Cognitive Immunity for Trustworthy LLM Agents: Auditable Failure Memory Against Repeated Safety Errors

Mian Zhang
Independent Researcher
China

Abstract

Trustworthy LLM agents should not merely apologize after a failure; they should convert safety-relevant failures into reusable, auditable defenses. We introduce cognitive immunity, a runtime failure-memory layer that extracts a scoped "antigen" descriptor from an observed failure, stores a bounded "antibody" rule with provenance, decay, and review metadata, and retrieves matching rules before later responses. The layer is model-agnostic, does not retrain the base model, and exposes why an intervention occurred. In a longitudinal protocol with 20 task templates, five rounds, three seeds, and four strategies, the recovered score-level record contains 1,200 strategy-task-round-seed events. Cognitive immunity lowers pooled severe-threshold repeat failure rate from 0.764 for no memory to 0.650; a paired seed-task bootstrap gives a mean difference of -0.106 with a 95% interval $[-0.211, -0.003]$. Reflexion retains the highest aggregate improvement score, so the result is a recurrence-reduction tradeoff rather than an all-metric win. Cognitive immunity is not a certified safety proof or a replacement for red teaming, access control, sandboxing, or human oversight. It is a falsifiable runtime mechanism for reducing recurrence of known observed failure classes while keeping interventions inspectable.

Keywords

trustworthy LLMs, secure LLM agents, failure memory, AI safety, auditability, repeat failure, sycophancy, runtime monitoring

1 Introduction

LLM agents increasingly operate across repeated user interactions, tool calls, and multi-step tasks. A correction may improve one answer while leaving the same failure class available to recur later. Session-local apologies and revisions therefore do not establish that a system has learned a durable safety lesson. We call this recurrence a *repeat failure*: after a relevant failure has been observed and corrected, a structurally similar error remains below the task's success threshold in a later round.

Persistent memory is not automatically safer. An unconstrained memory store can retain poisoned instructions, private user data, stale policies, or overgeneralized refusals. Recent work treats long-term agent memory as a lifecycle security surface, not merely a retrieval component [2, 4]. Agent benchmarks likewise expose prompt-injection, harmful agency, and repeated tool-policy interaction risks that a failure-memory layer must not claim to solve [1, 3, 8].

We study **cognitive immunity**, a bounded runtime layer that converts verified failure events into scoped and auditable prevention rules. Its contribution is deliberately narrow: reduce recurrence of known observed failure classes without hiding interventions inside an opaque prompt or retraining the base model. The mechanism records the source failure, matching features, rule scope, strength, decay state, intervention reason, and downstream outcome.

Object	Required property
Failure event	Evidence-bound task, action, feedback, and outcome tuple.
Antigen descriptor	Normalized failure class, scope, source, and severity.
Antibody rule	Bounded intervention, never an unrestricted hidden instruction.
Write gate	Provenance and trust check before storage or reinforcement.
Decay/review handle	Expiry, deletion, rollback, and human-review metadata.
Audit record	Trigger, rule, intervention, and post-intervention outcome.

Table 1: Runtime objects required for an auditable failure-memory layer.

The paper makes three contributions. First, it specifies a runtime failure-memory loop with write gates, scoped retrieval, decay, and review handles. Second, it reports a four-strategy longitudinal comparison with explicit repeat-failure denominators and score-level uncertainty estimates. Third, it treats memory poisoning, false intervention, overblocking, staleness, and cold-start exposure as first-class claim boundaries rather than afterthoughts.

2 Cognitive Immunity Layer

Let an interaction produce context o_t, action a_t, verifier outcome y_t, and failure metadata c_t. A failure event is

$$f_t = (o_t, a_t, y_t, c_t). \tag{1}$$

An extractor maps an eligible event into an antigen descriptor α_t and a bounded antibody rule b_t:

$$(\alpha_t, b_t) = B(f_t). \tag{2}$$

The descriptor identifies the failure class, evidence source, scope, severity, and matching features. The rule encodes a limited intervention such as requesting missing evidence, resisting unsupported user pressure, checking a contradiction, or routing an action for review.

A write gate admits the rule only when the failure evidence and provenance satisfy policy. At inference time, retrieval selects rules whose descriptor distance and strength pass explicit thresholds:

$$\mathcal{B}_q = \{b_i : d(\alpha_i, \alpha(q)) \le \rho,\ s_i \ge \tau,\ q \in \mathrm{scope}(b_i)\}. \tag{3}$$

Strength decays and can be reinforced only by verified recurrence:

$$s_i(t) = \exp[-\lambda(t - t_i)]\big(1 + \kappa \log(1 + r_i)\big). \tag{4}$$

Every applied rule produces an audit record containing descriptor id, rule id, source failure, match score, trigger reason, intervention, and outcome. Rules can expire, be deleted, or be routed for review. They do not override higher-priority system policy.

Operational loop. For a query q_t, the layer first retrieves candidate rules that pass the match threshold, then filters them by scope, expiry, failure class, and provenance. If eligible rules remain, it constructs a bounded intervention from the top matches and calls the unchanged base model. A task verifier or scoring rubric then produces an outcome. An observed safety-relevant failure may generate a new descriptor, but a rule is written only after the evidence passes the write gate. Otherwise, matched rules decay or are reinforced only by verified recurrence. The audit log records retrieved rule ids, the intervention, outcome, write/update/delete decision, decay state, and reason codes. This loop separates memory from the base model and makes each memory-mediated action reversible and reviewable.

Illustrative audit trace. Consider a later request that applies unsupported social pressure to repeat a previously observed sycophancy failure. The descriptor marks the failure class, evidence source, scope, and severity. The write gate rejects direct user-authored policy text and admits only a bounded rule such as "request supporting evidence and preserve documented contradictions." A later match can trigger that rule without overriding system policy. The resulting log links the source failure, match score, intervention, response outcome, and any review or deletion action. This trace illustrates the required semantics; it is not an additional empirical trial.

3 Evaluation Protocol

The longitudinal panel contains 20 task templates spanning hallucination, sycophancy, reasoning, and safety. Each task is evaluated over five rounds and three seeds under four strategies: No Memory, Self-Refine [5], Reflexion [6], and Cognitive Immunity. This gives 60 seed-task trajectories and 300 score events per strategy, or 1,200 score events across the complete four-strategy comparison. These are not 1,200 independent tasks or a raw-response transcript.

Scores use a 0–3 rubric. $R1$ and $R5$ are mean scores in the first and fifth rounds. For N matched seed-task trajectories, normalized improvement is

$$\mathrm{WQ} = \frac{1}{N}\sum_i \begin{cases} (s_{i,5} - s_{i,1})/(3 - s_{i,1}), & s_{i,1} < 3, \\ 0, & s_{i,1} = 3. \end{cases} \tag{5}$$

WQ remains a secondary quality signal. Repeat failure rate (RFR) conditions on a prior-failure opportunity:

$$\mathrm{RFR} = \frac{\sum_i \sum_{r=2}^{5} \mathbb{1}[s_{i,r-1} < 2]\mathbb{1}[s_{i,r} < 2]}{\sum_i \sum_{r=2}^{5} \mathbb{1}[s_{i,r-1} < 2]}. \tag{6}$$

We denote this pooled severe-threshold metric by $\mathrm{RFR}^{\mathrm{pooled}}_{\mathrm{severe}<2}$ and abbreviate it as RFR below. It measures recurrence conditional on a previous failure, rather than average task difficulty, and is not directly comparable to differently thresholded or trajectory-aggregated recurrence metrics.

The recovered score arrays cover seeds 42, 137, and 256 and reproduce the point estimates in Table 2. Wilson intervals use pooled prior-failure opportunities. Bootstrap intervals resample the 60 matched seed-task trajectories with replacement for 10,000 replicates using bootstrap seed 20260616. The paired delta compares each strategy with No Memory on the same seed-task unit.

Strategy	R1	R5	WQ ↑	RFR ↓	RFR 95% bootstrap
No Memory	1.78	1.72	0.067	0.764	[0.669, 0.843]
Self-Refine	1.73	1.62	0.100	0.803	[0.716, 0.870]
Reflexion	1.75	2.03	**0.217**	0.702	[0.602, 0.786]
Cognitive Immunity	**1.80**	**2.08**	0.158	**0.650**	[0.545, 0.742]

Table 2: Recovered score-level longitudinal results. Lower RFR is better; Reflexion has the highest WQ.

Recovered field	Value and evidence boundary
Score unit	One value per task, round, seed, and strategy.
Seeds	42, 137, and 256; matched to the recovered arrays.
Runner/judge label	`deepseek-v4-flash` in recovered scripts; no authoritative provider snapshot or run date.
Decoding	Temperature 0.0; top-p unspecified; token caps 1024 agent, 256 judge, and 200 feedback.
Rubric	Text recovered and hashed; calibration absent.
Artifact boundary	Score arrays and recomputation logic recovered; raw responses, judge rationales, and second-judge receipts absent.

Table 3: Recovered execution metadata. Script-level values do not substitute for model snapshots, score-date receipts, or judge calibration.

The recovered rubric copy has SHA-256 `DF2EA6A24E20B163 0BA72B87B0E4FFAD8088917A8C3D4555ACA166E9ED045F7E`. This identifies the rubric text; it does not establish calibration or independent adjudication.

4 Results and Interpretation

Cognitive Immunity has the lowest pooled RFR in the current record: $78/120 = 0.650$, compared with $107/140 = 0.764$ for No Memory. The paired seed-task RFR difference is -0.106 with a 95% bootstrap interval $[-0.211, -0.003]$. Its final-round score is also higher in this panel. The result supports recurrence reduction for known observed failure classes under the documented protocol.

The result is not an all-metric dominance claim. Reflexion has the highest WQ point estimate, and different methods trade immediate improvement against recurrence. Self-Refine does not improve RFR in this record. A trustworthy memory layer should therefore report both recurrence and utility, including cases where an intervention blocks correct behavior or introduces a new error.

The evidence package is score-level. It supports recomputation of R1, R5, WQ, RFR, and the local intervals from recovered arrays. Complete raw model responses, judge rationales, score-date receipts, and an independently calibrated second judge are not present. We therefore do not describe the result as raw-response replication, calibrated safety evidence, or a public benchmark certification.

5 Threat Model and Controls

Cognitive immunity targets known observed failure classes. It does not defend against arbitrary prompt injection, data exfiltration, malicious tool access, compromised infrastructure, or failures that have not yet produced a usable descriptor. The layer adds its own attack and reliability surface.

Memory should store abstract failure descriptors rather than unnecessary user content. Sensitive context requires minimization,

Risk	Failure mode	Required control
Memory poisoning	Untrusted evidence writes a malicious rule.	Provenance-aware write gate; review/delete path.
False intervention	A rule matches a non-equivalent case.	Scope and threshold checks; reason code.
Overblocking	Safety memory suppresses valid behavior.	Utility/tradeoff logging and bounded actions.
Staleness	A formerly valid rule persists after change.	Decay, expiry, versioning, rollback.
Cold start	No antibody exists before first failure.	Report first exposure outside recurrence suppression.
Audit gaps	Intervention cannot be traced to evidence.	Immutable ids, source links, outcomes, review handles.

Table 4: Failure-memory risks and minimum controls.

Reason code	Auditable meaning
KNOWN_FAILURE_MATCH	Current context matches a scoped observed failure class.
BOUNDED_INTERVENTION_APPLIED	A rule changed context within its declared scope.
WRITE_GATE_APPROVED	Verified evidence passed the provenance-aware write gate.
WRITE_GATE_REJECTED	Noisy, unsafe, or unverified evidence was not stored.
STALE_ANTIBODY_DECAYED	A rule weakened, expired, or was deleted after review.
FALSE_INTERVENTION_REVIEW	Suspected overblocking or autoimmunity was routed for review.
POISONING_GUARD_TRIGGERED	Untrusted input attempted to alter memory policy.

Table 5: Minimum reason codes for reviewing memory-mediated behavior. Names are protocol labels, not claims that every control has been empirically validated.

retention limits, deletion support, and access control. Retrieval must remain subordinate to system policy and domain-specific safety cases.

6 Auditability and Reproducibility

Auditability requires more than storing a natural-language reminder. Every intervention should be attributable to a verified source failure and should expose the decision that changed the model context. The minimum record is

$$\ell_t = (q_t, \mathcal{B}_q, u_t, y_t, o_t, w_t, v_t), \tag{7}$$

where $\mathcal{B}_q$ is the retrieved rule set, u_t is the bounded intervention, o_t is the evaluated outcome, w_t is the write/update/delete decision, and v_t contains rule and policy versions. The log should preserve identifiers or hashes rather than unnecessary private content.

Recomputation path. The recovered local artifact contains the three seed-level score arrays, a rubric-text copy, point-estimate recomputation code, manifest and hash records, and provenance sidecars. Its included script reproduces R1, R5, WQ, and RFR point estimates. A separate local statistical audit over the same arrays produced the Wilson and bootstrap intervals and the paired delta; that interval implementation is not part of the current artifact. This supports bounded table-level checking, not reconstruction of the original generations or judging dialogue. The absence of raw responses, judge rationales, authoritative model snapshots, score dates, calibration, and a second judge remains explicit in both the manuscript and artifact boundary.

Lifecycle controls. Rule review must cover creation, retrieval, reinforcement, decay, expiry, deletion, and rollback. A deployment should minimize retained user content, separate descriptor storage from policy authority, version write and retrieval gates, and test false interventions as well as prevented recurrence. These controls constrain the mechanism; they do not turn it into a complete long-term-memory security system.

7 Related Work

Self-Refine revises an answer through self-feedback, while Reflexion stores verbal feedback across trajectories [5, 6]. Cognitive Immunity focuses on safety-relevant recurrence and makes write, retrieval, decay, and intervention provenance explicit. ReAct and agent benchmarks establish broader reasoning-and-action settings [7]. AgentDojo, AgentHarm, and τ-bench expose prompt-injection, harmful agency, and repeated tool-policy interaction risks [1, 3, 8]; they motivate boundaries but are not claimed as evaluated P01 environments. Long-term-memory security and memory-poisoning work further motivates lifecycle controls around persistent memory [2, 4].

8 Limitations and Conclusion

The panel uses language-agent tasks and a fixed scoring pipeline, not physical deployment or regulated safety assessment. The current record lacks complete raw-response and judge-rationale receipts, independent human adjudication, and broad cross-model replication. The task set is small enough for workshop-scale analysis but not universal certification. The mechanism can also create autoimmunity through false or stale interventions.

Within those boundaries, the result is specific and falsifiable: a scoped, decaying, and auditable failure-memory layer reduces recurrence of known observed failure classes relative to no memory in the recovered longitudinal score record. Future work should add independent reproduction, multiple judges, poisoning stress tests, retention/deletion receipts, and deployment-specific false-intervention measurements.

References

[1] M. Andriushchenko et al. 2024. AgentHarm: A benchmark for measuring harmfulness of LLM agents. arXiv:2410.09024.

[2] P. Dash, T. Ge, A. Jain, T. Shah, and Z. Shang. 2026. From untrusted input to trusted memory: A systematic study of memory poisoning attacks in LLM agents. arXiv:2606.04329.

[3] E. Debenedetti, J. Zhang, M. Balunović, L. Beurer-Kellner, M. Fischer, and F. Tramèr. 2024. AgentDojo: A dynamic environment to evaluate prompt injection attacks and defenses for LLM agents. NeurIPS Datasets and Benchmarks.

[4] Z. Lin et al. 2026. A survey on long-term memory security in LLM agents: Attacks, defenses, and governance across the memory lifecycle. arXiv:2604.16548.

[5] A. Madaan et al. 2023. Self-Refine: Iterative refinement with self-feedback. NeurIPS.

[6] N. Shinn, F. Cassano, A. Gopinath, K. Narasimhan, and S. Yao. 2023. Reflexion: Language agents with verbal reinforcement learning. NeurIPS.

[7] S. Yao et al. 2023. ReAct: Synergizing reasoning and acting in language models. ICLR.

[8] S. Yao, N. Shinn, P. Razavi, and K. Narasimhan. 2024. τ-bench: A benchmark for tool-agent-user interaction in real-world domains. arXiv:2406.12045.