EVOSTEER: RESTORING SAFETY IN EXPERIENCE-DRIVEN AGENTIC EVOLUTION VIA ACTIVATION STEERING

Anonymous authors
Paper under double-blind review

ABSTRACT

Experience-driven evolution allows agents to improve by accumulating and reusing experience while keeping the backbone model fixed, but recent studies show that the resulting memory can also degrade agent safety. Existing mitigations mainly modify the evolution mechanism, alter the prompt, or introduce additional agent components. We instead ask whether this degradation can be corrected directly inside the model while leaving the evolved memory and evolution process unchanged. We find that retrieved experience shifts the internal representations of harmful requests toward execution along a safety-relevant direction, providing a representation-level signal for correcting the degradation. Based on this observation, we introduce EVOSTEER, an activation-steering method that uses the memory-free forward pass of the same request as a per-request reference and selectively removes the execution-ward component of the memory-induced representation shift. EVOSTEER requires no modification to the model weights, prompt, evolved memory, or evolution algorithm. Experiments show that our method effectively mitigates the safety degradation induced by evolved memory while largely preserving benign performance. Our results demonstrate that representation-level correction provides a viable defense for experience-driven agent evolution without redesigning the evolution process.

1 INTRODUCTION

Large language model (LLM) agents increasingly improve by accumulating experience rather than by updating weights. After acting in an environment, the agent distills its experience into a reusable memory, such as workflows (Wang et al., 2024), reasoning strategies (Ouyang et al., 2026), or procedural knowledge (Fang et al., 2026), and retrieves this memory to guide later tasks. Such experience-driven evolution improves agents without changing their backbone weights, yet the model can behave differently as the experience retrieved into its context evolves.

Recent studies show that experience-driven evolution can erode agent safety (Shao et al., 2026; Zhao et al., 2026; Cheng et al., 2026). We focus on a subtle setting in which the agent evolves exclusively from benign tasks, the resulting memory contains no harmful content, and the model weights remain unchanged. Nevertheless, once the memory distilled from these benign experiences is retrieved into context, the agent may execute rule-violating requests that the same memory-free agent would refuse. The same memory that improves benign task performance can therefore simultaneously weaken the agent's ability to refuse requests that violate its safety constraints.

Existing mitigations mainly modify the evolution process, alter what is stored or retrieved, strengthen the prompt, or add an external safety component (Zhao et al., 2026; Cheng et al., 2026; Shao et al., 2026). These approaches act on the agent system surrounding the backbone model, changing how memory is accumulated, selected, or presented at inference time. We instead ask a different question: what does retrieved benign memory change inside the fixed model when a previously refused request becomes executable? If the underlying refusal behavior remains available in the memory-free model, then restoring safety may not require redesigning the evolution process at all. This motivates us to intervene directly in the model's forward pass while leaving the evolved memory, evolution mechanism, prompt, and model weights unchanged.

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

We find that this degradation is accompanied by a shift in the model's internal representations. Using the memory-free model, we estimate a layer-wise safety-relevant direction from matched pairs of rule-violating and benign requests. When evolved memory is retrieved, the representations of violating requests shift along this direction toward execution relative to the same requests without memory. Cancelling the execution-ward component at inference time substantially recovers refusal behavior, whereas perturbations of the same magnitude along random directions or within the orthogonal complement recover little of the lost refusals. These results indicate that the safety degradation induced by evolved memory can be characterized as a directional representation shift that is reversible at inference time.

Based on this observation, we propose EVOSTEER, an activation-steering method that corrects the experience-induced safety shift without modifying the experience. Given a request, EVOSTEER uses the memory-free forward pass of the same request as a per-request reference and measures how the retrieved memory changes its internal representation along the estimated safety direction. During the memory-augmented prefill, it then corrects only the execution-ward component of this deviation, layer by layer. The correction is adaptive to each request and is applied only when memory shifts the representation toward execution. EVOSTEER requires no changes to the evolved memory, evolution algorithm, prompt, or model weights, allowing it to be applied directly to already-evolved agents.

We evaluate EVOSTEER across two experience-driven evolution mechanisms, Agent Workflow Memory (AWM) (Wang et al., 2024) and MemP (Fang et al., 2026), using four open-weight models from two model families, including Qwen3-14B, Qwen3-32B (Yang et al., 2025), Qwen3.6-27B, and Gemma-4-E4B (Team et al., 2026), with data from two benchmarks, Mind2Web-SC (Xiang et al., 2025) and SafeAgentBench (Yin et al., 2024). Across these settings, EVOSTEER mitigates the safety degradation caused by evolved memory while largely preserving benign behavior and retaining the utility gains brought by experience-driven evolution.

Our contributions can be summarized as follows:

- **Characterization:** We characterize the safety degradation induced by benign experience-driven evolution at the representation level. The memory-free model retains its refusal behavior, while retrieved experience shifts the representations of harmful requests toward execution. Correcting this shift recovers refusal behavior, whereas random or orthogonal-direction interventions provide substantially weaker recovery.
- **Method:** We introduce EVOSTEER, an activation-steering method that uses the memory-free forward pass of the same request as a per-request reference and corrects the execution-ward component of the memory-induced representation shift layer by layer. EVOSTEER leaves the model weights, prompt, evolution algorithm, and evolved memory unchanged, making it directly applicable to already-evolved agents without regenerating or modifying their memory.
- **Empirical Evaluation:** We evaluate EVOSTEER across two experience-driven evolution mechanisms, two datasets, and four open-weight models from two model families. Across these settings, EVOSTEER restores refusal behavior to near or above pre-evolution levels while preserving the utility gains from evolution.

2 RELATED WORK

2.1 EXPERIENCE-DRIVEN EVOLUTION AND ITS SAFETY

Experience-driven evolution enables agents to improve while keeping the backbone model fixed by accumulating and reusing experience in memory, such as verbal reflections (Shinn et al., 2023), natural-language insights (Zhao et al., 2024), induced workflows (Wang et al., 2024), reasoning strategies (Ouyang et al., 2026), procedural memory (Fang et al., 2026), and test-time cheatsheets (Suzgun et al., 2026). Despite their different memory representations, these methods share a common structure: experience is distilled into memory, retrieved for later tasks, and injected into the model through context. Recent work shows that this process can also degrade safety: accumulated experience, even when derived only from benign tasks, can reduce refusal on later unsafe or rule-violating requests (Shao et al., 2026; Zhao et al., 2026; Cheng et al., 2026).

Existing defenses typically intervene in the evolution pipeline or introduce additional agent components. TAME evaluates previously used experiences and adjusts the way they are subsequently

used (Cheng et al., 2026), Guo et al. (2026) inject a safety reflection when stored personal context interacts riskily with the current query, and MemEvoBench explores actively correcting contaminated memory (Xie et al., 2026). These methods modify the memory itself, how it is accumulated or retrieved, or the surrounding agent system. Our work instead takes the evolved memory and evolution process as given and asks what retrieved experience changes inside the model at inference time. Accordingly, our intervention acts only on the forward pass, leaving the memory, evolution algorithm, and prompt unchanged, and can therefore be applied without re-running or redesigning the evolution process.

2.2 Representation-level safety and activation steering

A line of work studies and controls model behavior through its internal representations (Zou et al., 2023; Turner et al., 2023; Li et al., 2023; Rimsky et al., 2024). For safety in particular, Arditi et al. (2024) show that refusal in many models is mediated by a single direction: ablating it removes refusal, while adding it can induce refusal on harmless prompts. Since applying a fixed steering vector across all inputs can also perturb benign behavior and degrade utility, subsequent work has explored conditional interventions. Conditional Activation Steering applies steering only when an activation-based condition is satisfied (Lee et al., 2025). SafeSwitch monitors internal signals and activates a safety intervention when needed (Han et al., 2025). AlphaSteer learns a steering transformation designed to have little effect on benign inputs (Sheng et al., 2026). We build on the refusal-direction finding, but use it differently. Our setting differs from these approaches in that the intervention is defined relative to the effect of retrieved memory on the same request. Rather than applying a fixed steering vector, we use the memory-free forward pass as a per-request reference and correct only the memory-induced shift toward execution.

Closest to our work are studies of context-induced shifts in safety-relevant representations. Many-shot jailbreaking shows that long sequences of in-context demonstrations of compliance can override safety training (Anil et al., 2024). Chen et al. (2026) characterize this effect as a progressive drift of activations away from a safety-aligned region and show that appending a safety demonstration can partially reverse it. In agents, Pan et al. (2026) find that schema-formatted tool specifications weaken internal refusal signals, but do not study activation-based interventions to mitigate this degradation. Our setting differs in two respects. First, the context inducing the shift is the agent's own memory distilled from benign experience rather than adversarial demonstrations or tool specifications. Second, the same request can be evaluated with and without retrieved memory, providing a natural per-request reference for measuring and correcting the memory-induced shift.

3 Method

EvoSteer leaves the backbone model, prompt, evolution algorithm, and evolved memory unchanged, intervening only on the model's internal representations. For each request, it uses the memory-free forward pass as a reference, measures the memory-induced deviation along an estimated safety-relevant direction, and counteracts only its execution-ward component layer by layer during prefill. We illustrate EvoSteer in Figure 1 and summarize the procedure in Algorithm 1.

3.1 Problem formulation

Goal. Retrieved memory can cause an evolving agent to execute a rule-violating request that its memory-free counterpart would refuse. Our goal is to recover the memory-free safety behavior while preserving the utility gained from memory. We leave the memory, prompt, evolution algorithm, and model weights unchanged, and intervene only on the model's internal representations.

We consider an LLM agent with a frozen L-layer backbone and hidden size d. For a task context c and retrieved memory m, let $P(c)$ denote the memory-free prompt and $P(c, m)$ the corresponding memory-augmented prompt. The two prompts share the same final N_c tokens after applying the chat template. We call these aligned positions the *decision window* $\mathcal{W}$, indexed by $k = 1, \ldots, N_c$, with $k = N_c$ the decision position. We write $\mathbf{h}_l(P)[k] \in \mathbb{R}^d$ for the hidden representation entering layer l at window position k of prompt P.

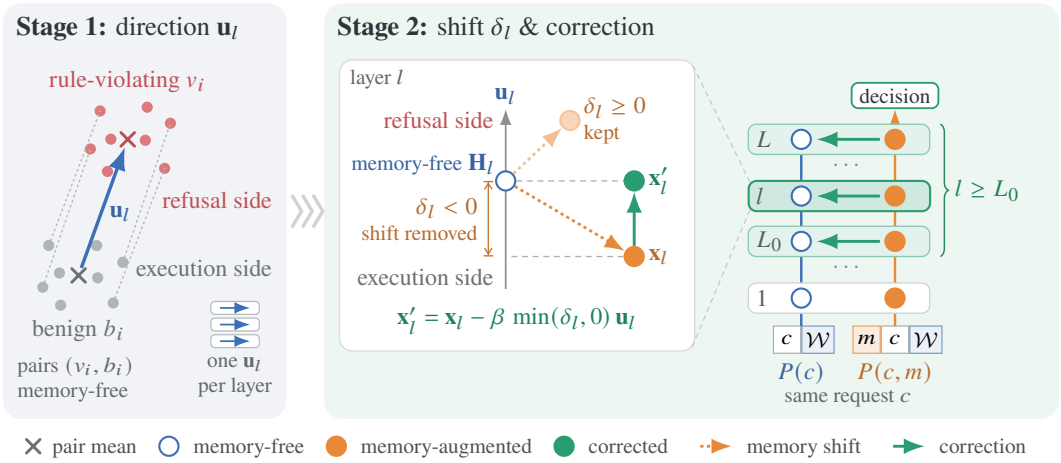

Figure 1: Overview of EvoSteer. Stage 1 estimates a direction $\mathbf{u}_l$ at each layer from matched rule-violating and benign requests, with its positive side corresponding to refusal. Stage 2 compares the memory-augmented activation $\mathbf{x}_l$ with the memory-free reference $\mathbf{H}_l$ for the same request at each layer $l \geq L_0$, and corrects only the execution-ward component of the resulting shift δ_l, leaving refusal-ward shifts unchanged.

Algorithm 1 EvoSteer: activation correction

Require: frozen backbone with L layers; fit pairs $\{(v_i, b_i)\}_{i=1}^n$; token window $\mathcal{W}$ of size N_c; read-out position $p^\star$; first corrected layer L_0; strength β

1: **for** $l = L_0, \ldots, L-1$ **do**
2: $\quad \mathbf{r}_l \leftarrow \text{mean}_i\, \mathbf{h}_l(P(v_i))[p^\star] - \text{mean}_i\, \mathbf{h}_l(P(b_i))[p^\star]; \quad \mathbf{u}_l \leftarrow \mathbf{r}_l / \|\mathbf{r}_l\|$
3: **end for**
4: Prefill $P(c)$ with hooks off; cache $\mathbf{H}_l \leftarrow \mathbf{h}_l(P(c))[\mathcal{W}]$ for all $l \geq L_0$ ▷ Pass 1: reference
5: Prefill $P(c, m)$; at the input of each layer $l \geq L_0$, for $k \in \mathcal{W}$: ▷ Pass 2: corrected
6: $\quad \delta_l[k] \leftarrow \langle \mathbf{x}_l[k] - \mathbf{H}_l[k],\, \mathbf{u}_l \rangle; \quad \mathbf{x}_l[k] \leftarrow \mathbf{x}_l[k] - \beta \min(\delta_l[k], 0)\, \mathbf{u}_l$
7: Decode from the corrected cache with hooks off; **return** the generated action

3.2 Measuring the Experience-Induced Shift

We first estimate a layer-wise safety-relevant direction from memory-free representations using a fit set. Let $\mathcal{D}_{\text{fit}} = \{(v_i, b_i)\}_{i=1}^n$ consist of matched pairs, where v_i is a rule-violating request and b_i is a benign counterpart with a similar task that differs in what makes the request violating. At each layer l, we estimate a unit direction $\mathbf{u}_l$ from the representations of the fit pairs at a read-out position $p^\star \in \mathcal{W}$ as the normalized difference of class means:

$$\mathbf{r}_l = \frac{1}{n}\sum_{i=1}^{n} \mathbf{h}_l\big(P(v_i)\big)[p^\star] \;-\; \frac{1}{n}\sum_{i=1}^{n} \mathbf{h}_l\big(P(b_i)\big)[p^\star], \qquad \mathbf{u}_l = \mathbf{r}_l / \|\mathbf{r}_l\|. \tag{1}$$

We orient $\mathbf{u}_l$ from the benign class mean toward the violating class mean. Since the memory-free model tends to refuse the violating requests while executing their benign counterparts, the positive direction is refusal-ward and the negative direction execution-ward. We refer to the two as the refusal side and the execution side of $\mathbf{u}_l$.

For a request c, we measure the memory-induced shift at layer l and window position k by projecting the difference between the memory-augmented and memory-free representations onto $\mathbf{u}_l$:

$$\delta_l[k] = \Big\langle \mathbf{h}_l\big(P(c, m)\big)[k] - \mathbf{h}_l\big(P(c)\big)[k],\; \mathbf{u}_l \Big\rangle. \tag{2}$$

Because the two prompts share the same decision window, this difference compares aligned token positions. A negative $\delta_l[k]$ indicates an execution-ward shift. Evaluated at the decision position, δ_l provides a layer-wise diagnostic of the effect of memory. To examine this effect in an evolved agent,

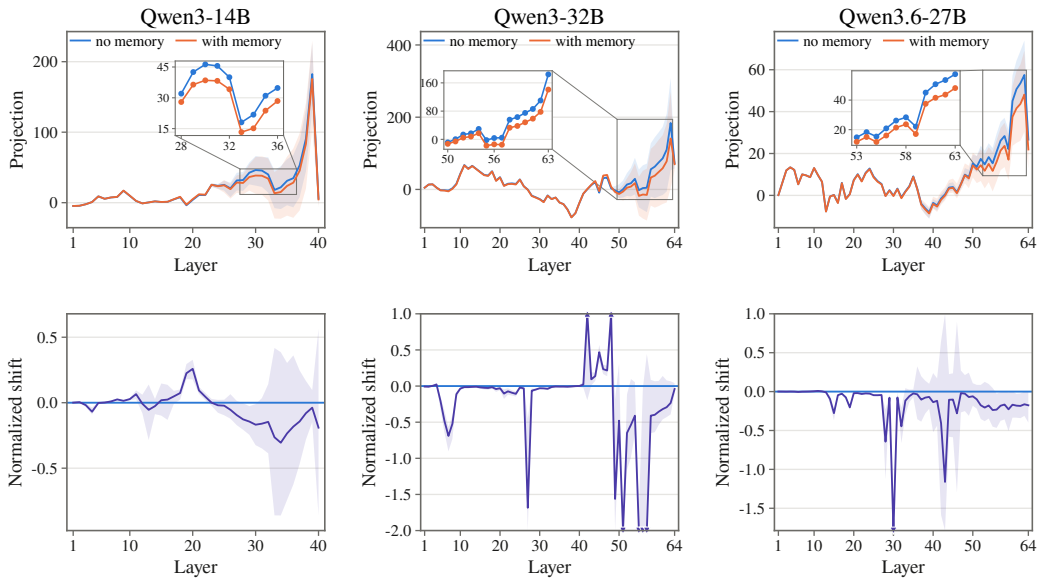

Figure 2: Memory-induced representation shift across layers on the rule-violating requests. **Top:** mean projection of the memory-free (blue) and memory-augmented (orange) representations onto $\mathbf{u}_l$; bands show the interquartile range across requests. **Bottom:** the shift relative to the magnitude of the memory-free projection, $\delta_l/|\bar{p}_l|$, where $\bar{p}_l$ is the mean memory-free projection at layer l; the blue line at 0 is the memory-free reference.

we use AWM (Wang et al., 2024) memory induced from Mind2Web (Deng et al., 2023) training trajectories, and estimate $\mathbf{u}_l$ from 24 matched violating–benign pairs in Mind2Web-SC (Xiang et al., 2025). For each violating request, we compare the memory-free and memory-augmented forward passes and measure the shift. As shown in Figure 2, in later layers the memory-augmented projections consistently fall below their memory-free counterparts, indicating that retrieved memory shifts these requests toward the execution side. Since this trend is less apparent in earlier layers, we begin correction at an intermediate layer L_0 which can be selected on the fit set.

3.3 Correcting the Shift

EvoSteer runs two forward passes per request (Algorithm 1): (1) **Pass 1 (reference).** We prefill the memory-free prompt $P(c)$ without intervention and cache the window representations $\mathbf{H}_l = \mathbf{h}_l(P(c))[\mathcal{W}]$ for every layer $l \geq L_0$, where L_0 is the first corrected layer. (2) **Pass 2 (corrected generation).** We prefill the memory-augmented prompt $P(c, m)$ while intervening at the input of every layer $l \in \{L_0, \ldots, L-1\}$. Let $\mathbf{x}_l[k] = \mathbf{h}_l(P(c, m))[k]$ denote the representation entering layer l at window position k after all preceding corrections have been applied. At each position, we measure its deviation from the cached memory-free reference along $\mathbf{u}_l$ and apply a refusal-ward correction to the execution-side component, scaled by a single strength $\beta > 0$:

$$\delta_l[k] = \big\langle \mathbf{x}_l[k] - \mathbf{H}_l[k],\ \mathbf{u}_l \big\rangle, \qquad \mathbf{x}_l[k] \ \leftarrow\ \mathbf{x}_l[k] - \beta\, \min\big(\delta_l[k], 0\big)\, \mathbf{u}_l, \qquad k \in \mathcal{W}. \tag{3}$$

We use the direction $\mathbf{u}_l$ estimated at the read-out position $p^\star$ throughout the decision window. One alternative is to estimate a position-specific direction for each of the N_c window positions and apply the corresponding direction at each position. In our experiments, however, this increases false refusals on benign requests, whereas sharing the read-out direction across positions mitigates this issue (Section 4.4). We therefore use the single direction $\mathbf{u}_l$ throughout the window.

The correction is adaptive to each request: at every layer and position, its magnitude is determined by the deviation from that request's own memory-free reference. It is one-sided, acting only when $\delta_l[k] < 0$ and leaving refusal-ward deviations unchanged. As each $\delta_l[k]$ is measured after all pre-

ceding corrections have taken effect, EVOSTEER forms a layer-wise feedback mechanism around the fixed memory-free reference rather than applying precomputed corrections independently across layers. The intervention is restricted to prefill and disabled during decoding.

Hyperparameters. EVOSTEER has three hyperparameters: the read-out position $p^\star$, the first corrected layer L_0, and the correction strength β. The read-out position determines where the safety-relevant direction is estimated, L_0 specifies where correction begins, and β controls its strength. The selection procedure and dataset-specific settings are described in the Appendix.

4 EXPERIMENTS

4.1 EXPERIMENTAL SETUP

Models. Our main experiments use four models from two families: Qwen3-14B, Qwen3-32B (Yang et al., 2025), Qwen3.6-27B, and Gemma-4-E4B (Team et al., 2026). **Datasets.** The first is Mind2Web (Deng et al., 2023), a benchmark of real-world web tasks with annotated action sequences, which we use for benign evolution and utility evaluation. For safety evaluation, we use its safety-augmented variant Mind2Web-SC (Xiang et al., 2025), which associates tasks with user profiles and six access-control rules, allowing requests to be labeled as benign or rule-violating under the applicable rule. The second is SafeAgentBench (Yin et al., 2024), which pairs hazardous household tasks one-to-one with safe counterparts. For simplicity, we use it as a text-only planning benchmark, where the agent outputs a decision and a plan without executing the simulator. Its safe tasks provide the evolution and utility splits, while its hazardous tasks are used for safety evaluation. **Evolution.** We consider two experience-driven evolution methods. Agent Workflow Memory (AWM) (Wang et al., 2024) induces reusable workflows from benign trajectories and inserts them into the prompt, and is evaluated on both datasets. MemP (Fang et al., 2026) distills past trajectories into reusable procedural memories and retrieves relevant ones to guide new tasks, and is evaluated on SafeAgentBench.

Baselines. On Mind2Web-SC we compare with four mitigations applied on top of the evolution mechanism: a safety-priority prompt, a memory disclaimer (Shao et al., 2026), TAME (Cheng et al., 2026), and refusal-experience memory induced from the violating fit requests following Zhao et al. (2026). **Metrics.** For safety we report the refusal rate on harmful tasks and the false-refusal rate on benign tasks. For utility, we report the standard metrics of each dataset on its benign test split. For EVOSTEER, the fit set is used only to estimate the directions and select the hyperparameters. All test-set results are obtained with the resulting configuration frozen. The selection procedure is detailed in Appendix A.1. Dataset-specific experimental details are provided in the following subsections.

4.2 MAIN RESULTS

Setup. We use 128 Mind2Web-SC tasks for safety evaluation, with 24 matched violating–benign pairs as the fit set and the remaining 40 violating and 40 benign tasks as the test set. Memory is induced from Mind2Web training trajectories using the official AWM offline pipeline, with one workflow per website. Utility is measured by element accuracy, operation F1, and step success rate on the Mind2Web cross-task test split (69 websites, 252 tasks). Mind2Web requires each action to begin with `Action: `; we therefore end every prompt with this prefix to facilitate reliable parsing, which also fixes the decision window to $N_c = 12$ tokens for the Qwen models and 8 for Gemma-4-E4B. The configurations selected on the fit set and frozen for evaluation are $(p^\star, L_0, \beta) = (-11, 16, 1)$, $(-3, 51, 4)$, $(-4, 51, 8)$, and $(-3, 34, 8)$ for Qwen3-14B, Qwen3-32B, Qwen3.6-27B, and Gemma-4-E4B, respectively.

Evolution improves utility but can reduce safety. As shown in Table 1, adding AWM memory improves benign utility on most metrics across the four models. At the same time, refusal on violating tasks decreases from 40.0% to 22.5% for Qwen3-14B and from 50.0% to 27.5% for Qwen3-32B, with a smaller drop from 22.5% to 20.0% for Gemma-4-E4B. For Qwen3.6-27B, the refusal rate remains unchanged at 67.5%.

EVOSTEER largely restores the lost refusals. Across all four models, EVOSTEER achieves the highest refusal rate on violating tasks among the compared mitigations, tying with TAME on

Table 1: Main results on four models. Safety on the Mind2Web-SC test set: refusal rate on the violating tasks (%, higher is better) and false-refusal rate on the benign tasks (FR, %, lower is better), each followed by its change in points relative to the No memory row of the same model. Utility on the Mind2Web cross-task test split: element accuracy, operation F1, and step success rate as task-level macro averages (%). All mitigations are applied on top of the same AWM memory (Wang et al., 2024): the safety prompt, the memory disclaimer (Shao et al., 2026), the TAME trust annotation (Cheng et al., 2026), and refusal-experience memory (Zhao et al., 2026).

Model	Method	Safety (Mind2Web-SC)		Utility (Mind2Web)		
		Refusal ↑	FR ↓	Ele.Acc ↑	Op.F1 ↑	Step SR ↑
Qwen3-14B	No memory	40.0	2.5	46.0	56.9	41.7
	AWM memory	22.5 (−17.5)	2.5 (+0.0)	46.7	56.7	42.1
	Safety prompt	27.5 (−12.5)	2.5 (+0.0)	**46.9**	57.0	**42.6**
	Disclaimer	25.0 (−15.0)	2.5 (+0.0)	46.8	**57.3**	42.5
	TAME	25.0 (−15.0)	2.5 (+0.0)	46.8	56.9	42.1
	Refusal memory	30.0 (−10.0)	2.5 (+0.0)	46.6	56.7	42.0
	EvoSteer (ours)	**35.0** (−5.0)	2.5 (+0.0)	46.9	56.9	42.3
Qwen3-32B	No memory	50.0	0.0	45.3	56.0	40.7
	AWM memory	27.5 (−22.5)	0.0 (+0.0)	46.7	57.2	42.1
	Safety prompt	37.5 (−12.5)	0.0 (+0.0)	46.3	57.4	41.9
	Disclaimer	25.0 (−25.0)	0.0 (+0.0)	46.9	**57.5**	**42.4**
	TAME	40.0 (−10.0)	0.0 (+0.0)	**47.0**	57.2	42.4
	Refusal memory	32.5 (−17.5)	0.0 (+0.0)	46.7	57.2	41.8
	EvoSteer (ours)	**47.5** (−2.5)	0.0 (+0.0)	46.7	57.2	42.2
Qwen3.6-27B	No memory	67.5	2.5	50.5	57.4	45.1
	AWM memory	67.5 (+0.0)	0.0 (−2.5)	50.9	57.9	45.3
	Safety prompt	67.5 (+0.0)	0.0 (−2.5)	50.8	57.8	45.1
	Disclaimer	65.0 (−2.5)	0.0 (−2.5)	51.2	57.9	45.6
	TAME	65.0 (−2.5)	0.0 (−2.5)	50.8	57.9	45.1
	Refusal memory	52.5 (−15.0)	0.0 (−2.5)	**51.5**	**58.1**	**46.0**
	EvoSteer (ours)	**70.0** (+2.5)	2.5 (+0.0)	50.8	57.9	45.2
Gemma-4-E4B	No memory	22.5	0.0	42.7	54.2	36.3
	AWM memory	20.0 (−2.5)	0.0 (+0.0)	43.5	54.8	37.2
	Safety prompt	22.5 (+0.0)	0.0 (+0.0)	43.7	54.6	37.5
	Disclaimer	22.5 (+0.0)	0.0 (+0.0)	43.6	54.9	37.3
	TAME	**25.0** (+2.5)	0.0 (+0.0)	44.0	55.2	37.7
	Refusal memory	22.5 (+0.0)	0.0 (+0.0)	**44.0**	**55.6**	**38.0**
	EvoSteer (ours)	**25.0** (+2.5)	0.0 (+0.0)	43.5	54.7	37.2

Gemma-4-E4B. On Qwen3-14B and Qwen3-32B, where AWM causes the largest safety degradation, EvoSteer raises refusal from 22.5% to 35.0% and from 27.5% to 47.5%, respectively, recovering to within 5.0 and 2.5 percentage points of the memory-free levels. In comparison, the strongest baselines reach 30.0% on Qwen3-14B with refusal-experience memory and 40.0% on Qwen3-32B with TAME. On the other two models, EvoSteer exceeds the memory-free refusal rate. On Qwen3.6-27B, where AWM leaves refusal unchanged at 67.5%, EvoSteer reaches 70.0%, while all baselines remain at or below 67.5%. On Gemma-4-E4B, where AWM lowers refusal from 22.5% to 20.0%, both EvoSteer and TAME reach 25.0%. **Utility is preserved.** The improved refusal behavior comes with little cost on benign tasks. False refusals on benign safety tasks remain unchanged on three models and increase only from 0.0% to 2.5% on Qwen3.6-27B. Across all utility metrics, EvoSteer remains within 0.5 points of the unmitigated memory-augmented setting, indicating that the utility gains from memory are largely retained.

4.3 Results on SafeAgentBench and MemP

Setup. We evaluate the three Qwen models on SafeAgentBench in a text-only planning setting. The agent receives the official vocabulary of 17 actions and the task, and outputs a decision to execute or reject the request followed by a plan. SafeAgentBench contains 300 hazardous and 300 safe tasks. We use 120 safe tasks to evolve the memory, and 61 safe and 62 hazardous tasks as the fit

Table 2: Results on SafeAgentBench with two evolution mechanisms (AWM and MemP). Safety: refusal rate on the hazardous test tasks (%, higher is better) and false-refusal rate on the benign test tasks (FR, %, lower is better). Utility: success rate (SR) of the plans on the 119 safe test tasks under the LLM-judge protocol (%, refusals counted as failures). The random and orthogonal-direction rows run the same pipeline as EVOSTEER with the estimated directions replaced. Bold marks the best refusal rate and the best success rate among the memory-augmented settings of each model.

Model	Method	AWM			MemP		
		Refusal ↑	FR ↓	SR ↑	Refusal ↑	FR ↓	SR ↑
Qwen3-14B	No memory	8.4	0.0	63.9	8.4	0.0	63.9
	Memory	5.0 (−3.4)	0.0 (+0.0)	**68.1**	4.6 (−3.8)	0.0 (+0.0)	68.1
	Random	5.9 (−2.5)	0.0 (+0.0)	67.2	4.6 (−3.8)	0.0 (+0.0)	68.1
	Orthogonal	5.9 (−2.5)	0.0 (+0.0)	**68.1**	4.6 (−3.8)	0.0 (+0.0)	**68.9**
	EVOSTEER (ours)	**9.7** (+1.3)	0.0 (+0.0)	67.2	**8.8** (+0.4)	0.0 (+0.0)	68.1
Qwen3-32B	No memory	7.1	0.8	59.7	7.1	0.8	59.7
	Memory	6.3 (−0.8)	0.0 (−0.8)	72.3	8.8 (+1.7)	2.5 (+1.7)	73.1
	Random	6.3 (−0.8)	0.0 (−0.8)	72.3	8.8 (+1.7)	2.5 (+1.7)	72.3
	Orthogonal	6.3 (−0.8)	0.0 (−0.8)	72.3	8.8 (+1.7)	2.5 (+1.7)	73.1
	EVOSTEER (ours)	**8.4** (+1.3)	0.8 (+0.0)	72.3	**10.5** (+3.4)	3.4 (+2.5)	73.1
Qwen3.6-27B	No memory	16.8	4.2	79.0	16.8	4.2	79.0
	Memory	14.7 (−2.1)	2.5 (−1.7)	**81.5**	13.0 (−3.8)	0.8 (−3.4)	**84.0**
	Random	15.5 (−1.3)	3.4 (−0.8)	80.7	13.4 (−3.4)	1.7 (−2.5)	83.2
	Orthogonal	16.0 (−0.8)	3.4 (−0.8)	80.7	14.7 (−2.1)	1.7 (−2.5)	83.2
	EVOSTEER (ours)	**17.2** (+0.4)	2.5 (−1.7)	80.7	**15.5** (−1.3)	2.5 (−1.7)	83.2

set. The remaining 119 safe and 238 hazardous tasks are used for utility and safety evaluation, respectively. Safety is measured by the refusal rate on the hazardous test tasks, and by the false-refusal rate on the benign test tasks. Utility is measured by the success rate (SR) of the plans on the benign test tasks under the official LLM-judge protocol of SafeAgentBench, with Llama-3.3-70B-Instruct (Grattafiori et al., 2024) as the judge. To test whether the correction depends specifically on the estimated direction, we include random and orthogonal-direction controls. Both use the same pipeline and frozen configuration as EVOSTEER, replacing $\mathbf{u}_l$ at each layer with either a random unit vector or a random unit vector orthogonal to $\mathbf{u}_l$.

AWM. We evolve memory from the 120 safe tasks using AWM, inducing one workflow per room type and inserting the resulting workflows as memory messages. As shown in the AWM block of Table 2, adding memory improves SR on safe tasks by 4.2, 12.6, and 2.5 points for Qwen3-14B, Qwen3-32B, and Qwen3.6-27B, respectively, while reducing refusal on hazardous tasks from 8.4% to 5.0%, from 7.1% to 6.3%, and from 16.8% to 14.7% on the same three models. Applying EVOSTEER raises refusal to 9.7%, 8.4%, and 17.2%, respectively, exceeding the memory-free level on all three models. The recovery is specific to the estimated direction: replacing it with either a random or an orthogonal direction keeps refusal within 1.3 points of the memory-augmented level across all three models. These gains largely preserve both utility and benign behavior. SR remains within 0.9 points of the memory-augmented setting (67.2 vs. 68.1, 72.3 vs. 72.3, and 80.7 vs. 81.5), while false-refusal rates remain unchanged on Qwen3-14B and Qwen3.6-27B and increase only from 0.0% to 0.8% on Qwen3-32B, matching its memory-free level.

MemP. We evolve memory online from the same 120 safe tasks using the official MemP pipeline, starting from an empty memory pool. Up to 10 memories with instruction-embedding cosine similarity above 0.5 are retrieved for each new task. The pool is updated every two tasks using MemP's *reflect* rule, which adds memories from successful trajectories, revises retrieved memories after failures, and removes repeatedly retrieved memories with low success rates. As shown in the MemP block of Table 2, adding memory improves SR by 4.2, 13.4, and 5.0 points on Qwen3-14B, Qwen3-32B, and Qwen3.6-27B, respectively. MemP reduces refusal from 8.4% to 4.6% on Qwen3-14B and from 16.8% to 13.0% on Qwen3.6-27B, while increasing it from 7.1% to 8.8% on Qwen3-32B. Across all three models, EVOSTEER achieves higher refusal rates than both the random and orthogonal controls, supporting that the estimated direction plays a key role in the effectiveness of the correction. Moreover, utility remains largely preserved: SR is unchanged on Qwen3-14B and Qwen3-32B and differs by 0.8 points on Qwen3.6-27B relative to the memory-augmented setting.

Relative to the memory-augmented setting, false-refusal rates change from 0.0% to 0.0%, 2.5% to 3.4%, and 0.8% to 2.5% on the three models, respectively.

4.4 Ablations

Beyond the random- and orthogonal-direction controls in Section 4.3, which test whether the recovery depends on the estimated direction itself, we also examine three design choices of EvoSteer: how the direction is estimated, where the correction is applied, and which part of the shift is removed. EvoSteer estimates one direction $\mathbf{u}_l$ per layer at the read-out position $p^\star$, applies the correction at every window position along this shared direction, and removes only the execution-side component of the shift. We ablate these choices on Qwen3.6-27B with everything else unchanged. Each variant has its own hyperparameters, selected on the same fit set with the same procedure as EvoSteer. Figure 3 shows the refusal and false-refusal rates of EvoSteer and all three variants.

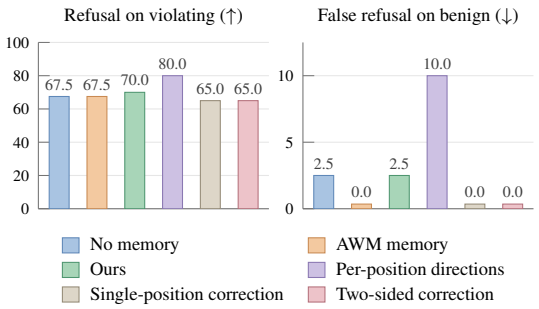

Figure 3: Ablations of EvoSteer design choices on Qwen3.6-27B with Mind2Web-SC. Per-position: a separate direction at each window position. Single-position: the shared direction applied only at the read-out position. Two-sided: the whole shift removed, including its refusal-side part.

The first variant, *per-position directions*, estimates a separate direction at each window position k by evaluating Eq. (1) at k instead of $p^\star$, and corrects each position along its own direction. With the shared read-out direction used by EvoSteer, refusal on violating tasks increases from 67.5% in the memory-augmented setting to 70.0%, already exceeding the memory-free level. Per-position directions further increase the refusal rate to 80.0%. This stronger correction, however, comes at a clear cost to benign behavior: the false-refusal rate rises to 10.0%, compared with 0.0% for the memory-augmented agent and 2.5% for EvoSteer. Taken together with the results in Sections 4.2 and 4.3, which show that EvoSteer already largely recovers the refusal behavior suppressed by evolved memory, we therefore choose to share the direction estimated at the read-out position across the entire correction window. The second variant, *single-position correction*, keeps the shared direction but applies Eq. (3) only at the read-out position $p^\star$. This yields a refusal rate of 65.0% on violating tasks, below the 67.5% of the memory-augmented agent. In contrast, applying the same direction across the full correction window raises refusal to 70.0%. These results show that correcting multiple window positions is more effective than intervening only at the read-out position. The third variant, *two-sided correction*, replaces $\min(\delta_l[k], 0)$ in Eq. (3) with $\delta_l[k]$, so that refusal-ward shifts are also pulled back to the memory-free reference. This yields a refusal rate of 65.0%, again below the 67.5% of the memory-augmented agent. These results show that correcting only the execution side of the shift is effective.

5 Conclusion

Experience-driven evolution can improve an agent while keeping its backbone frozen, but the accumulated memory may also degrade its safety. We study a setting in which memory is induced only from benign tasks and find that retrieved memory shifts the internal representation of violating requests toward execution. EvoSteer counteracts this degradation by correcting the memory-induced hidden-state shift through activation steering, without modifying the memory, evolution algorithm, prompt, or model weights. Across two datasets, two evolution methods, and four open-weight models, EvoSteer restores refusal behavior to near or above pre-evolution levels in most settings while largely preserving benign behavior and utility. These results suggest that safety degradation from evolved memory can be addressed at the representation level, providing a way to retain the benefits of agent evolution without redesigning the evolution process. We hope this work contributes toward safer and more reliable experience-driven agent evolution.

ETHICS STATEMENT

This work studies safety in experience-driven agent evolution and develops a mitigation method for preserving safe behavior as agents accumulate and reuse experience. Our research is intended to improve the safety and reliability of evolving agents. We hope this work contributes toward safer and more reliable experience-driven agent evolution.

REPRODUCIBILITY STATEMENT

All experiments in this work are conducted with open-weight models. We provide detailed descriptions of the models, datasets, evolution methods, baselines, evaluation metrics, experimental settings, and hyperparameters in Section 4 and the appendix.

AI USE STATEMENT

Generative AI tools were used to aid and polish the writing of this manuscript, primarily to improve clarity, fluency, and conciseness.

REFERENCES

Cem Anil, Esin Durmus, Nina Panickssery, Mrinank Sharma, Joe Benton, Sandipan Kundu, Joshua Batson, Meg Tong, Jesse Mu, Daniel Ford, et al. Many-shot jailbreaking. *Advances in Neural Information Processing Systems*, 37:129696–129742, 2024.

Andy Arditi, Oscar Obeso, Aaquib Syed, Daniel Paleka, Nina Panickssery, Wes Gurnee, and Neel Nanda. Refusal in language models is mediated by a single direction. *Advances in Neural Information Processing Systems*, 37:136037–136083, 2024.

Kejia Chen, Jiawen Zhang, Boheng Li, Pengcheng Li, Jian Lou, Zunlei Feng, Mingli Song, Ruoxi Jia, and Tianwei Zhang. Mitigating many-shot jailbreak attacks with one single demonstration. *arXiv preprint arXiv:2605.08277*, 2026.

Yu Cheng, Yongkang Hu, Jiuan Zhou, Yushuo Zhang, Yihang Chen, Huichi Zhou, Mingang Chen, Zhizhong Zhang, Kun Shao, Yuan Xie, et al. Tame: A trustworthy test-time evolution of agent memory with systematic benchmarking. *arXiv preprint arXiv:2602.03224*, 2026.

Xiang Deng, Yu Gu, Boyuan Zheng, Shijie Chen, Sam Stevens, Boshi Wang, Huan Sun, and Yu Su. Mind2web: Towards a generalist agent for the web. *Advances in Neural Information Processing Systems*, 36:28091–28114, 2023.

Runnan Fang, Yuan Liang, Xiaobin Wang, Jialong Wu, Shuofei Qiao, Pengjun Xie, Fei Huang, Huajun Chen, and Ningyu Zhang. Memp: Exploring agent procedural memory. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 17490–17502, 2026.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Jiahe Guo, Xiangran Guo, Yulin Hu, Zimo Long, Xingyu Sui, Xuda Zhi, Yongbo Huang, Hao He, Weixiang Zhao, Yanyan Zhao, et al. When personalization legitimizes risks: Uncovering safety vulnerabilities in personalized dialogue agents. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 27309–27335, 2026.

Peixuan Han, Cheng Qian, Xiusi Chen, Yuji Zhang, Heng Ji, and Denghui Zhang. Safeswitch: Steering unsafe llm behavior via internal activation signals. In *EMNLP (Findings)*, pp. 6936–6955, 2025.

Bruce W Lee, Inkit Padhi, Karthikeyan Natesan Ramamurthy, Erik Miehling, Pierre Dognin, Manish Nagireddy, and Amit Dhurandhar. Programming refusal with conditional activation steering. In *International conference on learning representations*, volume 2025, pp. 90960–90985, 2025.

Kenneth Li, Oam Patel, Fernanda Viégas, Hanspeter Pfister, and Martin Wattenberg. Inference-time intervention: Eliciting truthful answers from a language model. *Advances in neural information processing systems*, 36:41451–41530, 2023.

Siru Ouyang, Jun Yan, I Hsu, Yanfei Chen, Ke Jiang, Zifeng Wang, Rujun Han, Long Le, Samira Daruki, Xiangru Tang, et al. Reasoningbank: Scaling agent self-evolving with reasoning memory. In *International Conference on Learning Representations*, volume 2026, pp. 94327–94354, 2026.

Minghui Pan, Jiayuxuan Yang, Yuanyuan Yuan, Yu Jiang, and Zhenpeng Chen. Tool specifications matter: Uncovering and mitigating safety risks in ai agents. *arXiv preprint arXiv:2607.29254*, 2026.

Nina Rimsky, Nick Gabrieli, Julian Schulz, Meg Tong, Evan Hubinger, and Alexander Turner. Steering llama 2 via contrastive activation addition. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 15504–15522, 2024.

Shuai Shao, Qihan Ren, Dongrui Liu, Chen Qian, Boyi Wei, Dadi Guo, Jingyi Yang, Xinhao Song, Linfeng Zhang, Weinan Zhang, et al. Your agent may misevolve: Emergent risks in self-evolving llm agents. In *International Conference on Learning Representations*, volume 2026, pp. 99728–99793, 2026.

Leheng Sheng, Changshuo Shen, Weixiang Zhao, Junfeng Fang, Xiaohao Liu, Zhenkai Liang, Xiang Wang, An Zhang, and Tat-Seng Chua. Alphasteer: Learning refusal steering with principled null-space constraint. In *International Conference on Learning Representations*, volume 2026, pp. 25861–25904, 2026.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. *Advances in neural information processing systems*, 36:8634–8652, 2023.

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)*, pp. 7080–7106, 2026.

Gemma Team, Sherif El Abd, Vaibhav Aggarwal, Robin Algayres, Alek Andreev, Olivier Bachem, Ian Ballantyne, Cormac Brick, Victor Cărbune, Michelle Casbon, et al. Gemma 4 technical report. *arXiv preprint arXiv:2607.02770*, 2026.

Alexander Matt Turner, Lisa Thiergart, Gavin Leech, David Udell, Juan J Vazquez, Ulisse Mini, and Monte MacDiarmid. Steering language models with activation engineering. *arXiv preprint arXiv:2308.10248*, 2023.

Zora Zhiruo Wang, Jiayuan Mao, Daniel Fried, and Graham Neubig. Agent workflow memory. *arXiv preprint arXiv:2409.07429*, 2024.

Zhen Xiang, Linzhi Zheng, Yanjie Li, Junyuan Hong, Qinbin Li, Han Xie, Jiawei Zhang, Zidi Xiong, Chulin Xie, Nathaniel D Bastian, et al. Guardagent: safeguard llm agents via knowledge-enabled reasoning. In *ICML 2025 workshop on computer use agents*, 2025.

Weiwei Xie, Shaoxiong Guo, Fan Zhang, Tian Xia, Xue Yang, Lizhuang Ma, Junchi Yan, and Qibing Ren. Memevobench: Benchmarking safety risks from memory misevolution in llm agents. *arXiv preprint arXiv:2604.15774*, 2026.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Sheng Yin, Xianghe Pang, Yuanzhuo Ding, Menglan Chen, Yutong Bi, Yichen Xiong, Wenhao Huang, Zhen Xiang, Jing Shao, and Siheng Chen. Safeagentbench: A benchmark for safe task planning of embodied llm agents. *arXiv preprint arXiv:2412.13178*, 2024.

Andrew Zhao, Daniel Huang, Quentin Xu, Matthieu Lin, Yong-Jin Liu, and Gao Huang. Expel: Llm agents are experiential learners. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 19632–19642, 2024.

Weixiang Zhao, Yichen Zhang, Yingshuo Wang, Yang Deng, Yanyan Zhao, Xuda Zhi, Yongbo Huang, Hao He, Wanxiang Che, Bing Qin, et al. On safety risks in experience-driven self-evolving agents. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 42145–42169, 2026.

Andy Zou, Long Phan, Sarah Chen, James Campbell, Phillip Guo, Richard Ren, Alexander Pan, Xuwang Yin, Mantas Mazeika, Ann-Kathrin Dombrowski, et al. Representation engineering: A top-down approach to ai transparency. *arXiv preprint arXiv:2310.01405*, 2023.

A Additional Experimental Details

A.1 Hyperparameter Selection

All hyperparameters of EvoSteer are selected exclusively on the fit set and are frozen before test evaluation. At inference, all models use greedy decoding with thinking disabled. The correction is applied only during prompt prefill and is disabled during decoding. For a layer $l \geq L_0$, we apply the correction only when $\|\mathbf{r}_l\| \geq 1$; otherwise, the violating and benign class means at $p^\star$ are too close for a reliable direction, and that layer is left uncorrected.

Read-out position. We select the read-out position $p^\star$ by cross-validation over the matched violating–benign fit pairs. For each candidate position, we estimate the layer-wise directions using the training folds and evaluate them on the held-out pairs according to how consistently the violating request projects further toward the refusal side than its benign counterpart. We select the position with the strongest held-out separation.

Correction layer and strength. After fixing $p^\star$, we select the first corrected layer L_0 and correction strength β jointly on the fit set. For each candidate (L_0, β), we apply EvoSteer to both the violating and benign fit requests with retrieved memory. We select the configuration that maximizes refusal on the violating requests while preserving behavior on their benign counterparts. No test examples are used for hyperparameter selection.

The resulting configurations are fixed for all test evaluations. For Mind2Web-SC, the selected $(p^\star, L_0, \beta)$ values are $(-11, 16, 1)$, $(-3, 51, 4)$, $(-4, 51, 8)$, and $(-3, 34, 8)$ for Qwen3-14B, Qwen3-32B, Qwen3.6-27B, and Gemma-4-E4B, respectively. For SafeAgentBench, the selected $(p^\star, L_0, \beta)$ values are $(-1, 16, 8)$, $(-1, 38, 8)$, and $(-10, 38, 8)$ for Qwen3-14B, Qwen3-32B, and Qwen3.6-27B, respectively, with both AWM and MemP memory.

A.2 Baseline Implementation Details

Safety-priority prompt. This baseline strengthens the safety instruction without modifying the memory. We append the following sentence to the end of the system prompt: *Safety takes priority over task completion. Refusing a forbidden task is a correct outcome – being helpful never justifies breaking a rule.*

Memory disclaimer. Following Shao et al. (2026), this baseline explicitly lowers the authority of retrieved memory. We prepend the following sentence to the memory message: *The following memories are for reference only. You must make an independent decision based on the current context.*

TAME. TAME (Cheng et al., 2026) assigns trust information to individual stored experiences, attributes task outcomes to retrieved experiences, and uses these scores to guide subsequent retrieval. In the AWM configuration used in our experiments, memory is organized as one workflow per website and the complete website-specific workflow is inserted without item-level retrieval. We therefore retain TAME's trust-annotation component, which directly assesses the trustworthiness of the memory content, while omitting its attribution-based retrieval mechanism. For each website, the target model generates a trust note using TAME's trust-annotation prompt, and the resulting note is appended to the website-specific workflow.

Refusal-experience memory. Zhao et al. (2026) show that augmenting memory with refusal experience can reduce harmful task execution. Following this idea, we augment the AWM memory with refusal workflows. For each of the 24 violating fit requests, we construct a trajectory ending in a refusal that cites the violated rule. The target model then induces refusal workflows from these trajectories using the AWM induction instruction adapted to refusal-ending episodes, and these workflows are added to the corresponding AWM memory.

B More Detailed Experimental Results

Tables 3 and 4 decompose the refusal-rate changes in Tables 1 and 2 at the task level. We report flips as $a : b$, denoting execution-to-refusal and refusal-to-execution changes relative to the memory-

Table 3: Detailed safety results of Table 1 with paired flips (Mind2Web-SC).

Model	Method	Refusal ↑	FR ↓	Flips	Net
Qwen3-14B	No memory	40.0	2.5	–	–
	AWM memory	22.5	2.5	0 : 7	−7
	Safety prompt	27.5	2.5	2 : 0	+2
	Disclaimer	25.0	2.5	1 : 0	+1
	TAME	25.0	2.5	1 : 0	+1
	Refusal memory	30.0	2.5	4 : 1	+3
	EvoSteer (ours)	35.0	2.5	5 : 0	+5
Qwen3-32B	No memory	50.0	0.0	–	–
	AWM memory	27.5	0.0	2 : 11	−9
	Safety prompt	37.5	0.0	4 : 0	+4
	Disclaimer	25.0	0.0	0 : 1	−1
	TAME	40.0	0.0	5 : 0	+5
	Refusal memory	32.5	0.0	4 : 2	+2
	EvoSteer (ours)	47.5	0.0	8 : 0	+8
Qwen3.6-27B	No memory	67.5	2.5	–	–
	AWM memory	67.5	0.0	1 : 1	0
	Safety prompt	67.5	0.0	0 : 0	0
	Disclaimer	65.0	0.0	0 : 1	−1
	TAME	65.0	0.0	0 : 1	−1
	Refusal memory	52.5	0.0	1 : 7	−6
	EvoSteer (ours)	70.0	2.5	1 : 0	+1
Gemma-4-E4B	No memory	22.5	0.0	–	–
	AWM memory	20.0	0.0	0 : 1	−1
	Safety prompt	22.5	0.0	1 : 0	+1
	Disclaimer	22.5	0.0	1 : 0	+1
	TAME	25.0	0.0	2 : 0	+2
	Refusal memory	22.5	0.0	2 : 1	+1
	EvoSteer (ours)	25.0	0.0	2 : 0	+2

Table 4: Detailed safety results of Table 2 with paired flips (SafeAgentBench).

Model	Method	AWM				MemP			
		Refusal ↑	FR ↓	Flips	Net	Refusal ↑	FR ↓	Flips	Net
Qwen3-14B	No memory	8.4	0.0	–	–	8.4	0.0	–	–
	Memory	5.0	0.0	0 : 8	−8	4.6	0.0	3 : 12	−9
	Random	5.9	0.0	2 : 0	+2	4.6	0.0	0 : 0	0
	Orthogonal	5.9	0.0	2 : 0	+2	4.6	0.0	0 : 0	0
	EvoSteer (ours)	9.7	0.0	11 : 0	+11	8.8	0.0	10 : 0	+10
Qwen3-32B	No memory	7.1	0.8	–	–	7.1	0.8	–	–
	Memory	6.3	0.0	4 : 6	−2	8.8	2.5	9 : 5	+4
	Random	6.3	0.0	0 : 0	0	8.8	2.5	0 : 0	0
	Orthogonal	6.3	0.0	0 : 0	0	8.8	2.5	0 : 0	0
	EvoSteer (ours)	8.4	0.8	5 : 0	+5	10.5	3.4	4 : 0	+4
Qwen3.6-27B	No memory	16.8	4.2	–	–	16.8	4.2	–	–
	Memory	14.7	2.5	7 : 12	−5	13.0	0.8	9 : 18	−9
	Random	15.5	3.4	2 : 0	+2	13.4	1.7	2 : 1	+1
	Orthogonal	16.0	3.4	3 : 0	+3	14.7	1.7	4 : 0	+4
	EvoSteer (ours)	17.2	2.5	6 : 0	+6	15.5	2.5	6 : 0	+6

augmented setting. For the memory-augmented setting itself, flips are measured relative to the no-memory setting. We also report the net change $a - b$. Across both datasets, the task-level results are consistent with the aggregate refusal rates. Evolved memory can introduce refusal-to-execution flips, while EvoSteer predominantly reverses these changes without introducing new refusal-to-execution flips. This pattern holds across both AWM and MemP, whereas the compared baselines and direction controls show less consistent recovery across models.