# SCORE: Training-Free Agent Adaptation via Same-State Counterfactual Rule Editing

Anonymous ACL submission

## Abstract

Online agent learning typically uses next-state feedback to update policy parameters. We ask whether the same signal can instead update an editable external memory that conditions future inference. We introduce Same-State Counterfactual Rule Editing (SCORE), a training-free framework that converts interaction feedback into reusable decision rules. From each realized trajectory, SCORE recovers evaluative and directive hindsight from the observed next state, ranks reconstructed same-state candidates, and writes winner-loser memories into an external experience library. At test time, retrieved memories act as compact policy priors while the base model remains frozen. Across personalization, AIME24 tool use, a ReTool evaluation with Qwen3-4B-ReTool-SFT, and controlled tool-decision tasks, SCORE improves adaptation where feedback can be compressed into reusable rules, reduces tool-use overhead relative to retrieval-heavy baselines, and outperforms prompting, reflection, GRPO, and OpenClaw-RL on the Qwen3-4B-ReTool-SFT ReTool evaluation. These results show that realized next-state feedback can support editable context-space memory for online agent adaptation.

## 1 Introduction

Language-model agents increasingly operate in interactive settings where each action produces a *next state*: a user reply, a tool result, a terminal output, or a simulator response. Recent online agent reinforcement learning work shows that such next-state signals are rich enough to support continual policy improvement directly from interaction (Wang et al., 2026). In parallel, training-free optimization work suggests that context-space priors and experience libraries can realize part of this improvement at inference time (Cai et al., 2025).

This paper asks a concrete question: can the evaluative and directive signal recovered from next states update an external memory that conditions future behavior? We explore this question through SCORE, a training-free framework for online agent adaptation.

SCORE starts from one observation. In many interactive tasks, especially personalization and tool use, the learnable signal lies at a *decision boundary*: when to ask a clarifying question, when to respond in a natural tone, when to trust an intermediate result, and when to invoke a verification tool. These boundaries are often expressible as short reusable policy rules. If such rules are recovered from next-state signal and re-injected in later similar situations, external memory can carry part of the adaptation burden usually assigned to online finetuning.

The main challenge is that a live online agent usually produces one realized action, while group-based training-free optimization methods rely on relative comparisons within a candidate set. SCORE addresses this with *counterfactual group reconstruction*: for each learnable interaction, it recovers evaluative and directive hindsight from the observed next state, reconstructs a same-state candidate group containing the original action and several alternatives, ranks that group under a shared task-specific scorer, and writes reusable memory only when the resulting candidates show enough semantic separation.

SCORE therefore edits a compact experience library in context space, stores short trigger–action rules with explicit anti-patterns, and learns from same-state relative comparisons.

Our contributions are threefold: we formulate a *training-free framework for online agent adaptation* that updates a structured experience library from realized interaction feedback; we introduce a practical *same-state counterfactual rule-editing* pipeline that combines same-state group recon-

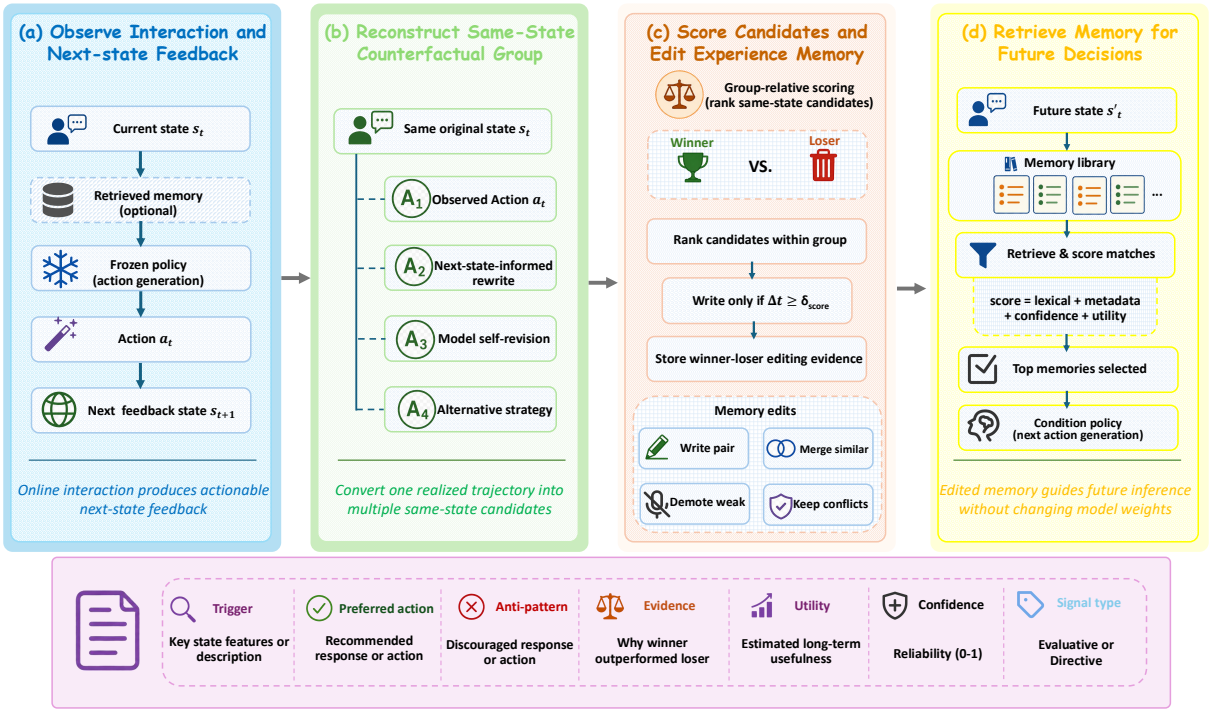


Figure 1: Overview of SCORE. The agent retrieves compact experience memories, acts with a frozen model, observes the next state, reconstructs a same-state counterfactual group, edits evaluative and directive memories, and reuses the updated library on later interactions. Utility statistics close the loop by reweighting memories after downstream success or failure.

struction, semantic-margin gating, and dual-signal memory writing; and we provide experiments on *personalization* and *stateful tool use*. SCORE improves several personalization and tool-choice settings across multiple models and outperforms prompting, reflection, GRPO, and OpenClaw-RL on a ReTool evaluation with Qwen3-4B-ReTool-SFT.

Our results show that *same-state counterfactual rule editing in context space is an effective training-free mechanism for online agent adaptation*, especially when the target behavior can be captured as stable reusable decision rules.

## 2 Related Work

### 2.1 Online Agent Reinforcement Learning

OpenClaw-RL argues that an agent can be trained "simply by talking" because next-state signals encode both reward-like evaluative information and text-like directive information (Wang et al., 2026). This perspective is closely related to earlier efforts on feedback-driven language model alignment, including RLHF (Ouyang et al., 2022; Bai et al., 2022) and preference optimization (Rafailov et al., 2023). Our work shares OpenClaw-RL's online interaction setting and moves the learned substrate into an external experience library.

### 2.2 Training-Free and Group-Based Optimization

Training-Free GRPO shows that group-relative comparison can be externalized into a token prior stored in an evolving experience library (Cai et al., 2025). GRPO itself emerged as a strong recipe for reasoning-oriented language model optimization, especially in mathematical domains (Shao et al., 2024). Standard PPO remains a canonical baseline for stable policy optimization (Schulman et al., 2017). Our method inherits the *group-relative* intuition from Training-Free GRPO and adapts it to online agent interaction where candidate groups are reconstructed from a single realized trajectory.

### 2.3 Reasoning, Reflection, and Self-Improvement

Recent work has explored various forms of language-model self-improvement, including chain-of-thought prompting (Wei et al., 2022), self-consistency (Wang et al., 2022), self-refinement (Madaan et al., 2023), verbal reinforcement (Shinn et al., 2023), and process supervision (Lightman et al., 2024). SCORE

contributes a same-state group reconstruction step for relative comparison and persists the extracted policy rule in an external library that can influence future trajectories.

### 2.4 Agents, Tools, and External Memory

Tool-using agents such as ReAct (Yao et al., 2022), Toolformer (Schick et al., 2023), WebGPT (Nakano et al., 2021), Voyager (Wang et al., 2023), and more recent agent frameworks (Park et al., 2023; Packer et al., 2024; Qin et al., 2024) have shown that external tools and memory can significantly extend model capability. Retrieval-augmented methods similarly externalize knowledge into external stores (Guu et al., 2020; Lewis et al., 2020; Borgeaud et al., 2022). SCORE treats the external store as an *online-updated policy library* whose entries are written, merged, and reweighted through interaction feedback.

### 2.5 Positioning

SCORE sits at the intersection of four threads: online agent RL, training-free optimization, tool-use policy learning, and editable external memory. Concretely, it instantiates training-free group-relative policy improvement for online agents by reconstructing counterfactual same-state groups and writing the resulting signal into an editable experience library.

## 3 Method

### 3.1 Overview

At interaction step $t$, the agent observes state $s_t$, takes action $a_t$, and then receives next state $s_{t+1}$. The base language model is frozen throughout. The only mutable object is an external experience library $\mathcal{E}$. Each interaction cycles through memory retrieval, frozen-policy action generation, next-state observation, hindsight signal recovery, counterfactual group reconstruction, same-state candidate ranking, and experience editing.

Figure 1 summarizes the closed loop from next-state feedback to memory editing and later retrieval-time behavior.

Algorithm 1 states the shared online update used across personalization and tool use, omitting the retrieval-calibration term used only in the controlled tool-choice ablation.

**Algorithm 1** SCORE Online Adaptation

**Require:** frozen policy $\pi$, library $\mathcal{E}$, retriever Ret, scorer $J$, threshold $\delta_{\text{score}}$

**for** $t = 1, \ldots, T$ **do**
  $\sigma_t \leftarrow \text{Summarize}(s_t)$
  $\mathcal{M}_t \leftarrow \text{TopK}(\text{Ret}(\sigma_t, \mathcal{E}))$
  $a_t \leftarrow \pi(s_t \mid \mathcal{M}_t)$
  $s_{t+1} \leftarrow \text{ObserveNext}(s_t, a_t)$
  $(v_t, d_t) \leftarrow \text{ExtractSignals}(s_{t+1})$
  $G_t \leftarrow \text{Reconstruct}(s_t, a_t)$
  **for all** $a \in G_t$ **do**
    $r(a) \leftarrow J(s_t, a)$
  **end for**
  $\Delta_t \leftarrow \max_{a \in G_t} r(a) - \min_{a \in G_t} r(a)$
  **if** $\Delta_t \geq \delta_{\text{score}}$ and $n_{\text{uniq}}(G_t) \geq 2$ **then**
    $a^+ \leftarrow \arg\max_{a \in G_t} r(a)$
    $a^- \leftarrow \arg\min_{a \in G_t} r(a)$
    $(e^+, e^-) \leftarrow \text{WritePair}(s_t, a^+, a^-, v_t, d_t)$
    $\mathcal{E} \leftarrow \text{EditLibrary}(\mathcal{E} \cup \{e^+, e^-\})$
  **end if**
  $\mathcal{E} \leftarrow \text{UpdateUtility}(\mathcal{E}, \mathcal{M}_t, s_{t+1})$
**end for**
**return** adapted experience library $\mathcal{E}$

### 3.2 Experience Representation and Library Edits

Each memory item stores a short policy rule with a trigger, a preferred action, an anti-pattern, a signal type, reliability statistics, and compact winner–loser evidence when available. This factorization makes the library editable at the rule level: SCORE can add winner–loser rule pairs, merge repeated support, preserve competing branches, and reweight rules through confidence and utility after later retrieval outcomes. The model weights remain fixed, so adaptation occurs only through the evolving state of the external library.

### 3.3 Dual-Signal Learning from Next States

Following OpenClaw-RL (Wang et al., 2026), we recover two distinct learning signals from $s_{t+1}$.

Evaluative signal indicates whether the action was successful, partially correct, or incorrect under task-specific criteria. In personalization, the evaluator focuses on user-preference satisfaction and answer quality. In tool choice, it also evaluates whether the agent made the correct call/no-call decision.

Directive signal extracts an explicit hindsight instruction, for example, to stop and map an existing result to the target quantity, or to issue exactly one verification call when a visible intermediate is likely off-target.

Separating these two signals matters in practice. Evaluative memories encode more global policy regularities, while directive memories carry

Table 1: Personalization test-set results across three model backbones, with 48 adaptation interactions and 200 evaluation examples per preference family. Values are run means over three runs. Avg Score is the mean final response score across the three preference families; Avg Comp. is the mean completion rate.

| Model | Method | Avg Score | Avg Comp. | Student | Teacher | Clarify-first |
|---|---|---|---|---|---|---|
| DeepSeek-Chat | Frozen Base | 0.7080 | 0.5744 | 0.7242 | 0.7865 | 0.6133 |
| | Raw History Memory | 0.7526 | 0.7067 | 0.7208 | 0.8082 | 0.7289 |
| | Self-Refine | 0.7836 | 0.6722 | 0.7275 | 0.7887 | 0.8347 |
| | Reflexion | 0.7912 | 0.7378 | 0.7287 | 0.8065 | 0.8382 |
| | MemoryBank | 0.7877 | 0.6978 | 0.7256 | 0.8037 | 0.8338 |
| | ExpeL | 0.7704 | 0.6378 | 0.7234 | 0.8055 | 0.7824 |
| | **SCORE** | **0.8442** | **0.8067** | **0.7305** | **0.9010** | **0.9011** |
| Qwen-Plus | Frozen Base | 0.7283 | 0.6222 | 0.6261 | 0.8413 | 0.7175 |
| | Raw History Memory | 0.6917 | 0.5989 | 0.5333 | 0.8533 | 0.6884 |
| | Self-Refine | **0.7814** | **0.6600** | 0.5997 | 0.8572 | **0.8874** |
| | Reflexion | 0.7204 | 0.5989 | 0.6160 | 0.8285 | 0.7167 |
| | MemoryBank | 0.7620 | 0.6244 | 0.6170 | 0.8535 | 0.8155 |
| | ExpeL | 0.7587 | 0.6111 | 0.6351 | 0.8323 | 0.8087 |
| | **SCORE** | 0.7793 | 0.6456 | **0.6491** | **0.8620** | 0.8268 |
| GPT-5.4-Nano | Frozen Base | 0.7791 | 0.7700 | **0.7305** | 0.7448 | **0.8620** |
| | Raw History Memory | 0.7646 | 0.7767 | 0.7202 | 0.7615 | 0.8120 |
| | Self-Refine | 0.7736 | 0.7833 | 0.7228 | 0.7408 | 0.8571 |
| | Reflexion | 0.7604 | 0.7378 | 0.7196 | 0.7743 | 0.7874 |
| | MemoryBank | 0.7634 | 0.8022 | 0.7114 | 0.7557 | 0.8233 |
| | ExpeL | 0.7131 | 0.7711 | 0.7186 | 0.5927 | 0.8280 |
| | **SCORE** | **0.7801** | **0.8500** | 0.7216 | **0.7780** | 0.8407 |

sharper local corrections.

### 3.4 Counterfactual Group Reconstruction

The key obstacle to adapting Training-Free GRPO ideas to online agents is that interaction naturally yields one realized action per state. We therefore reconstruct a synthetic same-state group containing the original response, a hindsight-conditioned rewrite, a self-revision, and a diverse alternative strategy.

All candidates are generated from the *same pre-action state*. The observed next state from the realized action supplies the evaluative and directive hindsight signals, while a shared task-specific scorer ranks the reconstructed same-state candidates before rule writing. Memory is written only when the candidate set shows enough semantic separation: we require a minimum score margin and at least two distinct candidate responses.

This mechanism gives the method a stronger learning signal than single-trajectory self-reflection. The system learns from *relative differences among same-state strategies*.

### 3.5 Minimal Formalization

Let $G_t = \{a_i\}_{i=1}^{m}$ denote the reconstructed same-state candidate group for state $s_t$, where one candidate is the realized action and the others are counterfactual rewrites.

We score each candidate with the same task-specific evaluation function:

$$r_i = J(s_t, a_i).$$

Here $J$ is the deterministic scorer used for the current task family to rank reconstructed same-state candidates. The directive and evaluative memory content is extracted from the observed next state of the executed action.

Memory is updated only when the candidate group contains real semantic separation:

$$\Delta_t = \max_i r_i - \min_i r_i,$$
$$g_t = \mathbf{1}[\Delta_t \geq \delta_{\text{score}}]\, \mathbf{1}[n_{\text{uniq}}(G_t) \geq 2].$$

Here $\delta_{\text{score}}$ is a minimum score margin and $n_{\text{uniq}}(G_t)$ counts distinct candidate responses. When $g_t = 1$, we write a preferred rule from the best candidate and an anti-pattern from the worst one, together with the directive and evaluative signals extracted from the observed next state.

For each memory item $e \in \mathcal{E}$, retrieval uses a utility-aware score:

$$\text{Ret}(s_t, e) = \lambda_{\text{lex}}\, s_{\text{lex}}(s_t, e) + \lambda_{\text{meta}}\, s_{\text{meta}}(s_t, e) + \lambda_{\text{conf}}\, C(e) + \lambda_{\text{util}}\, U(e).$$

The first two terms match the current state to the memory trigger and metadata, while $C(e)$ and

| Method | Avg Score | Avg Comp. | Student | Teacher | Clarify-first |
|---|---|---|---|---|---|
| SCORE w/o Group Recon. | 0.7087 | 0.5389 | 0.7109 | 0.8173 | 0.5978 |
| SCORE w/o Dir. Mem. | 0.7227 | 0.5544 | 0.7143 | 0.8057 | 0.6480 |
| SCORE w/o Eval. Mem. | 0.7055 | 0.5233 | 0.7173 | 0.8145 | 0.5848 |
| SCORE w/o Exp. Mem. | 0.6910 | 0.5622 | 0.7294 | 0.7883 | 0.5552 |
| SCORE w/o Mem. Editing | 0.8186 | **0.8078** | 0.7158 | 0.8688 | 0.8712 |
| SCORE | **0.8442** | 0.8067 | **0.7305** | **0.9010** | **0.9011** |

Table 2: DeepSeek-Chat personalization ablations. Values are run means. Each row removes one SCORE component under the same evaluation protocol.

| Method | Accuracy (%) ↑ |
|---|---|
| Frozen Base | 25.35 |
| ReAct | 30.56 |
| Reflexion | 31.60 |
| GRPO | 36.46 |
| OpenClaw-RL | 37.50 |
| SCORE | **41.67** |

Table 3: ReTool evaluation results with Qwen3-4B-ReTool-SFT on 96 evaluation problems under the shared code-interpreter protocol. Values are run means over three runs.

| Method | Pass@1 ↑ | Inv. Tool ↓ | Avg. Tool Calls |
|---|---|---|---|
| Frozen Base | 0.2556 | 0.8889 | 4.1444 |
| ReAct | 0.1778 | **0.7778** | 2.8333 |
| Raw Hist. | 0.2889 | 0.8889 | 4.3444 |
| Reflexion | 0.2111 | 0.8556 | 3.1333 |
| SCORE | **0.3222** | 0.8000 | 2.9566 |

Table 4: AIME24 public test-set comparison after 12 shared ReTool adaptation interactions; component ablations are deferred to Appendix A.2.1.

$U(e)$ represent stored confidence and empirical post hoc usefulness. We retrieve the top-$k$ memories under $\mathrm{Ret}$ and inject them as compact policy priors.

### 3.6 Utility-Aware Retrieval

Retrieval combines lexical similarity, task-state metadata, memory confidence, and post hoc usefulness. We also track a utility signal based on whether retrieved memories later participate in successful episodes. In controlled tool-choice analyses, we additionally evaluate a retrieval calibration term that emphasizes verification memories when the current state is ambiguous between direct completion and one more tool call. This term is reported as an ablation; the shared SCORE mechanism remains the same-state counterfactual rule-editing pipeline described above.

## 4 Experimental Setup

### 4.1 Benchmarks

**Personalization.** We use a personalization test suite with three preference families:

- a student-assistance preference for natural, non-formulaic homework help,
- a teacher-feedback preference for encouraging and pedagogically useful comments,
- a process preference that asks the assistant to clarify before giving the final answer.

We instantiate these three preference families over GSM8K; Appendix A.1.1 gives the profile wrappers and split construction. These families intentionally vary in what must be learned: stylistic naturalness, pedagogical tone, and interaction procedure. Each personalization setting uses 48 online adaptation interactions and 200 test examples per preference family.

**Public tool evaluation.** Following OpenClaw-RL's train/eval split, we use ReTool-style interaction traces (Feng et al., 2025) as the online adaptation prefix and AIME24 (Mathematical Association of America, 2026) as a public test set. Concretely, each method first receives 12 shared ReTool training interactions and is then evaluated on all 30 public AIME24 problems under the same code-interpreter protocol. Separately, Table 3 reports a native ReTool evaluation with Qwen3-4B-ReTool-SFT on 96 evaluation problems under the same code-interpreter protocol; this is the direct setting used for comparing SCORE against GRPO and OpenClaw-RL.

**Tool-choice stress tests.** We also evaluate stateful tool-use adaptation on a controlled tool-choice stress test. Each example consists of an original quantitative question plus a prior scratch state produced by a hypothetical previous attempt. The agent must decide whether:

Table 5: Tool-choice stress-test results across three model backbones; overuse/underuse breakdowns are deferred to Appendix A.2.3.

| Model | Method | Success ↑ | Decision ↑ |
|---|---|---|---|
| DeepSeek-Chat | Frozen Base | 0.4847 | 0.5884 |
| | ReAct | 0.3622 | 0.5663 |
| | Raw Hist. | 0.5901 | 0.7092 |
| | Reflexion | 0.3146 | 0.5765 |
| | MemBank | 0.6803 | 0.8129 |
| | ExpeL | 0.6922 | 0.8299 |
| | **SCORE** | **0.7602** | **0.8827** |
| Qwen-Plus | Frozen Base | 0.1582 | 0.3997 |
| | ReAct | 0.0595 | 0.4592 |
| | Raw Hist. | 0.2466 | 0.4456 |
| | Reflexion | 0.0476 | 0.4643 |
| | MemBank | 0.3622 | 0.5000 |
| | ExpeL | **0.4082** | **0.5510** |
| | **SCORE** | 0.3605 | 0.4949 |
| GPT-5.4-Nano | Frozen Base | 0.4235 | 0.5425 |
| | ReAct | 0.3231 | 0.5187 |
| | Raw Hist. | 0.3639 | 0.4779 |
| | Reflexion | 0.3282 | 0.4983 |
| | MemBank | 0.4099 | 0.5833 |
| | ExpeL | 0.3997 | 0.6054 |
| | **SCORE** | **0.5204** | **0.6990** |

- the visible intermediate is already sufficient for a direct finish, or
- the intermediate is off-target, partial, or otherwise insufficient, so one verification call is required.

This design explicitly measures decision boundaries rather than raw arithmetic alone. For this stress test, **raw history memory** retrieves prior interaction snippets without structured policy abstraction, while the structured-memory baselines extract reusable lessons or memory entries.

### 4.2 Metrics

We report deterministic task-specific metrics. Personalization uses test score and completion rate averaged across preference families. AIME24 uses **Pass@1**, invalid-tool rate, and average tool calls. Controlled tool choice uses success rate and decision accuracy, with overuse/underuse diagnostics in Appendix A.2.3.

### 4.3 Experimental Conditions

All experiments reported here are training-free with respect to the policy: the underlying model is never fine-tuned during the run. Only the external experience library is updated.

For the multi-backbone personalization and tool-choice comparisons, we use API-served DeepSeek-Chat (DeepSeek, 2026), Qwen-Plus (Alibaba Cloud, 2026), and GPT-5.4-Nano (OpenAI, 2026) through the same API interface. Within each backbone, all compared methods use the same backend configuration, prompts, evaluation files, and scoring scripts. The ablation tables use DeepSeek-Chat and state that backbone directly in their captions.

We use fixed decoding budgets, deterministic task-specific scorers, and shared aggregation scripts across compared methods within each backbone. Exact generation settings, scorer definitions, and multi-run aggregation details are deferred to Appendix A.1.1.

| Method | Success ↑ | Decision ↑ |
|---|---|---|
| SCORE w/o Group Recon. | 0.6905 | 0.8112 |
| SCORE w/o Dir. Mem. | 0.5714 | 0.6718 |
| SCORE w/o Eval. Mem. | 0.7143 | 0.8265 |
| SCORE w/o Exp. Mem. | 0.4507 | 0.5442 |
| SCORE w/o Mem. Editing | 0.4983 | 0.6037 |
| SCORE w/o State Prior | 0.6480 | 0.7806 |
| SCORE | **0.7602** | **0.8827** |

Table 6: DeepSeek-Chat tool-choice ablations; overuse/underuse breakdowns are deferred to Appendix A.2.3.

### 4.4 Compared Methods

We compare against prompting baselines, reflection/self-refinement baselines, retrieval-memory baselines, direct-experience writing, component ablations, and two parameter-update baselines on the Qwen3-4B-ReTool-SFT ReTool evaluation. Across these comparisons, the acting scaffold, prompts, evaluation files, and scoring pipeline are held fixed within each backbone. For training-free baselines, methods differ in how online experience is stored, edited, or reused. For parameter-update baselines, the adaptation substrate differs between parameter updates and external memory editing. Appendix A.1.2 lists the exact method definitions and implementation details.

## 5 Results

### 5.1 Personalization

Table 1 compares SCORE with external and retrieval-style personalization baselines under three API-served LLM backbones. Table 2 then separates method-comparison evidence from component evidence by removing individual parts of SCORE under the fixed ablation setting.

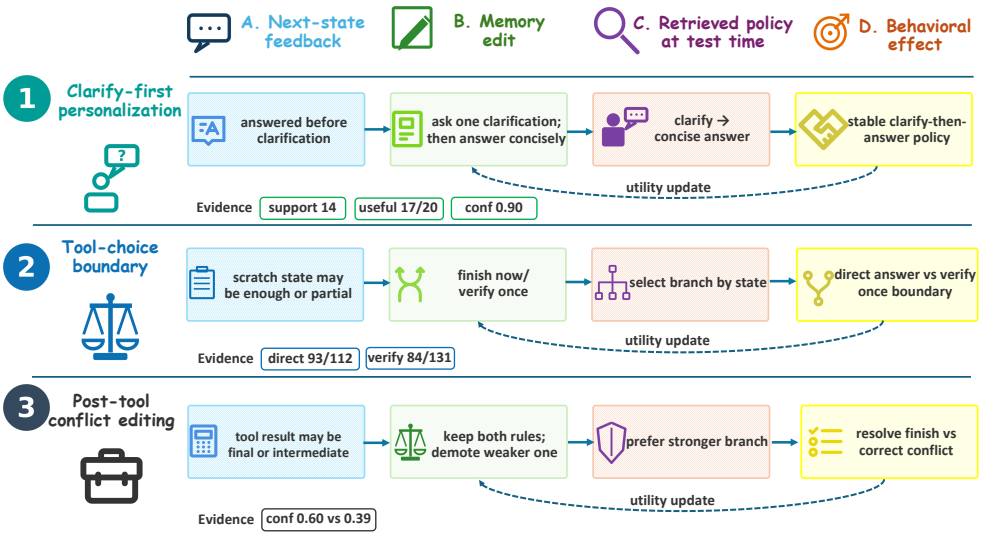


Figure 2: Representative case-study loops from memory snapshots and matched evaluation traces. Each lane shows how next-state feedback edits memory, how the edited rule is later retrieved, and how downstream behavior updates memory utility. Evidence chips report retrieval support, downstream usefulness, branch-specific counts, and post-update confidence.

Across the three hosted backbones, SCORE delivers the strongest personalization results on the DeepSeek-Chat and GPT-5.4-Nano blocks, with especially clear gains on the average and teacher-oriented metrics. The ablation table shows that these gains are driven by memory editing and dual-signal writing when the target behavior must be retained as a stable preference. On Qwen-Plus, SCORE is strongest on the behavior categories most directly tied to persistent preference memory, reinforcing the value of editable preference rules that can be retrieved and reused over time.

### 5.2 Tool Use

We first evaluate the native ReTool comparison that anchors the main parameter-update comparison. Table 3 reports the Qwen3-4B-ReTool-SFT results on 96 ReTool evaluation problems under the shared code-interpreter protocol. SCORE achieves the best accuracy on the same backbone, outperforming prompting, reflection, GRPO, and OpenClaw-RL while keeping the model weights frozen. This is the most direct evidence that context-space rule editing can compete with parameter-update baselines in the native ReTool setting.

We then test external validity on a public mathematical tool-use protocol rather than relying only on the native ReTool evaluation. Table 4 reports the AIME24 comparison after a shared ReTool adaptation prefix, while the corresponding component removals are deferred to Appendix A.2.1. SCORE achieves the best pass@1 among the no-training baselines while requiring substantially fewer tool calls than raw transcript retrieval. This result shows that structured rule editing transfers beyond the native ReTool evaluation and remains effective under a public mathematical tool-use protocol.

Beyond the public benchmark, Table 5 studies controlled tool/no-tool decisions under the same multi-backbone layout used for personalization. Table 6 then zooms in on the strongest-gain setting, DeepSeek-Chat, to identify which SCORE components are actually responsible for that gain, while Appendix A.2.3 reports the corresponding overuse/underuse diagnostic breakdown.

The controlled tool-choice results mirror the personalization pattern: structured rule editing is strongest on DeepSeek-Chat and GPT-5.4-Nano, where it improves both task success and decision accuracy. This multi-backbone comparison isolates the regimes in which compact, editable policy memory is most effective, and Table 6 shows that the resulting gain is driven mainly by persistent experience memory and memory editing rather than by retrieval calibration alone.

### 5.3 Mechanism Attribution

Table 6 shows that the DeepSeek-Chat gain depends on same-state counterfactual reconstruction with margin-based memory writing, the separation

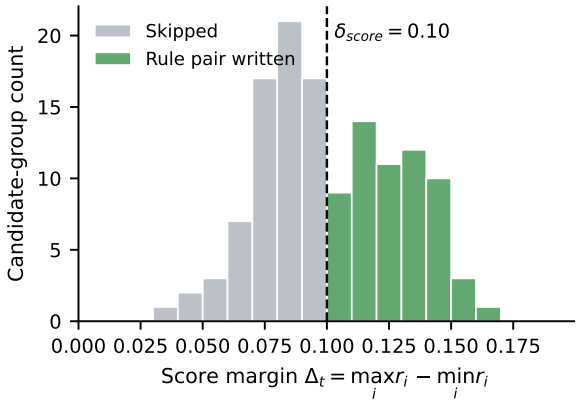


Figure 3: Selective rule writing by score margin. We plot the score margin $\Delta_t$ of reconstructed same-state candidate groups. SCORE writes a rule pair only when $\Delta_t \geq \delta_{\text{score}}$, filtering low-separation interactions before memory editing.

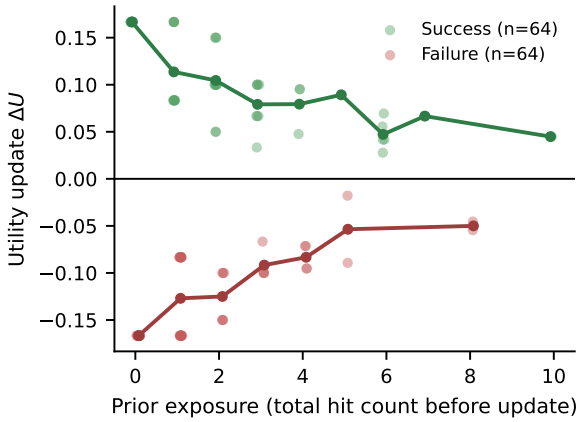


Figure 4: Utility update after retrieval outcome. Utility change $\Delta U$ versus prior exposure (total hit count before update). Successful retrievals receive positive updates, failed retrievals are demoted, and the update magnitude decreases as a memory item accumulates more previous use.

between evaluative and directive memory, and persistent editable experience memory with utility-aware retrieval.

Figure 2 provides qualitative memory snapshots and matched evaluation traces. Across personalization, controlled tool choice, and public mathematical tool use, SCORE converts feedback into local policy rules rather than raw transcripts: clarification preferences become two-stage directives, tool-choice memories separate finish-now from verify-once states, and post-tool branches are retained until utility feedback resolves conflicts.

To verify that SCORE's rule memory is not simply an append-only transcript store, we inspect the score margins of reconstructed same-state candidate groups. Figure 3 shows that memory writes occur only after the winner–loser margin passes the score gate, filtering low-separation interactions before rule editing.

Selective writing is only half of the memory-control loop. After a stored rule is retrieved and used, its downstream outcome updates that rule's utility and changes its future retrieval priority. Figure 4 plots utility updates after retrieval outcomes as a function of prior exposure: successful retrievals receive positive updates, failed retrievals are demoted, and the largest changes appear early in a memory item's lifetime. Together, Figures 3 and 4 show how SCORE controls memory quality at both stages: selective rule writing at entry and outcome-driven reweighting after use.

The largest drops come from removing experience memory or memory editing: success falls from $0.7602$ to $0.4507$ without experience memory and to $0.4983$ without editing. Removing only the state-aware retrieval prior is less damaging, leaving $0.6480$ success and $0.7806$ decision accuracy. Removing directive memory also hurts more than removing evaluative memory.

These quantitative drops match Figure 2 and the mechanisms in Figures 3 and 4. The gain is largest when the library preserves high-separation rule pairs at write time and reweights them after use, which is why the append-only and no-memory variants lose most of the advantage.

## 6 Conclusion

SCORE is a training-free framework for online agent adaptation that converts realized interaction feedback into structured experience-library updates. By combining counterfactual group reconstruction, dual-signal memory writing, and utility-aware retrieval, the method supports effective online adaptation through editable context-space memory. Across personalization, public tool use, controlled tool-choice evaluation, and a ReTool evaluation with Qwen3-4B-ReTool-SFT, SCORE shows that editable context-space memory can produce strong adaptation gains in these settings while keeping model weights frozen, including gains over prompting, reflection, GRPO, and OpenClaw-RL on the Qwen3-4B-ReTool-SFT ReTool evaluation. The results position SCORE as a context-space adaptation mechanism for reusable policy rules and point to richer memory representation as the next source of improvement.

## 7 Limitations

SCORE is designed for adaptation signals that can be compressed into reusable decision rules. This makes it well suited to personalization preferences and tool-use boundaries, but less ideal for cases where success depends on preserving exact local evidence or fine-grained field semantics. In such settings, richer memory representations or hybrid transcript-plus-rule memories may be preferable.

The method is training-free with respect to model weights, but not inference-free: counterfactual reconstruction, scoring, memory editing, and retrieval add runtime cost. This cost is most justified when edited memories are reused across future related decisions; for one-off tasks, simpler retrieval or prompting baselines may be more economical.

This study focuses on local decision-rule adaptation under fixed evaluation protocols. The results support context-space adaptation when feedback can be abstracted into reusable rules, while broader open-ended environments may require richer state abstraction and longer-horizon credit assignment.

## 8 Ethical Considerations

The reported experiments do not involve human subjects or private user data. The main ethical risk concerns future deployment of persistent interaction memory: systems that store online experience should use user-scoped isolation, filtering, auditability, and deletion controls.

## References


Alibaba Cloud. 2026. Supported models and capabilities overview - model studio - alibaba cloud - alibaba cloud model studio - alibaba cloud documentation center. Model Studio Documentation. Accessed 2026-05-21.

Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, et al. 2022. Constitutional ai: Harmlessness from ai feedback, 2022. *URL https://arxiv. org/abs/2212.08073*, 2212.

Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George Bm Van Den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, et al. 2022. Improving language models by retrieving from trillions of tokens. In *International conference on machine learning*, pages 2206–2240. PMLR.

Yuzheng Cai, Siqi Cai, Yuchen Shi, Zihan Xu, Lichao Chen, Yulei Qin, Xiaoyu Tan, Gang Li, Zongyi Li, Haojia Lin, et al. 2025. Training-free group relative policy optimization. *arXiv preprint arXiv:2510.08191*.

DeepSeek. 2026. Models & pricing | deepseek api docs. DeepSeek API Docs. Accessed 2026-05-21.

Jiazhan Feng, Shijue Huang, Xingwei Qu, Ge Zhang, Yujia Qin, Baoquan Zhong, Chengquan Jiang, Jinxin Chi, and Wanjun Zhong. 2025. Retool: Reinforcement learning for strategic tool use in llms. *arXiv preprint arXiv:2504.11536*.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Mingwei Chang. 2020. Retrieval augmented language model pre-training. In *International conference on machine learning*, pages 3929–3938. PMLR.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33:9459–9474.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2024. Let's verify step by step. In *International Conference on Learning Representations*, volume 2024, pages 39578–39601.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, et al. 2023. Self-refine: Iterative refinement with self-feedback. *Advances in neural information processing systems*, 36:46534–46594.

Mathematical Association of America. 2026. Maa invitational competitions – mathematical association of america. Accessed 2026-05-21.

Reiichiro Nakano, Jacob Hilton, Suchir Balaji, Jeff Wu, Long Ouyang, Christina Kim, Christopher Hesse, Shantanu Jain, Vineet Kosaraju, William Saunders, et al. 2021. Webgpt: Browser-assisted question-answering with human feedback. *arXiv preprint arXiv:2112.09332*.

OpenAI. 2026. Gpt-5.4 nano model | openai api. OpenAI API Documentation. Accessed 2026-05-21.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744.

Charles Packer, Vivian Fang, Shishir Patil, Kevin Lin, Sarah Wooders, and Joseph E. Gonzalez. 2024. Memgpt: Towards llms as operating systems. *arXiv preprint arXiv:2310.08560*.

Joon Sung Park, Joseph O'Brien, Carrie Jun Cai, Meredith Ringel Morris, Percy Liang, and Michael S Bernstein. 2023. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th annual acm symposium on user interface software and technology*, pages 1–22.

Yujia Qin, Shengding Hu, Yankai Lin, Weize Chen, Ning Ding, Ganqu Cui, Zheni Zeng, Xuanhe Zhou, Yufei Huang, Chaojun Xiao, et al. 2024. Tool learning with foundation models. *ACM Computing Surveys*, 57(4):1–40.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. *Advances in neural information processing systems*, 36:53728–53741.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. 2023. Toolformer: Language models can teach themselves to use tools. *Advances in neural information processing systems*, 36:68539–68551.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*.

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

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

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2022. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*.

Yinjie Wang, Xuyang Chen, Xiaolong Jin, Mengdi Wang, and Ling Yang. 2026. Openclaw-rl: Train any agent simply by talking. *arXiv preprint arXiv:2603.10165*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2022. React: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629*.

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

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

# A Appendix

## A.1 Reproducibility Details

### A.1.1 Detailed Experimental Conditions

For the multi-backbone personalization and tool-choice comparisons, we use API-served DeepSeek-Chat (DeepSeek, 2026), Qwen-Plus (Alibaba Cloud, 2026), and GPT-5.4-Nano (OpenAI, 2026). The ablation tables use DeepSeek-Chat and state that backbone directly in their captions.

For the main agent actions, we use temperature $= 0.25$ on personalization and temperature $= 0.20$ on tool-choice and tool-recovery episodes. Reflection-note generation uses temperature $= 0.0$, and the counterfactual candidate group uses fixed per-candidate temperatures from the rewrite specifications. The main response length cap is 256 tokens for personalization and tool choice; tool recovery uses a larger pre-tool budget and a shorter post-tool budget, implemented as 448 tokens before tool resolution and 224 tokens after tool resolution. Counterfactual candidate generation uses 192 tokens, and reflection-note generation uses 64 tokens.

All reported metrics are computed by deterministic task-specific scorers implemented in the evaluation code: personalization uses preference-aware rubric functions over correctness and style, tool choice uses rule-based scoring over final-answer correctness and tool-use correctness, and tool recovery uses rule-based scoring over stop/-correction behavior.

**Scorer definitions.** Let $C_{\text{ans}}(x, y) = \mathbb{I}[\text{norm}(x) = \text{norm}(y)]$ denote normalized exact-match correctness on the extracted final answer. For personalization, the student profile uses

$$S_{\text{student}} = 0.45\, C_{\text{ans}} + 0.55\, S_{\text{nat}},$$

where $S_{\text{nat}}$ is a deterministic plain-text student-style rubric. The teacher profile uses

$$S_{\text{teacher}} = R_{\text{teacher}}(x, c, y),$$

where $R_{\text{teacher}}$ is a deterministic grading-feedback rubric over the response $x$, the provided submission context $c$, and the reference answer $y$. The helper profile uses a clarify stage scored by a deterministic clarification rubric $S_{\text{clarify}}$, followed by an answer stage scored as

$$S_{\text{helper}} = 0.75\, C_{\text{ans}} + 0.25\, S_{\text{style}},$$

where $S_{\text{style}}$ is the brief-or-detailed helper-style rubric. Acceptance thresholds are fixed at $0.72$ for the student profile, $0.80$ for the teacher profile, $0.70$ for the clarify stage, and $0.68/0.65$ for brief/detailed helper answers.

For the public mathematical tool-use evaluation, let $I_{\text{FA}} = \mathbb{I}[\text{valid final-answer form}]$ and $I_{\text{TC}} = \mathbb{I}[\text{valid tool-call form}]$. The episode scorer is

$$S_{\text{tool}} = \text{clip}_{[0,1]}(0.1 + 0.7\, C_{\text{ans}} + 0.1\, I_{\text{FA}} + 0.1\, I_{\text{TC}})\,.$$

For controlled tool choice, malformed tool calls receive score $0$. A valid tool call receives $0.7$ when tool use is expected and $0$ otherwise. A no-tool response receives $0.1$ for a valid final-answer form and, when tool use is not expected, up to $0.9$ additional credit for a correct direct answer. Table 5 reports deterministic episode outcomes derived from this setting: *Success* requires a correct final answer, and *Decision* requires matching the gold tool-versus-no-tool decision.

For contract recovery, malformed tool calls receive $0$. In stop states, a valid extra tool call receives $0$, while a correctly formatted final answer that stops on the sufficient result receives full credit. In correction states, a valid repair tool call receives higher credit than an immediate final answer, and a correctly corrected final answer receives the top score; reusing the prior wrong result receives $0$. Table 8 reports the corresponding deterministic stop and correction outcomes.

The personalization benchmark reuses the GSM8K problem pool and applies three profile-specific interaction wrappers: a student-style homework-solution wrapper, a teacher-feedback wrapper over the question, reference answer, and a synthesized student submission, and a clarify-first helper wrapper that requires one clarifying question before the final answer. For each family, the first $48$ rows are used for adaptation and the next $200$ rows for evaluation.

For tool use, Table 4 reports the public AIME24 (Mathematical Association of America, 2026) comparison after the shared 12-example ReTool (Feng et al., 2025) adaptation prefix, Table 5 reports the tool-choice stress test, and Table 8 reports contract recovery.

### A.1.2 Compared Method Definitions

OpenClaw-RL's original paper reports **Binary RL**, **OPD**, and **Combined** variants for personalization, and **Integrated** versus **Outcome-only**

rewards for general agents; it does not include prompt-only external baselines in those main result tables. We therefore supplement that comparison with the strongest *no-training* baselines that can be executed in the same inference-only framework, and we align our ablations to the corresponding OpenClaw-RL signal families.

- **Frozen Base (ReAct-style)**: the standard acting scaffold with no online memory update.
- **ReAct**: an explicit tool-use prompting baseline with no memory update or experience editing; reported on the tool-use evaluations.
- **Raw History Memory**: a retrieval baseline that reuses unstructured previous interaction snippets.
- **Self-Refine**: an API-only iterative self-feedback and revision baseline following Madaan et al. (2023).
- **Reflexion**: an explicit verbal-reflection baseline with no persistent experience library.
- **MemoryBank**: a long-term memory retrieval baseline following the user-memory setting of Zhong et al. (2024).
- **ExpeL**: an experiential-learning baseline that extracts reusable natural-language lessons from prior interactions, following Zhao et al. (2024).
- **Directly Generated Experience**: removes counterfactual group comparison and writes experience directly.
- **Reward-only SCORE (Binary-RL analogue)**: removes directive hindsight signal.
- **Hint-only SCORE (OPD analogue)**: removes evaluative next-state signal.
- **Single-rollout Reflection (Reflexion-style)**: removes the persistent structured experience library and keeps only one-step verbal self-improvement.
- **Append-only SCORE**: removes memory editing, de-duplication, and conflict resolution.
- **SCORE w/o State Prior**: a tool-choice ablation that removes the state-aware retrieval term while keeping the learned memory bank fixed.
- **SCORE (Combined analogue)**: uses counterfactual group reconstruction, dual-signal memory writing, memory editing, and utility-aware retrieval; on the controlled tool-choice setting, we additionally report the effect of the state-aware retrieval term separately.

## A.2 Additional Results

### A.2.1 AIME24 Component Ablations

| Method | Pass@1 ↑ | Inv. Tool ↓ | Avg. Tool Calls |
|---|---|---|---|
| Direct Exp. | 0.2333 | 0.9111 | 4.0000 |
| Reward-only | <u>0.2444</u> | 0.9111 | 3.9444 |
| Hint-only | 0.2222 | 0.9111 | 4.1111 |
| 1-rollout Refl. | <u>0.2444</u> | <u>0.8333</u> | <u>3.5778</u> |
| Append-only | 0.2000 | 0.9333 | 4.0778 |
| SCORE | **0.3222** | **0.8000** | **2.9566** |

Table 7: AIME24 component ablations on the public test set, reported as run means over three runs. Every removal trails the full method on pass@1, indicating that the public-test gain is not recovered by any single partial component alone.

### A.2.2 Contract Recovery Results

This subsection reports the contract-grounded recovery evaluation. Table 8 reports overall, stop, and correction results.

| Method | Overall | Stop | Corr. |
|---|---|---|---|
| Frozen Base | 0.8983 | <u>0.9167</u> | 0.8800 |
| ReAct | 0.7867 | 0.8400 | 0.7333 |
| Raw Hist. Mem. | **0.9367** | **0.9333** | **0.9400** |
| Reflexion | 0.7250 | 0.8533 | 0.5967 |
| MemBank | <u>0.9133</u> | 0.9067 | <u>0.9200</u> |
| ExpeL | 0.8950 | 0.8933 | 0.8967 |
| SCORE | <u>0.9133</u> | 0.9133 | 0.9133 |

Table 8: Contract tool-recovery results. Overall averages the stop and correction subtasks.

### A.2.3 Tool-Choice Diagnostics

Table 6 in the main paper reports the component removals for the DeepSeek-Chat tool-choice setting. Here we provide the matching overuse/underuse breakdown used to interpret that table.

Table 9: Overuse/underuse diagnostic breakdown for the tool-choice stress test in Table 5. Lower is better for both metrics.

| | DeepSeek-Chat | | Qwen-Plus | | GPT-5.4-Nano | |
|---|---|---|---|---|---|---|
| Method | Overuse ↓ | Underuse ↓ | Overuse ↓ | Underuse ↓ | Overuse ↓ | Underuse ↓ |
| Frozen Base | 0.1514 | 0.2602 | 0.2364 | 0.3639 | **0.0306** | 0.4269 |
| ReAct | 0.1037 | 0.3299 | <u>0.0697</u> | 0.4711 | 0.0765 | 0.4048 |
| Raw Hist. | 0.0935 | 0.1973 | 0.2279 | 0.3265 | 0.0510 | 0.4711 |
| Reflexion | 0.1378 | 0.2857 | **0.0459** | 0.4898 | 0.0901 | 0.4116 |
| MemoryBank | **0.0000** | 0.1871 | 0.1956 | **0.3044** | <u>0.0323</u> | 0.3844 |
| ExpeL | **0.0000** | <u>0.1701</u> | 0.1241 | <u>0.3248</u> | 0.0629 | <u>0.3316</u> |
| **SCORE** | <u>0.0255</u> | **0.0918** | 0.1769 | 0.3282 | 0.0629 | **0.2381** |

## A.3 Qualitative Artifacts

### A.3.1 Prompt and Memory Snippets

This appendix shows compact prompt- and memory-level snippets behind the qualitative case-study evidence in Figure 2. We present them as colored boxes for readability rather than dense inline prose.

Artifact 1: Clarify-first preference memory written after repeated user feedback.

```
Trigger:
user expects clarification before the final answer

Stored prompt-facing memory:
- Ask a short clarifying question before answering.
- Then answer briefly and directly with only the key steps.

Observed reuse:
- The same two-stage interaction pattern is later retrieved as a stable behavior prior.
```

Artifact 2: Controlled tool-choice rules that separate direct-finish from verify-once states.

```
State trigger A:
candidate scratch result already resolves target
-> answer directly with a final answer

State trigger B:
multi-step state with no sufficient result yet
-> call the external tool once before the final answer

Observed reuse:
- Retrieval separates nearby decision boundaries instead of replaying raw transcript snippets.
```

Artifact 3: Conflict-edited post-tool branches retained in the AIME24 transfer setting.

```
Shared trigger:
tool-use state already has a tool result and must decide whether to finish or correct

Higher-utility branch:
- emit exactly one compact Python code block
- then wait for the interpreter result before any boxed final answer

Competing finish-now branch:
- use the available interpreter result
- do any short directly supported final derivation
- then finish inside the boxed answer block

Edited outcome:
- keep both branches
- demote the weaker branch with later utility feedback rather than collapsing them into one rule
```

## A.4 LLM Usage Statement

Large language models were used to polish the writing and improve the clarity of this manuscript. All technical content, experimental design, analysis, and conclusions are the original work of the authors.