When Should Agents Remember? Falsification-Gated Self-Evolution for LLM Agents

Runxuan Tang[1,*] **Haoyu Gao**[2,*] **Yuyan Ding**[3] **Junyi Yao**[4]
Zihao Zheng[4] **Zecheng Sheng**[5] **Liwei Hou**[6,†]
[1]Nanyang Technological University [2]Georgia Institute of Technology
[3]University of Washington [4]Washington University in St. Louis
[5]The University of Sydney [6]National University of Singapore
TANG0609@e.ntu.edu.sg gao.howard517@gmail.com
yuyand2@uw.edu j.yao@wustl.edu z.zihaogary@wustl.edu
zshe0457@uni.sydney.edu.au liweihou@u.nus.edu
[*]Equal contribution. [†]Corresponding author.

Abstract

LLM agents increasingly operate in long-horizon, verifier-rich environments, but current self-improvement mechanisms often write episode-derived lessons directly into memory, making local success an unreliable signal for reusable and safe knowledge. This paper addresses the core question of when an experience-derived behavioral update should be admitted into persistent agent state. We propose **Falsification-Gated Self-Evolution** (FGSE), a verifier-grounded framework that represents each candidate lesson as a structured hypothesis with an explicit precondition, behavior change, expected effect, and verifier. FGSE applies the hypothesis only in a temporary state, tests it on target-transfer, falsification, and archived-regression probes, and then commits, refines, or rejects it according to measured gain and risk. Across web, app, tool-use, and long-term memory benchmarks, FGSE achieves strong task performance, including 39.8 WebArena success rate, 78.8 AppWorld average completion, and 0.724/0.487 Pass^1 on τ-Retail/τ-Airline, while reducing falsification failures to 5.0%, regression damage to 1.3%, and harmful committed updates to 3.6%. These results suggest that testing memory updates as hypotheses supports more reliable long-term adaptation.

1 Introduction

Large language model (LLM) agents are increasingly expected to operate in long-horizon, interactive environments where they must reason, call tools, update internal context, and recover from mistakes [Yao et al., 2023]. This capability supports web navigation, app-level tool use, and user-facing assistance [Zhou et al., 2024, Trivedi et al., 2024, Yao et al., 2024]; embodied interaction additionally requires grounding language in visual and gestural context [Li et al., 2023]. Continual deployment requires agents to solve current tasks and safely learn reusable behavioral knowledge from experience.

Recent work has made substantial progress toward self-improving agents through self-feedback and reflection [Madaan et al., 2023], verbal reinforcement from past failures [Shinn et al., 2023], reusable skill memory [Wang et al., 2023, Li et al., 2026b], agentic long-term memory organization [Xu et al., 2025], and evolving symbolic contexts or playbooks [Zhou et al., 2025, Zhang et al., 2026f]. These methods show that storing lessons, memories, playbooks, or revised contexts can improve future performance without necessarily updating model weights. An episode-derived update may improve tasks that resemble the triggering episode while overgeneralizing to superficially similar but semantically different situations, conflicting with previously successful behaviors, or accumulating

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

as stale persistent memory. Even in verifier-rich environments, where task success can often be checked after execution, existing self-evolution mechanisms lack a principled criterion for assessing an update's transferability, retention value, and commitment risk. The central question is **how to govern the transition from temporary experience to persistent agent state** under transfer, safety, and regression constraints.

We study when an experience-derived behavioral update should enter an LLM agent's long-term memory. Our approach treats persistent updates as testable hypotheses. As shown in Fig. 1, our approach temporarily applies each candidate update, evaluates its transfer beyond the source episode, robustness to targeted counterexamples, and preservation of archived successful behavior, and then commits, refines, or rejects it.

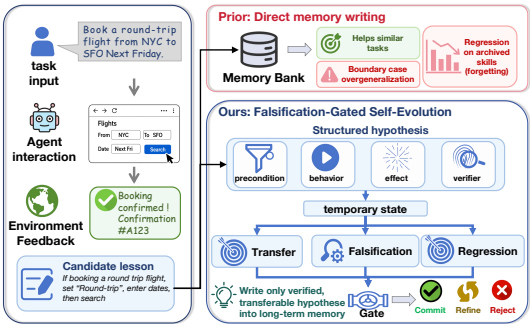


Figure 1: A locally useful episode-derived lesson may overgeneralize or regress archived skills if directly written into memory. FGSE tests each lesson as a structured hypothesis in a temporary state and commits only verified, transferable updates to long-term memory.

We make the following contributions: (**i**) We identify a core failure mode of self-evolving agents: target improvement alone is an unreliable signal for persistent memory writing, because many apparently useful updates encode unsafe shortcuts or overly broad applicability conditions. (**ii**) We introduce **Falsification-Gated Self-Evolution** (FGSE), a verifier-grounded framework that treats candidate updates as testable hypotheses and performs structured commit–refine–reject triage before long-term retention. (**iii**) Across web, app, tool-use, and long-term memory settings, FGSE improves task performance while reducing falsification failures, regression damage, harmful committed updates, and post-commit violations compared with reflection-, memory-, and context-evolution baselines.

2 Related Work

Interactive language agents and verifier-rich environments. LLM agents combine reasoning with external actions, from ReAct [Yao et al., 2023] to tool use [Qin et al., 2024, Chen and Liu, 2026, Xiao et al., 2026b], while LLM-derived traffic context also augments motion prediction [Zheng et al., 2024]. This progress has enabled realistic web, app, and tool-use benchmarks [Zhou et al., 2024, Trivedi et al., 2024, Yao et al., 2024, Debenedetti et al., 2024], where success can often be checked by executable validators, database states, user goals, or safety constraints. Healthcare-agent evaluation protocols additionally assess episode-level state continuity, evidence traceability, and escalation decisions [Yao et al., 2026]. Complementary work verifies and repairs synthetic web environments before agent training [Zhang et al., 2026a]. A YANG-grounded LLM agent supports multi-vendor OpenROADM management through schema retrieval and validation of generated configurations [Xiao et al., 2026c]. Complementary network-diagnosis work uses multi-head graph attention to jointly address alarm correlation and root alarm detection in optical transport networks [Xiao et al., 2025]. Medical-agent security combines risk perception, evidence constraints, and knowledge-consistency verification [Hu, 2026]. Broader security research studies quantitative vulnerability assessment [Shao, 2026b], SHAP-based explanations for network anomaly detection [Shao, 2026a], APT attribution over heterogeneous threat knowledge graphs [Wang et al., 2026], and provenance-graph anomaly detection [Ma et al., 2026]. FGSE builds on verifier-rich settings and applies verification to candidate behavioral updates before persistent commitment, testing transfer, falsification, and regression safety.

Memory and context-based self-improvement. Another line improves agents without updating model weights. Self-Refine [Madaan et al., 2023] and Reflexion [Shinn et al., 2023] revise future behavior through self-feedback or verbal reinforcement, while Voyager [Wang et al., 2023] stores reusable executable skills. Persistent-memory systems such as Generative Agents [Park et al., 2023], MemoryBank [Zhong et al., 2024], MemGPT [Packer et al., 2023], and A-Mem [Xu et al., 2025] organize memories for retrieval, reflection, and personalization, with LoCoMo [Maharana

et al., 2024] and LongMemEval [Wu et al., 2024] evaluating long-horizon memory utility. Recent context-evolution methods, including Symbolic Learning [Zhou et al., 2025], Lighthouse [Yang et al., 2025], and ACE [Zhang et al., 2026f], further revise prompts, playbooks, critiques, or contexts from execution feedback. These methods show that textual updates can improve agents, but they may overgeneralize local lessons, accumulate stale or conflicting memories, or degrade useful context during iterative rewriting, echoing concerns about unreliable self-correction [Huang et al., 2024, Liu et al., 2026b] and unsafe misevolution [Shao et al., 2026]. Adjacent retrieval studies distinguish semantic relevance from factual consistency in RAG [Ye et al., 2026] and show benefits of exemplar diversity under matched context budgets for multi-turn intent understanding [Lin et al., 2026]. FGSE treats each lesson as a structured, verifier-grounded hypothesis and commits it only after falsification and regression testing.

Capability-level contribution evidence. Xu et al. [2026c] propose an observable evidence standard for added planning capabilities, separating presence, opportunity to act, executed-plan change, and incremental operational consequence under controlled replay, with explicit treatment of equivalence, uncertainty, and operating scope. Their analysis requires capability-specific contribution evidence for acceptance, beyond aggregate policy gains or proxy diagnostics. For memory updates, contribution likewise depends on verified behavioral effects beyond storage or retrieval. FGSE evaluates each update against the current agent in a temporary state and requires target-transfer gain, falsification resistance, and regression preservation before commitment (Eq. (8)); subsequent activation requires a matching precondition and a violation record below threshold (Section 4.3). These conditions link contribution evidence to verification and bounded activation of memory updates.

Automated agent optimization and self-evolution. Beyond memory writing, recent work optimizes agent systems by searching over prompts, graph structures, workflows, or multi-agent state machines [Yang et al., 2024, Zhuge et al., 2024, Zhang et al., 2025a, Hu et al., 2025, Zhang et al., 2025b], and improves reasoning policies through test-time reinforcement learning, self-generated curricula, or tool-integrated co-evolution [Zuo et al., 2025, Tian et al., 2025, Huang et al., 2026a, Xia et al., 2025]. In Text-to-SQL, an autonomy-based taxonomy highlights execution-guided self-correction and modular agent collaboration [Su et al., 2027]. Continual adaptation also studies how to preserve knowledge under bounded resources: CoMem [Zhou et al., 2026] uses concept-graph memory and feature-space rehearsal for vision–language adaptation; InduceKV [Chen et al., 2026] externalizes multimodal adaptation into budgeted key–value memories; and Meta-UCF [Xiao et al., 2026d] generates task-conditioned LoRA updates with a shared hypernetwork. Compo-ReAlign [Xiao et al., 2026a] preserves compositional structure through reversible alignment and sensitivity-constrained updates, while Affordance-First Decomposition [Xu et al., 2026b] separates a stable affordance substrate from query-routed, conflict-aware adaptation in video–language learning. These lines address adaptation capacity, retention, and resource efficiency, alongside cross-domain studies of recoverable model pruning [Lu et al., 2026], multi-scale and attention-based visual processing [Zhang et al., 2023, 2024, 2026d], activation design and LLM-guided fusion for forecasting [Zhang et al., 2026b,c], and operational decision making through semantic evidence for inspection allocation [He et al., 2026], causal intervention ranking [Huang et al., 2026b], and accuracy–stability trade-offs in forecasting [Li et al., 2026a]. FGSE addresses a complementary update-level problem: it falsifies each episode-derived update before persistent commitment, preserving self-evolution plasticity while controlling unsafe transfer and regression.

3 Preliminary Study

A common self-improvement strategy is to summarize completed episodes into reflective text or experiential insights and store them in memory for future decision making [Shinn et al., 2023]. While such updates can help on similar tasks, they may overgeneralize, disrupt previously solved behaviors, or encode brittle shortcuts. We characterize these failure modes in a diagnostic study.

Diagnostic setup. For each completed episode, the base agent generates a free-form update u, such as a rule, reminder, or behavioral correction, which we inject into the memory prompt and compare against the original agent. For analysis, we evaluate u on four case groups: source replay, near transfer, boundary contrast, and archived success, corresponding to the original episode neighborhood, intended-pattern transfer, superficially similar non-trigger cases, and previously solved tasks. For

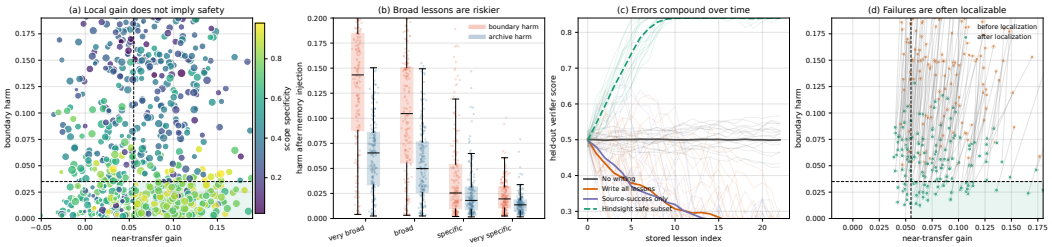


Figure 2: **Exploratory diagnostics for naive episode-derived self-updates. (a)** Many updates combine near-transfer gains with boundary harm. **(b)** Broad and underspecified updates produce substantially higher boundary and archive harm than more specific updates. **(c)** Sequentially writing free-form lessons into memory can accumulate errors, especially when updates are selected only by source-episode improvement. **(d)** A post-hoc scope-localization analysis shows that many harmful but useful updates can be repaired by narrowing their applicability, motivating an explicit refinement mechanism.

each group $\mathcal{C}$, we measure the average verifier-score change induced by u:

$$\Delta_{\mathcal{C}}(u) = \frac{1}{|\mathcal{C}|} \sum_{x \in \mathcal{C}} \left[J(\Theta \oplus u, x) - J(\Theta, x) \right], \tag{1}$$

where Θ is the original agent state and $\Theta \oplus u$ denotes the same agent with the free-form lesson u appended to memory. We report near-transfer gain $\Delta_{\text{near}}(u)$, boundary harm $[-\Delta_{\text{boundary}}(u)]_+$, and archive harm $[-\Delta_{\text{archive}}(u)]_+$. We also annotate each lesson with a coarse scope-specificity score in $[0, 1]$, where lower values indicate broader and less conditional updates.

Findings. Figure 2 shows that naive memory writing is useful but unreliable. Free-form lessons often improve near-transfer cases but can cause boundary harm, indicating that locally valid behaviors may be applied in inappropriate contexts. This failure is closely tied to scope: broad, underspecified lessons produce larger boundary and archive degradation than more specific ones, and sequentially writing them into memory accumulates errors despite short-term source gains. Post-hoc localization further shows that some harmful updates contain valid rules with overly broad applicability; narrowing their scope can retain transfer benefits while reducing harm. These findings motivate FGSE, which treats episode-derived lessons as structured, falsifiable hypotheses, tests them temporarily on target, boundary, and archived cases, and commits or refines them only when supported by evidence.

4 Method

4.1 Overview

We propose **Falsification-Gated Self-Evolution** (FGSE), a verifier-grounded framework for reliable long-term improvement of language agents. Given an episode, FGSE converts the observed success or failure into a structured behavioral hypothesis that specifies its precondition, behavior change, expected effect, and verifier. The hypothesis is applied only temporarily and evaluated on target-transfer probes and falsification probes. It is committed to persistent agent state only when it improves target cases, survives targeted falsification, and preserves previously solved behavior. Hypotheses with localized failures are refined once through precondition narrowing and reevaluated; hypotheses that still fail are rejected. Figure 3 summarizes the pipeline, with probe construction and refinement details provided in Appendix A.2 and Appendix A.3.

4.2 Falsification-Gated Self-Evolution

Verifier-grounded agent state. At episode t, the agent receives a task x_t and acts under persistent state

$$\Theta_t = (\mathcal{M}_t, \pi_t), \qquad J_v(\Theta, x) = \mathbb{E}_{\tau \sim p_\Theta(\cdot|x)} \left[v(x, \tau) \right]. \tag{2}$$

Here, $\mathcal{M}_t$ is the retrievable memory of committed evolution units, π_t is the current agent policy, $p_\Theta(\tau \mid x)$ is the trajectory distribution induced by state Θ on task x, and $v(x, \tau) \in [0, 1]$ is a verifier-

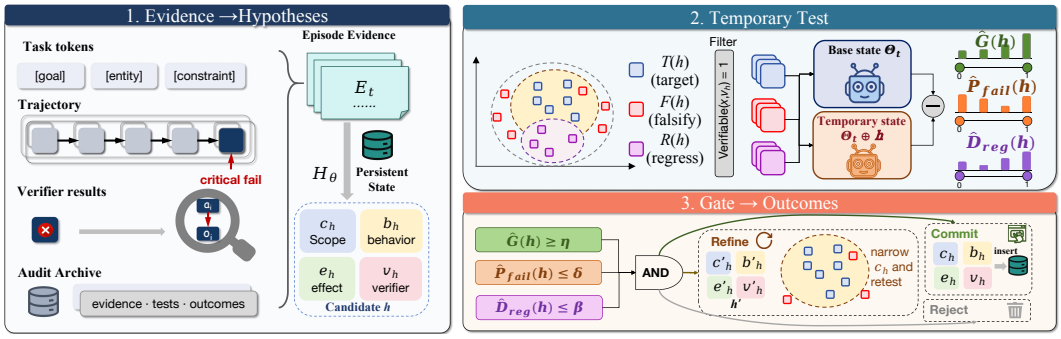


Figure 3: **Overview of Falsification-Gated Self-Evolution (FGSE).** FGSE first converts episode evidence into a structured behavioral hypothesis, temporarily evaluates it with target-transfer, falsification, and regression probes, and then commits, refines, or rejects the hypothesis according to verifier-grounded gain and risk criteria.

grounded score. The expectation accounts for stochastic decoding, tool uncertainty, and multi-step interaction. A completed episode yields a trajectory τ_t and verifier result $r_t = v_t(x_t, \tau_t)$, from which FGSE forms an evidence packet $E_t = (x_t, \tau_t, r_t, \mathcal{A}_t)$ using the audit archive $\mathcal{A}_t$.

Structured evolution hypothesis. Each candidate update is represented as

$$h = (c_h, b_h, e_h, v_h). \tag{3}$$

The precondition $c_h : \mathcal{Z} \to [0,1]$ assigns an applicability score to an interaction state $z = (x, \tau_{<\ell})$. The behavior transformation b_h maps the current policy to a modified policy $b_h[\pi_\Theta]$. The expected-effect checker e_h specifies the verifier-observable outcome that should hold after applying the update. The verifier v_h provides the task-level score used to evaluate improvement, failure, and regression. The proposer H_ϕ maps the evidence packet to a schema-constrained hypothesis, $h_t = H_\phi(E_t)$. The proposer generates candidates; the commitment gate determines their retention.

The hypothesis is activated only when its precondition is satisfied:

$$g_h(z) = \mathbb{I}[c_h(z) \geq \gamma_h], \qquad \pi_{\Theta_t \oplus h}(a \mid z) = (1 - \lambda_h g_h(z))\pi_{\Theta_t}(a \mid z) + \lambda_h g_h(z) b_h[\pi_{\Theta_t}](a \mid z). \tag{4}$$

Here, γ_h is the activation threshold, $\lambda_h \in [0,1]$ is the intervention strength, and $\Theta_t \oplus h$ denotes temporary application of h without changing persistent memory. The temporary policy remains a valid action distribution because Eq. (4) is a convex combination of two valid policies.

Verifier-grounded probes. For a candidate hypothesis h, FGSE constructs a target-transfer set $\mathcal{T}(h)$, a falsification set $\mathcal{F}(h)$, and a regression set $\mathcal{R}(h)$:

$$\mathcal{P}(h) = \mathcal{T}(h) \cup \mathcal{F}(h) \cup \mathcal{R}(h). \tag{5}$$

The target-transfer set contains cases where the hypothesis is expected to help. The falsification set contains cases designed to expose overgeneralization. The regression set $\mathcal{R}(h)$ contains archived cases that the current agent already solves and that must not be degraded. A probe x is admitted only when $\mathrm{Verifiable}(x, v_h) = 1$, ensuring that the gate is based on executable or externally checkable signals. FGSE builds falsification probes from boundary cases, precondition violations, regression cases, and decoys; their construction is described in Appendix A.2.

Temporary evaluation. FGSE evaluates h by comparing the original state Θ_t with the temporary state $\Theta_t \oplus h$ on the same probes:

$$\Delta_h(x) = J_{v_h}(\Theta_t \oplus h, x) - J_{v_h}(\Theta_t, x). \tag{6}$$

A positive $\Delta_h(x)$ means that the hypothesis improves verifier-grounded performance on x, while a negative value indicates harm. Empirical estimation under repeated rollouts is given in Appendix A.4.

Let $E_h(\Theta, x) \in \{0, 1\}$ denote whether the expected effect e_h is satisfied under state Θ on task x. FGSE computes

$$\begin{aligned}
\widehat{G}(h) &= \frac{1}{|\mathcal{T}(h)|} \sum_{x \in \mathcal{T}(h)} \Delta_h(x), \\
\widehat{P}_{\text{fail}}(h) &= \frac{1}{|\mathcal{F}(h)|} \sum_{x \in \mathcal{F}(h)} \mathbb{I}[\Delta_h(x) < -\epsilon \vee E_h(\Theta_t \oplus h, x) = 0], \\
\widehat{D}_{\text{reg}}(h) &= \frac{1}{|\mathcal{R}(h)|} \sum_{x \in \mathcal{R}(h)} [-\Delta_h(x)]_+.
\end{aligned} \tag{7}$$

Here, $\epsilon \geq 0$ is the tolerated verifier-score drop and $[\cdot]_+ = \max(\cdot, 0)$. The three statistics measure useful transfer, targeted falsification risk, and regression damage.

Commitment gate. The commitment decision is

$$\mathsf{Commit}(h) = \mathbb{I}\left[\widehat{G}(h) \geq \eta \wedge \widehat{P}_{\text{fail}}(h) \leq \delta \wedge \widehat{D}_{\text{reg}}(h) \leq \beta\right], \tag{8}$$

where $\eta > 0$ is the minimum target-transfer gain, $\delta \in [0, 1]$ is the maximum falsification rate, and $\beta \geq 0$ is the maximum regression damage. If $\widehat{G}(h) \geq \eta$ but failures are concentrated outside a separable sub-scope, FGSE performs one precondition-refinement pass and reevaluates the refined hypothesis. The refinement objective and separability criterion are given in Appendix A.3. A hypothesis that fails after refinement is rejected.

The persistent state is updated by

$$\Theta_{t+1} = \begin{cases} \Theta_t \oplus h, & \mathsf{Commit}(h) = 1, \\ \Theta_t, & \mathsf{Commit}(h) = 0. \end{cases} \tag{9}$$

A committed hypothesis is stored together with its precondition, behavior change, verifier, evaluated probes, verifier outcomes, and post-commit violation record. Rejected hypotheses are excluded from retrievable memory.

4.3 Runtime Use and Monitoring

During future inference, FGSE retrieves committed hypotheses whose preconditions match the current interaction state and whose violation records remain below threshold. Activated hypotheses modify the policy through the same behavior-transformation interface as Eq. (4). When multiple hypotheses are active, FGSE combines them by normalized precondition scores while preserving a nonzero base-policy contribution. Post-commit violations update a running violation score; hypotheses exceeding the violation threshold are removed from retrieval. The retrieval, combination, and monitoring rules are provided in Appendix A.5.

5 Theoretical Analysis

We analyze whether a hypothesis that passes the finite-probe commitment gate is reliable on the underlying target, falsification, and regression probe distributions. Fix a round t and condition on the pre-evaluation history, so the current state Θ_t, evidence packet, and finite candidate set $\mathcal{H}_t$ are fixed. If scope refinement is used, refinement is performed on a development split, while final commitment uses fresh probes.

For each $h \in \mathcal{H}_t$, let $\mathcal{D}_T(h)$, $\mathcal{D}_F(h)$, and $\mathcal{D}_R(h)$ denote the target-transfer, falsification, and regression probe distributions. We define

$$\begin{aligned}
G(h) &= \mathbb{E}_{x \sim \mathcal{D}_T(h)} [\Delta_h(x)], \\
P_{\text{fail}}(h) &= \mathbb{P}_{x \sim \mathcal{D}_F(h)} [\Delta_h(x) < -\epsilon \ \vee \ E_h(\Theta_t \oplus h, x) = 0], \\
D_{\text{reg}}(h) &= \mathbb{E}_{x \sim \mathcal{D}_R(h)} [[-\Delta_h(x)]_+].
\end{aligned} \tag{10}$$

These quantities measure useful transfer, falsification risk, and regression damage, respectively. Let $\text{LCB}_T(h)$, $\text{UCB}_F(h)$, and $\text{UCB}_R(h)$ be one-sided confidence bounds for these three quantities, computed from fresh final probes using empirical-Bernstein bounds for real-valued gains and Bernoulli-KL bounds for falsification events. Their finite-sample forms are given in Appendix B.2.

Algorithm 1 Falsification-Gated Self-Evolution

Require: Agent state Θ_t, task x_t, trajectory τ_t, verifier result r_t, audit archive $\mathcal{A}_t$
Ensure: Updated state Θ_{t+1}
Form evidence packet $E_t = (x_t, \tau_t, r_t, \mathcal{A}_t)$.
Propose structured hypothesis $h = H_\phi(E_t)$ with fields (c_h, b_h, e_h, v_h).
Construct $\mathcal{T}(h)$, $\mathcal{F}(h)$, and $\mathcal{R}(h)$ under verifier constraint $\mathrm{Verifiable}(x, v_h) = 1$.
Temporarily apply h and evaluate $\Delta_h(x)$ for all $x \in \mathcal{T}(h) \cup \mathcal{F}(h)$.
Compute $\widehat{G}(h)$, $\widehat{P}_{\mathrm{fail}}(h)$, and $\widehat{D}_{\mathrm{reg}}(h)$ using Eq. (7).
if $\mathsf{Commit}(h) = 1$ **then**
    Set $\Theta_{t+1} = \Theta_t \oplus h$ and record the evaluated evidence.
else if $\widehat{G}(h) \geq \eta$ and the failures satisfy the refinement criterion in Appendix A.3 **then**
    Refine the precondition, reevaluate the refined hypothesis, and apply Eq. (9).
else
    Reject h and set $\Theta_{t+1} = \Theta_t$.
end if

Theorem 1 (Generalization of the confidence-gated commitment rule)**.** *Suppose that, conditional on the pre-evaluation history, the final probes for each $h \in \mathcal{H}_t$ are sampled from $\mathcal{D}_T(h)$, $\mathcal{D}_F(h)$, and $\mathcal{D}_R(h)$. Then, with probability at least $1 - \alpha$, all $h \in \mathcal{H}_t$ simultaneously satisfy*

$$G(h) \geq \mathrm{LCB}_T(h), \qquad P_{\mathrm{fail}}(h) \leq \mathrm{UCB}_F(h), \qquad D_{\mathrm{reg}}(h) \leq \mathrm{UCB}_R(h). \tag{11}$$

Consequently, any hypothesis satisfying

$$\mathrm{LCB}_T(h) \geq \eta, \qquad \mathrm{UCB}_F(h) \leq \delta, \qquad \mathrm{UCB}_R(h) \leq \beta \tag{12}$$

also satisfies the population constraints

$$G(h) \geq \eta, \qquad P_{\mathrm{fail}}(h) \leq \delta, \qquad D_{\mathrm{reg}}(h) \leq \beta. \tag{13}$$

Proof sketch. Conditioning on the pre-evaluation history fixes $\mathcal{H}_t$ and prevents final probes from being reused for hypothesis construction. Empirical-Bernstein inequalities control the bounded target-transfer and regression quantities, while a Bernoulli-KL bound controls falsification risk. A union bound over hypotheses and statistics gives the simultaneous high-probability event, under which the confidence-gated rule implies the corresponding population constraints.

Corollary 1 (Probe complexity of safe commitment)**.** *Up to logarithmic factors in $|\mathcal{H}_t|$, $1/\alpha$, and the number of falsification probes, it suffices to use*

$$n_T = \widetilde{O}\left(\frac{1}{\zeta_T^2}\right), \qquad n_R = \widetilde{O}\left(\frac{1}{\zeta_R^2}\right), \qquad n_F = \widetilde{O}\left(\frac{1}{\mathrm{kl}(\delta_- \| \delta)}\right) \tag{14}$$

final target, regression, and falsification probes to ensure that any hypothesis with empirical target margin ζ_T, empirical regression margin ζ_R, and empirical falsification rate at most $\delta_- < \delta$ satisfies $G(h) \geq \eta$, $P_{\mathrm{fail}}(h) \leq \delta$, and $D_{\mathrm{reg}}(h) \leq \beta$ with probability at least $1 - \alpha$. The exact finite-sample statement is proved in Appendix B.7.

6 Experiments

6.1 Experimental Setup

Datasets. We evaluate FGSE on verifier-grounded agentic tasks where updates can be tested before persistent commitment. The benchmark suite covers web and browser control with WebArena [Zhou et al., 2024] and BrowserGym [Le Sellier de Chezelles et al., 2025], app and API interaction with AppWorld [Trivedi et al., 2024] and τ-bench [Yao et al., 2024], adversarial tool-use robustness with AgentDojo [Debenedetti et al., 2024], and long-term memory reliability with LongMemEval [Wu et al., 2024] and LoCoMo [Maharana et al., 2024]. We also include verifier-rich coding and reasoning diagnostics on HumanEval [Chen et al., 2021], MBPP [Austin et al., 2021], SWE-bench Verified [Jimenez et al., 2024, Chowdhury et al., 2024], GSM8K [Cobbe et al., 2021], and MATH [Hendrycks et al., 2021]. Details are provided in App. C.

Table 1: **Main comparison on web, app, and tool-interaction benchmarks.** For AppWorld, Avg. is the mean of Test-Normal and Test-Challenge task-goal completion. Best and second-best scores in each column are shown in **bold** and underlined, respectively.

Method	Backbone / Setting	WebArena SR ↑	AppWorld Normal ↑	AppWorld Challenge ↑	AppWorld Avg. ↑	τ-Retail Pass^1 ↑	τ-Airline Pass^1 ↑
Web / browser-control agents: methods primarily designed or evaluated for browser navigation and web interaction.							
GPT-4 Agent [Zhou et al., 2024]	GPT-4	14.4	43.2	25.4	34.3	0.553	0.378
AutoEval [Pan et al., 2024]	GPT-4	20.2	45.8	28.7	37.3	0.579	0.391
BrowserGym [Le Sellier de Chezelles et al., 2025]	GPT-4o	24.3	46.5	29.1	37.8	0.585	0.402
$\text{CER}_{\text{online}}$ [Liu et al., 2025]	GPT-4o	33.2	51.9	33.4	42.7	0.622	0.424
$\text{CER}_{\text{offline}}$ [Liu et al., 2025]	GPT-4o	33.4	52.6	34.1	43.4	0.631	0.429
$\text{CER}_{\text{hybrid}}$ [Liu et al., 2025]	GPT-4o	36.7	53.0	35.5	44.3	0.638	0.437
App / API-interaction agents: methods primarily evaluated on app-level tool use, database-state goals, and API execution.							
IPFunCall [Trivedi et al., 2024]	GPT-4o	11.8	32.1	18.0	25.1	0.482	0.309
FullCodeRefl [Trivedi et al., 2024]	GPT-4o	12.7	33.9	19.2	26.6	0.497	0.318
PlanExec [Trivedi et al., 2024]	GPT-4o	15.9	44.6	19.7	32.2	0.534	0.347
ReAct [Yao et al., 2023, Trivedi et al., 2024]	GPT-4o	18.1	48.8	30.2	39.5	0.558	0.325
ReAct + ICL [Zhang et al., 2026f]	DeepSeek-V3.1	19.6	64.3	46.0	55.2	0.615	0.408
ReAct + GEPA [Agrawal et al., 2026, Zhang et al., 2026f]	DeepSeek-V3.1	20.1	64.9	46.0	55.5	0.620	0.415
ReAct + DC [Suzgun et al., 2026, Zhang et al., 2026f]	DeepSeek-V3.1	20.6	65.5	52.3	58.9	0.630	0.421
ReAct + $\text{ACE}_{\text{offline}}$ [Zhang et al., 2026f]	DeepSeek-V3.1	25.7	76.2	57.3	66.8	0.654	0.448
ReAct + $\text{ACE}_{\text{online}}$ [Zhang et al., 2026f]	DeepSeek-V3.1	26.4	69.6	66.7	68.2	0.665	0.456
Tool-user interaction agents: methods centered on multi-turn user simulation, tool calling, and domain-policy execution.							
Tool-Calling [Yao et al., 2024]	GPT-4o	17.3	50.4	31.6	41.0	0.604	0.420
Tool-Calling [Yao et al., 2024]	Claude-3.5-Sonnet	18.6	54.2	34.7	44.5	0.692	0.460
Act [Yao et al., 2024]	GPT-4o	15.8	47.7	28.2	38.0	0.545	0.365
ReAct [Yao et al., 2023, 2024]	GPT-4o	16.4	48.1	29.0	38.6	0.521	0.325
Long-term memory agents: methods primarily designed for persistent memory, conversational memory, or memory-augmented retrieval.							
Offline Reading [Wu et al., 2024]	GPT-4o oracle	3.5	12.4	6.3	9.4	0.180	0.121
Coze [Wu et al., 2024]	GPT-4o	9.4	24.0	12.9	18.5	0.410	0.270
ChatGPT [Wu et al., 2024]	GPT-4o	13.8	43.0	26.0	34.5	0.560	0.390
GPT-4o + Chain-of-Note [Wu et al., 2024]	long-context	14.6	45.0	28.5	36.8	0.575	0.399
ChatGPT [Wu et al., 2024]	GPT-4o-mini	12.2	36.5	20.9	28.7	0.520	0.348
ReadAgent [Xu et al., 2025]	GPT-4o-mini	7.4	25.3	12.6	19.0	0.420	0.260
MemoryBank [Xu et al., 2025]	GPT-4o-mini	6.8	22.9	10.4	16.7	0.390	0.247
MemGPT [Xu et al., 2025]	GPT-4o-mini	10.7	29.7	16.2	23.0	0.450	0.286
LoCoMo [Maharana et al., 2024, Xu et al., 2025]	GPT-4o-mini	11.5	31.6	17.3	24.5	0.463	0.300
A-Mem [Xu et al., 2025]	GPT-4o-mini	14.9	38.2	22.5	30.4	0.544	0.368
FGSE (Ours)	GPT-4o-mini	38.6 ± 1.8	82.7 ± 2.2	66.1 ± 2.0	74.4 ± 1.6	0.707 ± 0.015	0.478 ± 0.012
FGSE (Ours)	GPT-4o	39.2 ± 1.5	84.8 ± 1.3	68.3 ± 1.8	76.6 ± 1.8	0.716 ± 0.012	0.471 ± 0.010
FGSE (Ours)	DeepSeek-V3.1	**39.8** ± 1.2	**88.2** ± 1.5	**69.4** ± 1.0	**78.8** ± 1.7	**0.724** ± 0.010	**0.487** ± 0.008

Evaluation metrics. We report each benchmark's official task metric, including success rate, task-goal completion, pass rate or pass^k, utility and security success, QA accuracy/F1, pass@1, resolved rate, and exact-match accuracy. To evaluate safe self-evolution, we additionally report target-transfer gain $\widehat{G}$, falsification failure rate $\widehat{P}_{\text{fail}}$, regression damage $\widehat{D}_{\text{reg}}$, post-commit violation rate, and harmful update rate.

Baselines. We compare FGSE with non-evolving agents, including direct prompting and ReAct [Yao et al., 2023], as well as self-update baselines such as Self-Refine [Madaan et al., 2023], Reflexion [Shinn et al., 2023], Voyager-style skill memory [Wang et al., 2023], and naive episodic memory. We further compare with recent self-evolving or agent-optimization methods, including Symbolic Learning [Zhou et al., 2025], AlphaLLM [Tian et al., 2024], GPTSwarm [Zhuge et al., 2024], AFlow [Zhang et al., 2025a], ADAS [Hu et al., 2025], MetaAgent [Zhang et al., 2025b], A-Mem [Xu et al., 2025], Lighthouse [Yang et al., 2025], and ACE [Zhang et al., 2026f]. For methods requiring workflow search, external data, or model updates, we separate controlled comparisons from native-setting results, since FGSE modifies only the persistent agent state and commitment rule.

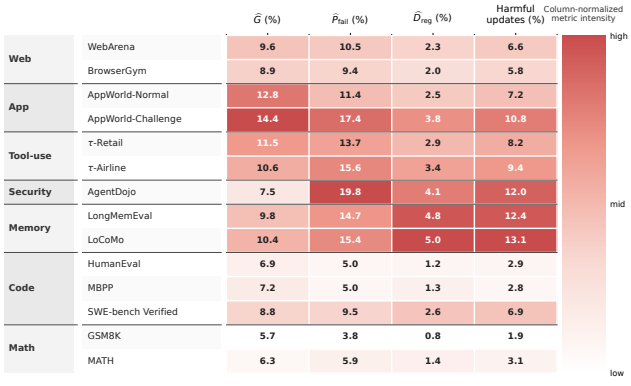
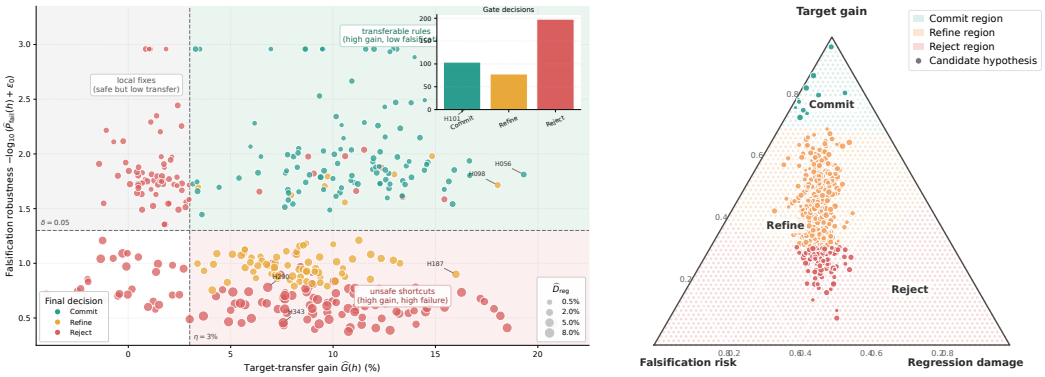


Figure 4: **Domain-level diagnostic of self-evolution risk.**

Implementation details. The default backbone is a ReAct-style executor with persistent memory retrieval. FGSE proposes at most $B = 3$ candidate hypotheses per episode and evaluates each using target-transfer, falsification, and regression probes with default budgets $|\mathcal{T}(h)| = 8$, $|\mathcal{F}(h)| = 16$,



(a) Transfer–Falsification profile **(b)** Decision simplex

Figure 5: **Interpreting FGSE update selection. (a)** Transfer–Falsification volcano plot for candidate hypotheses. The x-axis shows target-transfer gain $\widehat{G}$, the y-axis shows $-\log_{10}(\widehat{P}_{\text{fail}} + \epsilon_0)$, point size encodes regression damage $\widehat{D}_{\text{reg}}$, and color indicates the final FGSE decision. The dashed lines mark the useful-transfer and falsification thresholds. **(b)** Decision simplex over normalized target gain, falsification risk, and regression damage. The shaded area shows the gate-induced decision regions.

and $|\mathcal{R}(h)| = 16$. We tune only η, δ, and β on the development split, using defaults $\eta = 0.03$, $\delta = 0.05$, $\beta = 0.01$, $\epsilon = 0$, $\gamma_h = 0.5$, and $\lambda_h = 0.8$. Stochastic settings use three random seeds, and we report mean performance with bootstrap confidence intervals. Refer to App. C for more details.

6.2 Main Results

Overall comparison. Table 1 shows that FGSE improves over memory-writing and reflection-style baselines across verifier-rich web, app, tool-use, and memory benchmarks. The gains are especially clear on verifier-rich settings such as WebArena, AppWorld, and τ-bench, where tool-state and database-state validators provide reliable evidence for pre-commitment testing. Appendix D.1 further shows that FGSE improves long-term memory reliability while reducing falsification failures, regression damage, and harmful committed updates.

Domain-level update diagnostics. Beyond task success, Figure 4 reports update-centric metrics across domains, including $\widehat{G}$, $\widehat{P}_{\text{fail}}$, $\widehat{D}_{\text{reg}}$, and harmful update rate. AppWorld, τ-bench, AgentDojo, and long-term memory benchmarks show both meaningful gains and higher safety risks, indicating greater vulnerability to overgeneralized self-updates. Coding and math tasks exhibit lower falsification and regression risks due to cleaner executable or exact-match verifiers, while SWE-bench remains regression-prone because repository-level changes are highly contextual. These results support verifier-grounded falsification over commitment based only on target improvement.

6.3 Ablation and Mechanism Analysis

Single-factor ablation. We isolate FGSE components by removing or replacing one component at a time and report both task performance and update reliability. Blue values in parentheses denote absolute changes from full FGSE, with negative values indicating drops for higher-is-better metrics and positive values indicating worse lower-is-better metrics. Tables 2 and 6 show that each component contributes. Removing structured hypotheses causes the largest drop, reducing WebArena SR by 4.2 points, AppWorld Avg. by 5.9 points, and LoCoMo Avg. R-L by 4.78 points, confirming the need for scoped, verifier-aware updates. Target-transfer probes improve useful generalization, as $\widehat{G}$ drops from 12.9% to 9.8% without them, while falsification probes prevent unsafe overgeneralization, with $\widehat{P}_{\text{fail}}$ increasing from 5.0% to 18.6% and harmful updates from 3.6% to 10.6% when removed. The regression archive reduces $\widehat{D}_{\text{reg}}$ from 5.2% to 1.3%, and scope refinement, post-commit monitoring, and executable verifiers further improve the transfer-safety trade-off.

Transferability of candidate updates. We assess whether episode-derived updates capture reusable rules that transfer beyond the triggering case. We therefore evaluate each candidate before commit-

Table 2: **Single-factor ablation on task-level performance.**

Variant	WebArena SR ↑	AppWorld Avg. ↑	τ-Retail Pass^1 ↑	LongMemEval Acc. ↑	LoCoMo Avg. R-L ↑
Full FGSE	**39.8**	**78.8**	**0.724**	**0.748**	**37.62**
w/o structured hypothesis	35.6 (-4.2)	72.9 (-5.9)	0.664 (-0.060)	0.692 (-0.056)	32.84 (-4.78)
w/o target-transfer probes	36.8 (-3.0)	74.9 (-3.9)	0.684 (-0.040)	0.710 (-0.038)	34.72 (-2.90)
w/o falsification probes	37.4 (-2.4)	75.8 (-3.0)	0.697 (-0.027)	0.713 (-0.035)	35.10 (-2.52)
w/o regression archive	38.2 (-1.6)	76.9 (-1.9)	0.706 (-0.018)	0.719 (-0.029)	35.87 (-1.75)
w/o scope refinement	37.1 (-2.7)	75.4 (-3.4)	0.690 (-0.034)	0.716 (-0.032)	34.96 (-2.66)
LLM-judge gate	36.7 (-3.1)	74.1 (-4.7)	0.681 (-0.043)	0.707 (-0.041)	34.18 (-3.44)
w/o post-commit monitoring	39.0 (-0.8)	77.2 (-1.6)	0.713 (-0.011)	0.726 (-0.022)	35.92 (-1.70)

ment using three temporary statistics: target-transfer gain $\widehat{G}(h)$, falsification failure rate $\widehat{P}_{\text{fail}}(h)$, and regression damage $\widehat{D}_{\text{reg}}(h)$. Figure 5(a) plots candidates in a Transfer–Falsification volcano plot, with $\widehat{G}(h)$ on the x-axis, $-\log_{10}(\widehat{P}_{\text{fail}}(h)+\epsilon_0)$ on the y-axis, point size encoding $\widehat{D}_{\text{reg}}(h)$, and color indicating the gate decision. The plot reveals three groups: committed updates in the upper right with high transfer and low risk, unsafe shortcuts in the lower right, and safe but low-transfer local fixes in the upper left. In this run, FGSE commits $104/380$ candidates, achieving average $\widehat{G} = 10.77\%$, $\widehat{P}_{\text{fail}} = 1.28\%$, and $\widehat{D}_{\text{reg}} = 0.60\%$. The 101 rejected candidates that exceed the gain threshold η show much higher falsification and regression risk, with $\widehat{P}_{\text{fail}} = 22.82\%$ and $\widehat{D}_{\text{reg}} = 4.85\%$. Refined candidates lie between these groups, suggesting that many useful but unsafe updates require scope narrowing before commitment.

Commit–refine–reject decision boundary. To examine the gate, we map each candidate into a simplex over normalized target gain, falsification risk, and regression damage. For each candidate h, we normalize the three statistics across candidates and project them onto simplex coordinates (g_h, f_h, r_h) with $g_h + f_h + r_h = 1$. Figure 5(b) shows the joint role of the three statistics: COMMIT lies in the high-gain, low-risk region, REJECT concentrates near the high-risk, high-regression edge, and REFINE forms a transition band between them. This pattern indicates that FGSE performs structured triage, committing gain-dominant updates, refining repairable but ambiguous ones, and rejecting risk-dominant candidates.

7 Conclusion

We studied reliable self-evolution for long-horizon LLM agents, focusing on when episode-derived lessons should enter persistent memory. FGSE treats each candidate update as a structured, verifier-grounded hypothesis and commits it only after target-transfer, falsification, and regression testing. Our results show that safe adaptation requires evaluating usefulness, boundary failures, and regression on previously solved behavior alongside source-task improvement. Future work can extend falsification-gated update control to multimodal agents [Xu et al., 2026a, Dong et al., 2026, Liu et al., 2026a, Zhang et al., 2026e, Kang et al., 2026], collaborative tool-use systems [Fan et al., 2026], and settings where verifiers must be learned or approximated.

Acknowledgments and Disclosure of Funding

We thank Airon Tech for providing computing resources and data-processing support for this work. The authors declare no competing interests.

References

Lakshya A. Agrawal, Shangyin Tan, Dilara Soylu, Noah Ziems, Rishi Khare, Krista Opsahl-Ong, Arnav Singhvi, Herumb Shandilya, Michael J. Ryan, Meng Jiang, Christopher Potts, Koushik Sen, Alex Dimakis, Ion Stoica, Dan Klein, Matei Zaharia, and Omar Khattab. GEPA: Reflective prompt evolution can outperform reinforcement learning. In *International Conference on Learning Representations (ICLR)*, 2026. URL `https://proceedings.iclr.cc/paper_files/paper/2026/hash/0e9e708b6f48e14fd0ac29e167413f76-Abstract-Conference.html`.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, et al. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*, 2021.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde De Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

Qianyu Chen, Ziteng Feng, Canran Xiao, and Runxuan Tang. InduceKV: Fixed-footprint continual adaptation of multimodal LLMs via inducing KV memories. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2026. URL https://arxiv.org/abs/2607.02010.

Qibai Chen and Zeming Liu. SemVerBench: Benchmarking LLM comprehension of version-constraint resolution semantics, 2026. URL https://arxiv.org/abs/2609.11180.

Neil Chowdhury, James Aung, Chan Jun Shern, Oliver Jaffe, Dane Sherburn, Giulio Starace, Evan Mays, Rachel Dias, Marwan Aljubeh, Mia Glaese, Carlos E. Jimenez, John Yang, Leyton Ho, Tejal Patwardhan, Kevin Liu, and Aleksander Madry. Introducing SWE-bench Verified. OpenAI, August 2024. URL https://openai.com/index/introducing-swe-bench-verified/. Published August 13, 2024.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Edoardo Debenedetti et al. AgentDojo: A dynamic environment to evaluate prompt injection attacks and defenses for LLM agents. In *Advances in Neural Information Processing Systems (NeurIPS), Datasets and Benchmarks Track*, 2024. URL https://openreview.net/forum?id=m1YYAQjO3w. Spotlight presentation.

Kairui Ding, Boyuan Chen, Yuchen Su, Huan-ang Gao, Bu Jin, Chonghao Sima, Xiaohui Li, Wuqiang Zhang, Paul Barsch, Hongyang Li, and Hao Zhao. Hint-AD: Holistically aligned interpretability in end-to-end autonomous driving. In *Proceedings of The 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pages 3742–3765. PMLR, 2025. URL https://proceedings.mlr.press/v270/ding25a.html.

Guangyuan Dong, Chuang Liu, Haoyu Wang, Yangchen Zeng, Jiaqi Zhang, Li Jiuxing, Xiaoyang Yu, Pinlong Zhao, Yuchao Hou, Ziwei Li, Zheng Lin, Alexander Lim Han Yang, and Yusen Wu. When images look right and retrieve wrong: Coverage-guided cross-scale re-indexing for knowledge-faithful generative perception. *arXiv preprint arXiv:2608.20810*, 2026. URL https://arxiv.org/abs/2608.20810.

Enguang Fan, Anfeng Peng, Matthew Caesar, Jae Kim, Josh Eckhardt, Greg Kimberly, and Denis Osipychev. Towards effective swarm-based GPS spoofing detection in disadvantaged platforms. In *Proceedings of the 2023 IEEE Military Communications Conference (MILCOM)*, pages 722–728. IEEE, 2023. doi: 10.1109/MILCOM58377.2023.10356314.

Enguang Fan, Yifan Chen, Zihan Shan, Klara Nahrstedt, Matthew Caesar, and Jae Kim. Communication-aware multi-agent reinforcement learning for decentralized cooperative UAV deployment, 2026. URL https://arxiv.org/abs/2603.16141.

Jiani He, Dingyan Shang, Yihua Xu, Shiqi Huang, Yan Lyu, Jize Li, and Shangjing Tang. Semantic signal-assisted inspection and recovery allocation in reverse logistics, 2026. URL https://arxiv.org/abs/2609.02116.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*, 2021.

Saisai Hu. Research on security enhancement methods for adversarial robust large language model intelligent agents for medical decision-making tasks. In *2026 7th International Seminar on Artificial Intelligence, Networking and Information Technology (AINIT)*, pages 599–603. IEEE, 2026. URL https://arxiv.org/abs/2605.08257.

Shengran Hu et al. Automated design of agentic systems. In *International Conference on Learning Representations (ICLR)*, 2025. URL https://openreview.net/forum?id=t9U3LW7JVX.

Chengsong Huang et al. R-Zero: Self-evolving reasoning LLM from zero data. In *International Conference on Learning Representations (ICLR)*, 2026a. URL https://arxiv.org/abs/2508.05004.

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

Shiqi Huang, Jiani He, Dingyan Shang, Yihua Xu, Jize Li, Yan Lyu, and Lakshmi Muraleedharan Nair. DACRI: Decision-aware causal intervention ranking for critical supply chains, 2026b. URL https://arxiv.org/abs/2608.11154.

Carlos E Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik Narasimhan. Swe-bench: Can language models resolve real-world github issues? In *12th International Conference on Learning Representations, ICLR 2024*, 2024.

Zhaolu Kang, Tailong Luo, Lei Wei, Guangyuan Dong, Sihan Liu, Yingjie He, Siheng Wang, Shuaibo Li, Rongchao Zhang, Zhen Tian, Richeng Xuan, and Zhichao Hu. DRILL: Training world models to improve policies, not predict pixels. *arXiv preprint arXiv:2610.30814*, 2026. URL https://arxiv.org/abs/2610.30814. Accepted at NeurIPS 2026.

Thibault Le Sellier de Chezelles, Maxime Gasse, Alexandre Lacoste, Massimo Caccia, Alexandre Drouin, Léo Boisvert, Megh Thakkar, Tom Marty, Rim Assouel, Sahar Omidi Shayegan, Lawrence Keunho Jang, Xing Han Lù, Ori Yoran, Dehan Kong, Frank F. Xu, Siva Reddy, Graham Neubig, Quentin Cappart, Russ Salakhutdinov, and Nicolas Chapados. The BrowserGym ecosystem for web agent research. *Transactions on Machine Learning Research*, 2025. ISSN 2835-8856. URL https://openreview.net/forum?id=5298fKGmv3. Expert Certification.

Jize Li, Jiani He, Dishu Yang, Dingyan Shang, Jingjing Liu, and Shiqi Huang. Accuracy-preserving stability regularization for large-scale retail demand forecasting, 2026a. URL https://arxiv.org/abs/2607.13331.

Wei Li, Rui Shao, Jie He, Lingsen Zhang, Ziwei Liu, and Liqiang Nie. UniMPA: A unified memory-prediction-action model via action-grounded transition modeling. *arXiv preprint arXiv:2609.11875*, 2026b.

Yang Li, Xiaoxue Chen, Hao Zhao, Jiangtao Gong, Guyue Zhou, Federico Rossano, and Yixin Zhu. Understanding embodied reference with Touch-Line Transformer. In *International Conference on Learning Representations (ICLR)*, 2023. URL https://arxiv.org/abs/2210.05668.

Zhiming Lin, Canran Xiao, and Kai Zhao. Beyond more context: Retrieval diversity boosts multi-turn intent understanding. In *Proceedings of the ACM Web Conference 2026*, pages 2320–2329. Association for Computing Machinery, 2026. doi: 10.1145/3774904.3792567. URL https://doi.org/10.1145/3774904.3792567.

Jiaxin Liu, Ding Zhong, Yue Wang, Zhidong Yang, Zhaolu Kang, Guangyuan Dong, Qishi Zhan, Pengcheng Fang, and Aofan Liu. Dual-pathway circuits of object hallucination in vision-language models. *arXiv preprint arXiv:2605.13156*, 2026a. URL https://arxiv.org/abs/2605.13156. Accepted at NeurIPS 2026.

Jingjing Liu, Dishu Yang, Hongbo Qin, and Hansong Zhang. Passing tests, missing semantics: Evidence of interpretive lock-in in plan-based DataFrame code generation, 2026b. URL https://arxiv.org/abs/2610.15797.

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

Yao Lu, Tengfei Ma, Zeyu Wang, Zhuangzhi Chen, Dongwei Xu, Yun Lin, Qi Xuan, and Guan Gui. FCOS: A two-stage recoverable model pruning framework for automatic modulation recognition. *IEEE Transactions on Cognitive Communications and Networking*, 2026.

Mingjun Ma, Saisai Hu, Tiantian Zhu, Yitian Lin, Jiachen Sun, Zhengqiu Weng, Dehao Li, and Yajie Zhang. MuSK: Multi-scale knowledge learning for provenance-graph anomaly detection. *Computer Networks*, page 112728, 2026. doi: 10.1016/j.comnet.2026.112728. URL https://doi.org/10.1016/j.comnet.2026.112728.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. Self-Refine: Iterative refinement with self-feedback. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023. URL https://openreview.net/forum?id=S37hOerQLB.

Adyasha Maharana et al. Evaluating very long-term conversational memory of LLM agents, 2024. URL https://arxiv.org/abs/2402.17753.

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

Jiayi Pan, Yichi Zhang, Nicholas Tomlin, Yifei Zhou, Sergey Levine, and Alane Suhr. Autonomous evaluation and refinement of digital agents. In *First Conference on Language Modeling*, 2024.

Joon Sung Park, Joseph O'Brien, Carrie Jun 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*, pages 1–22, 2023.

Yujia Qin et al. ToolLLM: Facilitating large language models to master 16000+ real-world APIs. In *International Conference on Learning Representations (ICLR)*, 2024. URL https://openreview.net/forum?id=dHng2O0Jjr.

Shuai Shao et al. Your agent may misevolve: Emergent risks in self-evolving LLM agents. In *International Conference on Learning Representations (ICLR)*, 2026. URL https://openreview.net/pdf?id=Fd1jgQQW28.

Wanru Shao. Interpretable ensemble learning for network traffic anomaly detection: A SHAP-based explainable AI framework for embedded systems security. In *2026 14th International Symposium on Digital Forensics and Security (ISDFS)*, pages 1–4. IEEE, 2026a. doi: 10.1109/ISDFS69419.2026.11459069. URL https://doi.org/10.1109/ISDFS69419.2026.11459069.

Wanru Shao. Policy-driven vulnerability risk quantification framework for large-scale cloud infrastructure data security. In *Proceedings of the 2026 5th International Conference on Cyber Security, Artificial Intelligence and Digital Economy*, pages 40–45, New York, NY, USA, 2026b. Association for Computing Machinery. doi: 10.1145/3819836.3819842. URL https://doi.org/10.1145/3819836.3819842.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023. URL https://openreview.net/forum?id=vAElhFcKW6.

Yiyun Su, Huiying Zhu, Yu Tian, Changruo Zhao, Zujun Peng, Yuting Liu, Luyan Zhang, Liang Fan, and Baihua Li. Agentic-SQL Taxonomy: The research of autonomous and interactive Text-to-SQL with LLMs. In *Advanced Intelligent Computing Technology and Applications*, volume 16655 of *Lecture Notes in Computer Science*, pages 323–334. Springer Nature Singapore, 2027. doi: 10.1007/978-981-92-3438-7_27. URL https://doi.org/10.1007/978-981-92-3438-7_27.

Mirac Suzgun, Mert Yuksekgonul, Federico Bianchi, Dan Jurafsky, and James Zou. Dynamic cheatsheet: Test-time learning with adaptive memory. In *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7080–7106, 2026.

Wanxin Tian et al. SEEA-R1: Tree-structured reinforcement fine-tuning for self-evolving embodied agents. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025. URL https://openreview.net/forum?id=dAwKePZvcN.

Ye Tian et al. ALPHALLM: Toward self-improvement of large language models via imagination, searching, and criticizing. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024. URL https://openreview.net/pdf?id=tPdJ2qHkOB.

Harsh Trivedi et al. AppWorld: A controllable world of apps and people for benchmarking interactive coding agents. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2024. URL https://arxiv.org/abs/2407.18901.

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, 2023. URL https://openreview.net/forum?id=ehfRiF0R3a.

Mengxiao Wang and Nitesh Saxena. Trading agents or market crashers? dissecting robustness and security failures in academic financial LLM trading schemes, 2026. URL https://arxiv.org/abs/2609.19705.

Zijun Wang, Saisai Hu, Tiantian Zhu, Wenrui Cheng, Zhengqiu Weng, Haiting Chen, Xiangyang Zheng, and Suyu Zhang. Trident: Dual-stream APT attribution over heterogeneous threat knowledge graphs. *Computers & Security*, 171:105094, 2026. doi: 10.1016/j.cose.2026.105094. URL https://doi.org/10.1016/j.cose.2026.105094.

D. Wu et al. LongMemEval: Benchmarking chat assistants on long-term memory, 2024. URL https://arxiv.org/abs/2410.10813.

Peng Xia et al. Agent0: Unleashing self-evolving agents from zero data via tool-integrated reasoning, 2025. URL https://arxiv.org/abs/2511.16043.

Canran Xiao, Tianxiang Xu, SiYuan Ma, Yiyang Jiang, Haoyu Gao, and Yuhan Wu. Reversible primitive–composition alignment for continual vision–language learning. In *International Conference on Learning Representations (ICLR)*, 2026a. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/8f0d446441a938d9de420a8ab8d7fd36-Abstract-Conference.html.

Linqi Xiao, Venkateswarlu Gudepu, and Andrea Fumagalli. Joint alarm correlation and root alarm detection in optical transport networks with graph-attention. In *2025 IEEE 30th International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD)*, pages 1–6. IEEE, 2025. doi: 10.1109/CAMAD67323.2025.11229899.

Linqi Xiao, Venkateswarlu Gudepu, and Andrea Fumagalli. CORA: A conversational AI-agent for intelligent automation of OpenROADM multi-vendor optical transport networks. In *Applications of Machine Learning 2026*, volume 14218 of *Proceedings of SPIE*, page 142180Y. SPIE, 2026b. doi: 10.1117/12.3113861.

Linqi Xiao, Aparaajitha Gomathinayakam Latha, Venkateswarlu Gudepu, and Andrea Fumagalli. A YANG-grounded LLM agent supporting multi-vendor OpenROADM optical transport networks. In *Proceedings of the ACM SIGCOMM 2026 Conference*, pages 2109–2115. Association for Computing Machinery, 2026c. doi: 10.1145/3789240.3822606.

Shilin Xiao, Tianxiang Xu, Canran Xiao, Weihao Luo, Liwei Hou, and Chuangxin Zhao. Meta-UCF: Unified task-conditioned LoRA generation for continual learning in large language models. In *International Conference on Learning Representations (ICLR)*, 2026d. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/12202970782399ee67981dc5269c3b8a-Abstract-Conference.html.

Junhao Xu, Zheng Liu, Mengqi Zhang, Guangyuan Dong, and Zhumin Chen. Lever can move the earth: Towards adaptive semantic capacity balance for image-text retrieval. *IEEE Transactions on Multimedia*, pages 1–14, 2026a. doi: 10.1109/TMM.2026.3723619. URL https://doi.org/10.1109/TMM.2026.3723619.

Mengzhu Xu, Hanzhi Liu, Ningkang Peng, Qianyu Chen, and Canran Xiao. Affordance-first decomposition for continual learning in video-language understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3908–3919, June 2026b. URL https://openaccess.thecvf.com/content/CVPR2026/html/xu_Affordance-First_Decomposition_for_Continual_Learning_in_Video-Language_Understanding_CVPR_2026_paper.html.

Wujiang Xu et al. A-Mem: Agentic memory for LLM agents. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025. URL https://openreview.net/forum?id=FiM0M8gcct.

Yihua Xu, Yan Lyu, Jize Li, Jiani He, You Lin, Yidan Hu, and Rubing Qiang. When does an added planning capability create operational value? an observable evidence standard, 2026c. URL https://arxiv.org/abs/2610.27305.

Chengrun Yang, Xuezhi Wang, Yifeng Lu, Hanxiao Liu, Quoc V. Le, Denny Zhou, and Xinyun Chen. Large language models as optimizers. In *International Conference on Learning Representations (ICLR)*, 2024. URL https://openreview.net/forum?id=Bb4VGOWELI.

Dishu Yang, Jingjing Liu, and Jize Li. Balancing privacy, utility, and safety in LLM alignment through preference optimization, 2026a. URL https://arxiv.org/abs/2608.30141.

Dishu Yang, Qi Su, Hongbo Qin, and Hansong Zhang. Measurement-first auditing of agentic leaderboards: Contamination susceptibility, matched-control re-evaluation, and scorer validation, 2026b. URL https://arxiv.org/abs/2610.05830.

Ruihan Yang et al. The lighthouse of language: Enhancing LLM agents via critique-guided improvement. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025. URL https://openreview.net/forum?id=D88yhLFzDR.

Junyi Yao, Baichuan Li, Zihao Zheng, and Jiayu Long. Toward workflow-aware benchmarking for healthcare NLP agents, 2026. URL https://arxiv.org/abs/2609.00296.

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

Shunyu Yao et al. τ-bench: A benchmark for tool-agent-user interaction in real-world domains, 2024. URL https://arxiv.org/abs/2406.12045.

Hua Ye, Siyuan Chen, Ziqi Zhong, Canran Xiao, Haoliang Zhang, Yuhan Wu, and Fei Shen. Seeing through the conflict: Transparent knowledge conflict handling in retrieval-augmented generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 34423–34431, 2026. doi: 10.1609/aaai.v40i40.40740. URL https://ojs.aaai.org/index.php/AAAI/article/view/40740.

Chenghao Zhang, Yuxi Cheng, Saisai Hu, Canran Xiao, Yikai Mao, and Dan Roth. Training needs trustworthy worlds: Verified synthetic web environments for agent learning. *arXiv preprint arXiv:2608.21898*, 2026a. URL https://arxiv.org/abs/2608.21898.

Fan Zhang, Gongguan Chen, Hua Wang, Jinjiang Li, and Caiming Zhang. Multi-scale video super-resolution transformer with polynomial approximation. *IEEE Transactions on Circuits and Systems for Video Technology*, 33(9):4496–4506, 2023.

Fan Zhang, Gongguan Chen, Hua Wang, and Caiming Zhang. CF-DAN: Facial-expression recognition based on cross-fusion dual-attention network. *Computational Visual Media*, 10(3):593–608, 2024. doi: 10.1007/s41095-023-0369-x.

Fan Zhang, Shiming Fan, and Hua Wang. Rethinking activation function: A simple path to efficient and accurate time series forecasting. *IEEE Transactions on Knowledge and Data Engineering*, 2026b.

Fan Zhang, Shiming Fan, and Hua Wang. TimeSAF: Towards LLM-guided semantic asynchronous fusion for time series forecasting. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 26267–26284. Association for Computational Linguistics, 2026c. doi: 10.18653/v1/2026.acl-long.1208. URL https://aclanthology.org/2026.acl-long.1208/.

Fan Zhang, Zhiwei Gu, and Hua Wang. Decoding with structured awareness: Integrating directional, frequency-spatial, and structural attention for medical image segmentation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 12421–12429, 2026d. doi: 10.1609/aaai.v40i15.38235.

Jiaqi Zhang, Zheng Pang, Mengting Li, Yiqi Wang, Guangyuan Dong, Chao Xue, Yusen Wu, Zihao Li, Huy Phan, Sicheng Zhao, Björn Schuller, and Jiachen Luo. Modality disentangled learning for incomplete multimodal emotion recognition: A primitive memory distillation perspective. In Hung-Yi Lee, Sunipa Dev, Jessy Li, and Desmond Elliott, editors, *Findings of the Association for Computational Linguistics: EMNLP 2026*, pages 47534–47552, Budapest, Hungary, October 2026e. Association for Computational Linguistics. ISBN 979-8-89176-462-0. doi: 10.18653/2026.findings-emnlp.2220. URL https://aclanthology.org/2026.findings-emnlp.2220/.

Jiayi Zhang et al. AFlow: Automating agentic workflow generation. In *International Conference on Learning Representations (ICLR)*, 2025a. URL https://openreview.net/forum?id=z5uVAKwmjf. Oral presentation.

Qizheng Zhang, Changran Hu, Shubhangi Upasani, Boyuan Ma, Fenglu Hong, Vamsidhar Kamanuru, Jay Rainton, Chen Wu, Mengmeng Ji, Hanchen Li, Urmish Thakker, James Zou, and Kunle Olukotun. Agentic context engineering: Evolving contexts for self-improving language models. In *International Conference on Learning Representations (ICLR)*, 2026f. URL https://arxiv.org/abs/2510.04618.

Yaolun Zhang et al. MetaAgent: Automatically constructing multi-agent systems based on finite state machines. In *International Conference on Machine Learning (ICML)*, 2025b. URL https://openreview.net/forum?id=vOxaD3hhPt.

Xiaoji Zheng, Lixiu Wu, Zhijie Yan, Yuanrong Tang, Hao Zhao, Chen Zhong, Bokui Chen, and Jiangtao Gong. Large language models powered context-aware motion prediction in autonomous driving. In *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 980–985. IEEE, 2024. URL https://arxiv.org/abs/2403.11057.

Zihao Zheng, Junyi Yao, Baichuan Li, and Jiayu Long. Auditing escalation reliability in healthcare LLM agents: A perturbation-controlled benchmark of workflow-state failures, 2026. URL https://arxiv.org/abs/2610.21270.

Wanjun Zhong, Lianghong Guo, Qiqi Gao, He Ye, and Yanlin Wang. Memorybank: Enhancing large language models with long-term memory. In *Proceedings of the AAAI conference on artificial intelligence*, volume 38(17), pages 19724–19731, 2024.

Heng Zhou, Jing Tang, Jusheng Zhang, Yanshu Li, Canran Xiao, Liwei Hou, Zong Ke, and Jiawei Yao. CoMem: Compositional concept-graph memory for vision–language adaptation. In *International Conference on Learning Representations (ICLR)*, 2026. URL https://openreview.net/forum?id=xp7wDU9JBW.

Shuyan Zhou, Frank F. Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Yonatan Bisk, Daniel Fried, Uri Alon, and Graham Neubig. WebArena: A realistic web environment for building autonomous agents. In *International Conference on Learning Representations (ICLR)*, 2024. URL https://openreview.net/forum?id=oKn9c6ytLx.

Wangchunshu Zhou et al. Symbolic learning enables self-evolving agents. *AI Open*, 2025. URL https://www.sciencedirect.com/science/article/pii/S2666651025000208.

Mingchen Zhuge et al. GPTSwarm: Language agents as optimizable graphs. In *International Conference on Machine Learning (ICML)*, 2024. URL https://openreview.net/forum?id=uTC9AFXIhg. Oral presentation.

Yuxin Zuo et al. TTRL: Test-time reinforcement learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025. URL https://openreview.net/forum?id=VuVhgEiu20.

Appendix	Second-Level Headings	
Appendix A **Additional Technical Details**	A.1	Details of the Exploratory Preliminary Study
	A.2	Verifier-Grounded Probe Construction
	A.3	Precondition Refinement
	A.4	Empirical Estimation Under Stochastic Rollouts
	A.5	Runtime Retrieval, Combination, and Monitoring
	A.6	Audit Record
	A.7	Detailed Algorithm with Refinement
Appendix B **Proofs for the Commitment Gate Generalization**	B.1	Conditional Setup
	B.2	Confidence Bounds Used by the Commitment Gate
	B.3	Concentration Tools
	B.4	Uniform Confidence Event
	B.5	Proof of Theorem 1
	B.6	Exact Probe Complexity Statement
	B.7	Proof of Corollary 1
	B.8	Remarks on Adaptivity and Scope Refinement
Appendix C **Additional Experimental Details**	C.1	Benchmark Details
	C.2	Evaluation Metrics
	C.3	Fair Comparison Protocol
	C.4	Baseline Implementation Details
	C.5	FGSE Hyperparameters
	C.6	Ablation Studies
	C.7	Compute Resources
Appendix D **Additional Experimental Results**	D.1	Additional Reliability and Ablation Results
	D.2	Update-Selection and Falsification Analysis
	D.3	Mechanistic Analysis of Long-Term Self-Evolution
	D.4	Cost and Scalability
Appendix E **Broader Impacts**	-	
Appendix F **LLM Usage**	-	

A Additional Technical Details

A.1 Details of the Exploratory Preliminary Study

The preliminary study in Section 3 examines *naive episode-derived memory writing*. The agent directly summarizes a completed episode into a free-form lesson and writes it into persistent memory.

Episode-derived lessons. For each completed episode, the base agent observes the task instruction x, its interaction trajectory τ, and the verifier outcome r. It then produces a natural-language lesson

$$u = \mathrm{SummarizeLesson}(x, \tau, r), \tag{15}$$

which may take the form of a rule, reminder, correction, or behavioral heuristic. The updated agent state is obtained by simply appending u to the memory context:

$$\Theta \oplus u = (\mathcal{M} \cup \{u\}, \pi), \tag{16}$$

where $\mathcal{M}$ denotes the current memory and π denotes the unchanged base policy. This direct memory-writing procedure is a common self-improvement baseline that exposes unsafe updates.

Probe groups. To analyze the effect of a lesson u, we evaluate the original agent Θ and the memory-augmented agent $\Theta \oplus u$ on four groups of diagnostic cases. The *source-replay* group $\mathcal{C}_{\mathrm{src}}(u)$

contains the original episode and minor perturbations that preserve the same intended solution. The *near-transfer* group $\mathcal{C}_{\mathrm{near}}(u)$ contains tasks that share the same underlying behavioral pattern but differ in surface form, entities, or context. The *boundary-contrast* group $\mathcal{C}_{\mathrm{bdry}}(u)$ contains cases that are superficially similar to the source episode but should not trigger the same lesson. The *archived-success* group $\mathcal{C}_{\mathrm{arch}}(u)$ contains previously solved cases sampled from the agent's historical archive. The first two groups measure whether the lesson captures a useful pattern, while the latter two groups measure whether it overgeneralizes or interferes with existing behavior.

Verifier-score changes. For a case x, we estimate the verifier-grounded performance of an agent state Θ as

$$J(\Theta, x) = \frac{1}{L} \sum_{\ell=1}^{L} v(x, \tau^{(\ell)}), \qquad \tau^{(\ell)} \sim p_{\Theta}(\cdot \mid x), \tag{17}$$

where $v(x, \tau) \in [0, 1]$ is the task verifier and L is the number of repeated rollouts. For deterministic tasks, $L = 1$ is sufficient. The effect of lesson u on a case group $\mathcal{C}$ is

$$\Delta_{\mathcal{C}}(u) = \frac{1}{|\mathcal{C}|} \sum_{x \in \mathcal{C}} \left[J(\Theta \oplus u, x) - J(\Theta, x) \right]. \tag{18}$$

We report source gain $\Delta_{\mathrm{src}}(u)$, near-transfer gain $\Delta_{\mathrm{near}}(u)$, boundary harm

$$H_{\mathrm{bdry}}(u) = \left[-\Delta_{\mathcal{C}_{\mathrm{bdry}}}(u) \right]_+, \tag{19}$$

and archive harm

$$H_{\mathrm{arch}}(u) = \left[-\Delta_{\mathcal{C}_{\mathrm{arch}}}(u) \right]_+. \tag{20}$$

A lesson is considered locally useful if it improves near-transfer cases, but it is considered unsafe if it also induces large boundary or archive harm.

Scope-specificity annotation. To understand why some useful lessons are harmful, we annotate each free-form lesson with a coarse scope-specificity score $s(u) \in [0, 1]$. A lesson receives a higher score when it contains explicit applicability conditions, contextual triggers, exception clauses, or task-specific constraints. It receives a lower score when it is phrased as a broad heuristic without clear conditions. This analysis-only score tests whether overgeneralization is associated with underspecified lesson scope; update selection in the preliminary study proceeds without it.

Sequential memory-writing analysis. To study error accumulation, we simulate sequential memory writing over a stream of candidate lessons. After writing k lessons, the agent state is

$$\Theta_k = \Theta_0 \oplus \{u_1, \ldots, u_k\}, \tag{21}$$

and its held-out verifier score is measured on a fixed evaluation set $\mathcal{E}$:

$$S_k = \frac{1}{|\mathcal{E}|} \sum_{x \in \mathcal{E}} J(\Theta_k, x). \tag{22}$$

We compare three diagnostic memory-writing strategies: writing all lessons, writing only lessons that improve the source episode, and a hindsight safe subset. The hindsight subset keeps lessons with high near-transfer gain and low boundary/archive harm, and is used only as an oracle diagnostic upper bound. It is not available to the agent during deployment. This comparison isolates the main observation that source improvement alone is insufficient for stable long-term self-evolution.

Post-hoc scope localization. For lessons that are useful but harmful, we conduct a post-hoc localization analysis. Given a lesson u with positive near-transfer gain but high boundary harm, we inspect the boundary failures and rewrite the lesson into a narrower version u' by adding an explicit applicability condition. We then reevaluate u' on the same near-transfer and boundary-contrast groups. We summarize the effect of localization by the transfer-retention ratio

$$R_{\mathrm{retain}}(u') = \frac{\Delta_{\mathrm{near}}(u')}{\Delta_{\mathrm{near}}(u) + \varepsilon}, \tag{23}$$

and the boundary-harm reduction ratio

$$R_{\mathrm{reduce}}(u') = \frac{H_{\mathrm{bdry}}(u) - H_{\mathrm{bdry}}(u')}{H_{\mathrm{bdry}}(u) + \varepsilon}, \tag{24}$$

where ε is a small constant for numerical stability. This separate diagnostic motivates explicit applicability conditions and refinement of useful but overly broad candidates in the main method.

Connection to the method. The preliminary study supports three design choices. First, testing episode-derived updates before persistence can expose unsafe free-form lessons. Second, contrastive and archived cases expose boundary failures that source or near-transfer cases can miss. Third, scope refinement can repair some harmful lessons while retaining their useful content. These observations motivate the structured hypothesis representation, temporary evaluation, verifier-grounded commitment, and scope refinement introduced in Section 4.

A.2 Verifier-Grounded Probe Construction

FGSE constructs probes from three candidate sources: archived solved tasks, verifier-preserving perturbations of the triggering episode, and language-model-generated candidates filtered by executable verifiers. The verifier filter is applied before evaluation:

$$\mathrm{Verifiable}(x, v_h) = 1 \quad \Longleftrightarrow \quad v_h(x, \tau) \text{ returns a stable score for every evaluated rollout } \tau. \tag{25}$$

Tasks that cannot be checked by v_h are excluded from the gate.

The falsification set is decomposed as

$$\mathcal{F}(h) = \mathcal{F}_{\mathrm{bd}}(h) \cup \mathcal{F}_{\mathrm{pv}}(h) \cup \mathcal{F}_{\mathrm{rg}}(h) \cup \mathcal{F}_{\mathrm{dc}}(h). \tag{26}$$

Here, $\mathcal{F}_{\mathrm{bd}}$ contains boundary probes near the activation threshold of c_h, $\mathcal{F}_{\mathrm{pv}}$ contains precondition-violation probes that are similar to the claimed scope but should not activate the rule, $\mathcal{F}_{\mathrm{rg}}$ contains regression probes from previously solved cases, and $\mathcal{F}_{\mathrm{dc}}$ contains decoy probes that are superficially similar to target cases but require different behavior.

For each probe type $q \in \{\mathrm{bd}, \mathrm{pv}, \mathrm{rg}, \mathrm{dc}\}$, FGSE assigns a risk score

$$\rho_q(x; h) = \begin{cases} \exp(-|c_h(x) - \gamma_h|/\sigma)\,, & q = \mathrm{bd}, \\ (1 - c_h(x))\ \mathrm{sim}(x, h), & q = \mathrm{pv}, \\ \mathbb{I}[J_{v_h}(\Theta_t, x) \geq \zeta]\ \mathrm{sim}(x, h), & q = \mathrm{rg}, \\ \mathrm{sim}(x, h)\, \mathbb{I}[c_h(x) < \gamma_h], & q = \mathrm{dc}. \end{cases} \tag{27}$$

In Eq. (27), $\sigma > 0$ controls the width of the boundary region, $\mathrm{sim}(x, h)$ measures semantic or structural similarity between x and the triggering context of h, and ζ is the score threshold for considering an archived case solved.

The selected probes are

$$\mathcal{F}_q(h) = \mathrm{TopK}_{x \in \mathcal{X}_q} \left\{ \rho_q(x; h) : \mathrm{Verifiable}(x, v_h) = 1 \right\}, \tag{28}$$

where $\mathcal{X}_q$ is the candidate pool for probe type q. Boundary probes test whether the hypothesis changes behavior too aggressively near its activation threshold. Precondition-violation probes test whether the rule fires outside its intended condition. Regression probes measure damage to cases already solved by Θ_t. Decoy probes test whether superficial similarity causes the wrong behavior change.

The target-transfer set $\mathcal{T}(h)$ is selected from verifiable candidates satisfying $c_h(x) \geq \gamma_h$. FGSE prioritizes candidates that match the precondition while differing from the triggering task in surface form, tool trace, or problem instance. This tests transfer beyond memorization of the original episode.

A.3 Precondition Refinement

Refinement is used when a hypothesis improves target probes but fails on falsification probes that can be separated by a more precise precondition. FGSE forms the safe and falsifying sets

$$\begin{aligned} \mathcal{P}_h^+ &= \{x \in \mathcal{T}(h) \cup \mathcal{F}(h) : \Delta_h(x) \geq 0,\ E_h(\Theta_t \oplus h, x) = 1\}, \\ \mathcal{P}_h^- &= \{x \in \mathcal{F}(h) : \Delta_h(x) < -\epsilon\ \vee\ E_h(\Theta_t \oplus h, x) = 0\}. \end{aligned} \tag{29}$$

The refined precondition is obtained by

$$c_h' = \arg\min_{c \in \mathfrak{C}} \left[\Omega(c) + \lambda_+ \sum_{x \in \mathcal{P}_h^+} \ell(1, c(x)) + \lambda_- \sum_{x \in \mathcal{P}_h^-} \ell(0, c(x)) \right]. \tag{30}$$

Here, $\mathfrak{C}$ is the precondition class, $\Omega(c)$ penalizes overly complex conditions, ℓ is binary cross-entropy, and $\lambda_+, \lambda_- \geq 0$ weight safe and falsifying probes.

FGSE triggers refinement only when the following separability score is below a threshold:

$$\mathrm{Sep}(h) = \min_{c \in \mathfrak{C}} \left[\frac{1}{|\mathcal{P}_h^+|} \sum_{x \in \mathcal{P}_h^+} \mathbb{I}[c(x) < \gamma_h] + \frac{1}{|\mathcal{P}_h^-|} \sum_{x \in \mathcal{P}_h^-} \mathbb{I}[c(x) \geq \gamma_h] \right] \leq \alpha_{\mathrm{sep}}. \tag{31}$$

A refined hypothesis is $h' = (c_h', b_h, e_h, v_h)$. FGSE performs one refinement pass. The refined hypothesis is then evaluated by the same equations used in the main method. If it fails the gate after refinement, it is rejected.

A.4 Empirical Estimation Under Stochastic Rollouts

The verifier-grounded score in Eq. (2) is estimated with repeated rollouts:

$$\widehat{J}_v(\Theta, x) = \frac{1}{n} \sum_{j=1}^{n} v(x, \tau^{(j)}), \qquad \tau^{(j)} \sim p_\Theta(\cdot \mid x). \tag{32}$$

The empirical update effect is

$$\widehat{\Delta}_h(x) = \widehat{J}_{v_h}(\Theta_t \oplus h, x) - \widehat{J}_{v_h}(\Theta_t, x). \tag{33}$$

In experiments with deterministic verifiers and deterministic decoding, $n = 1$ is sufficient. With stochastic decoding or nondeterministic tool responses, FGSE uses paired random seeds for Θ_t and $\Theta_t \oplus h$ when the environment supports seed control. This reduces variance in $\widehat{\Delta}_h(x)$ because both states are evaluated under matched rollout conditions.

For finite probe budgets, FGSE reports the conservative falsification score

$$S_{\mathrm{fail}}(h) = \widehat{P}_{\mathrm{fail}}(h) + \kappa |\mathcal{F}(h)|^{-1/2}, \tag{34}$$

where $\kappa \geq 0$ controls the penalty for small falsification sets. This score is used for analysis and ablations. The main gate in Eq. (8) uses the empirical falsification rate $\widehat{P}_{\mathrm{fail}}(h)$.

A.5 Runtime Retrieval, Combination, and Monitoring

At inference time, FGSE retrieves committed hypotheses whose preconditions match the current interaction state $z = (x, \tau_{<\ell})$:

$$\mathcal{H}_t(z) = \{h \in \mathcal{M}_t : c_h(z) \geq \gamma_h,\ u_t(h) \leq u_{\max}\}, \tag{35}$$

where $u_t(h)$ is the post-commit violation score and $u_{\max}$ is the maximum allowed violation level.

When $\mathcal{H}_t(z)$ is nonempty, FGSE assigns normalized weights

$$w_h(z) = \frac{\exp(c_h(z)/\rho)}{\sum_{h' \in \mathcal{H}_t(z)} \exp(c_{h'}(z)/\rho)}, \qquad h \in \mathcal{H}_t(z), \tag{36}$$

where $\rho > 0$ is the temperature. The runtime policy is

$$\pi_t^{\mathrm{FGSE}}(a \mid z) = (1 - \lambda(z)) \pi_{\Theta_t}(a \mid z) + \lambda(z) \sum_{h \in \mathcal{H}_t(z)} w_h(z) b_h[\pi_{\Theta_t}](a \mid z), \tag{37}$$

where $\lambda(z) \in [0, 1]$ is the total intervention strength. This form preserves a nonzero base-policy contribution even when several hypotheses are active.

After a committed hypothesis is activated, FGSE records whether its expected effect is violated:

$$V_t(h, x_t) = \mathbb{I}\left[E_h(\Theta_t, x_t) = 0 \vee J_{v_h}(\Theta_t, x_t) < \zeta_h\right], \tag{38}$$

where ζ_h is the minimum acceptable verifier score associated with hypothesis h. The violation score is updated by

$$u_{t+1}(h) = \rho_u u_t(h) + (1 - \rho_u) V_t(h, x_t), \tag{39}$$

where $\rho_u \in [0, 1)$ controls the memory of previous violations. If $u_{t+1}(h) > u_{\max}$, the hypothesis is removed from the active retrieval set. Removed hypotheses remain in the audit archive and are not used for future action selection.

A.6 Audit Record

Each committed hypothesis is stored with an audit record

$$\mathrm{Record}(h) = \Big(h, \mathcal{T}(h), \mathcal{F}(h), \mathcal{R}(h), \{\Delta_h(x)\}_{x\in\mathcal{P}(h)}, \widehat{G}(h), \widehat{P}_{\mathrm{fail}}(h), \widehat{D}_{\mathrm{reg}}(h), u_t(h)\Big). \quad (40)$$

The audit record supports post-hoc inspection of why a rule was committed, which probes were used to falsify it, and whether the rule later accumulated violations.

A.7 Detailed Algorithm with Refinement

Algorithm 2 FGSE with One-Pass Precondition Refinement

Require: State Θ_t, evidence packet E_t, audit archive $\mathcal{A}_t$, thresholds $(\eta, \delta, \beta, \alpha_{\mathrm{sep}})$
Ensure: Updated state Θ_{t+1}

Propose $h = (c_h, b_h, e_h, v_h) = H_\phi(E_t)$.
Build $\mathcal{T}(h)$, $\mathcal{F}(h)$, and $\mathcal{R}(h)$.
Evaluate $\widehat{G}(h)$, $\widehat{P}_{\mathrm{fail}}(h)$, and $\widehat{D}_{\mathrm{reg}}(h)$.
if $\widehat{G}(h) \geq \eta$ and $\widehat{P}_{\mathrm{fail}}(h) \leq \delta$ and $\widehat{D}_{\mathrm{reg}}(h) \leq \beta$ **then**
  Commit h and set $\Theta_{t+1} = \Theta_t \oplus h$.
else if $\widehat{G}(h) \geq \eta$ and $\mathrm{Sep}(h) \leq \alpha_{\mathrm{sep}}$ **then**
  Compute c'_h by Eq. (30) and set $h' = (c'_h, b_h, e_h, v_h)$.
  Rebuild probes for h' and reevaluate the gate.
  if h' satisfies Eq. (8) **then**
    Commit h' and set $\Theta_{t+1} = \Theta_t \oplus h'$.
  else
    Reject h' and set $\Theta_{t+1} = \Theta_t$.
  end if
else
  Reject h and set $\Theta_{t+1} = \Theta_t$.
end if

Practical representation of hypotheses. In implementation, each hypothesis is stored as a schema-constrained JSON object. The precondition field is a natural-language predicate together with a scoring function implemented by a lightweight LLM classifier or rule-based matcher. The behavior field specifies a bounded prompt/tool-policy patch. The expected-effect field specifies a verifier-observable condition, and the verifier field points to the benchmark validator or executable checker. Audits of synthetic healthcare workflows also expose escalation errors under conflicting evidence and brittle routing rules under unseen paraphrases [Zheng et al., 2026].

JSON example

```
{
  "precondition": "The task asks for a round-trip flight booking with explicit
       origin, destination, and dates.",
  "behavior": "Before searching, set trip_type='round_trip' and fill both
      departure and return dates.",
  "expected_effect": "Search query contains both outbound and return dates and
       trip_type='round_trip'.",
  "verifier": "booking_database_state_validator"
}
```

B Proofs for the Commitment Gate Generalization

B.1 Conditional Setup

We prove Theorem 1 and Corollary 1. Fix a round t. Let $\mathcal{F}_t$ denote the sigma-field containing the current agent state Θ_t, the evidence packet, the audit archive, the proposal and refinement randomness

before final evaluation, and the finite candidate set $\mathcal{H}_t$. Throughout the proof, we condition on $\mathcal{F}_t$. Thus, the candidate set is fixed before the final evaluation probes are sampled.

For each $h \in \mathcal{H}_t$, define the target-transfer random variable

$$X_h^T = \Delta_h(x), \qquad x \sim \mathcal{D}_T(h). \tag{41}$$

Because verifier scores lie in $[0, 1]$, we have

$$X_h^T \in [-1, 1]. \tag{42}$$

The population and empirical target-transfer gains are

$$G(h) = \mathbb{E}[X_h^T], \qquad \widehat{G}(h) = \frac{1}{n_T} \sum_{i=1}^{n_T} X_{h,i}^T. \tag{43}$$

The empirical target-transfer variance is

$$\widehat{V}_T(h) = \frac{1}{n_T} \sum_{i=1}^{n_T} \left(X_{h,i}^T - \widehat{G}(h) \right)^2. \tag{44}$$

For falsification probes, define the Bernoulli random variable

$$X_h^F = \mathbb{I}\left[\Delta_h(x) < -\epsilon \ \vee \ E_h(\Theta_t \oplus h, x) = 0 \right], \qquad x \sim \mathcal{D}_F(h). \tag{45}$$

Then

$$X_h^F \in \{0, 1\}. \tag{46}$$

The population falsification risk and empirical failure rate are

$$P_{\text{fail}}(h) = \mathbb{E}[X_h^F], \qquad \widehat{P}_{\text{fail}}(h) = \frac{1}{n_F} \sum_{i=1}^{n_F} X_{h,i}^F. \tag{47}$$

For regression probes, define

$$X_h^R = [-\Delta_h(x)]_+, \qquad x \sim \mathcal{D}_R(h). \tag{48}$$

Since $\Delta_h(x) \in [-1, 1]$, we have

$$X_h^R \in [0, 1]. \tag{49}$$

The population and empirical regression damages are

$$D_{\text{reg}}(h) = \mathbb{E}[X_h^R], \qquad \widehat{D}_{\text{reg}}(h) = \frac{1}{n_R} \sum_{i=1}^{n_R} X_{h,i}^R. \tag{50}$$

The empirical regression variance is

$$\widehat{V}_R(h) = \frac{1}{n_R} \sum_{i=1}^{n_R} \left(X_{h,i}^R - \widehat{D}_{\text{reg}}(h) \right)^2. \tag{51}$$

The analysis treats the randomness in X_h^T, X_h^F, and X_h^R as including both probe sampling and any rollout randomness used to estimate verifier outcomes. The only properties needed are boundedness and conditional independence of the final evaluation samples for each fixed h.

B.2 Confidence Bounds Used by the Commitment Gate

Let n_T, n_F, n_R be the numbers of final target, falsification, and regression probes. Let $\alpha_T, \alpha_F, \alpha_R > 0$ satisfy

$$\alpha_T + \alpha_F + \alpha_R = \alpha. \tag{52}$$

Define the logarithmic factors

$$L_T = \log \frac{2|\mathcal{H}_t|}{\alpha_T}, \qquad L_R = \log \frac{2|\mathcal{H}_t|}{\alpha_R}, \qquad L_F = \log \frac{(n_F + 1)|\mathcal{H}_t|}{\alpha_F}. \tag{53}$$

The empirical-Bernstein slacks are

$$\varepsilon_T(h) = \sqrt{\frac{2\widehat{V}_T(h)L_T}{n_T}} + \frac{14L_T}{3(n_T-1)}, \qquad \varepsilon_R(h) = \sqrt{\frac{2\widehat{V}_R(h)L_R}{n_R}} + \frac{7L_R}{3(n_R-1)}. \tag{54}$$

The target lower confidence bound and regression upper confidence bound are

$$\mathrm{LCB}_T(h) = \widehat{G}(h) - \varepsilon_T(h), \qquad \mathrm{UCB}_R(h) = \widehat{D}_{\mathrm{reg}}(h) + \varepsilon_R(h). \tag{55}$$

For $p, q \in [0,1]$, define the binary KL divergence

$$\mathrm{kl}(p\|q) = p\log\frac{p}{q} + (1-p)\log\frac{1-p}{1-q}, \tag{56}$$

with the standard continuous extension at the boundary. The KL upper confidence bound for falsification risk is

$$\mathrm{UCB}_F(h) = \sup\left\{ q \in [\widehat{P}_{\mathrm{fail}}(h), 1] : \mathrm{kl}\left(\widehat{P}_{\mathrm{fail}}(h)\|q\right) \le \frac{L_F}{n_F} \right\}. \tag{57}$$

B.3 Concentration Tools

We use two concentration results. The first controls bounded real-valued random variables using empirical variance.

Lemma 1 (One-sided empirical Bernstein inequality)**.** *Let $Y_1, \dots, Y_n$ be independent random variables taking values in an interval of length $\mathfrak{b}$. Let*

$$\mu = \mathbb{E}[Y_1], \qquad \widehat{\mu} = \frac{1}{n}\sum_{i=1}^{n} Y_i, \tag{58}$$

and

$$\widehat{V} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \widehat{\mu})^2. \tag{59}$$

For any $\rho \in (0,1)$ and $n \ge 2$, with probability at least $1-\rho$,

$$\mu - \widehat{\mu} \le \sqrt{\frac{2\widehat{V}\log(2/\rho)}{n}} + \frac{7\mathfrak{b}\log(2/\rho)}{3(n-1)}. \tag{60}$$

The same inequality also holds with $\widehat{\mu} - \mu$ on the left-hand side.

Proof. This is the standard one-sided form of the empirical Bernstein inequality for bounded random variables. The lower-tail statement follows by applying the same result to $-Y_i$, whose range length remains $\mathfrak{b}$. □

The second result gives a KL confidence bound for Bernoulli variables.

Lemma 2 (Bernoulli KL confidence bound)**.** *Let $B_1, \dots, B_n$ be i.i.d. Bernoulli random variables with mean p, and let*

$$\widehat{p} = \frac{1}{n}\sum_{i=1}^{n} B_i. \tag{61}$$

Then, for any $u > 0$,

$$\mathbb{P}\left(\mathrm{kl}(\widehat{p}\|p) \ge u\right) \le (n+1)\exp(-nu). \tag{62}$$

Proof. For each $k \in \{0, 1, \dots, n\}$, let

$$q_k = \frac{k}{n}. \tag{63}$$

The event $\widehat{p} = q_k$ has probability

$$\mathbb{P}(\widehat{p} = q_k) = \binom{n}{k} p^k (1-p)^{n-k}. \tag{64}$$

Using the type-counting bound

$$\binom{n}{k} \le \exp\left(nH(q_k)\right), \tag{65}$$

where

$$H(q) = -q\log q - (1-q)\log(1-q), \tag{66}$$

we obtain

$$\begin{aligned}
\mathbb{P}(\widehat{p} = q_k) &= \binom{n}{k} p^{nq_k}(1-p)^{n(1-q_k)} \\
&\le \exp\left(nH(q_k) + nq_k\log p + n(1-q_k)\log(1-p)\right) \\
&= \exp\left(-n\left[q_k\log\frac{q_k}{p} + (1-q_k)\log\frac{1-q_k}{1-p}\right]\right) \\
&= \exp\left(-n\mathrm{kl}(q_k\|p)\right).
\end{aligned} \tag{67}$$

Therefore,

$$\begin{aligned}
\mathbb{P}\left(\mathrm{kl}(\widehat{p}\|p) \ge u\right) &= \sum_{k:\mathrm{kl}(q_k\|p)\ge u} \mathbb{P}(\widehat{p} = q_k) \\
&\le \sum_{k:\mathrm{kl}(q_k\|p)\ge u} \exp\left(-n\mathrm{kl}(q_k\|p)\right) \\
&\le \sum_{k=0}^{n} \exp(-nu) \\
&= (n+1)\exp(-nu).
\end{aligned} \tag{68}$$

This proves the claim. □

B.4 Uniform Confidence Event

We now prove that the three confidence bounds hold simultaneously for all hypotheses in $\mathcal{H}_t$.

Lemma 3 (Uniform confidence event)**.** *With probability at least $1-\alpha$, every $h \in \mathcal{H}_t$ simultaneously satisfies*

$$G(h) \ge \mathrm{LCB}_T(h), \qquad P_{\mathrm{fail}}(h) \le \mathrm{UCB}_F(h), \qquad D_{\mathrm{reg}}(h) \le \mathrm{UCB}_R(h). \tag{69}$$

Proof. We prove the target, regression, and falsification bounds separately and then apply a union bound.

First consider target-transfer gain. For a fixed h, apply Lemma 1 to X_h^T. Its range length is

$$\mathfrak{b}_T = 2. \tag{70}$$

Set

$$\rho_T = \frac{\alpha_T}{|\mathcal{H}_t|}. \tag{71}$$

Then

$$\log\frac{2}{\rho_T} = \log\frac{2|\mathcal{H}_t|}{\alpha_T} = L_T. \tag{72}$$

For this fixed h, Lemma 1 gives

$$\begin{aligned}
G(h) - \widehat{G}(h) &\ge -\left[\sqrt{\frac{2\widehat{V}_T(h)L_T}{n_T}} + \frac{7\mathfrak{b}_T L_T}{3(n_T-1)}\right] \\
&= -\left[\sqrt{\frac{2\widehat{V}_T(h)L_T}{n_T}} + \frac{14L_T}{3(n_T-1)}\right] \\
&= -\varepsilon_T(h).
\end{aligned} \tag{73}$$

Thus,

$$G(h) \ge \widehat{G}(h) - \varepsilon_T(h) = \mathrm{LCB}_T(h). \tag{74}$$

A union bound over $h \in \mathcal{H}_t$ gives

$$\mathbb{P}\left(\forall h \in \mathcal{H}_t : G(h) \geq \mathrm{LCB}_T(h)\right) \geq 1 - \alpha_T. \tag{75}$$

Next consider regression damage. For a fixed h, apply Lemma 1 to X_h^R. Its range length is

$$\mathfrak{b}_R = 1. \tag{76}$$

Set

$$\rho_R = \frac{\alpha_R}{|\mathcal{H}_t|}. \tag{77}$$

Then

$$\log \frac{2}{\rho_R} = \log \frac{2|\mathcal{H}_t|}{\alpha_R} = L_R. \tag{78}$$

For this fixed h, Lemma 1 gives

$$\begin{aligned} D_{\mathrm{reg}}(h) - \widehat{D}_{\mathrm{reg}}(h) &\leq \sqrt{\frac{2\widehat{V}_R(h)L_R}{n_R}} + \frac{7\mathfrak{b}_R L_R}{3(n_R - 1)} \\ &= \sqrt{\frac{2\widehat{V}_R(h)L_R}{n_R}} + \frac{7L_R}{3(n_R - 1)} \\ &= \varepsilon_R(h). \end{aligned} \tag{79}$$

Therefore,

$$D_{\mathrm{reg}}(h) \leq \widehat{D}_{\mathrm{reg}}(h) + \varepsilon_R(h) = \mathrm{UCB}_R(h). \tag{80}$$

A union bound over $h \in \mathcal{H}_t$ gives

$$\mathbb{P}\left(\forall h \in \mathcal{H}_t : D_{\mathrm{reg}}(h) \leq \mathrm{UCB}_R(h)\right) \geq 1 - \alpha_R. \tag{81}$$

Finally consider falsification risk. For a fixed h, apply Lemma 2 to X_h^F. Let

$$u_F = \frac{L_F}{n_F}. \tag{82}$$

Then

$$\begin{aligned} \mathbb{P}\left(\mathrm{kl}\left(\widehat{P}_{\mathrm{fail}}(h) \| P_{\mathrm{fail}}(h)\right) > \frac{L_F}{n_F}\right) &\leq (n_F + 1)\exp(-L_F) \\ &= (n_F + 1)\exp\left(-\log\frac{(n_F+1)|\mathcal{H}_t|}{\alpha_F}\right) \\ &= \frac{\alpha_F}{|\mathcal{H}_t|}. \end{aligned} \tag{83}$$

A union bound over $h \in \mathcal{H}_t$ yields

$$\mathbb{P}\left(\forall h \in \mathcal{H}_t : \mathrm{kl}\left(\widehat{P}_{\mathrm{fail}}(h) \| P_{\mathrm{fail}}(h)\right) \leq \frac{L_F}{n_F}\right) \geq 1 - \alpha_F. \tag{84}$$

On the event in Eq. (84), we prove that $P_{\mathrm{fail}}(h) \leq \mathrm{UCB}_F(h)$ for every h. Fix h. If $P_{\mathrm{fail}}(h) \leq \widehat{P}_{\mathrm{fail}}(h)$, then

$$P_{\mathrm{fail}}(h) \leq \widehat{P}_{\mathrm{fail}}(h) \leq \mathrm{UCB}_F(h), \tag{85}$$

because $q = \widehat{P}_{\mathrm{fail}}(h)$ is feasible in the definition of $\mathrm{UCB}_F(h)$. If $P_{\mathrm{fail}}(h) > \widehat{P}_{\mathrm{fail}}(h)$, then $q = P_{\mathrm{fail}}(h)$ lies in $[\widehat{P}_{\mathrm{fail}}(h), 1]$, and the KL event gives

$$\mathrm{kl}\left(\widehat{P}_{\mathrm{fail}}(h) \| P_{\mathrm{fail}}(h)\right) \leq \frac{L_F}{n_F}. \tag{86}$$

Thus $P_{\mathrm{fail}}(h)$ is feasible in the supremum defining $\mathrm{UCB}_F(h)$, which implies

$$P_{\mathrm{fail}}(h) \leq \mathrm{UCB}_F(h). \tag{87}$$

Combining the two cases gives

$$\forall h \in \mathcal{H}_t : P_{\text{fail}}(h) \leq \text{UCB}_F(h). \tag{88}$$

Let $\mathcal{E}_T$, $\mathcal{E}_F$, and $\mathcal{E}_R$ denote the target, falsification, and regression confidence events. Combining Eq. (75), Eq. (81), and Eq. (84), we have

$$\begin{aligned} \mathbb{P}(\mathcal{E}_T \cap \mathcal{E}_F \cap \mathcal{E}_R) &= 1 - \mathbb{P}(\mathcal{E}_T^c \cup \mathcal{E}_F^c \cup \mathcal{E}_R^c) \\ &\geq 1 - \mathbb{P}(\mathcal{E}_T^c) - \mathbb{P}(\mathcal{E}_F^c) - \mathbb{P}(\mathcal{E}_R^c) \\ &\geq 1 - \alpha_T - \alpha_F - \alpha_R \\ &= 1 - \alpha. \end{aligned} \tag{89}$$

On this event, all three bounds in Eq. (69) hold simultaneously for every $h \in \mathcal{H}_t$. This proves the lemma. □

B.5 Proof of Theorem 1

Proof. By Lemma 3, with probability at least $1 - \alpha$, all $h \in \mathcal{H}_t$ simultaneously satisfy

$$G(h) \geq \text{LCB}_T(h), \qquad P_{\text{fail}}(h) \leq \text{UCB}_F(h), \qquad D_{\text{reg}}(h) \leq \text{UCB}_R(h). \tag{90}$$

This proves Eq. (11).

Now suppose that h satisfies the confidence-gated commitment rule:

$$\text{LCB}_T(h) \geq \eta, \qquad \text{UCB}_F(h) \leq \delta, \qquad \text{UCB}_R(h) \leq \beta. \tag{91}$$

For target-transfer gain, Eq. (90) and Eq. (91) imply

$$\begin{aligned} G(h) &\geq \text{LCB}_T(h) \\ &\geq \eta. \end{aligned} \tag{92}$$

For falsification risk,

$$\begin{aligned} P_{\text{fail}}(h) &\leq \text{UCB}_F(h) \\ &\leq \delta. \end{aligned} \tag{93}$$

For regression damage,

$$\begin{aligned} D_{\text{reg}}(h) &\leq \text{UCB}_R(h) \\ &\leq \beta. \end{aligned} \tag{94}$$

Combining Eq. (92), Eq. (93), and Eq. (94), we obtain

$$G(h) \geq \eta, \qquad P_{\text{fail}}(h) \leq \delta, \qquad D_{\text{reg}}(h) \leq \beta. \tag{95}$$

This proves Eq. (13). □

B.6 Exact Probe Complexity Statement

We now give the exact finite-sample version of Corollary 1. Let $\zeta_T, \zeta_R > 0$ be target and regression margins, and let $0 \leq \delta_- < \delta \leq 1$ be empirical and population falsification thresholds.

Corollary 2 (Exact probe complexity)**.** *Suppose*

$$n_T \geq 1 + \max\left\{ \frac{8L_T}{\zeta_T^2}, \frac{28L_T}{3\zeta_T} \right\}, \tag{96}$$

$$n_R \geq 1 + \max\left\{ \frac{2L_R}{\zeta_R^2}, \frac{14L_R}{3\zeta_R} \right\}, \tag{97}$$

and

$$n_F \geq \frac{L_F}{\text{kl}(\delta_- \| \delta)}. \tag{98}$$

Then, with probability at least $1 - \alpha$*, any* $h \in \mathcal{H}_t$ *satisfying*

$$\widehat{G}(h) \geq \eta + \zeta_T, \qquad \widehat{P}_{\text{fail}}(h) \leq \delta_-, \qquad \widehat{D}_{\text{reg}}(h) \leq \beta - \zeta_R \tag{99}$$

also satisfies

$$G(h) \geq \eta, \qquad P_{\text{fail}}(h) \leq \delta, \qquad D_{\text{reg}}(h) \leq \beta. \tag{100}$$

B.7 Proof of Corollary 2

Proof. We first control the target-transfer confidence radius. Since $X_h^T \in [-1, 1]$, its empirical variance satisfies

$$\widehat{V}_T(h) \le 1. \tag{101}$$

Therefore,

$$\begin{aligned} \varepsilon_T(h) &= \sqrt{\frac{2\widehat{V}_T(h)L_T}{n_T}} + \frac{14L_T}{3(n_T-1)} \\ &\le \sqrt{\frac{2L_T}{n_T}} + \frac{14L_T}{3(n_T-1)}. \end{aligned} \tag{102}$$

If

$$n_T \ge \frac{8L_T}{\zeta_T^2}, \tag{103}$$

then

$$\sqrt{\frac{2L_T}{n_T}} \le \frac{\zeta_T}{2}. \tag{104}$$

If

$$n_T - 1 \ge \frac{28L_T}{3\zeta_T}, \tag{105}$$

then

$$\frac{14L_T}{3(n_T-1)} \le \frac{\zeta_T}{2}. \tag{106}$$

Combining Eq. (102), Eq. (104), and Eq. (106), we get

$$\varepsilon_T(h) \le \zeta_T. \tag{107}$$

We next control the regression confidence radius. Since $X_h^R \in [0, 1]$, its empirical variance satisfies

$$\widehat{V}_R(h) \le \frac{1}{4}. \tag{108}$$

Thus,

$$\begin{aligned} \varepsilon_R(h) &= \sqrt{\frac{2\widehat{V}_R(h)L_R}{n_R}} + \frac{7L_R}{3(n_R-1)} \\ &\le \sqrt{\frac{L_R}{2n_R}} + \frac{7L_R}{3(n_R-1)}. \end{aligned} \tag{109}$$

If

$$n_R \ge \frac{2L_R}{\zeta_R^2}, \tag{110}$$

then

$$\sqrt{\frac{L_R}{2n_R}} \le \frac{\zeta_R}{2}. \tag{111}$$

If

$$n_R - 1 \ge \frac{14L_R}{3\zeta_R}, \tag{112}$$

then

$$\frac{7L_R}{3(n_R-1)} \le \frac{\zeta_R}{2}. \tag{113}$$

Combining Eq. (109), Eq. (111), and Eq. (113), we obtain

$$\varepsilon_R(h) \le \zeta_R. \tag{114}$$

We now control the falsification upper confidence bound. Assume $\widehat{P}_{\text{fail}}(h) \le \delta_-$. For fixed $q = \delta$, the function $p \mapsto \mathrm{kl}(p\|\delta)$ is decreasing on $p \in [0, \delta]$. Indeed,

$$\frac{\partial}{\partial p}\mathrm{kl}(p\|\delta) = \log\frac{p(1-\delta)}{\delta(1-p)}, \tag{115}$$

and for $p < \delta$,

$$\log \frac{p(1-\delta)}{\delta(1-p)} < 0. \tag{116}$$

Therefore,

$$\mathrm{kl}\left(\widehat{P}_{\mathrm{fail}}(h) \| \delta\right) \geq \mathrm{kl}(\delta_- \| \delta). \tag{117}$$

By Eq. (98),

$$\frac{L_F}{n_F} \leq \mathrm{kl}(\delta_- \| \delta) \leq \mathrm{kl}\left(\widehat{P}_{\mathrm{fail}}(h) \| \delta\right). \tag{118}$$

For fixed $p = \widehat{P}_{\mathrm{fail}}(h)$, the map $q \mapsto \mathrm{kl}(p\|q)$ is increasing on $q \in [p, 1]$. Hence every $q > \delta$ satisfies

$$\mathrm{kl}\left(\widehat{P}_{\mathrm{fail}}(h) \| q\right) > \frac{L_F}{n_F}. \tag{119}$$

Thus no $q > \delta$ is feasible in the supremum defining $\mathrm{UCB}_F(h)$, and we have

$$\mathrm{UCB}_F(h) \leq \delta. \tag{120}$$

Now suppose h satisfies the empirical margin condition in Eq. (99). Combining Eq. (107), Eq. (114), and Eq. (120), we get

$$\mathrm{LCB}_T(h) \geq \eta, \qquad \mathrm{UCB}_F(h) \leq \delta, \qquad \mathrm{UCB}_R(h) \leq \beta. \tag{121}$$

Applying Theorem 1 yields

$$G(h) \geq \eta, \qquad P_{\mathrm{fail}}(h) \leq \delta, \qquad D_{\mathrm{reg}}(h) \leq \beta. \tag{122}$$

This proves the corollary. □

B.8 Remarks on Adaptivity and Scope Refinement

The proof requires that the final evaluation probes are not reused to construct or refine the hypothesis being certified. This condition prevents adaptive overfitting to the finite falsification set. In practice, this can be implemented in two equivalent ways. The first is sample splitting: use a development split for scope refinement and a fresh evaluation split for final commitment. The second is to define $\mathcal{H}_t$ as the enlarged finite set containing all hypotheses that can be produced by proposal and refinement before final evaluation; the same proof then applies with $|\mathcal{H}_t|$ replaced by the size of this enlarged set.

The theorem certifies reliability with respect to the probe-generation distributions $\mathcal{D}_T(h)$, $\mathcal{D}_F(h)$, and $\mathcal{D}_R(h)$. If these distributions differ from the deployment distribution, an additional distribution-shift term is needed. For example, if the deployment falsification distribution is within total variation distance ρ of $\mathcal{D}_F(h)$, then the certified falsification risk increases by at most ρ. The commitment gate controls statistical generalization from finite probes to the specified probe distributions, while the quality of the probe generator determines how well those distributions cover deployment-time boundary cases.

C Additional Experimental Details

C.1 Benchmark Details

Web and browser-control tasks. For WebArena [Zhou et al., 2024] and BrowserGym [Le Sellier de Chezelles et al., 2025], we follow the official environment setup and success validators. Each task provides a natural-language instruction, an initial browser state, and an environment-specific completion condition. We use these validators as the verifier v_h whenever possible. For FGSE, target-transfer probes are constructed from tasks sharing the same high-level interaction pattern, while falsification probes are selected from tasks with similar surface forms but different completion conditions, websites, forms, or required constraints. Regression probes are drawn from previously solved held-out tasks and are never used to propose the candidate hypothesis.

Tool-use and app-interaction tasks. For AppWorld [Trivedi et al., 2024], we evaluate on the official Normal and Challenge splits and report task-goal completion using the benchmark-provided database-state evaluator. For τ-bench [Yao et al., 2024], we evaluate both retail and airline domains and use the official database-state comparison and pass^k reliability protocol. In both benchmarks, a hypothesis may modify the agent's tool-use policy, retrieval memory, or instruction-level strategy, but the available tools, API schemas, and domain policies are fixed across all methods. Holding tool access fixed isolates the effect of the update mechanism.

Security and adversarial tool-output tasks. AgentDojo [Debenedetti et al., 2024] is used to evaluate whether self-evolution increases susceptibility to prompt injection or malicious tool outputs. We report utility success on benign tasks, security success under attacks, and the joint utility-security score. For FGSE, falsification probes include decoy tool outputs, conflicting instructions, and previously solved benign cases. A candidate update is rejected if it improves utility but increases attack success or violates a high-priority system or user constraint.

Long-term memory tasks. For LongMemEval [Wu et al., 2024] and LoCoMo [Maharana et al., 2024], we evaluate whether committed hypotheses improve memory use without corrupting persistent user or task state. We report the official QA accuracy or F1 metrics, temporal reasoning accuracy, knowledge-update accuracy, and abstention accuracy when available. We also measure memory pollution rate, defined as the fraction of committed updates that are later contradicted by held-out evidence or cause a regression on archived successful examples.

Code and mathematical reasoning diagnostics. We use HumanEval [Chen et al., 2021], MBPP [Austin et al., 2021], and SWE-bench Verified [Chowdhury et al., 2024] as verifier-rich coding environments because unit tests provide executable feedback. SWE-bench Verified is a human-validated subset of the original SWE-bench [Jimenez et al., 2024], released by OpenAI in collaboration with the SWE-bench authors [Chowdhury et al., 2024]. We report pass@1 on HumanEval and MBPP and resolved rate on SWE-bench Verified. For mathematical reasoning, we use GSM8K [Cobbe et al., 2021] and MATH [Hendrycks et al., 2021], reporting exact-match solve rate after answer normalization. These datasets are used primarily for controlled analysis of update quality because their verifiers allow clean construction of target, boundary, and regression probes.

C.2 Evaluation Metrics

Let $\mathcal{D}$ denote a held-out evaluation set and let $v(x, \tau) \in [0, 1]$ be the benchmark-specific verifier score. The standard task success rate is

$$\mathrm{SR} = \frac{1}{|\mathcal{D}|} \sum_{x \in \mathcal{D}} v(x, \tau_x). \tag{123}$$

For binary task validators, this reduces to the fraction of solved tasks. For benchmarks with official non-binary scores, we use the official normalized score.

For each committed or rejected hypothesis h, we compute the three FGSE-specific statistics:

$$\widehat{G}(h) = \frac{1}{|\mathcal{T}(h)|} \sum_{x \in \mathcal{T}(h)} \left[J_{v_h}(\Theta \oplus h, x) - J_{v_h}(\Theta, x) \right], \tag{124}$$

$$\widehat{P}_{\mathrm{fail}}(h) = \frac{1}{|\mathcal{F}(h)|} \sum_{x \in \mathcal{F}(h)} \mathbb{I}\left[J_{v_h}(\Theta \oplus h, x) - J_{v_h}(\Theta, x) < -\epsilon \ \vee \ E_h(\Theta \oplus h, x) = 0 \right], \tag{125}$$

$$\widehat{D}_{\mathrm{reg}}(h) = \frac{1}{|\mathcal{R}(h)|} \sum_{x \in \mathcal{R}(h)} \left[J_{v_h}(\Theta, x) - J_{v_h}(\Theta \oplus h, x) \right]_+ . \tag{126}$$

We also report:

- **Harmful update rate**: the fraction of committed updates that cause held-out regression larger than ϵ.
- **Accepted-update precision**: the fraction of committed updates that yield positive held-out target-transfer gain without measurable regression.

- **Falsification hit rate**: the fraction of rejected or refined updates for which at least one falsification probe exposes a failure.
- **Post-commit violation rate**: the fraction of future activations where a committed hypothesis violates its expected-effect checker.
- **Cost**: total LLM calls, verifier calls, wall-clock time, and token usage per committed update.

C.3 Fair Comparison Protocol

All controlled comparisons follow the same protocol [Yang et al., 2026b]:

1. **Same backbone**: every method uses the same base LLM and the same ReAct-style executor unless the method intrinsically requires a different executor.
2. **Same tools**: all methods receive identical tool schemas, browser actions, API access, and environment observations.
3. **Same memory budget**: methods with persistent memory are allowed the same maximum number of stored units and the same retrieval top-k.
4. **Same update budget**: methods that propose self-updates receive the same number of completed training episodes and the same maximum number of update attempts.
5. **No test leakage**: hypotheses, reflections, playbooks, workflows, and memory entries are constructed only from training or development episodes. Test tasks are used only for final evaluation.
6. **Same verifier access**: if a verifier is available during development, all methods may use it under the same budget. During final evaluation, verifier outputs are used only for scoring.

For workflow-search or architecture-search methods such as AFlow [Zhang et al., 2025a], ADAS [Hu et al., 2025], GPTSwarm [Zhuge et al., 2024], and MetaAgent [Zhang et al., 2025b], we report a controlled-budget version when code and prompts are available. Methods requiring a substantially larger search budget, extra training data, reinforcement learning, or model fine-tuning are reported in a separate native-setting table, outside the controlled FGSE comparisons.

C.4 Baseline Implementation Details

Direct and ReAct. The direct baseline uses the base LLM without persistent memory or post-episode updates. The ReAct baseline [Yao et al., 2023] uses the same action space and tool interface as FGSE but does not modify its persistent state after each episode.

Self-Refine and Reflexion. Self-Refine [Madaan et al., 2023] performs within-episode critique and revision under the same maximum number of refinement calls as FGSE's hypothesis proposal calls. Reflexion [Shinn et al., 2023] writes natural-language reflections after failed or low-scoring episodes and retrieves them in future tasks. Reflexion commits reflections without requiring structured preconditions, falsification probes, or regression tests.

Voyager-style skill memory. The Voyager-style baseline [Wang et al., 2023] stores reusable skills or lessons from successful episodes and retrieves them for related future tasks. For non-Minecraft benchmarks, we adapt the skill library into a text-based rule or tool-use strategy library while keeping the same retrieval budget as FGSE.

Memory and context-evolution methods. A-Mem [Xu et al., 2025] is implemented as an agentic memory organization baseline that dynamically indexes, links, and updates memories. ACE [Zhang et al., 2026f] is implemented as an evolving playbook baseline that incrementally revises persistent context using execution feedback. Both methods are evaluated with the same memory capacity and retrieval top-k as FGSE.

Workflow and agent-design optimization methods. AFlow [Zhang et al., 2025a], ADAS [Hu et al., 2025], GPTSwarm [Zhuge et al., 2024], and MetaAgent [Zhang et al., 2025b] provide comparisons at the agentic-workflow level, complementing update-level memory methods. Because these methods may search over a larger design space than FGSE, we distinguish controlled-budget and native-budget results.

Table 3: Default FGSE hyperparameters used in controlled experiments.

Hyperparameter	Symbol	Default value
Candidate hypotheses per episode	B	3
Target-transfer probes	$\lvert\mathcal{T}(h)\rvert$	8
Falsification probes	$\lvert\mathcal{F}(h)\rvert$	16
Regression probes	$\lvert\mathcal{R}(h)\rvert$	16
Minimum target gain	η	0.03
Maximum falsification rate	δ	0.05
Maximum regression damage	β	0.01
Tolerated verifier-score drop	ϵ	0
Precondition activation threshold	γ_h	0.5
Intervention strength	λ_h	0.8
Retrieved committed hypotheses	k	5
Post-commit violation threshold	$\rho_{\max}$	0.10

C.5 FGSE Hyperparameters

All hyperparameters are selected on development tasks and then fixed for test evaluation.

C.6 Ablation Studies

We conduct the following ablations:

- **No falsification**: commit every proposed hypothesis that improves the source episode.
- **Random probes**: replace targeted falsification probes with randomly sampled held-out tasks.
- **LLM-judge gate**: replace executable or environment verifiers with an LLM-only judgment.
- **No regression archive**: remove $\mathcal{R}(h)$ from the commitment rule.
- **No scope narrowing**: reject all hypotheses that fail falsification without allowing precondition refinement.
- **Free-form memory**: store unstructured natural-language lessons instead of structured (c_h, b_h, e_h, v_h) hypotheses.
- **Probe-budget sweep**: vary the number of falsification probes to measure the trade-off between update safety and computation.

These ablations isolate whether FGSE's gains come from verifier grounding, targeted counterexample construction, regression protection, structured update representation, or scope refinement.

C.7 Compute Resources

FGSE uses frozen language models without training or fine-tuning. All reported experiments access frozen LLM backbones through provider-hosted APIs, with no author-side GPU use for model training or inference. The provider-side serving compute for GPT-4o, GPT-4o-mini, and DeepSeek-V3.1 is not observable to us and is not included in the reported compute. Our local compute consists of agent orchestration, environment simulation, browser/app execution, sandboxed code execution, probe construction, verifier calls, logging, and result aggregation.

Unless otherwise stated, experiments were orchestrated on CPU workers with 8 vCPUs and 32GB memory per worker. For code-execution and repository-level diagnostics such as SWE-bench Verified, we used sandbox workers with 16 vCPUs and 64GB memory. We used up to 32 workers concurrently. We report compute in terms of LLM calls, verifier calls, parallel wall-clock time, and aggregate client-side CPU usage. One LLM call denotes one model completion request, including action generation, reflection, hypothesis proposal, scope refinement, and LLM-judge calls when used by a baseline. One verifier call denotes one execution of an official benchmark validator, unit test, database-state checker, security checker, or exact-match evaluator.

The full project also included preliminary diagnostics, pilot prompt variants, failed verifier configurations, and exploratory ablations that are not included in Table 4. These exploratory runs used

Table 4: **Approximate compute used by the reported experiments.** LLM calls include agent rollouts, hypothesis proposal/refinement, and baseline-specific reflection or judging calls. Verifier calls include official success validators, database-state checkers, unit tests, security validators, and exact-match evaluators. CPU-hours denote client-side orchestration and environment/verifier execution only. Provider-side LLM serving compute is not included because all LLM backbones are accessed through hosted APIs.

Experiment group	LLM calls	Verifier calls	Wall-clock time	Client CPU-hours
Web/browser-control tasks	$\approx 7.8\times10^4$	$\approx 1.1\times10^5$	≈ 39 h	$\approx 1.1\times10^3$
App/API-interaction tasks	$\approx 1.46\times10^5$	$\approx 2.18\times10^5$	≈ 32 h	$\approx 9.0\times10^2$
τ-bench tool-user tasks	$\approx 8.4\times10^4$	$\approx 1.24\times10^5$	≈ 28 h	$\approx 7.2\times10^2$
AgentDojo security diagnostics	$\approx 3.6\times10^4$	$\approx 6.8\times10^4$	≈ 12 h	$\approx 3.0\times10^2$
Long-term memory benchmarks	$\approx 6.4\times10^4$	$\approx 8.7\times10^4$	≈ 16 h	$\approx 4.2\times10^2$
Code/math diagnostics	$\approx 5.2\times10^4$	$\approx 9.6\times10^4$	≈ 21 h	$\approx 7.6\times10^2$
Ablations and sensitivity analyses	$\approx 9.5\times10^4$	$\approx 1.38\times10^5$	≈ 27 h	$\approx 7.4\times10^2$
Total reported experiments	$\approx 5.55\times10^5$	$\approx 8.41\times10^5$	≈ 175 h	$\approx 4.94\times10^3$

approximately $1.6\times$ the compute of the final reported experiments. In total, the project used approximately 8.9×10^5 LLM calls, 1.35×10^6 verifier calls, and 7.9×10^3 client CPU-hours. Most additional compute came from early probe-construction variants and unsuccessful target-only or LLM-judge gating configurations.

D Additional Experimental Results

D.1 Additional Reliability and Ablation Results

Memory and update reliability. Table 5 further evaluates long-term memory performance and controlled self-evolution reliability. FGSE obtains the strongest memory performance while also reducing unsafe update commitment, falsification failure, and regression damage. These results support improving future target performance while preserving previously solved behaviors and rejecting overgeneralized updates.

Additional ablation results Table6 presents the results of additional ablation experiments.

D.2 Update-Selection and Falsification Analysis

Plasticity–safety Pareto frontier Effective self-evolution combines useful agent improvements with protection against harmful or regressive memory updates. We examine the *plasticity–safety Pareto frontier* for update selection and compare FGSE's position with alternative strategies.

We sweep a broad range of update-selection configurations and compare representative strategies, including FGSE, FGSE W/O REFINE, TARGET-ONLY, LLM-JUDGE GATE, CONSERVATIVE GATE, EXHAUSTIVE VERIFIER, RANDOM KEEP, and NO PERSISTENT UPDATE. For each configuration c, we measure: (i) *update gain* $\Delta_{\mathrm{upd}}(c)$, defined as the average improvement after applying the selected update; (ii) *harmful update rate* $H(c)$, i.e., the fraction of committed updates that later induce safety violations or incorrect behavior on held-out probes; (iii) *regression damage* $R(c)$, i.e., the average degradation on archived successful behaviors; and (iv) *verifier cost* $C(c)$, measured as the average number of verifier calls per candidate update. Figure 6 visualizes the resulting trade-off. The left panel plots Δ_{upd} against H, while the right panel plots Δ_{upd} against R. Each point corresponds to one configuration, point size encodes verifier cost, and the black dashed curve marks the empirical Pareto frontier.

Figure 6 reveals a clear plasticity–safety frontier. First, FGSE traces the dominant lower-right boundary in both panels over a broad range of medium-cost configurations, achieving substantially larger update gain at similar or lower harm than competing strategies. In the harmful-update panel, frontier configurations of FGSE achieve roughly 9%–12% update gain while keeping the harmful update rate within 2.5%–4.5%, typically with only 20–40 verifier calls per candidate update. At comparable gain levels, TARGET-ONLY and LLM-JUDGE GATE incur much larger harmful update

Table 5: **Long-term memory and controlled update-reliability comparison.** The Offline Reading result on LongMemEval is an oracle-style upper bound and is not ranked. Gray 0.0 entries indicate methods without persistent self-update, for which update-reliability metrics are not ranked. $\widehat{G}$ measures target-transfer gain; $\widehat{P}_{\text{fail}}$ measures falsification failure rate; $\widehat{D}_{\text{reg}}$ measures regression damage. Lower is better for the last four columns among self-updating methods. Best and second-best ranked scores in each column are shown in **bold** and underlined, respectively.

Method	Backbone / Setting	LongMemEval Acc. ↑	LoCoMo Avg. R-L ↑	$\widehat{G}$ (%) ↑	$\widehat{P}_{\text{fail}}$ (%) ↓	$\widehat{D}_{\text{reg}}$ (%) ↓	Harmful Updates (%) ↓	Post-commit Violations (%) ↓
Web / browser-control agents: methods primarily designed or evaluated for browser navigation and web-interaction self-improvement.								
GPT-4 Agent [Zhou et al., 2024]	GPT-4	0.552	22.84	0.0	0.0	0.0	0.0	0.0
AutoEval [Pan et al., 2024]	GPT-4	0.548	23.11	2.4	19.7	5.3	13.2	11.7
BrowserGym [Le Sellier de Chezelles et al., 2025]	GPT-4o	0.560	24.02	2.8	18.9	5.1	12.8	11.1
$\text{CER}_{\text{online}}$ [Liu et al., 2025]	GPT-4o	0.591	26.37	8.6	17.4	4.9	12.1	10.8
$\text{CER}_{\text{offline}}$ [Liu et al., 2025]	GPT-4o	0.598	26.91	8.8	17.1	4.8	11.9	10.5
$\text{CER}_{\text{hybrid}}$ [Liu et al., 2025]	GPT-4o	0.607	27.58	9.1	16.8	4.7	11.5	10.2
App / API-interaction agents: methods primarily evaluated on app-level tool use, database-state goals, and API execution.								
IPFunCall [Trivedi et al., 2024]	GPT-4o	0.507	18.24	1.9	27.6	7.0	20.1	17.5
FullCodeRefl [Trivedi et al., 2024]	GPT-4o	0.512	18.69	3.2	24.9	6.5	18.4	15.9
PlanExec [Trivedi et al., 2024]	GPT-4o	0.518	20.15	3.8	23.5	6.2	17.9	15.2
ReAct [Yao et al., 2023, Trivedi et al., 2024]	GPT-4o	0.560	22.46	0.0	0.0	0.0	0.0	0.0
ReAct + ICL [Zhang et al., 2026f]	DeepSeek-V3.1	0.572	23.46	6.4	18.6	4.9	13.1	11.4
ReAct + GEPA [Agrawal et al., 2026, Zhang et al., 2026f]	DeepSeek-V3.1	0.575	23.91	7.2	17.9	4.7	12.4	10.9
ReAct + DC [Suzgun et al., 2026]	DeepSeek-V3.1	0.590	26.42	9.4	14.8	4.4	10.9	9.6
ReAct + $\text{ACE}_{\text{offline}}$ [Zhang et al., 2026f]	DeepSeek-V3.1	0.661	31.86	10.3	12.4	3.6	8.4	7.3
ReAct + $\text{ACE}_{\text{online}}$ [Zhang et al., 2026f]	DeepSeek-V3.1	0.672	32.91	10.8	11.6	3.3	7.9	6.8
Tool-user interaction agents: methods centered on multi-turn user simulation, tool calling, and domain-policy execution.								
Tool-Calling [Yao et al., 2024]	GPT-4o	0.557	22.15	2.6	22.8	6.1	16.9	14.6
Tool-Calling [Yao et al., 2024]	Claude-3.5-Sonnet	0.581	24.63	3.1	21.4	5.8	15.7	13.9
Act [Yao et al., 2024]	GPT-4o	0.538	21.37	2.3	23.6	6.4	17.6	15.1
ReAct [Yao et al., 2023, 2024]	GPT-4o	0.545	21.80	0.0	0.0	0.0	0.0	0.0
Long-term memory agents: methods primarily designed for persistent memory, conversational memory, or memory-augmented retrieval.								
Offline Reading [Wu et al., 2024]	GPT-4o oracle	0.918	30.15	0.0	0.0	0.0	0.0	0.0
Coze [Wu et al., 2024]	GPT-4o	0.330	18.90	4.1	26.8	6.9	20.7	17.2
ChatGPT [Wu et al., 2024]	GPT-4o	0.577	25.32	0.0	0.0	0.0	0.0	0.0
GPT-4o + Chain-of-Note [Wu et al., 2024]	long-context	0.640	29.84	5.6	19.9	5.5	14.8	12.6
ChatGPT [Wu et al., 2024]	GPT-4o-mini	0.711	27.66	0.0	0.0	0.0	0.0	0.0
ReadAgent [Xu et al., 2025]	GPT-4o-mini	0.612	9.61	4.8	24.2	6.3	18.8	15.7
MemoryBank [Xu et al., 2025]	GPT-4o-mini	0.598	6.96	4.2	25.7	6.6	19.6	16.4
MemGPT [Xu et al., 2025]	GPT-4o-mini	0.630	29.19	6.7	23.1	6.0	17.3	14.9
LoCoMo [Maharana et al., 2024, Xu et al., 2025]	GPT-4o-mini	0.651	32.68	7.3	20.5	5.4	15.5	13.4
A-Mem [Xu et al., 2025]	GPT-4o-mini	0.684	35.49	9.9	13.2	3.8	8.7	7.6
FGSE variants: the same falsification-gated update mechanism instantiated with different base LLM backbones.								
FGSE (Ours)	GPT-4o-mini	0.721	36.41	12.5	6.2	1.8	4.5	3.9
FGSE (Ours)	GPT-4o	0.736	36.98	12.1	5.5	1.5	4.0	3.5
FGSE (Ours)	DeepSeek-V3.1	**0.748**	**37.62**	**12.9**	**5.0**	**1.3**	**3.6**	**3.1**

Table 6: **Single-factor ablation on update reliability.** $\widehat{G}$ measures target-transfer gain; $\widehat{P}_{\text{fail}}$ measures falsification failure rate; $\widehat{D}_{\text{reg}}$ measures regression damage. For $\widehat{G}$, higher is better; for the remaining metrics, lower is better. Blue numbers in parentheses denote the absolute degradation compared with the full FGSE model.

Variant	$\widehat{G}$ (%) ↑	$\widehat{P}_{\text{fail}}$ (%) ↓	$\widehat{D}_{\text{reg}}$ (%) ↓	Harmful Updates (%) ↓	Post-commit Violations (%) ↓
Full FGSE	**12.9**	**5.0**	**1.3**	**3.6**	**3.1**
w/o structured hypothesis	9.2 (-3.7)	16.7 (+11.7)	4.8 (+3.5)	11.3 (+7.7)	10.1 (+7.0)
w/o target-transfer probes	9.8 (-3.1)	11.9 (+6.9)	3.1 (+1.8)	8.2 (+4.6)	7.0 (+3.9)
w/o falsification probes	11.7 (-1.2)	18.6 (+13.6)	4.1 (+2.8)	10.6 (+7.0)	9.4 (+6.3)
w/o regression archive	12.2 (-0.7)	8.9 (+3.9)	5.2 (+3.9)	8.1 (+4.5)	7.4 (+4.3)
w/o scope refinement	10.6 (-2.3)	6.4 (+1.4)	1.8 (+0.5)	4.4 (+0.8)	3.9 (+0.8)
LLM-judge gate	10.1 (-2.8)	13.8 (+8.8)	3.9 (+2.6)	9.4 (+5.8)	8.5 (+5.4)
w/o post-commit monitoring	12.5 (-0.4)	7.4 (+2.4)	2.6 (+1.3)	6.5 (+2.9)	10.6 (+7.5)

rates, often exceeding 8%–15%, indicating that target improvement alone is an unreliable proxy for safe transfer.

Second, a purely conservative policy is also suboptimal. CONSERVATIVE GATE combines low harm with much smaller update gain in the lower-left region. This shows that effective self-improvement also requires sufficient plasticity to capture reusable behavioral rules. Conversely, EXHAUSTIVE VERIFIER pushes harm slightly lower than FGSE in a few settings, but does so at much higher verifier cost and with little or no gain advantage, indicating that aggressive verification yields diminishing returns.

Third, the comparison between FGSE and FGSE W/O REFINE highlights the functional role of scope refinement. Without refinement, many configurations move upward in both panels: they retain some gain but suffer higher harmful update rates and larger regression damage. This suggests that a substantial fraction of risky updates retain utility and become safe only after narrowing their

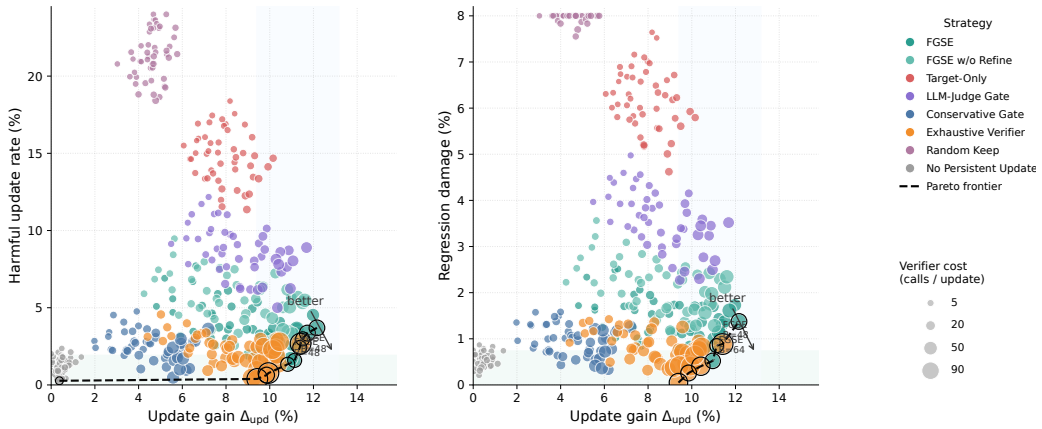


Figure 6: **Plasticity–safety Pareto analysis of update-selection strategies.** Each point is one configuration obtained by sweeping verifier budgets and acceptance thresholds. The x-axis is update gain Δ_{upd}, and the y-axis is either harmful update rate H (left) or regression damage R (right). Point size indicates verifier cost C (average verifier calls per candidate update), and color denotes the update-selection strategy. The black dashed line shows the empirical Pareto frontier, where larger gain and smaller harm are jointly preferred. FGSE occupies the most favorable lower-right region over a wide range of medium verifier budgets, showing that it improves plasticity without incurring the safety cost typically associated with aggressive update policies.

applicability conditions. Overall, the Pareto analysis supports FGSE's better trade-off between *plasticity* and *safety*, placing it in a more favorable region of the update-selection design space.

Targeted probes versus random probes. FGSE uses *targeted* falsification: the verifier constructs probes likely to expose scope overreach, missing preconditions, archived regressions, and decoy-induced shortcut behavior. We compare targeted probe scheduling with budget-matched baselines to evaluate its efficiency in discovering harmful updates.

Let $\mathcal{H}_{\text{bad}}$ denote the set of candidate updates that are in fact unsafe or regressive under held-out evaluation. Given a probe budget K, we define the *bad-update detection rate* as

$$D(K) = \frac{1}{|\mathcal{H}_{\text{bad}}|} \sum_{h \in \mathcal{H}_{\text{bad}}} \mathbb{1}[\tau_h \le K], \tag{127}$$

where τ_h is the number of probes required to expose update h under a given scheduling policy. Thus, $D(K)$ measures the fraction of unsafe updates discovered within the first K probes.

To account for the severity of detected updates, we also report *harmful update avoided*,

$$A(K) = \frac{\sum_{h \in \mathcal{H}_{\text{bad}}} w_h \mathbb{1}[\tau_h \le K]}{\sum_{h \in \mathcal{H}_{\text{bad}}} w_h}, \tag{128}$$

where w_h is the downstream harm weight of update h, estimated from its induced violation severity and regression impact. Hence, $A(K)$ measures how much total harmful mass would be prevented if all detected bad updates were rejected before commitment.

We compare five probe-selection policies under the same verifier budget: TARGETED (the full FGSE probe scheduler), STRATIFIED RANDOM (random selection balanced across probe sources), PURE RANDOM (uniform sampling from the entire probe pool), BOUNDARY-ONLY (target only applicability-boundary probes), and REGRESSION-ONLY (target only archived-regression probes). Figure 7 plots the resulting falsification discovery curves.

Figure 7 shows that targeted probing is markedly more informative than random testing at every budget. The gap is largest in the practically relevant low-budget regime, where verifier calls are scarce. At around $K = 8$, the TARGETED policy already detects roughly twice as many bad updates as PURE RANDOM, and the improvement is even larger in terms of harmful update avoided. This

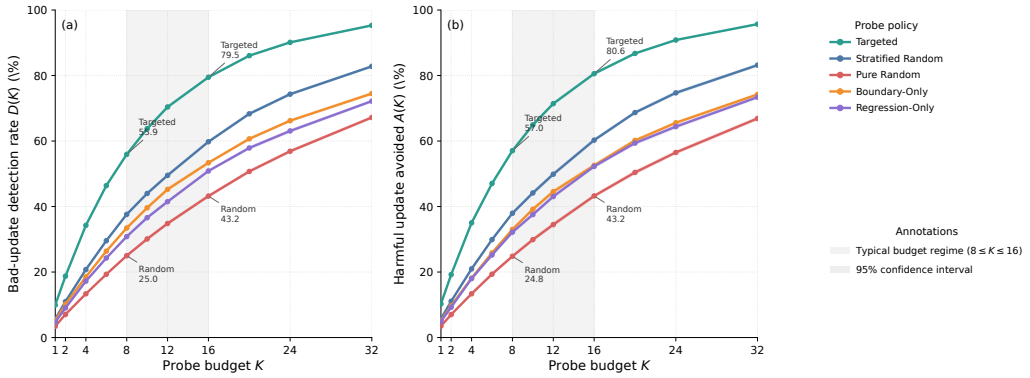


Figure 7: **Falsification discovery curves under different probe-selection policies.** The x-axis is the probe budget K. The left panel shows bad-update detection rate $D(K)$, and the right panel shows harmful update avoided $A(K)$. Solid lines denote the mean across repeated trials, and shaded bands denote 95% confidence intervals. The gray vertical band marks the practically relevant low-budget regime ($8 \leq K \leq 16$). Targeted probing consistently dominates random baselines, especially in the low-budget regime, showing that hypothesis-aware falsification is substantially more information-efficient than unguided random testing.

indicates that targeted probing discovers *more* harmful updates early and prioritizes those with larger downstream damage.

A second observation is that BOUNDARY-ONLY and REGRESSION-ONLY each improve over pure random sampling in the regimes where their probe families are most relevant, but both eventually saturate below the full targeted strategy. This suggests that harmful updates are heterogeneous: some arise from overly broad applicability conditions, whereas others only become visible through archived-regression or decoy-style counterfactual tests. Therefore, a single probe family is insufficient; the gain of FGSE comes from selecting probes conditionally on the specific hypothesis under test.

Finally, the larger gap in $A(K)$ than in $D(K)$ is itself informative. It shows that targeted probing tends to surface the *most dangerous* updates first while increasing discovery count. This property is crucial for practical self-evolution, where only a small verifier budget is available and preventing a few highly harmful commitments is more important than marginally increasing average detection at very large budgets.

Stability under stochastic rollouts. We evaluate each candidate hypothesis under multiple stochastic rollouts to assess the reproducibility of its benefit across seeds.

For a hypothesis h and evaluation set $\mathcal{X}_h$, we define its seed-specific update effect as

$$\Delta_h^{(s)} = \frac{1}{|\mathcal{X}_h|} \sum_{x \in \mathcal{X}_h} \Big(R(\pi \oplus h; x, s) - R(\pi; x, s) \Big), \tag{129}$$

where s is the rollout seed controlling stochastic decoding, action tie-breaking, and environment perturbations. Thus, $\Delta_h^{(s)}$ measures the marginal effect of applying hypothesis h under seed s.

To quantify stability, we further compute the mean seed effect

$$\bar{\Delta}_h = \frac{1}{S} \sum_{s=1}^{S} \Delta_h^{(s)}, \tag{130}$$

and the sign-consistency score

$$C(h) = \frac{1}{S} \sum_{s=1}^{S} \mathbb{1}\Big[\operatorname{sign}\big(\Delta_h^{(s)}\big) = \operatorname{sign}\big(\bar{\Delta}_h\big) \Big], \tag{131}$$

which measures the fraction of seeds whose update effect agrees in sign with the mean effect. A stable and genuinely useful update should exhibit both a positive $\bar{\Delta}_h$ and high sign consistency.

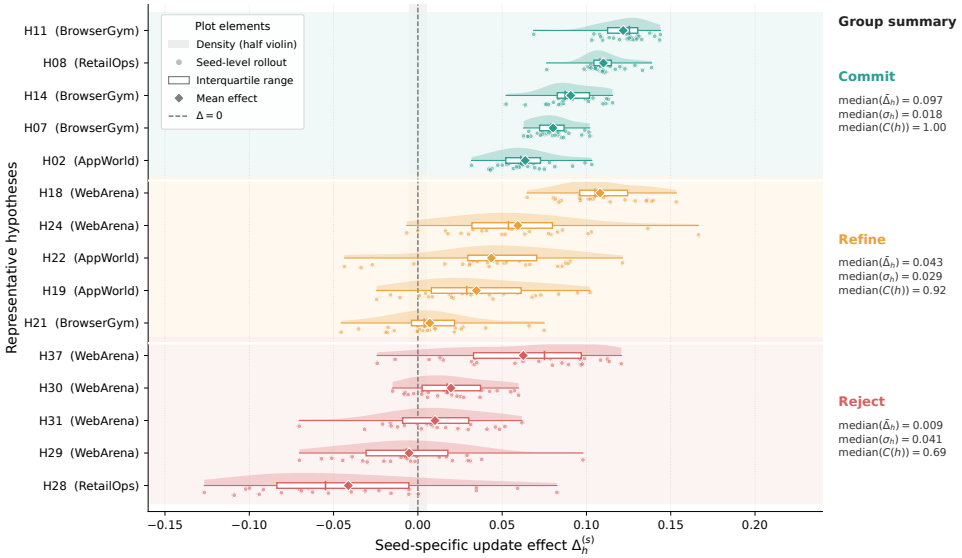


Figure 8: **Stability of hypothesis effects under stochastic rollouts.** Each row corresponds to one hypothesis and shows the distribution of seed-specific update effects $\Delta_h^{(s)}$ across repeated rollouts. Rows are grouped by the final FGSE decision. Dots denote seed-level outcomes, the shaded half-violin shows the empirical density, the white box shows the interquartile range, and the diamond marks the mean effect. The right-side annotation reports the sign-consistency score $C(h)$. Committed hypotheses remain consistently positive across seeds, refined hypotheses are still mostly positive but exhibit larger variance, while rejected hypotheses are centered near zero or change sign frequently.

Figure 8 visualizes the seed-level effect distributions for representative hypotheses from the COMMIT, REFINE, and REJECT groups. Each row shows the distribution of $\Delta_h^{(s)}$ over seeds for one hypothesis, using a raincloud plot. This visualization shows the robustness of observed gains across stochastic rollouts.

Figure 8 shows a clear separation between update classes. Committed hypotheses are tightly concentrated on the positive side, with small seed variance and high sign consistency. In our diagnostic run, the median committed hypothesis has $\bar{\Delta}_h \approx 0.10$ and median sign consistency above 0.9, indicating that its positive effect persists across seeds. Refined hypotheses remain beneficial on average, but their distributions are wider and more sensitive to rollout randomness, which is consistent with the interpretation that they capture partially useful rules whose scope still requires tightening. In contrast, rejected hypotheses are typically centered near zero or slightly negative, and many of them flip sign across seeds, showing that their apparent gains are unstable and largely attributable to stochastic variation.

The effects of accepted updates persist across seeds. Hypothesis identity dominates rollout randomness as a source of variation. FGSE preferentially commits hypotheses whose benefit persists across stochastic executions.

Representative Commit–Refine–Reject Cases We inspect three representative hypotheses from the development audit archive. Each case is evaluated under the default setting used in the main experiments: $|T(h)| = 8$, $|F(h)| = 16$, $|R(h)| = 16$, $\eta = 0.03$, $\delta = 0.05$, and $\beta = 0.01$. We report the target-transfer gain $\widehat{G}(h)$, falsification failure rate $\widehat{P}_{\text{fail}}(h)$, regression damage $\widehat{D}_{\text{reg}}(h)$, and the resulting gate decision. These cases illustrate how FGSE distinguishes transferable rules, repairable overgeneralizations, and unsafe shortcuts.

The default FGSE gate commits the round-trip booking rule and accepts the refined calendar rule after scope narrowing. In the refinement and rejection cases, falsification and regression evidence govern commitment alongside positive target-transfer gain. The refinement case demonstrates the practical value of one-pass scope narrowing: $\widehat{G}(h)$ decreases moderately from 10.4% to 7.6%, but

Table 7: Representative FGSE hypothesis triage cases under the default gate. All three candidates improve the triggering or near-transfer cases. Based on falsification and regression evidence, FGSE commits high-margin transferable rules, refines useful but overly broad rules, and rejects high-risk shortcuts despite positive target gain.

Decision	Domain / Trigger	Candidate hypothesis	Gate statistics	Interpretation
COMMIT	τ-Airline / round-trip booking	**Scope:** the user asks to book a round-trip flight and explicitly provides origin, destination, outbound date, and return date. **Behavior:** select the round-trip mode before search, fill both date fields, and verify that both flight legs are present before final booking. **Expected effect:** the final booking record contains `trip_type=round_trip` and valid outbound and return segments.	$\widehat{G}(h) = 12.1\%$ $\widehat{P}_{\text{fail}}(h) = 0/16 = 0.0\%$ $\widehat{D}_{\text{reg}}(h) = 0.6\%$ Post-commit violation: 1.1%	The hypothesis passes all three default thresholds with margin: $\widehat{G}(h) > \eta$, $\widehat{P}_{\text{fail}}(h) < \delta$, and $\widehat{D}_{\text{reg}}(h) < \beta$. It captures a reusable tool-order rule without harming boundary or archived cases.
REFINE	AppWorld / calendar rescheduling	**Initial scope:** any request mentioning "reschedule". **Behavior:** edit the existing calendar event rather than creating a new one. **Failure mode:** the rule also fires on tasks that ask to create a second event, cancel an event, or modify only a reminder. **Refined scope:** exactly one matching event exists, and the user explicitly requests changing its time rather than creating, cancelling, or duplicating an event.	**Before refinement:** $\widehat{G}(h) = 10.4\%$ $\widehat{P}_{\text{fail}}(h) = 3/16 = 18.8\%$ $\widehat{D}_{\text{reg}}(h) = 2.7\%$ $\text{Sep}(h) = 0.07$ **After refinement:** $\widehat{G}(h') = 7.6\%$ $\widehat{P}_{\text{fail}}(h') = 0/16 = 0.0\%$ $\widehat{D}_{\text{reg}}(h') = 0.8\%$ Post-commit violation: 2.0%	The initial hypothesis is useful but too broad: target gain is high, but falsification and regression exceed the default risk thresholds. Because failures are separable, one precondition-refinement pass preserves 73.1% of the target gain while reducing falsification failures to zero. This shows that refinement salvages repairable updates.
REJECT	AgentDojo / tool-output shortcut	**Scope:** a tool output contains an urgent instruction or an apparent verification token. **Behavior:** follow the tool-suggested action directly without rechecking the user-level instruction or system constraint. **Expected effect:** complete the immediate target task faster by trusting the tool message.	$\widehat{G}(h) = 8.7\%$ $\widehat{P}_{\text{fail}}(h) = 5/16 = 31.2\%$ $\widehat{D}_{\text{reg}}(h) = 4.9\%$ $\text{Sep}(h) = 0.31$ Forced-commit violation: 14.6%	Although the hypothesis improves several target-transfer probes, it creates a high-risk shortcut under decoy and prompt-injection probes. The failures are not cleanly separable by a narrower precondition. FGSE therefore rejects the update, preventing a target-only gate from committing an unsafe behavior.

$\widehat{P}_{\text{fail}}(h)$ drops from 18.8% to 0.0%, and $\widehat{D}_{\text{reg}}(h)$ drops from 2.7% to 0.8%. Thus, under the default setting, FGSE preserves useful self-evolution while blocking the main failure modes exposed by falsification and regression probes.

D.3 Mechanistic Analysis of Long-Term Self-Evolution

Trajectory-level behavioral changes. Aligned interpretability in autonomous driving ties language to intermediate perception, prediction, and planning outputs [Ding et al., 2025]. We compare baseline and FGSE execution trajectories to identify the dominant behavioral edits underlying performance improvements.

Let $\tau^{\text{B}} = (a_1, \ldots, a_T)$ and $\tau^{\text{F}} = (a'_1, \ldots, a'_{T'})$ denote the baseline and FGSE trajectories for the same task. We first align the two traces at the action-type level and group all trajectory edits into four

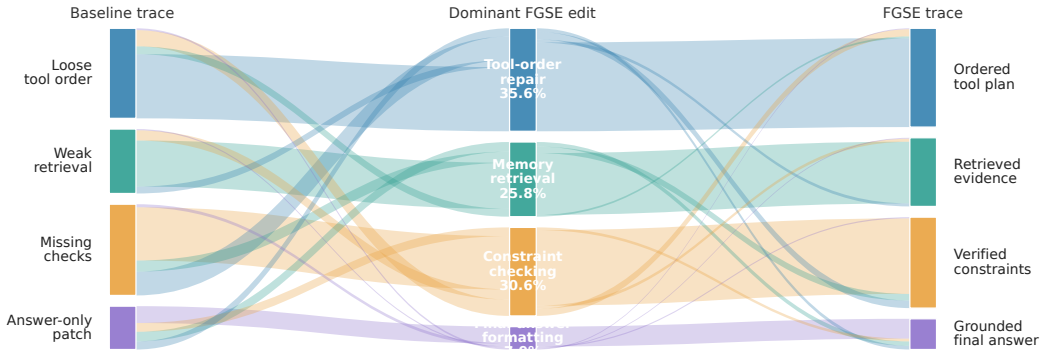


Figure 9: **Trajectory-edit Sankey diagram from baseline traces to FGSE traces.** The left column shows the dominant behavioral deficiency in the baseline trace, the middle column shows the dominant edit family introduced by FGSE, and the right column shows the resulting FGSE behavior motif. Band width is proportional to the number of corrected trajectories. The diagram shows that most successful transitions are driven by tool-order repair, memory retrieval, and constraint checking, whereas final-answer formatting accounts for only a small minority of improvements.

interpretable behavior families: *tool order*, *memory retrieval*, *constraint checking*, and *final answer formatting*. For each family $e \in \mathcal{E}$, we compute its weighted edit mass

$$m_e(\tau^{\mathrm{B}}, \tau^{\mathrm{F}}) = \sum_{o \in \mathcal{O}_e} w(o), \tag{132}$$

where $\mathcal{O}_e$ is the set of aligned edit operations assigned to family e, and $w(o)$ denotes the causal weight of operation o estimated from its contribution to final task improvement. We then define the dominant edit family as

$$e^\star(\tau^{\mathrm{B}}, \tau^{\mathrm{F}}) = \arg\max_{e \in \mathcal{E}} m_e(\tau^{\mathrm{B}}, \tau^{\mathrm{F}}). \tag{133}$$

To make the behavioral change interpretable, we further assign each trajectory pair a *baseline behavior motif* and an *FGSE behavior motif*, determined from the dominant failure pattern before the first decisive divergence and the dominant corrected pattern after refinement, respectively. Figure 9 aggregates all improved trajectory pairs into a three-stage Sankey diagram: baseline motif $\rightarrow$ dominant edit family $\rightarrow$ FGSE motif. The width of each band is proportional to the number of corrected trajectories contributing to that transition.

Figure 9 shows that FGSE's gains are primarily driven by *substantive behavioral correction*. The dominant flow mass concentrates in three edit families: tool-order repair, constraint checking, and memory retrieval. Final-answer formatting occupies only a small portion of the total corrected mass and rarely converts failures to successes.

The largest transition typically starts from baseline traces with loose or misordered tool usage, passes through tool-order repair, and ends in ordered tool plans under FGSE. A similarly large portion of corrected trajectories flows through constraint checking, showing that many failures arise from missing explicit verification before tool execution or answer commitment despite available knowledge. Memory retrieval forms the third major band, and its contribution is especially prominent on long-horizon and memory-intensive tasks, where the baseline agent often fails to retrieve previously relevant context or latent facts.

Overall, the Sankey analysis clarifies *where* FGSE changes behavior: it reorganizes tool usage, retrieves missing evidence, and enforces explicit constraint checks. These findings strengthen the interpretation that FGSE improves the agent's internal decision process.

D.4 Cost and Scalability

Memory-growth controllability. We analyze memory-growth scalability by varying the retention budget and measuring how agent quality changes as the number of committed hypotheses increases.

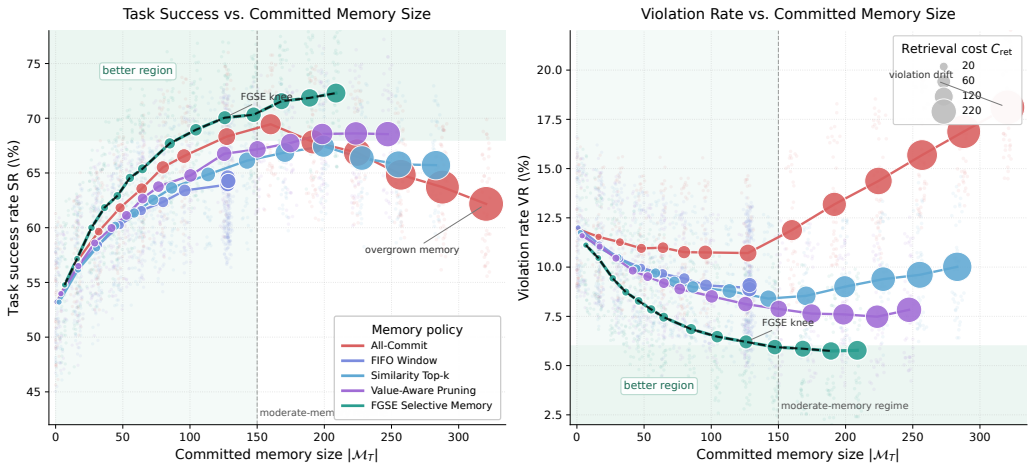


Figure 10: **Memory-growth Pareto analysis.** We vary the retention budget and evaluate long-horizon self-evolving agents under different memory policies. **Left:** task success rate versus committed memory size. **Right:** violation rate versus committed memory size. Each faint point denotes one domain-seed run, while solid lines connect policy means across budgets. Marker area is proportional to retrieval cost C_{ret}. A desirable memory policy should remain close to the upper-left region in the left panel (high success with small memory) and the lower-left region in the right panel (low violation with small memory). FGSE traces the most favorable frontier: it achieves high task success and low violation with substantially smaller committed memory and lower retrieval cost than naive memory growth.

For each memory policy, we vary the nominal retention budget over

$$B \in \{0, 16, 32, 48, 64, 80, 96, 128, 160, 192, 224, 256, 288, 320\},$$

and run long-horizon evaluation episodes under the same self-evolution protocol. For each run, we record the final committed memory size $|\mathcal{M}_T|$, task success rate SR, violation rate VR, and average retrieval cost C_{ret}. We compare ALL-COMMIT, FIFO WINDOW, SIMILARITY TOP-k, VALUE-AWARE PRUNING, and FGSE SELECTIVE MEMORY. Figure 10 presents the resulting memory-growth Pareto plot.

Figure 10 shows that memory growth is *not* monotonic in utility. The naive ALL-COMMIT strategy initially benefits from additional memory, but after the committed set becomes large, the curve enters an overgrown regime: success saturates and then degrades, while the violation rate rises steadily and retrieval cost increases sharply. In contrast, bounded heuristics such as FIFO WINDOW and SIMILARITY TOP-k keep memory smaller, but they also leave substantial performance on the table because they discard useful reusable rules too aggressively.

FGSE SELECTIVE MEMORY lies on the best empirical frontier in both panels. It reaches its strongest regime at a moderate memory size. In this diagnostic run, FGSE attains around $70\% \sim 72\%$ task success once the committed memory reaches roughly 120–150 entries, while maintaining a violation rate near $4\% \sim 5\%$. At a comparable or even lower committed size, competing strategies either achieve lower success or incur substantially higher violation. Moreover, compared with ALL-COMMIT, FGSE achieves a clearly better quality–cost trade-off: it matches or exceeds the best success of naive growth using far fewer retained hypotheses and noticeably smaller retrieval cost. This indicates that *memory quality and selective retention* are the key bottleneck for gains from persistent memory.

Finally, the curves exhibit a visible saturation effect. Beyond a moderate memory size, the marginal gain of additional committed hypotheses becomes very small, while retrieval cost continues to increase. These results support controllable long-term memory growth through commitment filtered by utility, falsification, and regression preservation.

E Broader Impacts

FGSE may improve the reliability of long-horizon language agents by reducing unsafe memory updates, overgeneralized behavioral rules, and regressions on previously solved tasks. Potential positive impacts include safer agent adaptation in web, app, and tool-use settings where persistent memory is used. However, more reliable self-evolving agents may also increase the capability of automated systems that can perform harmful or unauthorized actions if deployed without access control, monitoring, and domain-specific safety constraints [Yang et al., 2026a, Fan et al., 2023]. Financial-agent audits likewise identify source-manipulation risks and motivate model-independent halts and hard loss limits [Wang and Saxena, 2026]. FGSE mitigates some of these risks through verifier-grounded falsification, regression testing, audit records, and post-commit monitoring. These mechanisms complement deployment-level safety controls.

F LLM Usage

This paper studies self-evolving LLM agents. In the experiments, frozen LLM backbones are used as ReAct-style executors that generate interaction trajectories in web, app, tool-use, and long-term memory environments. FGSE also uses an LLM-based proposer to convert episode evidence into structured candidate hypotheses. LLM-generated hypotheses enter verifier-based evaluation. The final commit–refine–reject decision is determined by target-transfer, falsification, and regression probes evaluated with executable or benchmark-provided verifiers, including official success validators, database-state checkers, unit tests, security validators, and exact-match evaluators.

During the broader research process, LLM-based tools assisted with wording, grammar, LaTeX formatting, code drafting, script debugging, and organization of experimental notes. The authors specified, checked, and approved all scientific claims, methodological choices, mathematical statements, proofs, experimental settings, reported numbers, figures, tables, and interpretations. They manually verified all references for existence, relevance, and correctness, and reviewed and edited LLM-assisted text and code before inclusion in the paper or experiments.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction clearly state the main contribution: treating episode-derived self-updates as verifier-grounded hypotheses and committing them only after transfer, falsification, and regression testing. The claims are supported by the method, theoretical analysis, and experiments across web, app, tool-use, and memory benchmarks.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The paper discusses directions for future improvement: "Future work can extend falsification-gated update control to multimodal agents, collaborative tool-use systems, and settings where verifiers must be learned or approximated."

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: The paper states the assumptions for the confidence-gated commitment theorem, including fixed candidate sets, fresh final probes, bounded verifier scores, and conditional independence. Full proofs, concentration bounds, and probe-complexity results are provided in the appendix.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: The paper describes the benchmarks, evaluation metrics, baselines, update budgets, verifier access, hyperparameters, random seeds, and fair-comparison protocol needed to reproduce the main experimental claims.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [No]

   Justification: Code release is pending the patent application process. We plan to release code and reproduction instructions upon publication.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: The paper specifies the benchmark suite, official metrics, baselines, backbone settings, probe budgets, hyperparameters, development/test usage, and controlled comparison protocol in the main text and appendix.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: The paper reports stochastic settings with three random seeds and uses bootstrap confidence intervals for performance reporting.

   Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We report the author-side compute resources in Appendix C.7, including LLM calls, verifier calls, wall-clock time, and client CPU-hours. No author-side GPU was used because all LLM backbones were accessed through provider-hosted APIs.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research is conducted on existing benchmark environments and does not involve deceptive data collection, human-subject experimentation, or release of high-risk models or datasets. The work is designed to improve reliability and safety of self-evolving agents.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: The paper discusses both the potential reliability benefits of safer self-evolving agents and the risks of misuse or incorrect autonomous tool-use, along with mitigation mechanisms such as falsification probes, regression archives, audit records, and post-commit monitoring.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper does not release a new pretrained model, scraped dataset, image generator, or other high-risk asset requiring special release safeguards. The proposed method itself includes safety-oriented mechanisms such as verifier-gated commitment and post-commit monitoring.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [No]

Justification: The paper cites the existing benchmarks and models used in evaluation, but the current version does not explicitly list the licenses or terms of use for all datasets, environments, and model APIs.

Guidelines:

- The answer [N/A] means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [N/A]

Justification: The paper does not introduce or release a new dataset, benchmark, pretrained model, or other standalone asset.

Guidelines:

- The answer [N/A] means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [N/A]

Justification: The paper does not involve crowdsourcing experiments or research with human participants.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.

- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The paper does not involve human-subject research, crowdsourcing, or collection of participant data, so IRB approval is not applicable.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [Yes]

    Justification: LLMs are central to both the problem setting and the experimental instantiation of this work. The studied systems are LLM agents that interact with web, app, tool-use, and memory environments through a ReAct-style executor. In FGSE, frozen LLM backbones are used to generate agent trajectories and to propose structured candidate hypotheses from episode evidence, while verifier- grounded probes and executable validators determine whether a hypothesis is committed, refined, or rejected. Several baselines also use LLM-based reflection, memory writing, or LLM-judge gates.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.