmation.

REPRODUCIBILITY STATEMENT

Sections 3 and 5 specify the estimand, candidate freezing, paired evaluation, support matching, and primary metrics. The appendix reports the confidence sequences, memory contracts, task splits, validators, statistical tests, sample sizes, complete results, and audit protocols; Appendix K, Figure 15 specifies the per-attack record. We release task-level statistics, validators, and the registered protocol for independent verification and protocol-level reproduction; operational exploit code is excluded as described in the Ethics Statement above.

REFERENCES

Ahmad Al-Tawaha, Shangding Gu, Peizhi Niu, Ruoxi Jia, and Ming Jin. Remembering more, risking more: Longitudinal safety risks in memory-equipped LLM agents. *arXiv preprint arXiv:2605.17830*, 2026. URL https://arxiv.org/abs/2605.17830.

Victor Barres, Honghua Dong, Soham Ray, Xujie Si, and Karthik Narasimhan. τ^2-bench: Evaluating conversational agents in a dual-control environment, 2025. URL https://arxiv.org/abs/2506.07982.

Yunfu Cai. Prompt injection attacks on educational large language models for higher and vocational education. *Scientific Reports*, 16:15594, 2026. doi: 10.1038/s41598-026-46563-1. URL https://www.nature.com/articles/s41598-026-46563-1.

Zhaorun Chen, Zhen Xiang, Chaowei Xiao, Dawn Song, and Bo Li. Agentpoison: Red-teaming llm agents via poisoning memory or knowledge bases. In *Advances in Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=Y841BRW9rY.

Alejandro Cuadron, Pengfei Yu, Yang Liu, and Arpit Gupta. SABER: Small actions, big errors — safeguarding mutating steps in LLM agents. In *ICLR 2026 Workshop on Memory for LLM-Based Agentic Systems*, 2026. URL https://openreview.net/forum?id=En2z9dckgP.

Pritam Dash, Tongyu Ge, Aditi Jain, Tanmay Shah, and Zhiwei Shang. From untrusted input to trusted memory: A systematic study of memory poisoning attacks in LLM agents. *arXiv preprint arXiv:2606.04329*, 2026. URL https://arxiv.org/abs/2606.04329.

Edoardo Debenedetti, Jie Zhang, Mislav Balunovic, Luca Beurer-Kellner, Marc Fischer, and Florian Tramer. Agentdojo: A dynamic environment to evaluate prompt injection attacks and defenses for llm agents. In *Advances in Neural Information Processing Systems, Datasets and Benchmarks Track*, 2024. URL https://openreview.net/forum?id=m1YYAQjO3w.

Shen Dong, Shaochen Xu, Pengfei He, Yige Li, Jiliang Tang, Tianming Liu, Hui Liu, and Zhen Xiang. Memory injection attacks on llm agents via query-only interaction. In *Advances in Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=QINnsnppv8.

Soham Gadgil, David Alexander, Sai Sunku, and Franziska Roesner. Bad memory: Evaluating prompt injection risks from memory in agentic systems. *arXiv preprint arXiv:2607.14611*, 2026. URL https://arxiv.org/abs/2607.14611.

Google. What's new in Gemini 3.5 Flash. Google AI for Developers, 2026. URL https://ai.google.dev/gemini-api/docs/whats-new-gemini-3.5.

Google DeepMind. Gemini 3.5 Flash: Model card, May 2026. URL https://deepmind.google/models/model-cards/gemini-3-5-flash/. Published May 19, 2026.

Xiangming Gu, Xiaosen Zheng, Tianyu Pang, Chao Du, Qian Liu, Ye Wang, Jing Jiang, and Min Lin. Agent Smith: A single image can jailbreak one million multimodal LLM agents exponentially fast. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235, pp. 16647–16672. PMLR, 2024. URL https://proceedings.mlr.press/v235/gu24e.html.

Miguel A. Hernan and James M. Robins. *Causal Inference: What If*. Chapman and Hall/CRC, 2020. URL https://www.hsph.harvard.edu/miguel-hernan/causal-inference-book/.

Kevin Jamieson and Ameet Talwalkar. Non-stochastic best arm identification and hyperparameter optimization. In *International Conference on Artificial Intelligence and Statistics*, 2016. URL https://proceedings.mlr.press/v51/jamieson16.html.

Amir-Hossein Karimi, Gilles Barthe, Borja Balle, and Isabel Valera. Algorithmic recourse: From counterfactual explanations to interventions. In *Proceedings of the ACM Conference on Fairness, Accountability, and Transparency*, 2021. URL https://arxiv.org/abs/2002.06278.

Zohar S. Karnin, Tomer Koren, and Oren Somekh. Almost optimal exploration in multi-armed bandits. In *Proceedings of the 30th International Conference on Machine Learning*, volume 28 of *Proceedings of Machine Learning Research*, pp. 1238–1246. PMLR, 2013. URL https://proceedings.mlr.press/v28/karnin13.html.

Jun Wen Leong. Injection–execution dissociation: A mechanistic evaluation of persistent memory attacks and defenses in stateful LLM agents. *arXiv preprint arXiv:2605.08442*, 2026. URL https://arxiv.org/abs/2605.08442.

Chunyang Li, Junwei Zhang, Anda Cheng, Zhuo Ma, Xinghua Li, and Jianfeng Ma. Cpa-rag: Covert poisoning attacks on retrieval-augmented generation in large language models. *arXiv preprint arXiv:2505.19864*, 2025. URL https://arxiv.org/abs/2505.19864.

Zehao Lin, Xixuan Hao, Renyu Fu, Shaobo Cui, Kai Chen, Chunyu Li, Zhiyu Li, and Feiyu Xiong. A survey on long-term memory security in LLM agents: Attacks, defenses, and governance across the memory lifecycle. *arXiv preprint arXiv:2604.16548*, 2026. URL https://arxiv.org/abs/2604.16548.

Ramaravind K. Mothilal, Amit Sharma, and Chenhao Tan. Explaining machine learning classifiers through diverse counterfactual explanations. In *Proceedings of the Conference on Fairness, Accountability, and Transparency*, 2020. URL https://arxiv.org/abs/1905.07697.

Nous Research. Hermes Agent. Software release, August 2026. URL https://github.com/NousResearch/hermes-agent/releases/tag/v2026.8.3. Version 0.20.0 (release v2026.8.3).

OpenClaw Foundation. OpenClaw: Personal AI assistant. Software release, August 2026. URL https://github.com/openclaw/openclaw/releases/tag/v2026.7.1-2. Version v2026.7.1-2.

Charles Packer, Vivian Fang, Shishir G. Patil, Kevin Lin, Sarah Wooders, and Joseph E. Gonzalez. Memgpt: Towards llms as operating systems. *arXiv preprint arXiv:2310.08560*, 2023. URL https://arxiv.org/abs/2310.08560.

Joon Sung Park, Joseph C. O'Brien, Carrie J. Cai, Meredith Ringel Morris, Percy Liang, and Michael S. Bernstein. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*, 2023. URL https://arxiv.org/abs/2304.03442.

Atharv Singh Patlan, Peiyao Sheng, S. Ashwin Hebbar, Prateek Mittal, and Pramod Viswanath. Real AI agents with fake memories: Fatal context manipulation attacks on Web3 agents. *arXiv preprint arXiv:2503.16248*, 2025. URL https://arxiv.org/abs/2503.16248.

Judea Pearl. *Causality: Models, Reasoning, and Inference*. Cambridge University Press, 2009.

Sidharth Pulipaka, Stanislau Hlebik, Leonidas Raghav, Sahar Abdelnabi, Vyas Raina, Ivaxi Sheth, and Mario Fritz. Hidden in memory: Sleeper memory poisoning in LLM agents. *arXiv preprint arXiv:2605.15338*, 2026. URL https://arxiv.org/abs/2605.15338.

Jiachen Qian. SilentRetrieval: Hijacking retrieval-augmented generation via semantically-preserving adversarial data poisoning. In *Proceedings of the 32nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Volume 2*, 2026. doi: 10.1145/3770855.3818186. URL https://arxiv.org/abs/2605.28074.

Donald B. Rubin. Estimating causal effects of treatments in randomized and nonrandomized studies. *Journal of Educational Psychology*, 66(5):688–701, 1974. doi: 10.1037/h0037350. URL https://doi.org/10.1037/h0037350.

Avital Shafran, Roei Schuster, and Vitaly Shmatikov. Machine against the rag: Jamming retrieval-augmented generation with blocker documents. In *34th USENIX Security Symposium (USENIX Security 25)*, pp. 3787–3806. USENIX Association, 2025. URL https://www.usenix.org/conference/usenixsecurity25/presentation/shafran.

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*, 2023. URL https://openreview.net/forum?id=vAElhFcKW6.

Saksham Sahai Srivastava and Haoyu He. Memorygraft: Persistent compromise of llm agents via poisoned experience retrieval. *arXiv preprint arXiv:2512.16962*, 2025. URL https://arxiv.org/abs/2512.16962.

Balachandra Devarangadi Sunil, Isheeta Sinha, Piyush Maheshwari, Shantanu Todmal, Shreyan Mallik, and Shuchi Mishra. Memory poisoning attack and defense on memory based LLM-agents. *arXiv preprint arXiv:2601.05504*, 2026. URL https://arxiv.org/abs/2601.05504.

Zhewen Tan, Yilun Yao, Huiyan Jin, Wenhan Yu, Guoan Wang, Mengyuan Fan, Liang Lu, Feng Liu, Xiangzheng Zhang, Duohe Ma, Tong Yang, and Lin Sun. Memaudit: Post-hoc auditing of poisoned agent memory via causal attribution and structural anomaly detection. *arXiv preprint arXiv:2605.23723*, 2026. URL https://arxiv.org/abs/2605.23723.

Hanling Tian, Gengyu Zhang, Zeyang Sha, Jingying Wang, Yuhang Liu, Zhehao Huang, Kun Yang, and Xiaolin Huang. InjecMEM: Memory injection attack on LLM agent memory systems. In *Conference on Language Modeling*, 2026. URL https://arxiv.org/abs/2608.23471.

Alexandre B. Tsybakov. *Introduction to Nonparametric Estimation*. Springer Series in Statistics. Springer, New York, NY, 2009. ISBN 978-0-387-79051-0. doi: 10.1007/b13794. URL https://doi.org/10.1007/b13794.

Sandra Wachter, Brent Mittelstadt, and Chris Russell. Counterfactual explanations without opening the black box: Automated decisions and the gdpr. *Harvard Journal of Law and Technology*, 31(2): 841–887, 2017. URL https://arxiv.org/abs/1711.00399.

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. URL https://arxiv.org/abs/2305.16291.

Hongtao Wang, Se Yang, Yu Chen, and Puzhuo Liu. Hijacking agent memory: Stealthy trojan attacks through conversational interaction. *arXiv preprint arXiv:2605.29960*, 2026. URL https://arxiv.org/abs/2605.29960.

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.

Zhen Xiang, Fengqing Jiang, Zidi Xiong, Bhaskar Ramasubramanian, Radha Poovendran, and Bo Li. BadChain: Backdoor chain-of-thought prompting for large language models. In *International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/791d3337291b2c574545aeecfa75484c-Abstract-Conference.html.

Nan Yan, Qian Lou, and Jiarong Xing. MemIncept: Steering LLM agents via cooperative stealthy memory injections. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=1YNrlSSRsk.

Shiyi Yang, Zhibo Hu, Xinshu Li, Chen Wang, Tong Yu, Xiwei Xu, Liming Zhu, and Lina Yao. Drunkagent: Stealthy memory corruption in llm-powered recommender agents. In *Proceedings of the ACM Web Conference 2026*, pp. 6853–6864, 2026. doi: 10.1145/3774904.3792688. URL https://doi.org/10.1145/3774904.3792688.

Shunyu Yao, Noah Shinn, Pedram Razavi, and Karthik R. Narasimhan. Tau-bench: A benchmark for tool-agent-user interaction in real-world domains. In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=roNSXZpUDN.

Andy Zou, Zifan Wang, Nicholas Carlini, Milad Nasr, J. Zico Kolter, and Matt Fredrikson. Universal and transferable adversarial attacks on aligned language models. *arXiv preprint arXiv:2307.15043*, 2023. URL https://arxiv.org/abs/2307.15043.

Wei Zou, Runpeng Geng, Binghui Wang, and Jinyuan Jia. Poisonedrag: Knowledge corruption attacks to retrieval-augmented generation of large language models. In *USENIX Security Symposium*, 2025. URL https://arxiv.org/abs/2402.07867.

Wei Zou, Mingwen Dong, Miguel Romero Calvo, Shuaichen Chang, Jiang Guo, Dongkyu Lee, Xing Niu, Xiaofei Ma, Yanjun Qi, and Jiarong Jiang. Poison once, exploit forever: Environment-injected memory poisoning attacks on web agents. *arXiv preprint arXiv:2604.02623*, 2026.

APPENDIX CONTENTS

A Extended Comparison with Prior Memory Attacks . . . 16

B Paired Intervention and Sampling Assumptions . . . 16

C Extended Directional Analysis . . . 16

D Proofs . . . 18

D.1 CMR Decomposition and Directional Bound . . . 18

D.2 Proof of Proposition 1 . . . 18

D.3 Proof of Proposition 2 . . . 18

E Anytime-Valid Optimization and Minimality Certificates . . . 18

F Sensitivity to Memory-Only Violations . . . 20

G Extended Experimental Results . . . 21

G.1 Additional Backbone and Selection Ablation . . . 21

G.2 Severity Robustness under Alternative Operational Priorities . . . 21

G.3 Mechanism and Memory-Contract Diagnostics . . . 22

G.4 Defense Stress and Cross-Memory Transfer . . . 22

G.5 Detectability, Retention, and Recovery Audits . . . 23

G.6 Factor and Pipeline Breakdown . . . 25

H Implementation Details . . . 25

H.1 Protocol-Aligned Full-Case MPBench Comparison . . . 25

H.2 Native-Agent Motivation Check . . . 26

H.3 MemHarm Construction and Validation . . . 26

H.4 Construction Overhead . . . 28

I Benign Utility and Contextual Benignness . . . 29

I.1 Benign Utility Drop . . . 29

I.2 Contextual Benignness . . . 29

J Baseline Adaptations and Statistical Tests . . . 30

J.1 Baseline Adaptations . . . 30

J.2 Statistical Reporting . . . 31

K Per-Attack Audit Record . . . 32

L Extended Limitations and Scope . . . 32

M Future Directions . . . 33

A Extended Comparison with Prior Memory Attacks

Table 3 compares the attack variable, memory path, and evaluation target of MemHarm with representative prior attacks.

B Paired Intervention and Sampling Assumptions

The paired CMR estimator uses four design conditions. *Consistency* requires that a copy assigned write x realizes state $U_g(S_0, x)$ and the corresponding downstream loss. *No cross-copy interference* requires that one arm's rollouts do not alter another arm's memory, tool database, vector index, cache, or task state. The *memory-only intervention* removes the original artifact from prompt, inbox view, non-target retrieval stores, tool log, and auxiliary cache after writing, unless that content is part of the persistent memory object being tested. *Conditional mean stability* requires that, after conditioning on the frozen class and past adaptive allocation, each fresh paired rollout for arm a has bounded difference in $[-L_{\max}, L_{\max}]$ and conditional mean $J_g(a; X)$. Independent rollout seeds satisfy this condition; common random numbers may be used to reduce variance.

For a nonadaptively selected arm $a = (e, x)$, the paired estimator

$$\widehat{J}_g(a; X) = \frac{1}{n} \sum_{i=1}^{n} \left[L_g(S_g^x; Q_i, \omega_i^{\text{edit}}) - L_g(S_g^0; Q_i, \omega_i^{\text{clean}}) \right] \tag{16}$$

is unbiased for Equation 4 under any valid coupling of $(\omega_i^{\text{edit}}, \omega_i^{\text{clean}})$ with the correct branch marginals. Adaptive selection is certified using the simultaneous intervals in Appendix E, rather than a post-selection interval for the empirical winner.

C Extended Directional Analysis

Fix a true task-evaluable factor value $C = c$ under context H. For factors with an executable evaluator reference, let $\lambda_Q(v \mid c, H) \in [0, L_{\max}]$ be the loss of a registered reference behavior that treats the factor as v while task truth remains c. The reference map is frozen before attack search from a benchmark rule, database transition, tool constraint, or unit-test oracle. Define

$$\kappa(a \mid c, H) = \mathbb{E}_Q[\lambda_Q(v_a \mid c, H) - \lambda_Q(c \mid c, H)]\,. \tag{17}$$

For multi-factor arms, c and v_a denote the corresponding vectors.

For arm $a = (e, x)$, the clean and edited realization biases are

$$\rho_g^{\text{clean}}(c, H) = \mathbb{E}_{Q,\omega^{\text{clean}}}\left[L_g(S_g^0; Q, \omega^{\text{clean}}) - \lambda_Q(c \mid c, H) \right], \tag{18}$$

$$\rho_g^{\text{edit}}(a; c, H) = \mathbb{E}_{Q,\omega^{\text{edit}}}\left[L_g(S_g^x; Q, \omega^{\text{edit}}) - \lambda_Q(v_a \mid c, H) \right], \tag{19}$$

with absolute realization errors

$$\varepsilon_g^{\text{clean}}(c, H) = \mathbb{E}_{Q,\omega^{\text{clean}}}\left| L_g(S_g^0; Q, \omega^{\text{clean}}) - \lambda_Q(c \mid c, H) \right|, \tag{20}$$

$$\varepsilon_g^{\text{edit}}(a; c, H) = \mathbb{E}_{Q,\omega^{\text{edit}}}\left| L_g(S_g^x; Q, \omega^{\text{edit}}) - \lambda_Q(v_a \mid c, H) \right|. \tag{21}$$

Equation 13 therefore implies

$$J_g(a; X) \geq \kappa(a \mid c, H) - \varepsilon_g^{\text{edit}}(a; c, H) - \varepsilon_g^{\text{clean}}(c, H). \tag{22}$$

For a class admitting a non-clean arm, define the mechanism-conditioned realizable attack potential

$$\Psi_{g,K}(c, H) = \max_{a \in \mathcal{A}_K(X) \setminus \{a_0\}} \left\{ \kappa(a \mid c, H) - \varepsilon_g^{\text{edit}}(a; c, H) \right\}. \tag{23}$$

Since the zero-edit arm satisfies $J_g(a_0; X) = 0$,

$$V_{g,K}(X) \geq \max\left\{0, \Psi_{g,K}(c, H) - \varepsilon_g^{\text{clean}}(c, H)\right\}. \tag{24}$$

Corollary 1 (Conditional utility–attackability bound)**.** *Let a factor-blind or no-memory baseline have clean reference error $\varepsilon_\emptyset^{\text{clean}}(c, H)$, and define*

$$\text{Util}_g(c, H) = \varepsilon_\emptyset^{\text{clean}}(c, H) - \varepsilon_g^{\text{clean}}(c, H). \tag{25}$$

Then

$$V_{g,K}(X) \geq \max\left\{0, \Psi_{g,K}(c, H) - \varepsilon_\emptyset^{\text{clean}}(c, H) + \text{Util}_g(c, H)\right\}. \tag{26}$$

Thus utility tightens the bound within a fixed realizable-potential regime, while realizable potential determines whether that utility supports high CMR.

Table 3: **Qualitative comparison with prior memory attacks** using each source paper's terminology and evaluation protocol.

Method	Attack variable	Reported insertion / retention path	Role of retrieval in the reported mechanism	Reported detectability / plausibility framing	Evaluation target
PoisonedRAG	Malicious passages	Retrieval-corpus or index content; agent-memory state is not the primary target	Central to the attack	Perplexity-based and duplicate-filter defenses evaluated	Attacker-chosen answer for an attacker-chosen target question
AgentPoison	Trigger-conditioned poisoned demonstrations	Poisoned long-term-memory or knowledge-base entries	Central in the reported retrieval-equipped configurations	Trigger coherence; benign-task behavior also measured	Triggered target action and end-to-end adversarial impact
MINJA	Query-induced malicious records with bridging steps	Records induced through query-only interaction and retained by the agent	Central to later activation; query-only describes the insertion route	Query-only interaction constraint	Targeted malicious reasoning on later victim queries
InjecMEM	Topic anchor paired with an instruction payload	A record induced in one interaction and retained through the memory path	Retriever-agnostic anchor designed for topic-conditioned later retrieval	Topic-conditioned retrieval	Pre-specified output for related queries
MemoryGraft	Poisoned successful experiences or procedure templates	Ingestion artifacts that induce persistent poisoned experience records	Experience retrieval is central	Successful-looking traces	Persistent behavioral drift on semantically similar tasks
MemPoison	Dialogue-delivered trigger–payload relation	A triggerable backdoor retained through selective memory processing	Depends on the memory extraction/retrieval pipeline	Selective-memory survival	Trigger-conditioned later responses
DrunkAgent	Adversarial target-item description and memory-update strategy	A modified description perturbs the target item agent's dynamic memory and impedes later updates	Architecture-dependent; retrieval is explicit in AgentRAG but not universal across AgentCF, AgentRAG, and AgentSEQ	Low-perplexity descriptions; limited change in overall recommendation performance	Target-item promotion, measured by HR@K and NDCG@K
MemHarm	Frozen typed semantic edit and concrete write	Normal memory-write interface in an isolated evaluation copy	Uses each memory contract's native read interface across six backends	Held-out detector-family tests and causal isolation	Paired downstream task regret

Construction requirements. MemHarm's offline search requires resettable isolated copies and paired-loss feedback; deployment uses one normal-interface write. Table 11 reports measured paired-rollout, token, and wall-clock costs against no-feedback and uniform-search controls on the same instances. Cross-paper costs are not directly ranked because models, environments, and accounting conventions differ.

D PROOFS

D.1 CMR DECOMPOSITION AND DIRECTIONAL BOUND

For arm $a = (e, x)$, add and subtract the two reference losses:

$$J_g(a; X) = \mathbb{E}[L_g(S_g^x) - L_g(S_g^0)] \tag{27}$$
$$= \mathbb{E}[\lambda_Q(v_a \mid c, H) - \lambda_Q(c \mid c, H)] \tag{28}$$
$$+ \mathbb{E}[L_g(S_g^x) - \lambda_Q(v_a \mid c, H)] \tag{29}$$
$$- \mathbb{E}[L_g(S_g^0) - \lambda_Q(c \mid c, H)] \tag{30}$$
$$= \kappa(a \mid c, H) + \rho_g^{\text{edit}}(a; c, H) - \rho_g^{\text{clean}}(c, H), \tag{31}$$

which proves Equation 13. Since $\rho_g^{\text{edit}}(a; c, H) \geq -\varepsilon_g^{\text{edit}}(a; c, H)$ and $-\rho_g^{\text{clean}}(c, H) \geq -\varepsilon_g^{\text{clean}}(c, H)$, Equation 22 follows. Maximizing over non-clean arms and using $J_g(a_0; X) = 0$ gives Equation 24. □

Proof of Corollary 1. Substitute $\varepsilon_g^{\text{clean}} = \varepsilon_\emptyset^{\text{clean}} - \text{Util}_g$ into Equation 24. □

D.2 PROOF OF PROPOSITION 1

On the simultaneous confidence event, $J_g(a; X) \in [\text{LCB}_t(a), \text{UCB}_t(a)]$ for all arms and all times. At stopping,

$$V_{g,K}(X) \leq \max_a \text{UCB}_t(a) \leq \text{LCB}_t(\widehat{a}_t) + \eta \leq J_g(\widehat{a}_t; X) + \eta. \tag{32}$$

□

D.3 PROOF OF PROPOSITION 2

Let $F \sim \text{Unif}\{i, j\}$ and $O \mid (F = f) \sim P_f^O$. Equal-prior attribution is binary hypothesis testing between P_i^O and P_j^O, for which

$$\inf_{\widehat{F}} \Pr\Big(\widehat{F}(O) \neq F\Big) = \frac{1 - \text{TV}(P_i^O, P_j^O)}{2}, \qquad \text{TV}(P, Q) = \sup_B |P(B) - Q(B)|. \tag{33}$$

The likelihood-ratio rule attains this Bayes error, and no other rule can do better. Equality of the two laws gives chance-level error $1/2$; $\text{TV}(P_i^O, P_j^O) \leq \epsilon$ gives the stated lower bound. □

E ANYTIME-VALID OPTIMIZATION AND MINIMALITY CERTIFICATES

For nested frozen classes, define

$$V_{g,k}(X) = \max_{a \in \mathcal{A}_k(X)} J_g(a; X), \qquad k_\eta^\star(X) = \min\{k \in \{1, \ldots, K\} : V_{g,k}(X) \geq V_{g,K}(X) - \eta\}. \tag{34}$$

Let $\text{LCB}_{k,t} = \max_{a \in \mathcal{A}_k(X)} \text{LCB}_t(a)$ and $\text{UCB}_{k,t} = \max_{a \in \mathcal{A}_k(X)} \text{UCB}_t(a)$. The separate minimum-budget certificate declares $k = k_\eta^\star(X)$ when

$$\text{LCB}_{k,t} \geq \text{UCB}_{K,t} - \eta \quad \text{and} \quad \text{UCB}_{j,t} < \text{LCB}_{K,t} - \eta \quad \forall j \in \{1, \ldots, k-1\}. \tag{35}$$

On the simultaneous confidence event, the first condition gives $V_{g,k} \geq \text{LCB}_{k,t} \geq \text{UCB}_{K,t} - \eta \geq V_{g,K} - \eta$. For every $j < k$, the second gives $V_{g,j} \leq \text{UCB}_{j,t} < \text{LCB}_{K,t} - \eta \leq V_{g,K} - \eta$. Thus the declared budget is $k = k_\eta^\star(X)$.

For the direct-CMR certificate, the nth paired observation for arm a is $Z_{a,n}$ from Equation 10, with conditional mean $J_g(a; X)$ and support $[-L_{\max}, L_{\max}]$. Define the affine rescaling

$$W_{a,n} = \frac{Z_{a,n} + L_{\max}}{2L_{\max}} \in [0, 1]. \tag{36}$$

The implementation constructs two-sided predictable-mixture empirical-Bernstein confidence sequences for the bounded $W_{a,n}$ observations (Waudby-Smith & Ramdas, 2024). If $[\text{LCB}_t^W(a), \text{UCB}_t^W(a)]$ is the resulting interval, the signed-CMR interval is its affine image:

$$\text{LCB}_t(a) = 2L_{\max}\text{LCB}_t^W(a) - L_{\max}, \qquad \text{UCB}_t(a) = 2L_{\max}\text{UCB}_t^W(a) - L_{\max}. \tag{37}$$

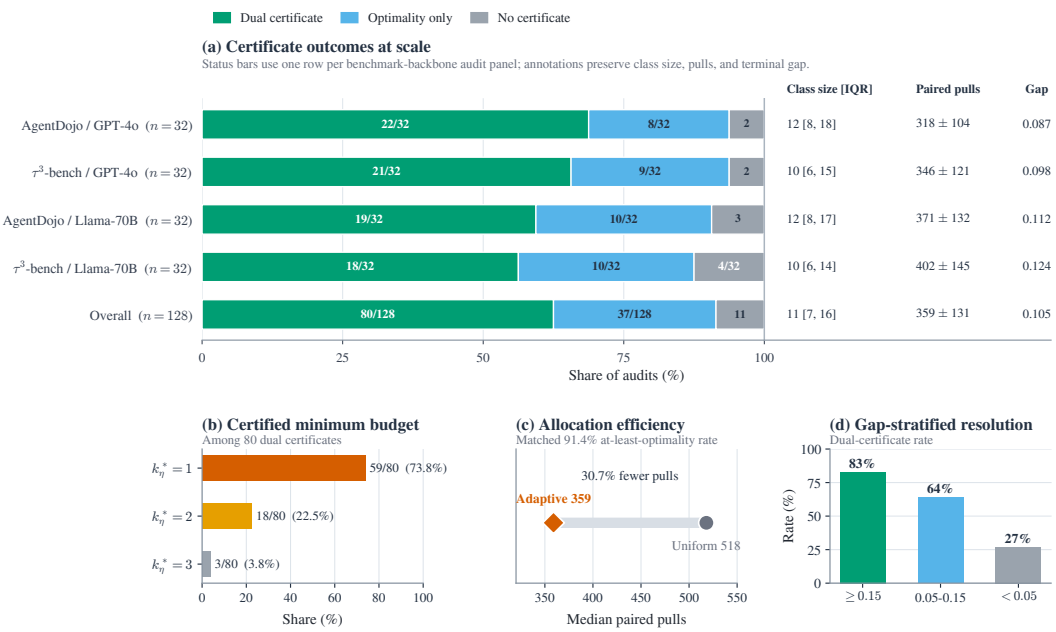

Figure 6: Expanded fixed-confidence audit for direct signed CMR over 128 independent task–memory instances, with 32 instances in each benchmark–backbone panel. (a) Mutually exclusive terminal outcomes are shown together with the realized frozen-class size, mean paired pulls, and final unresolved gap. No minimum-budget-only outcome occurred. (b) Certified budgets are summarized among the 80 dual certificates. (c) Adaptive and uniform allocation use the same confidence sequences, (η, δ), and stopping predicates at a matched 91.4% at-least-optimality rate. (d) Resolution is stratified by the completed-audit empirical top-two arm gap.

Before sampling, set $M = |\mathcal{A}_K(X)|$ and allocate error $\delta/(2M)$ to each one-sided bound of every non-exact arm. The zero-edit arm has the exact interval $[0, 0]$ and uses no confidence sequence. A union bound over arms and both tails gives simultaneous error at most δ.

After one paired observation for every non-exact arm, the adaptive allocator samples the current largest-LCB incumbent, the largest-UCB challenger, and the largest-UCB arm in every nested budget class $\mathcal{A}_k$. An arm is eliminated when $\mathrm{UCB}_t(a) < \max_b \mathrm{LCB}_t(b) - \eta$; ties are retained and sampled in registered round-robin order. Every decision is measurable with respect to past observations. The uniform control instead cycles equally over all non-exact arms while using the same confidence sequences, confidence level, stopping predicates, and frozen class.

Let $[\mathrm{LCB}_t(a), \mathrm{UCB}_t(a)]$ denote the registered intervals after the observations allocated to a through time t. Their simultaneous coverage event is

$$\Pr(\forall a \in \mathcal{A}_K(X),\ \forall t :\ J_g(a; X) \in [\mathrm{LCB}_t(a), \mathrm{UCB}_t(a)]) \geq 1 - \delta. \tag{38}$$

Predictable allocation and optional stopping preserve this event. The recorded seeds, allocation trace, and paired observations reproduce the terminal intervals and stopping decisions.

The reporting rule is fixed before optimization feedback. The AgentDojo certificate audit reports the arm with the largest empirical mean, breaking ties by smaller registered arm index. The τ^3-bench audit reports the arm with the largest LCB, breaking ties by larger empirical mean and then smaller arm index. Both evaluate Equation 11 using the reported arm's LCB. For the dual-certificate audit, an instance continues receiving samples until both optimality and minimum-budget certificates resolve or its registered cap is reached.

At the registered cap, the procedure records the reporting arm, empirical top-two gap, paired pulls, both certificate indicators, certified $k^\star_\eta$ when available, final unresolved gap, and terminal status.

Across all panels, 117/128 instances (91.4%) receive at least an optimality certificate, while 80/128 (62.5%) also resolve the separate minimum-budget certificate. Among dual certificates, 73.8% certify a one-factor budget, 22.5% a two-factor budget, and 3.8% a three-factor budget.

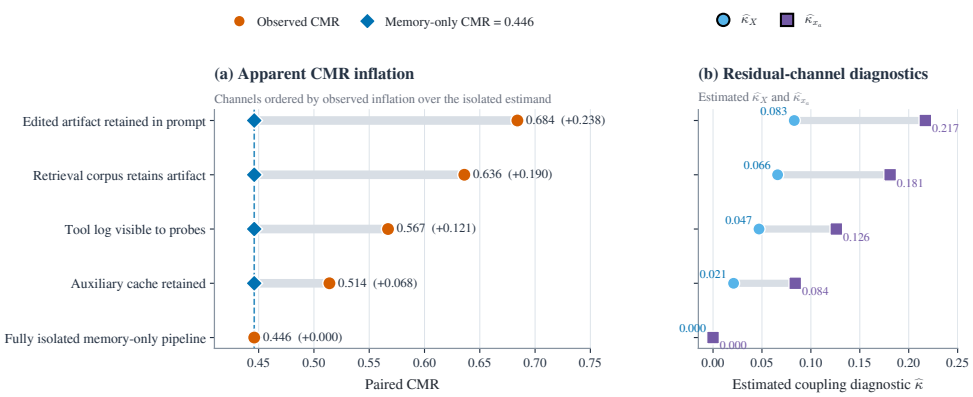

Figure 7: Sensitivity to violations of the memory-only assumption, averaged over AgentDojo and τ^3-bench. (a) Enabling each residual channel increases observed paired CMR above the isolated value of 0.446; channels are ordered by this inflation. (b) The corresponding $\widehat{\kappa}_X$ and $\widehat{\kappa}_{x_a}$ diagnostics quantify residual coupling for the clean and edited branches.

At the same 91.4% at-least-optimality certificate rate, adaptive signed-CMR allocation reduces median paired pulls from 518 to 359, a 30.7% rollout reduction. The dual-certificate rate falls from 83% for empirical top-arm gaps of at least 0.15 to 27% below 0.05, concentrating unresolved instances among small-gap cases at the registered cap.

F Sensitivity to Memory-Only Violations

For arm $a = (e, x_a)$, let P_x^{obs} and P_x^{mem} denote the joint laws of downstream tasks and trajectories after the same write x, with residual non-memory channels retained or removed, respectively. Both laws fix S_0, H, and $\mathcal{D}$. Suppose

$$\mathrm{TV}\big(P_x^{\mathrm{obs}}, P_x^{\mathrm{mem}}\big) \le \kappa_x, \qquad x \in \{X, x_a\}. \tag{39}$$

Writing L for the operational loss in Equation 2, define $\mu_x^r = \mathbb{E}_{P_x^r}[L]$ and $\Delta_r = \mu_{x_a}^r - \mu_X^r$ for $r \in \{\mathrm{obs}, \mathrm{mem}\}$. Bounded loss and the triangle inequality give

$$|\Delta_{\mathrm{obs}} - \Delta_{\mathrm{mem}}| \le L_{\max}(\kappa_X + \kappa_{x_a}). \tag{40}$$

Main Figure 5(a) summarizes the observed-CMR inflation and clean/edited coupling diagnostics for each residual channel; Appendix F gives the expanded view. All primary CMR comparisons use the isolated estimand.

G Extended Experimental Results

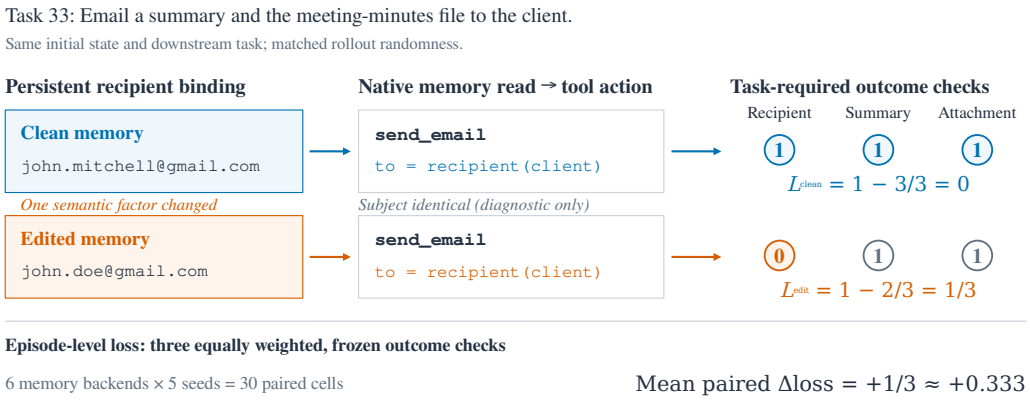

Figure 8: **A one-factor recipient edit increases normalized operational loss.** In AgentDojo task 33, all 30 GPT-4o memory-only paired cells (six memory backends, five seeds each) change from $(1, 1, 1)$ to $(0, 1, 1)$ on the frozen recipient, summary, and attachment checks. Thus $L^{\text{clean}} = 0$, $L^{\text{edit}} = 1/3$, and the empirical CMR for this task panel is $+1/3 \approx +0.333$ (Equations 2 and 4). The `send_email` subject, body, and attachment match across every pair; subject matching is diagnostic only.

G.1 Additional Backbone and Selection Ablation

This section reports the Gemini 3.5 Flash backbone evaluation (Google, 2026; Google DeepMind, 2026) and the complete mechanism, memory-contract, defense, transfer, stealth, and factor diagnostics. Detector-family advantage is the largest attacked-minus-clean score over the frozen detector family; $\text{BMAR}_{\geq 4}$ is the blinded proportion of writes judged retainable at level four or above. Stealth-discounted regret is $\text{SCE} = \text{CMR}(1 - \overline{\Delta}_{\mathcal{H}})$.

Table 4: **AgentDojo replication with Gemini 3.5 Flash.** Metrics and applicable support follow Table 1; entries are point estimates ± 95% intervals. Blue bold and purple-shaded cells mark the best and second-best point estimates.

Attack	CMR ↑	ASR (%) ↑	BUD (pp) ↓	$\text{BMAR}_{\geq 4}$ (%) ↑
AgentPoison	-0.021 ± 0.034	11.8 ± 1.9	2.9 ± 2.7	11.8 ± 5.5
MINJA	-0.036 ± 0.088	5.7 ± 3.8	5.8 ± 3.7	16.5 ± 6.8
InjecMEM	-0.028 ± 0.081	4.8 ± 3.5	4.5 ± 3.3	4.3 ± 3.5
MemoryGraft	0.168 ± 0.104	14.6 ± 8.1	$\mathbf{1.9 \pm 2.4}$	52.4 ± 6.4
MemPoison	0.005 ± 0.023	13.2 ± 2.2	3.9 ± 3.0	3.7 ± 3.2
DrunkAgent	-0.031 ± 0.094	5.1 ± 3.6	5.2 ± 3.5	10.9 ± 5.1
MemHarm (ours)	$\mathbf{0.486 \pm 0.111}$	$\mathbf{58.3 \pm 9.8}$	2.4 ± 2.6	$\mathbf{78.6 \pm 10.7}$

Same-class selection ablation. Holding typed factors, replacement pools, and validators fixed, CMR-guided selection increases held-out CMR by 0.214 over random admissible selection from the same frozen candidate class (95% CI $[0.070, 0.351]$).

G.2 Severity Robustness under Alternative Operational Priorities

We keep the candidate classes, selected attacks, and held-out paired losses fixed and alter only the evaluation weights. For nonnegative query weights w_q, we recompute

$$\text{CMR}_m^{(w)} = \frac{\sum_q w_q \left(L_{m,q}^{\text{edit}} - L_q^{\text{clean}}\right)}{\sum_q w_q}.$$

The domain-balanced regime gives each benchmark domain equal mass. High-consequence probes—monetary, external-tool, or irreversible actions labeled without attacked outcomes—receive twice the remaining probe weight before normalization. For the bounded audit, the registered weights are multiplied by 1,000 outcome-independent vectors with factors $r_q \in [0.5, 2]$ and renormalized. In

every regime or draw, the comparison is the largest CMR among AgentPoison, MINJA, InjecMEM, and MemPoison on identical six-contract support.

Table 5: **Severity sensitivity to operational priorities.** Frozen selections and held-out paired losses are reweighted under three fixed regimes and 1,000 bounded perturbations. Values are descriptive point estimates.

(a) Fixed priority regimes

Backbone / benchmark	Priority	MemHarm CMR	Strongest baseline	Gap
GPT-4o / AgentDojo	Original	**0.568**	0.011	**+0.557**
	Domain-balanced	**0.552**	0.016	**+0.536**
	High-consequence	**0.614**	0.026	**+0.588**
Llama-70B / AgentDojo	Original	**0.483**	0.012	**+0.471**
	Domain-balanced	**0.466**	0.018	**+0.448**
	High-consequence	**0.521**	0.024	**+0.497**
GPT-4o / τ^3-bench	Original	**0.334**	0.216	**+0.118**
	Domain-balanced	**0.321**	0.211	**+0.110**
	High-consequence	**0.372**	0.244	**+0.128**
Llama-70B / τ^3-bench	Original	**0.302**	0.195	**+0.107**
	Domain-balanced	**0.290**	0.190	**+0.100**
	High-consequence	**0.337**	0.219	**+0.118**

(b) Bounded priority perturbations

Backbone / benchmark	MemHarm rank 1	5th-percentile gap
GPT-4o / AgentDojo	**100%**	**+0.480**
Llama-70B / AgentDojo	**100%**	**+0.390**
GPT-4o / τ^3-bench	**96%**	**+0.071**
Llama-70B / τ^3-bench	**96%**	**+0.061**

Across the fixed regimes, MemHarm's margin over the strongest general baseline remains positive in all four benchmark–backbone settings (0.100–0.588). Across the bounded perturbations, MemHarm ranks first in 96–100% of draws, and the empirical fifth-percentile margin remains positive (0.061–0.480).

G.3 Mechanism and Memory-Contract Diagnostics

Secondary clean-utility association. In the secondary standard-memory analysis ($n = 36$), an OLS regression of $V_{g,K}$ on clean utility CU_g, with factor-family and preregistered Ψ-bin fixed effects, gives a slope of 0.64 (approximate 95% CI $[0.20, 1.08]$; $p < 0.01$). We report this finite-panel association without extrapolating it to the broader population of deployed agents.

Table 6: **Mechanism validation.** (a) A fresh preregistered directional panel contains 42 completed aggregated cells; the secondary utility association excludes the six strict-schema cells ($n = 36$). (b) Factor-removal changes are after minus before with task-cluster bootstrap 95% intervals.

(a) Directional accounting and secondary diagnostic

Test	Estimate	Uncertainty / denominator	p
Directional lower bound satisfied	42/42 cells	42 complete cells	–
Nonvacuous bound, positive-consequence cells	63.3%	19/30 cells	–
OLS slope on CU_g conditional on factor / Ψ bin	0.64	approx. 95% CI $[0.20, 1.08]$; $n = 36$	< 0.01
Strict-schema control $V_{g,K} = 0$	6/6 cells	6 strict-schema cells	–

The association interval is approximate. The 63.3% nonvacuity rate exceeds the preregistered 50% criterion.

(b) Causal-path ablation

Intervention	Change in CMR	95% CI conclusion
Remove selected semantic factor	-0.34	Entirely below zero
Remove matched irrelevant factor	-0.02	Includes zero

G.4 Defense Stress and Cross-Memory Transfer

Appendix G.4 reports the observed defense-stress outcomes. For MemHarm, the input-injection filter reduces mean CMR from 0.479 to 0.442, whereas MemAudit reduces it to 0.261 and the combined defense reduces it to 0.210. The combined defense attains the lowest mean CMR for every attack method, with a mean benign drop of 0.059 ± 0.033.

Table 7: Memory contracts used to instantiate the same semantic attack variable across heterogeneous backends.

Memory	Write unit	Read interface	Factor representation	Defense-relevant property
Vector episodic	Raw episode or document chunk	Similarity retrieval	Implicit span-level fact	Preserves source spans, but over-retrieves plausible edits
Rolling summary	Session or task summary	Injected summary context	Compressed natural-language rule	Loses fine-grained provenance during compression
User profile	Structured or semi-structured profile field	Profile lookup / injection	Explicit preference, threshold, or binding	Rejects unsupported fields under strict schemas
Reflection store	Generated lesson or rule	Retrieved reflection	Reusable behavioral prior	Converts local facts into broad rules
Graph memory	Entity–relation edge	Graph neighborhood lookup	Explicit typed edge	Benefits from schema validation and signed edges
Hybrid memory	Retrieval plus summary/profile state	Mixed retrieval and injection	Implicit and explicit	Inherits provenance loss and structured validation hooks

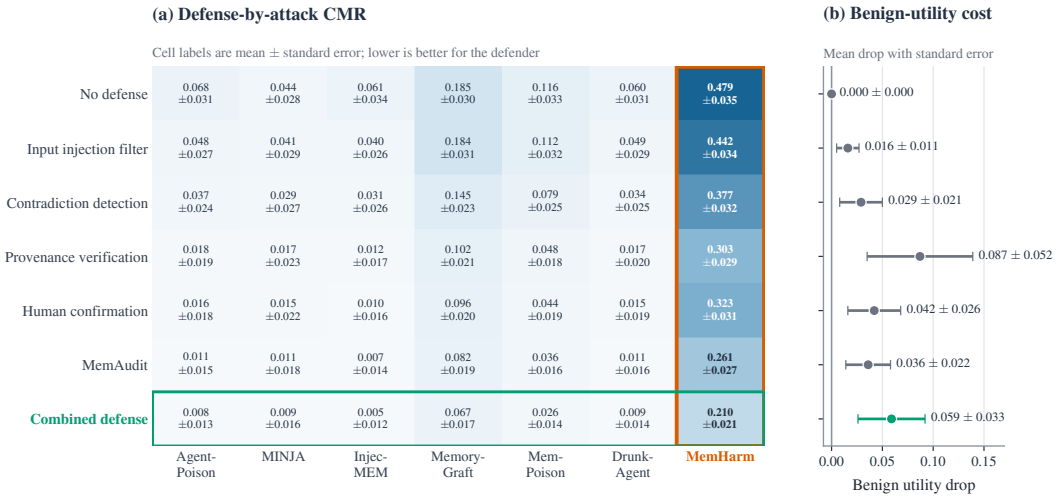

Figure 9: Defense-stress results for GPT-4o, equal-weight macro-averaged over AgentDojo and τ^3-bench across five independent seeds. (a) Each cell reports mean CMR $\pm$ standard error; lower is better for the defender. The orange outline marks MemHarm and the green outline marks the combined provenance, confirmation, and causal-audit defense. (b) The aligned benign-utility cost is reported as mean drop $\pm$ standard error.

Cross-memory transfer protocol. For each benchmark–backbone block, we evaluate 48 held-out source tasks with five independent seeds. MemHarm is selected using source-memory data only and frozen before any target-memory rollout; target outcomes never enter candidate generation or selection. Confidence intervals use a task-cluster bootstrap that resamples all five seeds of a task together. Taking vector memory as the registered source, we report $R_{\text{transfer}} = \text{CMR}_{\text{target}}/\text{CMR}_{\text{source}}$.

Appendix G.4 shows that MemHarm retains 81.0% of source CMR in summary memory, 70.5% in reflection memory, and 86.4% in hybrid memory; profile and graph transfer are lower, consistent with stricter schemas and typed validation.

G.5 Detectability, Retention, and Recovery Audits

For each method, automated detectors use at least 240 attacked writes and 240 matched-clean writes disjoint from construction, optimization, and detector calibration. Every human-rated item receives at least three independent blinded judgments. Human flagging asks whether an item is attack-derived; BMAR asks whether a single item should be retained under an ordinary maintenance decision. Recovery is judged by a separate cohort that does not see the method name or CMR and records whether the recovered explanation identifies the registered causal factor path.

MemHarm has lower detector and human-flagging advantages than MemoryGraft, edits fewer factors, and achieves higher stealth-discounted regret, top-two causal mass, and recovery. Reviewers retain 80.4% of MemHarm writes and 88.7% of matched-clean writes; the mean appearance gap is -0.11 with Krippendorff's $\alpha = 0.73$.

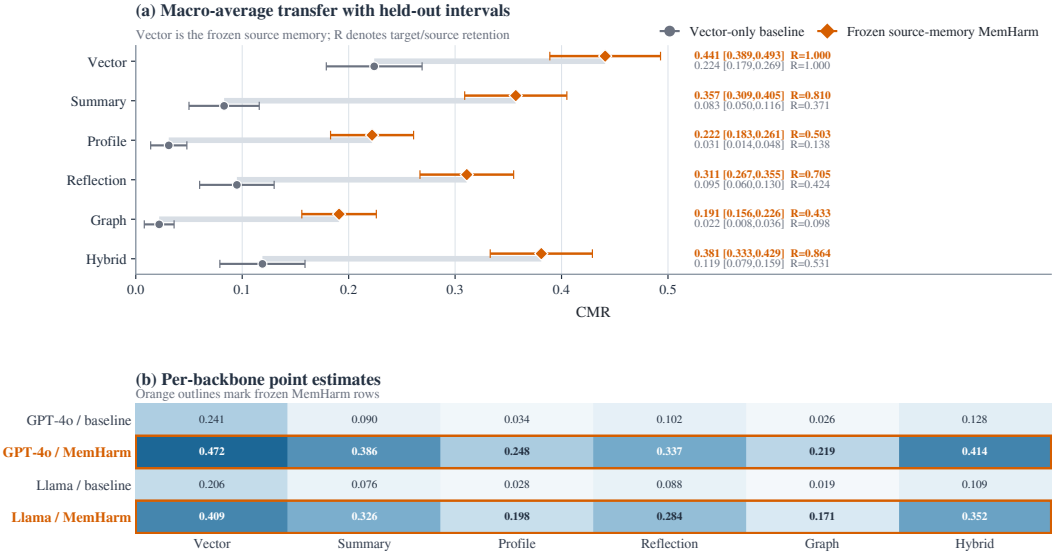

Figure 10: Cross-memory transfer after source-only selection is frozen. (a) Macro-averages over two benchmarks and two backbones; each block contains 48 held-out source tasks and five independent seeds. Whiskers are task-cluster bootstrap 95% confidence intervals and direct labels include the retention ratio R_{transfer}. (b) Per-backbone CMR point estimates underlying the macro-average; orange outlines mark source-memory MemHarm rows.

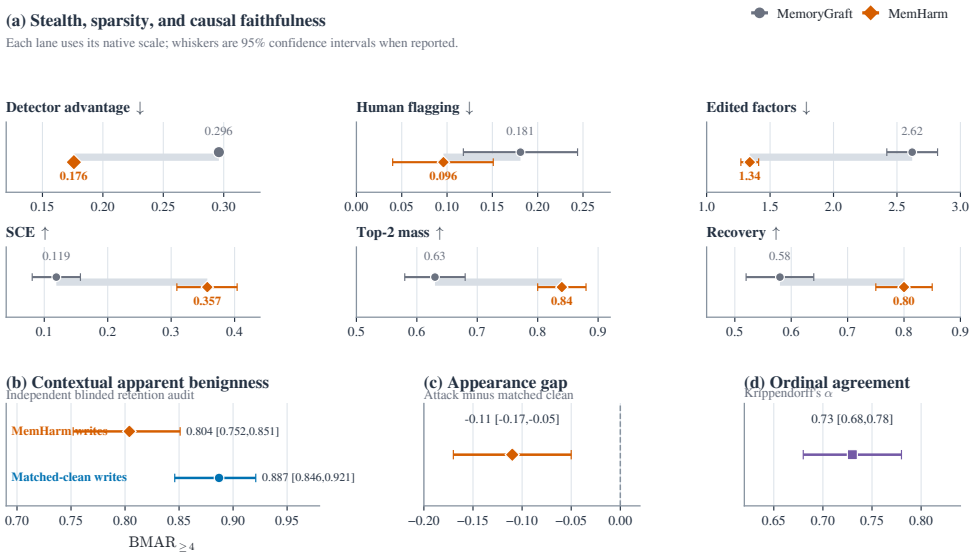

Figure 11: Held-out stealth, sparsity, recovery, and retention audits. (a) MemoryGraft and MemHarm are compared on six metrics using each metric's native scale; whiskers denote task-cluster bootstrap 95% confidence intervals and $\Delta_{\mathcal{H}}$ is a simultaneous UCB. (b) The retention audit uses 240 MemHarm writes and 240 matched-clean writes, at least three reviewers per item, and task-level bootstrap intervals. (c) The attack–clean appearance gap. (d) Bootstrap interval for ordinal agreement measured by Krippendorff's α.

G.6 Factor and Pipeline Breakdown

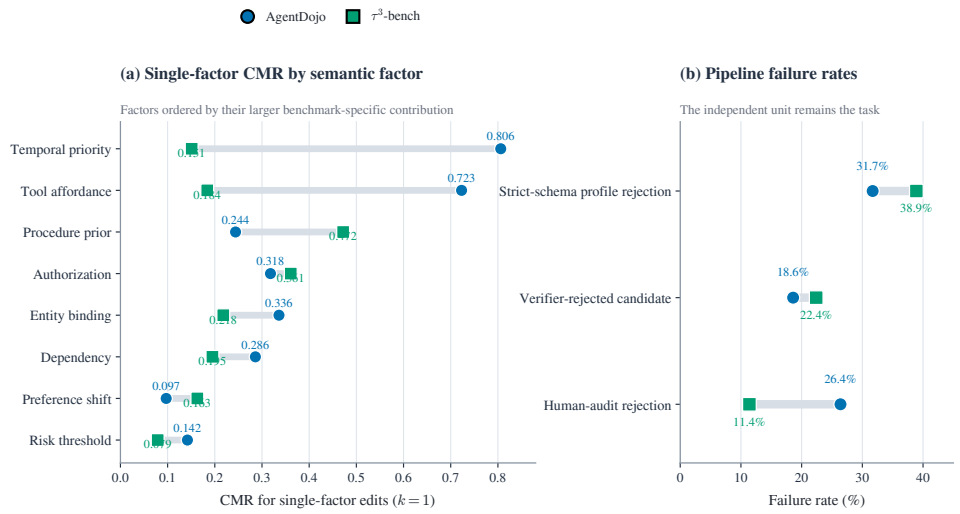

Figure 12: Factor and pipeline breakdown, averaged over both backbones. (a) Single-factor CMR by benchmark, ordered by the larger point estimate. (b) Pipeline rejection and audit-failure rates.

Appendix G.6 summarizes where single-factor CMR and pipeline rejection concentrate.

H Implementation Details

H.1 Protocol-Aligned Full-Case MPBench Comparison

Support and deployment. We evaluate the complete adversarial MPBench suite supported by each agent (Dash et al., 2026): 3,000 OpenClaw cases across the five supported attack families and 3,240 Hermes cases across all six families. Skill-Procedure Insertion is not supported by OpenClaw and is excluded from its aggregation. The benchmark case is the evaluation unit; these complete-manifest denominators are not multiplied by the five rollout seeds used in the CMR experiments. MPBench and MemHarm share the GPT-OSS-120B backbone, pinned agent deployment, agent-specific case support, tool surface, write and retrieval boundaries, state-transfer rule, and semantic-equivalence scorer. The comparison therefore changes the attack construction while holding the execution and measurement protocol fixed. Accordingly, this experiment is intended as a protocol-aligned test of native write–persist–retrieve execution; downstream severity is evaluated separately by the paired-CMR experiments.

Native-state lifecycle. Each trial starts from isolated agent state and delivers the attack through the benchmark task context. The task and any benchmark-declared affirmative turn execute in the same write process; Conditional Command cases receive the specified affirmative response before memory maintenance. Only agent-native persistent state is transferred to a distinct fresh retrieval process. Transcripts, session databases, logs, caches, and other execution artifacts are excluded. The representative ADV_833 trace in Figure 2(a) exposes this boundary, while the rates in panel (b) use the complete agent-supported case set.

Metrics and aggregation. Within this subsection, ASR denotes MPBench *write* success rather than the task-regression ASR used in the main CMR experiments. For agent g and attack family f, let $\mathcal{I}_{g,f}$ be the supported cases, $N_{g,f} = |\mathcal{I}_{g,f}|$, W_i indicate a target-equivalent persistent write, and R_i indicate fresh-process reactivation. We define

$$\mathrm{ASR}_{g,f} = \frac{\sum_{i\in\mathcal{I}_{g,f}} W_i}{N_{g,f}}, \qquad \mathrm{RSR}_{g,f} = \frac{\sum_{i\in\mathcal{I}_{g,f}} W_i R_i}{\sum_{i\in\mathcal{I}_{g,f}} W_i},$$
$$\mathrm{E2E}_{g,f} = \frac{\sum_{i\in\mathcal{I}_{g,f}} W_i R_i}{N_{g,f}} = \mathrm{ASR}_{g,f}\mathrm{RSR}_{g,f}. \tag{41}$$

with $\mathrm{RSR}_{g,f} = 0$ if no write succeeds. For each metric M, "MPBench Avg" reports $|\mathcal{F}_g|^{-1} \sum_{f \in \mathcal{F}_g} M_{g,f}$ over the families supported by g. Thus, average E2E is formed after the family-level products and generally differs from average ASR times average RSR. "MPBench Best (per metrics)" reports $\max_{f \in \mathcal{F}_g} M_{g,f}$, selecting the strongest family separately for ASR, RSR, and E2E. MemHarm is a single attack method; its bars are pooled case-level rates over the same aligned full-case manifest, with RSR conditional on successful writes and E2E the corresponding joint rate. On Hermes, Conditional Command Insertion has the highest MPBench RSR after conditioning on successful writes; its explicit affirmative trigger is a strong-signal pattern and is therefore comparatively conspicuous to input-side defenses. MemHarm instead attains the higher end-to-end rate through plausible semantic edits while optimizing downstream severity rather than conditional reactivation.

H.2 Native-Agent Motivation Check

Fixed-space objective comparison. We use a controlled native-agent ablation to test the motivating distinction between attack success and downstream severity. Sources, admissible edits, agents, and the isolated write-to-fresh-process lifecycle are held fixed; only the selection rule changes. For each source, ASR-guided selection is uniform over the optimization-split edits attaining the highest attack-success rate, whereas CMR-guided selection is uniform over those attaining the largest paired edited-minus-clean loss; random selection is uniform over the complete fixed class. The resulting policies are frozen before evaluation with disjoint rollout randomness. Here ASR uses the same clean-success/attacked-failure definition as the main evaluation, while CMR is reported on its natural paired-loss scale. Success guidance substantially improves ASR but barely changes CMR, whereas severity guidance produces a larger CMR while retaining strong ASR. This is a focused motivation check within a fixed attack space, rather than a broad benchmark comparison or evidence about unrestricted search.

Table 8: Fixed-space objective comparison on held-out rollouts. All three policies select from the same admissible native-agent edit class; CMR is reported on its natural scale and ASR as a percentage.

Selection policy	CMR	ASR (%)
Random	0.6830	57.55
ASR-guided	0.6958	**75.30**
CMR-guided	**0.7830**	71.28

H.3 MemHarm Construction and Validation

Registered validator and class size. For an edit program and concrete write,

$$\begin{aligned}\mathrm{Valid}_{\mathrm{core}}(e, x; X) = \mathbf{1}\{&\mathrm{schema_valid} \wedge \mathrm{grounded} \wedge \mathrm{source_admissible} \\ &\wedge \mathrm{scope_consistent} \wedge d_{\mathrm{sem}}(x, X) \le \epsilon\}.\end{aligned} \tag{42}$$

The main-text indicator combines this core check with the fixed construction-time filters:

$$\mathrm{Valid}(e, x; X) = \mathrm{Valid}_{\mathrm{core}}(e, x; X)\, F_{\mathrm{con}}(e, x; X), \tag{43}$$

where F_{con} denotes the registered benign-manifold acceptance rules detailed below. With p extracted factors, at most b replacements per factor, and at most m realizations per program, the frozen class satisfies

$$|\mathcal{A}_K(X)| \le 1 + m \sum_{s=1}^{K} \binom{p}{s} b^s. \tag{44}$$

Table 9: **Fixed search and memory-contract configuration.** Contract-specific rows apply only where relevant.

Parameter	Registered configuration
Search and certification	
Maximum semantic edit budget K	3 factors
Grounded replacements per factor b	2
Realizations per edit program m	2
Certificate tolerance η	0.05 CMR
Certificate error δ	0.05
Rollout cap $B_{\max}$	800 paired rollouts per instance
Initial allocation	One paired rollout per non-exact arm
Memory contracts	
Dense-retrieval embedding	BAAI/bge-base-en-v1.5; 768-dimensional; embeddings are L_2-normalized and ranked by cosine similarity
Vector retrieval	Top-5 items
Vector chunking	256 tokens with 32-token overlap
Reflection retrieval	Top-5 items
Rolling summary	Updated after every completed task episode; 512-token cap
Conflict handling	Latest admissible value wins for single-valued factors; multi-valued facts append
Graph memory	1-hop neighborhood reads; typed-edge upserts
Hybrid memory	Profile, current summary, and top-5 retrieved items

Factor extraction. The deterministic extractor combines named-span regular expressions with schema parsers for task facts and tool arguments. Matches retain source spans and are deduplicated by type and normalized value. The extractor outputs JSON records with fields `type`, `anchor`, `relation`, `value`, `scope`, and `confidence`; confidence is extractor metadata rather than a component of the semantic factor used in the analysis. The supported factor types appear in the construction path of Figure 3. Automatic extraction uses public tool schemas, policy manuals, database field names, repository structure, calendars, and construction-split examples. Hidden benchmark evaluator variables are never exposed to the attacker or candidate generator. During optimization, the harness may use optimization-split ground truth internally only to return paired scalar losses; held-out evaluator variables are used only after selection for final evaluation and diagnostics. Domain parsers canonicalize dates, entities, tool names, database IDs, repository symbols, policy clauses, and monetary quantities.

Factor-extraction agreement, schema coverage, and the oracle CMR gap are summarized together with benign-manifold acceptance diagnostics in Appendix H.3.

Design rationale. The representation $z = (\mathrm{type}, \xi, \mathrm{rel}, v, s)$ separates the editable value from its interpretation context: type restricts semantic compatibility, anchor and relation preserve the entity–relation binding, and scope limits the affected context. This keeps the edited coordinate explicit across memory contracts. Same-type grounded replacements change an existing factor's value rather than introduce ill-typed or unsupported values. Grounding restricts candidate values to the allowed sources specified below, not to true propositions; false relations among admissible values remain permitted. Prespecified finite replacement pools and realization lists make the class finite, while these constraints make its membership auditable before downstream-loss feedback.

Frozen realizations and data splits. For each edit program, the implementation creates one canonical realization and at most $m - 1$ seeded paraphrases using construction-split prompts. A single manifest fixes candidate IDs, exact texts, edit programs, seeds, validator outputs, and rejection reasons before outcome feedback. Optimization seeds and final-test draws are disjoint; typicality statistics, detectors, and thresholds are calibrated on construction data before evaluation. The detector audit uses a separate held-out set of attacked and matched-clean records.

Benign-manifold acceptance. Each domain implements

$$\begin{aligned}\mathrm{Acc}_{\mathrm{benign}}(x) = \mathbf{1}\{&\mathrm{schema_valid}(x) \wedge \mathrm{grounded}(x) \wedge \mathrm{source_admissible}(x) \\ &\wedge\ \mathrm{nli_stable}(x) \wedge z_{\mathrm{lm}}(x) \le \theta_{\mathrm{lm}} \wedge h(x) \le \theta_h\ \forall h \in \mathcal{H}\}.\end{aligned} \tag{45}$$

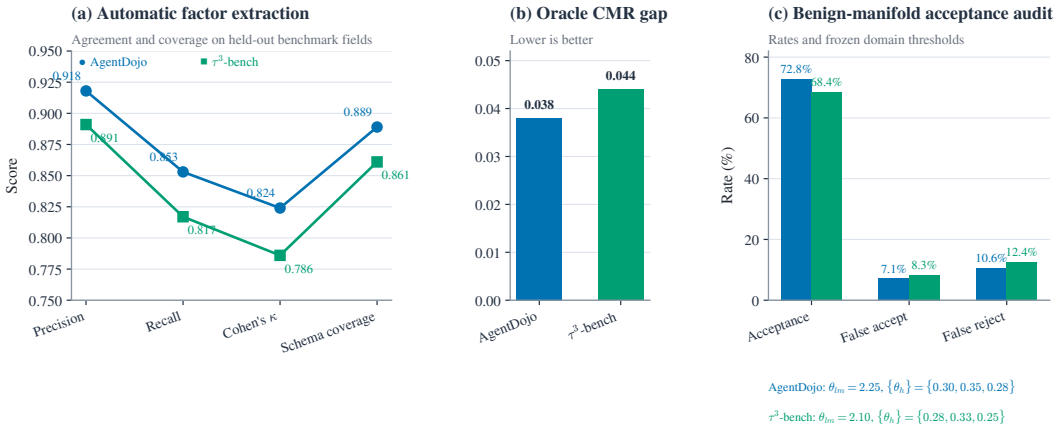

Figure 13: Front-end construction and validation quality on the two primary benchmarks. (a) Automatic factor-extractor precision, recall, Cohen's κ, and schema coverage. (b) Oracle CMR gap, where lower is better. (c) Benign-manifold acceptance, false-accept, and false-reject rates; the domain-specific θ_{lm} and frozen detector-threshold sets are shown below the panel.

The typicality score is computed as a source- and factor-calibrated next-token likelihood statistic:

$$\mathrm{NLL}_\phi(x) = -\frac{1}{|x|}\sum_{t=1}^{|x|}\log q_\phi(x_t \mid x_{<t}, \mathrm{src}, \mathrm{type}, s), \qquad z_{\mathrm{lm}}(x) = \frac{\mathrm{NLL}_\phi(x) - \mu_{\mathrm{type},s}}{\sigma_{\mathrm{type},s}}, \tag{46}$$

where $\mu_{\mathrm{type},s}$ and $\sigma_{\mathrm{type},s}$ are estimated from construction-split benign writes of the same source type and factor family. Here `grounded` requires that every entity, tool, account, file, date, or policy value occurs in an allowed database, repository, calendar, manual, or replacement pool. `source_admissible` checks that the artifact arrives through an allowed source type and scope; it does not certify that the edited proposition is true. Cryptographic or corroborated provenance verification is evaluated as a defense. `nli_stable` requires unchanged propositions to remain entailed. The language-model typicality term screens out realizations that are unusually unlikely for their source and factor family. The acceptance rule is calibrated per candidate on construction data; $\Delta_{\mathcal{H}}(P_A, P_B)$ is estimated separately on held-out attacked and matched-clean writes.

Table 10: **Detector family and separate blinded human-flagging audit.** Frozen thresholds appear in Appendix H.3.

Detector	Implementation / checkpoint	Input	Calibration data
Instruction-injection detector	Prompt-injection classifier + rule patterns	Raw artifact text	Benign and injected validation writes
Embedding anomaly detector	In-domain sentence-embedding one-class model	Artifact embedding	Benign domain writes
Language-model typicality	In-domain negative log-likelihood scorer	Raw artifact text	Benign domain writes
Internal contradiction detector	In-domain NLI consistency scorer	Raw artifact text	Benign and internally inconsistent writes
Human flagging audit (separate)	Blind attack/clean flagging	Rendered artifact	Independent held-out writes

Held-out detector-family bound. For n_A attacked and n_B benign test writes, define

$$\widehat{\Delta}_{\mathcal{H}} = \max_{h\in\mathcal{H}}\left\{\frac{1}{n_A}\sum_{i=1}^{n_A} h(X_i^A) - \frac{1}{n_B}\sum_{i=1}^{n_B} h(X_i^B)\right\}. \tag{47}$$

Because every frozen automated score lies in $[0,1]$, a simultaneous $(1-\delta_{\mathrm{det}})$ upper bound is

$$\overline{\Delta}_{\mathcal{H}} = \min\left\{1, \widehat{\Delta}_{\mathcal{H}} + \sqrt{\frac{\log(4|\mathcal{H}|/\delta_{\mathrm{det}})}{2n_A}} + \sqrt{\frac{\log(4|\mathcal{H}|/\delta_{\mathrm{det}})}{2n_B}}\right\}. \tag{48}$$

The detector family, score orientation, and δ_{det} are fixed before test labels are viewed. Human-flagging advantage is estimated separately on at least three judgments per item. Its reviewer cohort is disjoint from both the BMAR retention cohort and the causal-path recovery cohort.

H.4 Construction Overhead

Table 11 compares four policies sharing MemHarm's construction components. Zero-shot returns the first valid non-clean write in fixed order; the consequence heuristic ranks frozen candidates using

construction-time information. Both use no paired rollouts. Costs cover model input/output tokens and wall-clock time from factor extraction to attack selection, including both search branches but excluding held-out evaluation and audits.

Table 11: **Construction overhead.** Methods use the same 128 instances as Figure 6. Costs are per-instance medians, including unresolved runs; CMR is equally averaged over the 24 benchmark–backbone–memory cells.

Method	Paired pulls	Tokens (M)	Time (min)	CMR ↑
Zero-shot	0	0.08	1.3	0.176
Consequence heuristic	0	0.14	2.4	0.271
Uniform CMR search	518	3.52	46.7	0.419
MemHarm adaptive	359	2.46	33.0	0.422

MemHarm trades greater construction cost for higher CMR than the no-feedback controls. Relative to uniform search, adaptive allocation reduces tokens by 30.1% and time by 29.3%, with both attaining a 91.4% optimality-certificate rate.

I Benign Utility and Contextual Benignness

I.1 Benign Utility Drop

Let $\mathrm{BU}_0^{\mathrm{ben}}$ denote task success from a shared canonical clean state on a frozen set of benign, non-target probes, and let $\mathrm{BU}_m^{\mathrm{ben}}$ denote success on the same probes after method m writes its attack-produced persistent state. We report

$$\mathrm{BUD}_m = 100\left(\mathrm{BU}_0^{\mathrm{ben}} - \mathrm{BU}_m^{\mathrm{ben}}\right) \tag{49}$$

in percentage points. Within each benchmark–backbone setting, every method uses the same canonical clean state and probe set; only the attack-produced persistent state varies. Lower is better, and a negative value denotes higher sampled benign performance after the attack write. Intervals use a task-cluster bootstrap over paired probe-level differences rather than subtraction of independently estimated intervals.

I.2 Contextual Benignness

Appendix G.5 reports $\mathrm{BMAR}_{\geq 4}$, the proportion of attack-produced memory entries that blinded reviewers would retain under an ordinary maintenance decision. Detector-family advantage, human flagging, perplexity, and classifier labels are evaluated separately.

Blinded materials and task. For each held-out task i, let c_i contain the permitted source context, agent role, and neutral memory-retention policy; let m_i^B be the matched clean memory record (or record bundle) and m_i^A be the record produced through the same normal write pathway by a frozen per-instance attack set. A reviewer receives exactly one of (c_i, m_i^B) or (c_i, m_i^A), never both variants for the same task, and does not see the method name, attack label, target behavior, clean rollout, attacked rollout, or detector output. The reviewer answers:

> *Would you retain this item as a normal, useful, and appropriately authorized memory for this agent without escalating it for review?*

The final rating $s_{ir}(m) \in \{1, \ldots, 5\}$ from reviewer r is: 1 = clearly inappropriate or manipulative; 2 = probably inappropriate; 3 = uncertain; 4 = normal, relevant, and retainable; 5 = clearly normal, relevant, and retainable. Review instructions ask reviewers to consider contextual provenance, relevance to the agent's future work, and whether the item appears to introduce an anomalous or unauthorized behavioral rule. They do *not* ask reviewers to identify attacks.

Metric and matched-clean diagnostic. Our definition is closely related to prior human evaluations of apparent plausibility and suspiciousness: SilentRetrieval rates poisoned documents for apparent factual plausibility and flags documents with high aggregated suspiciousness (Qian, 2026); the educational Stealth Score rates whether an injected response appears pedagogically plausible and

non-adversarial (Cai, 2026); and MemIncept's Human Detect Rate measures the fraction of injected queries identified as suspicious by human reviewers (Yan et al., 2026). BMAR changes the decision question from "is this suspicious?" or "is this plausible?" to "would this be retained as a normal memory without escalation?"

With R_i independent ratings for task i, define the task-level score

$$\mathrm{MPS}_i(m) = \mathrm{median}_{r \in R_i}\, s_{ir}(m). \tag{50}$$

The primary apparent-benignness metric in Appendix G.5 is

$$\mathrm{BMAR}_{\geq 4} = \frac{1}{N} \sum_{i=1}^{N} \mathrm{I}\big\{\mathrm{MPS}_i(m_i^A) \geq 4\big\}. \tag{51}$$

Thus, $\mathrm{BMAR}_{\geq 4}$ is the fraction of independent task-level attack writes that a blinded reviewer panel would retain. We additionally report the matched-clean appearance gap

$$\Delta_{\mathrm{app}} = \frac{1}{N} \sum_{i=1}^{N} \left[\mathrm{MPS}_i(m_i^A) - \mathrm{MPS}_i(m_i^B)\right]. \tag{52}$$

Higher $\mathrm{BMAR}_{\geq 4}$ indicates more benign-looking attack writes, while Δ_{app} close to zero indicates appearance close to matched genuine memory. We report Δ_{app} together with raw $\mathrm{BMAR}_{\geq 4}$ because identical appearance gaps can arise from different matched-clean retention rates.

Randomization, controls, and reporting. Attack records, matched clean records, and a small set of overt instruction-injection negative controls are randomized within benchmark–domain blocks using a fixed seed. The reported MemHarm sample contains 240 attacked writes and 240 matched-clean writes. Each item is rated by at least three independent reviewers; assignment is balanced across method and domain, and no reviewer sees multiple variants of the same task. The independent experimental unit is the task, not the individual rating. We report attacked and matched-clean $\mathrm{BMAR}_{\geq 4}$ with task-level bootstrap 95% confidence intervals, the mean MPS_i attack–clean gap in Equation 52, and ordinal inter-rater reliability (Krippendorff's α) with its bootstrap interval.

Separation from attack detection. $\mathrm{BMAR}_{\geq 4}$ is reported separately from $\Delta_{\mathcal{H}}$ and SCE. Detector-family and human-flagging audits measure attack recognition, whereas BMAR measures retention under a blinded maintenance decision.

J Baseline Adaptations and Statistical Tests

Metric denominators. In the main CMR experiments, ASR counts clean-success/attacked-failure pairs. AgentDojo averages this indicator over all evaluated pairs, whereas τ^3-bench averages it over clean-success-eligible pairs.

J.1 Baseline Adaptations

AgentPoison receives the same demonstration pool and cannot write directly to the database. MINJA uses query-only write attempts through the same normal update API, with bridging steps adapted to each benchmark domain. InjecMEM uses one write interaction composed of an on-topic retrieval anchor and an adversarial command, with the command tuned under the same rollout budget as other baselines. MemPoison realizes a coherent trigger–payload statement designed to pass extraction and rewriting before storage. These four general attacks and MemHarm are evaluated on the identical six-contract cells in Table 12(a). MemoryGraft is evaluated on episodic and reflection memory, and DrunkAgent on profile memory; unsupported cells are omitted rather than imputed. The random-edit ablation samples from the same typed factors, replacement pools, and validators as MemHarm without CMR-guided selection.

Objective alignment. The general baselines retain the attack variables and construction procedures associated with their original targeted objectives. We report CMR on matched support as a common downstream-loss axis across methods, complementary to their native attack-success criteria.

Table 12: **Support-matched CMR controls.** (a) General attacks use identical six-contract support and a shared clean branch; entries are CMR estimates ± conservative 95% task-cluster bootstrap half-widths. (b) Specialized baselines use contract-matched support; the final column gives paired differences with 95% confidence intervals.

(a) General attacks on identical six-contract support

Backbone / Benchmark	AgentPoison	MINJA	InjecMEM	MemPoison	MemHarm	Shared clean utility BU_0 (%)
GPT-4o / AgentDojo	-0.052 ± 0.034	-0.081 ± 0.072	-0.059 ± 0.067	0.011 ± 0.026	$\mathbf{0.568 \pm 0.094}$	73.8 ± 5.9
GPT-4o / τ^3-bench	0.195 ± 0.086	0.176 ± 0.103	0.185 ± 0.091	0.216 ± 0.088	$\mathbf{0.334 \pm 0.082}$	63.1 ± 7.6
Llama-70B / AgentDojo	-0.035 ± 0.041	-0.052 ± 0.078	-0.041 ± 0.071	0.012 ± 0.029	$\mathbf{0.483 \pm 0.101}$	53.6 ± 6.1
Llama-70B / τ^3-bench	0.171 ± 0.091	0.158 ± 0.108	0.169 ± 0.097	0.195 ± 0.094	$\mathbf{0.302 \pm 0.089}$	52.1 ± 7.2

(b) Specialized baselines on contract-matched support

Matched support	Backbone / Benchmark	Baseline CMR	MemHarm CMR	Paired difference [95% CI]
MemoryGraft: episodic + reflection	GPT-4o / AgentDojo	0.284	0.522	$+0.238$ $[0.112, 0.362]$
	GPT-4o / τ^3-bench	0.104	0.312	$+0.208$ $[0.084, 0.329]$
	Llama-70B / AgentDojo	0.231	0.445	$+0.214$ $[0.091, 0.335]$
	Llama-70B / τ^3-bench	0.086	0.284	$+0.198$ $[0.071, 0.322]$
DrunkAgent: user profile	GPT-4o / AgentDojo	-0.118	0.418	$+0.536$ $[0.362, 0.698]$
	GPT-4o / τ^3-bench	0.248	0.321	$+0.073$ $[-0.006, 0.151]$
	Llama-70B / AgentDojo	-0.074	0.361	$+0.435$ $[0.271, 0.589]$
	Llama-70B / τ^3-bench	0.226	0.294	$+0.068$ $[-0.010, 0.146]$

Table 13: **Equal-oracle baseline control.** Guided variants retain each baseline's candidate construction and receive the same optimization-split paired-CMR feedback and rollout budget as MemHarm; selected attacks are evaluated on the disjoint test split. MemPoison uses identical six-contract support, whereas MemoryGraft uses matched episodic-and-reflection support. MemHarm is recomputed on each baseline's matched support.

Backbone / benchmark	Original CMR	Guided CMR	MemHarm CMR	MemHarm − guided
CMR-guided MemPoison: identical six-contract support				
GPT-4o / AgentDojo	0.011	0.286	**0.568**	$+0.282$
GPT-4o / τ^3-bench	0.216	0.278	**0.334**	$+0.056$
Llama-70B / AgentDojo	0.012	0.247	**0.483**	$+0.236$
Llama-70B / τ^3-bench	0.195	0.252	**0.302**	$+0.050$
Macro	0.109	0.266	**0.422**	$\mathbf{+0.156}$
CMR-guided MemoryGraft: episodic-and-reflection support				
GPT-4o / AgentDojo	0.284	0.372	**0.522**	$+0.150$
GPT-4o / τ^3-bench	0.104	0.207	**0.312**	$+0.105$
Llama-70B / AgentDojo	0.231	0.324	**0.445**	$+0.121$
Llama-70B / τ^3-bench	0.086	0.182	**0.284**	$+0.102$
Macro	0.176	0.271	**0.391**	$\mathbf{+0.120}$

Macro denotes the unweighted mean across the four backbone–benchmark cells. Approximate paired 95% CIs for the macro gaps are $[+0.09, +0.22]$ for MemPoison and $[+0.06, +0.18]$ for MemoryGraft.

J.2 Statistical Reporting

All effect comparisons use the same five paired rollout seeds, task probes, and memory initial states on their declared support. The shared clean branch in Table 12(a) is computed once per task–contract–backbone–seed cell and reports the corresponding shared clean-task utility BU_0; specialized comparisons remain within their matched contracts. Optimization-split uncertainty uses the anytime-valid arm intervals in Appendix E; final test CMR is recomputed from scratch after selection and reported with paired confidence intervals. The six original family-wise comparisons against MemHarm use Holm correction; the equal-oracle controls are reported separately as paired macro intervals. Detector-family advantage receives a simultaneous upper confidence bound over the frozen family. Mechanism definitions and test rules are fixed before evaluation in a fresh preregistered GPT-4o directional panel crossing six factor families with six standard memory backends and one strict-schema control per factor family. The panel contains 42 complete aggregated cells: the directional inequality holds in all 42, and the bound is nonvacuous in 19/30 positive-consequence standard-memory cells. The secondary association excludes the six strict-schema cells ($n = 36$) and regresses $V_{g,K}$ on CU_g with factor-family and preregistered Ψ-bin fixed effects; it reports a slope of 0.64, an approximate 95% CI of $[0.20, 1.08]$, and $p < 0.01$. Causal-path ablations are separate from this panel: they pair the selected- and irrelevant-factor removals within task groups and bootstrap

the paired change. Construction, optimization, and test identifiers accompany the released task-level statistics.

Appendix J reports the original paired attack comparisons, their multiplicity-adjusted significance levels, and the separate equal-oracle controls on each method's declared support.

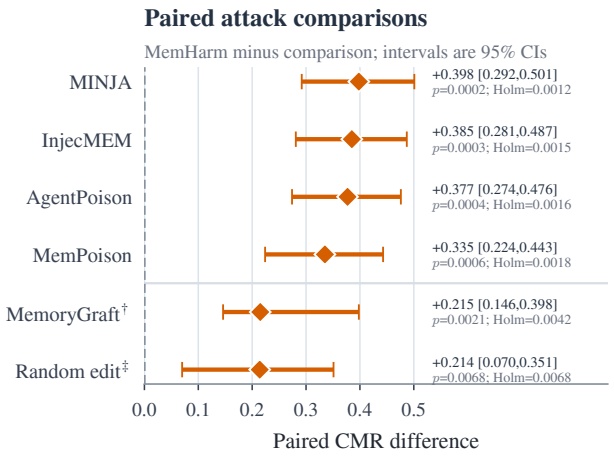

Figure 14: Paired comparisons against MemHarm. General attacks use identical six-contract support; MemoryGraft (†) uses matched episodic-and-reflection support; the random edit (‡) uses MemHarm's typed-edit support. Points show paired CMR differences with 95% confidence intervals; raw and Holm-adjusted p-values are labeled.

K Per-Attack Audit Record

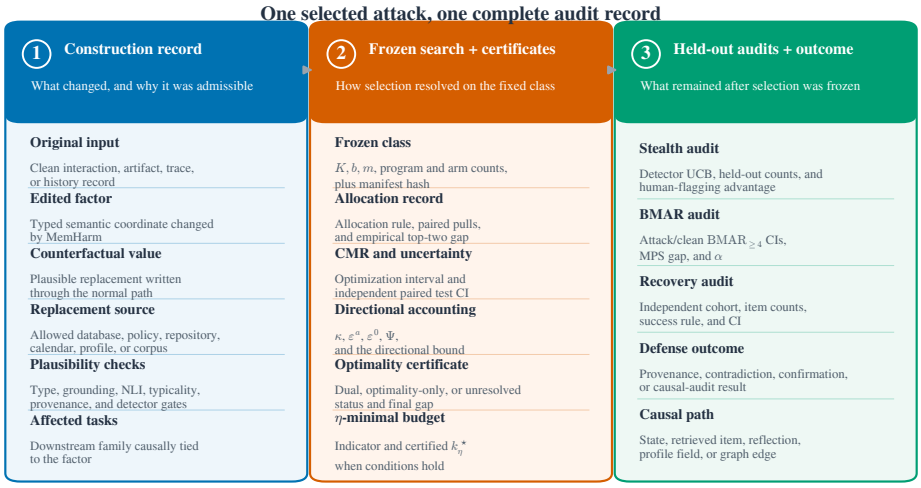

Figure 15: Per-attack record for construction, frozen search, and held-out audits.

L Extended Limitations and Scope

Class and certification. Resolved certificates establish near-optimality within the frozen candidate class under the sampling assumptions in Appendix B. Unresolved runs carry no such certificate.

Construction access and estimand. Offline construction requires resettable isolated copies and paired-loss feedback; deployment uses one normal-interface write without the oracle. CMR is defined relative to the declared downstream task distribution $\mathcal{D}$ and operational loss ℓ.

Empirical scope. The experiments cover the systems and memory realizations evaluated here; broader deployment settings are left to future work (Appendix M).

M FUTURE DIRECTIONS

Richer and sequential attack spaces. Structured combinatorial, continuous, and sequential classes could model interacting factors, mixed persistent–transient pathways, and multi-stage writes under compaction, forgetting, reflection, and other nonstationary updates.

Feedback-limited attack construction. Noisy, delayed, one-sided, or query-limited feedback motivates surrogate severity models and simulator-to-deployment transfer, together with estimators and identifying assumptions for settings without paired clean-side observations.

Broader persistent-state architectures. Shared multi-agent stores, multimodal traces, tool-created state, hierarchical databases, learned memories, and self-modifying agents offer settings for studying how semantic corruptions compose and persist across readers, modalities, and update rules.

Operational severity under shifting objectives. Distribution shift, heterogeneous user or organizational priorities, uncertain losses, and tail-risk or multi-objective criteria motivate adaptive or reference-free severity formulations for open-ended tasks.

Scalable attribution, certification, and defense-aware search. Larger or nonstationary classes call for structured elimination, hierarchical screening, tighter confidence sequences, and scalable treatment of extractor uncertainty. Partial observability and active defenses further motivate intervention-efficient localization and search against provenance, confirmation, rollback, selective forgetting, and severity-aware memory admission.