ACTIVE FORGETTING FOR MULTI-TURN LLM AGENT

Anonymous authors
Paper under double-blind review

ABSTRACT

Multi-turn LLM agents rely on interaction histories to integrate evolving instructions, observations, and actions into subsequent decisions. However, retaining the full history can reinforce outdated assumptions in earlier model responses, while indiscriminate deletion can remove information essential for reasoning and tool execution. Existing approaches to history removal, compression, and memory management generally lack an explicit account of these dependencies, leaving unresolved what agents should retain and what they can safely forget. We introduce *ACTive Forgetting (ACTF)*, a training-free context selection mechanism that determines retention through explicit dependencies among interaction records. ACTF removes unreferenced assistant responses and superseded state while preserving current evidence and its required prerequisites, yielding a compact context satisfying the declared retention and dependency constraints. On the frozen Lost-in-Conversation suite, merged ACTF improves accuracy from 44.2% to 61.0%. A separate rendering audit reaches 61.7% while using 18% of full-history input tokens; controlled evidence-delivery and final-answer evaluations retain the gain. Complementary tool-use and state-update experiments show why selective retention matters: forgetting unsupported model outputs can improve reasoning, whereas preserving required interaction records avoids the failures caused by indiscriminate deletion. These findings highlight active forgetting as a promising approach to building more reliable and efficient LLM agents.

1 INTRODUCTION

Large language model (LLM) agents increasingly tackle complex tasks through multi-turn interactions (Yao et al., 2023), where each decision is conditioned on an evolving history of user instructions, model responses, tool calls, and external observations. This history allows agents to integrate new information and adapt their behavior over time, but simply accumulating more context does not necessarily improve decision making. Prior work has shown that LLMs can struggle to integrate information distributed across conversational turns (Kwan et al., 2024), maintain instructions and conversational consistency over extended interactions (Deshpande et al., 2025), and revise earlier assumptions as new information becomes available (Laban et al., 2026). These findings suggest that interaction history is not merely a passive record of past exchanges: what remains in context can directly shape, and sometimes distort, an agent's subsequent reasoning and actions.

Despite this, typical LLM agents provide the model with the full interaction history at every step (Lindenbauer et al., 2025; Huang et al., 2026), effectively allowing past records to persist indefinitely unless the context window is explicitly modified. Yet history elements differ substantially in both role and lifetime. An earlier model response may contain a provisional interpretation that becomes obsolete after later evidence arrives, whereas a user constraint, tool result, or executed action may remain necessary because subsequent decisions depend on it. Thus, the value of a history element is not an intrinsic property of that message alone. It also depends on its temporal and functional relationships with other records in the interaction. This leads to a fundamental question for multi-turn LLM agents: *which parts of the interaction history should remain available for future decisions, and which can be safely forgotten?*

Existing approaches address history management from several complementary directions. Some *clean or reset* conversational history, for example by removing prior assistant responses, consolidating context when uncertainty increases, or training against history-cleaned trajectories (Mohammad Khalid et al., 2025; Huang et al., 2026; Zheng et al., 2026). Others *compress or restructure*

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

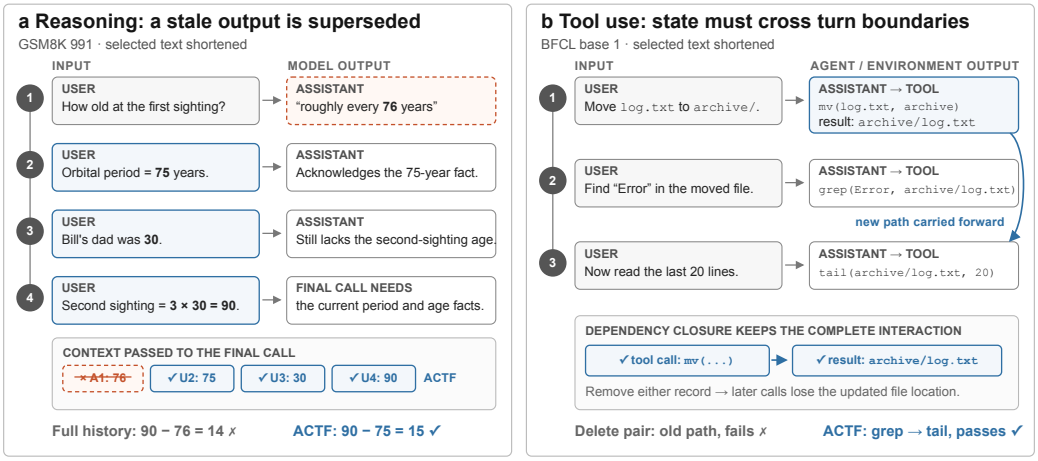

Figure 1: **Case Study.** Each row shows the input and output for one turn, while the lower strip shows the context carried forward. *Left:* ACTF removes an unsupported assistant-generated value and recomputes it from user evidence. *Right:* ACTF retains a state-changing tool call and its result because subsequent turns depend on the updated file location. Text is shortened for display.

long trajectories, typically by summarizing completed subgoals or compacting accumulated context (Hu et al., 2025; Li et al., 2026b). More recent methods make memory management part of the agent policy, allowing the model to learn operations such as retaining, summarizing, or discarding information (Yu et al., 2026b; Zhang et al., 2026). These approaches establish that modifying history can improve multi-turn performance, but they largely treat retention, deletion, or compression as operations on individual records or coarse context segments. What remains less explored is the *dependency structure* among heterogeneous interaction records: removing one element may be beneficial when it represents stale state, yet harmful when another retained element relies on it. Without explicitly accounting for such dependencies, a history-management strategy may either preserve obsolete information or discard prerequisites needed for future reasoning and action.

Figure 1 illustrates this tension. In reasoning, retaining stale intermediate conclusions can be harmful: an earlier assistant response may contain an unverified value that is later superseded by user evidence, yet remains available to contaminate subsequent computation. In tool use, however, indiscriminately deleting earlier assistant history can produce a failure mode: removing an executed tool call or its result may leave a later action without the record it depends on, making the interaction no longer executable. Together, these examples suggest that forgetting should not be decided independently for each message. Instead, effective forgetting must distinguish state that has been superseded and state that remains a prerequisite for retained interaction records. Motivated by this observation, we propose *ACTive Forgetting (ACTF)*, a training-free context selection mechanism that explicitly captures dependencies encoded by chat and tool-calling protocols. ACTF removes superseded or released state, retains records required by the current interaction, and recursively restores their prerequisites through dependency closure. In this way, ACTF can discard stale information without breaking the dependency structure needed for subsequent decisions. More broadly, ACTF reframes interaction history from an ever-growing transcript into a dynamically maintained agent state, providing a principled foundation for more reliable and context-efficient long-horizon interaction.

Contributions: 1) We formulate *context selection* for multi-turn LLM agents and identify a central principle: whether a history element should be retained is *dependency-dependent*, rather than determined by its content or message type alone. This perspective explains why removing prior model outputs can improve reasoning while the same operation can disrupt tool use. 2) We introduce ACTF, a training-free context selection mechanism that removes obsolete state while preserving protocol-level dependencies, and show that the resulting context provides a compact solution satisfying the required retention and dependency constraints. 3) We evaluate ACTF on fully scored Lost-in-Conversation, LongBench, MultiChallenge, three BFCL multi-turn subsets, and synthetic state-update conversations. On the frozen Lost-in-Conversation suite, merged ACTF improves accuracy from 0.442 to 0.610; a separate rendering audit reaches 0.617 at 18% of full-history input, while complete BFCL runs test retention of interaction records required for downstream execution.

2 Related Work

Multi-Turn History Management. Multi-turn LLMs can degrade as earlier assumptions, instructions, or model outputs persist into later decisions. Laban et al. (2026) introduce sharded instruction delivery and show that assumptions formed early in a conversation can persist across later turns, while MultiChallenge (Deshpande et al., 2025) evaluates related failures in instruction retention, inference memory, editing, and self-coherence. Broader remedies modify how agents process conversational context, including interleaved reasoning and acting (Yao et al., 2023), recap and snowball prompting (Laban et al., 2026), prompt and harness optimization (Agrawal et al., 2026; Lee et al., 2026; Zhang et al., 2025), clarification learning (Chen et al., 2025), intent mediation (Liu et al., 2026), dialogue-state tracking (Wu et al., 2019), reinforcement learning (Chen et al., 2026b; Li et al., 2026a), learned gates (Xie, 2026), running summaries (Singh et al., 2026; Wang et al., 2025), and learned context operators (Shen et al., 2026). Closest to our work, Huang et al. (2026) study assistant-history omission on real conversations and Lost-in-Conversation, comparing complete omission, one-sentence summaries, and last-exchange retention, and learning a per-turn choice between full and omitted history. ERGO (Mohammad Khalid et al., 2025) uses entropy changes to trigger prompt rewriting and restart generation in a new conversation, while MAIGO (Zheng et al., 2026) studies self-contamination through direct history removal across math, action, database, and code tasks and trains on history-cleaned references. Chen et al. (2026a) instead distill supervision from a single-turn view, and Wan et al. (2026) combine context preference learning with inference-time context management to mitigate imitation of earlier responses. In contrast, ACTF performs dependency-aware context selection at inference time: it removes unreferenced assistant responses and superseded state while preserving retained evidence together with its required prerequisites. Our evaluations further isolate when and why selective forgetting works.

Context and Memory Management. A related line of work studies how agent context should be compressed, retained, or organized over time. Production APIs distinguish transient reasoning content from persistent dialogue and impose retention requirements around tool use (DeepSeek, 2026; OpenAI, 2026; Anthropic, 2026); context-management mechanisms can additionally clear older tool results (Anthropic, 2025). Observation masking retains reasoning and actions while replacing older tool observations (Lindenbauer et al., 2025), illustrating that appropriate retention depends on the interaction workflow. Other methods compress histories by folding or summarizing prior steps (Sun et al., 2025; Ye et al., 2025; Kang et al., 2026), maintain rolling memory (Luo et al., 2026), or organize information through hierarchical and persistent memory systems (Packer et al., 2023; Li et al., 2025; Park et al., 2023; Yu et al., 2026a; Shinn et al., 2023; Lee et al., 2024). TokenPilot (Xu et al., 2026) jointly considers segment lifetime and prefix-cache continuity. ACTF is complementary to these relevance- and efficiency-oriented approaches: rather than estimating importance alone, it explicitly models dependencies among interaction records to determine which records can be safely removed and which must be retained together.

Token-level context selection. A separate literature reduces context or KV-cache cost at the token level. H2O and SnapKV select cache entries using attention statistics (Zhang et al., 2023; Li et al., 2024), StreamingLLM preserves attention sinks together with a recent context window (Xiao et al., 2024), and RocketKV and KVzip reduce cache size through token-selection or reconstruction criteria (Behnam et al., 2025; Kim et al., 2025). Serving systems further optimize cache allocation and reuse (Kwon et al., 2023; Zheng et al., 2024; Cheng et al., 2025). These methods primarily optimize token-level retention or serving efficiency, whereas ACTF asks a different question: whether an interaction record remains semantically necessary for the agent's next decision. Our equal-budget experiment therefore compares message-level retention against an attention-ranking baseline, directly testing whether token importance is sufficient for dependency-sensitive history selection.

3 Methodology

In this section, we introduce *ACTive Forgetting (ACTF)*, a dependency-aware context selection mechanism that treats interaction history as dynamically maintained agent state. The key idea is that whether a record can be forgotten depends not only on whether it remains active, but also on whether other retained records depend on it. ACTF therefore separates context selection into two complementary operations: identifying the records that should remain directly active and restoring any prerequisites required to keep the resulting context structurally complete. Concretely, it first

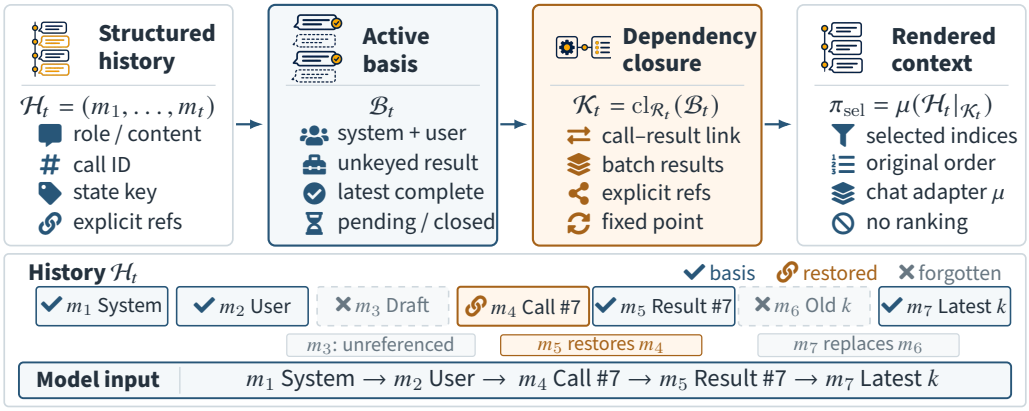

Figure 2: ACTF's context-selection flow at a decision point. The active basis $\mathcal{B}_t$ retains eligible system, user, tool, and state records; dependency closure restores required calls and references to form $\mathcal{K}_t$. In the example, blue checkmarks mark directly retained messages, the orange link marks call m_4 restored for result m_5, and dashed gray cards mark the unreferenced draft m_3 and state m_6 superseded by m_7. The final row shows exactly what the model receives, in original order.

constructs an *active basis*, expands this basis through protocol-level dependency closure, and then renders the retained records in their original chronological order. This design allows ACTF to remove stale or superseded information, including unsupported assistant outputs, while preserving the interaction dependencies required for subsequent reasoning and tool execution.

3.1 Context Selection

A multi-turn interaction induces a sequence of messages, each represented using fields supported by standard chat and tool-calling protocols. Specifically, each message m records its role $\rho(m)$ and content and, when applicable, the identifier of the tool call associated with the message. We additionally consider two optional fields supplied by the environment or client: a state key, which identifies the state variable reported or modified by a tool result, and a set of explicit references to earlier messages that the current message uses. Formally, we represent a message as $m = \big(\rho(m), \mathrm{content}(m), \mathrm{call}(m), \mathrm{key}(m), \mathrm{ref}(m)\big), \quad \rho(m) \in \{\mathsf{sys}, \mathsf{user}, \mathsf{asst}, \mathsf{tool}\}$, where $\mathrm{ref}(m)$ denotes the earlier messages identified by these references. We denote the interaction history available at turn t by $\mathcal{H}_t = (m_1, \ldots, m_t)$. A rendering policy π maps the interaction history to the message sequence presented to the model (with parameters θ), which then generates $y \sim p_\theta(\cdot \mid \pi(\mathcal{H}_t))$. Let Q denote the task score, $|\cdot|$ the rendered input-token count, and C an expected input-token budget per model call. We formulate context selection as:

$$\pi^\star = \arg\max_{\pi} \ \mathbb{E}\big[Q(y)\big] \quad \text{subject to} \quad \mathbb{E}\big[|\pi(\mathcal{H}_t)|\big] \le C. \tag{1}$$

The full-history policy takes $\pi_{\mathrm{full}}(\mathcal{H}_t) = \mathcal{H}_t$. Equation 1 describes the evaluation tradeoff rather than an optimization solved at inference time. ACTF applies one deterministic selection rule using the fields available before the next model call, and we evaluate its task performance and token cost. This distinction is important because context selection cannot always be reduced to independently ranking messages under a token budget. Existing methods often prioritize content using signals such as attention, recency, or message length, but such rankings do not capture dependencies between interaction records. For example, retaining a tool result may require retaining the corresponding tool call that produced it. ACTF explicitly models these dependencies, enabling it to remove obsolete state while preserving the structural consistency of the retained context.

3.2 Active Forgetting

ACTF constructs the retained context in two stages: *Direct Retention* and *Dependency Closure*. Direct Retention identifies the records that remain active, while Dependency Closure restores any additional records required to support them. We detail these two stages below.

Direct Retention. ACTF first constructs a conservative basis of messages that should be retained before dependency closure. By default, system instructions, user messages, and tool results are kept unless they have been safely superseded or explicitly released. Let $\mathrm{key}(m)$ denote the validated environment-state key associated with message m, and let $\mathrm{key}(m) = \varnothing$ when no such key is available. The state key identifies observations that refer to the same underlying state and therefore determines whether replacement is safe. Specifically, when multiple complete and successful observations share the same key k, ACTF retains only the latest one, denoted by last_k. In contrast, a message with $\mathrm{key}(m) = \varnothing$ cannot be reliably matched to a newer observation and is therefore retained by default. The environment may also explicitly indicate that records associated with a completed operation can be released. Let $\mathcal{S}_t$ denote the set of state keys observed up to turn t. We define the directly retained basis as

$$\mathcal{B}_t = \Big(\{m : \rho(m) \in \{\mathsf{sys}, \mathsf{user}, \mathsf{tool}\},\ \mathrm{key}(m) = \varnothing\} \cup \{\mathrm{last}_k\}_{k \in \mathcal{S}_t}\Big) \setminus \mathcal{D}_t \ \cup \ \mathcal{P}_t, \tag{2}$$

where $\mathcal{D}_t$ is the set of tool records explicitly marked as releasable, and $\mathcal{P}_t \subseteq \mathcal{H}_t$ is the set of assistant messages containing unresolved tool calls. Thus, Eq. 2 retains all unkeyed records together with the latest valid observation for each keyed state, removes records in $\mathcal{D}_t$, and always retains messages containing unresolved calls in $\mathcal{P}_t$ until their results arrive. This construction is intentionally conservative. A failed or partial observation cannot replace an earlier usable observation, and an executed state-changing operation remains in context until the environment explicitly signals that its record can be released. ACTF therefore forgets a record only when its replacement or release is supported by protocol-level evidence.

Dependency Closure. The directly retained basis $\mathcal{B}_t$ identifies the records that remain active, but it may not be self-contained: some retained records depend on earlier messages that are not selected directly. For example, a retained tool result requires the assistant message that issued the corresponding call, while a retained message may explicitly refer to an earlier record. Dependency Closure restores such prerequisites so that the selected context remains structurally complete. We write (m_i, m_j) to mean that retaining m_i requires retaining m_j. For tool interactions, let

$$\mathcal{E}_t = \{(m_i, m_j) : \rho(m_i) = \mathsf{tool},\ \mathrm{call}(m_i) \in \mathrm{calls}(m_j)\}\,,$$

where $\mathrm{calls}(m_j)$ denotes the set of tool calls issued by assistant message m_j. Thus, $\mathcal{E}_t$ links each tool result to the assistant message that produced it. A completed call–result interaction is kept together in both directions, while explicit references introduce additional prerequisite links. We therefore define the protocol-level prerequisite relation as

$$\mathcal{R}_t = \mathcal{E}_t \cup \mathcal{E}_t^{-1} \cup \{(m_i, m_j) : \mathrm{id}(m_j) \in \mathrm{ref}(m_i)\}\,. \tag{3}$$

The first two terms preserve matched tool calls and results together, and the last term follows an explicit reference from a retained message to the earlier message it uses. Starting from $\mathcal{B}_t$, ACTF repeatedly follows $\mathcal{R}_t$ and restores every required prerequisite. This process continues until no additional message needs to be added. The resulting set $\mathcal{K}_t$ is therefore the smallest dependency-closed superset of $\mathcal{B}_t$:

$$\mathcal{K}_t = \mathcal{B}_t \cup \{m_j : (m_i, m_j) \in \mathcal{R}_t,\ m_i \in \mathcal{K}_t\}\,. \tag{4}$$

Intuitively, Direct Retention determines which records remain active, whereas Dependency Closure restores any additional records required to support them. For example, retaining a tool result may restore the assistant call that produced it. After the two selection stages, the messages in $\mathcal{K}_t$ are returned to their original chronological order and passed through the chat-template adapter μ:

$$\pi_{\mathrm{sel}}(\mathcal{H}_t) = \mu(\mathcal{H}_t|_{\mathcal{K}_t})\,. \tag{5}$$

Appendix A.1 provides a worked example tracing the complete selection process in Eqs. 2–5, from direct retention through dependency closure to the final rendered context.

Instantiation across settings. The same context-selection rule applies across our evaluation settings, with the available protocol metadata determining which dependencies can be instantiated. In sharded-reasoning tasks, only message roles are available, so system instructions and accumulated user evidence are retained directly, while unreferenced prior assistant responses can be omitted. Tool-use environments additionally expose call identifiers, allowing ACTF to preserve call–result dependencies and conservatively retain unresolved interactions. Stateful environments further expose state keys and completion records, enabling superseded observations to be replaced while preserving earlier records that remain dependencies of the current state. Environment-specific mappings and implementation details are provided in Appendix A.1.

Algorithm 1 Dependency Closure for ACTF Context Selection

Require: History $\mathcal{H}_t$, directly retained basis $\mathcal{B}_t$, prerequisite relation $\mathcal{R}_t$

$\mathcal{K} \leftarrow \mathcal{B}_t$
$\mathcal{Q} \leftarrow \text{queue}(\mathcal{B}_t)$
while $\mathcal{Q}$ is not empty **do**
  $m_i \leftarrow \text{pop}(\mathcal{Q})$
  for all m_j such that $(m_i, m_j) \in \mathcal{R}_t$ **do**
    if $m_j \notin \mathcal{K}$ **then**
      $\mathcal{K} \leftarrow \mathcal{K} \cup \{m_j\}$
      $\text{push}(\mathcal{Q}, m_j)$
    end if
  end for
end while
return $\mu(\mathcal{H}_t|_{\mathcal{K}})$

3.3 Property and Online Implementation

Call a message set *admissible* if it contains the directly retained basis $\mathcal{B}_t$ and is closed under the prerequisite relation $\mathcal{R}_t$. Equation 4 returns the smallest admissible superset of $\mathcal{B}_t$. Consequently, under positive additive message costs, $\mathcal{K}_t$ is the unique minimum-cost admissible message set satisfying the declared retention and dependency constraints. This guarantee is defined with respect to the protocol metadata available to ACTF and the dependencies represented by that metadata. Appendix A.1 gives the formal statement, proof, and relevant special cases. The closure can be maintained online using three indices: the latest eligible observation for each state key, the call and results associated with each call identifier, and reverse adjacency lists for explicit message references. At each decision point, Algorithm 1 performs a worklist traversal initialized from $\mathcal{B}_t$. Each reachable dependency edge is visited once, giving $O(t+|\mathcal{R}_t|)$ worst-case time and memory; sorting or filtering by message index then preserves the original chronological order. The procedure requires no additional model call. Unresolved calls remain pending, and incomplete metadata therefore leads to conservative retention rather than an unsafe deletion. Further validation and environment-specific state-replacement rules are given in Appendix A.1.

4 Experiments

We evaluate ACTF across multi-turn reasoning, question answering, tool use, and state-update tasks using Lost-in-Conversation (LiC), MultiChallenge, LongBench HotpotQA, BFCL, and controlled synthetic settings. Additional results, including model transfer, complete cross-benchmark evaluations, detailed tool-use and state-update analyses, and evidence-delivery and final-answer controls, are provided in Appendices B-D.

4.1 Experimental setup

Benchmarks. The primary reasoning benchmark is Lost-in-Conversation (LiC) (Laban et al., 2026): 308 conversations comprising 103 mathematics, 100 code, and 105 tool-action instances. LiC reveals a task through shards, so earlier responses precede complete evidence. Each of nine frozen policies has three outcomes per instance (924 rows), averaged before the equally weighted instance mean. Breadth comes from all 200 four-chunk LongBench HotpotQA (Bai et al., 2024) instances, three 30-instance synthetic tasks, and all 266 MultiChallenge (Deshpande et al., 2025) conversations. Tool-use comparisons use the 200 fully scored instances in each of three BFCL (Patil et al., 2025) multi-turn subsets. Six 40-task mechanism families, each evaluated with three seeds and five policies, test declared references and state updates. Other frozen tool populations were attempted but had unresolved official scores; Appendix C.3.1 audits their coverage without using them in performance comparisons. Appendix C gives populations, templates, graders, and policy-level results for the included benchmarks.

Models and execution controls. Qwen2.5-14B-Instruct is primary. We add Qwen2.5-7B/32B, Llama-3.1-8B, Mistral-7B-v0.2, Gemma-2-9B, Qwen1.5-MoE-A2.7B, and six closed models. Ev-

Table 1: Primary Lost-in-Conversation comparisons with Qwen2.5-14B: 308 IDs, three rollouts, 924 outcomes per row. Δ is against full history; SD is across rollout means. Huang adaptive, ERGO, and Context-Agent are cross-task adaptations; all fixed-policy rows are reported in Appendix Table 6.

Policy	Rollout 1	Rollout 2	Rollout 3	Mean	Δ	SD
Full history	0.435	0.435	0.455	0.442	0.000	0.011
Huang adaptive	0.490	0.536	0.510	0.512	+0.070	0.023
ERGO (adapted)	0.435	0.435	0.448	0.439	-0.002	0.007
Context-Agent (adapted)	0.380	0.364	0.360	0.368	-0.074	0.010
ACTF	0.578	0.620	0.565	0.588	+0.146	0.029
ACTF, merged	**0.601**	**0.627**	**0.604**	**0.610**	**+0.169**	0.014

ery checkpoint is frozen: ACTF has no learned selector and changes only the next rendered context. Within each benchmark, policies share task IDs, evidence, prompts, tool schemas, decoding settings, and graders. Simulated user turns may differ because the simulator observes prior assistant responses, so pairing is by task ID. Selection preserves message order; adjacent user turns are merged only where marked. Implementation details are in Appendices A and D.2.

Baselines. Full history is the reference; single turn measures model aptitude rather than a deployable policy. Reasoning baselines (i) mark earlier responses or recap user turns, (ii) use placeholders or one-sentence summaries, retain the latest exchange, or clip two rounds, and (iii) remove assistant history at the final or every turn; current-request-only additionally tests the cost of removing user evidence. Omission, summary, and last-exchange are local fixed-policy adaptations of Huang et al. (2026), while recap follows Laban et al. (2026). We also adapt two training-free methods, ERGO (Mohammad Khalid et al., 2025) and Context-Agent (Hu et al., 2026), and fit a Huang-style full/omitted-history selector on disjoint calibration tasks. These are cross-task comparisons on the frozen LiC IDs and grader, not copied scores or reproductions of the original learned systems. For tools, deletion removes prior calls and results, results-only omits producing calls, and observation masking replaces old observations while retaining reasoning and calls (Lindenbauer et al., 2025). Call/result/text ablations identify state-carrying records, while length, recency, and attention provide matched-budget controls. Table 7 records local adaptations and unmeasured learned systems.

Metrics and uncertainty. For instance scores q_i, the headline mean is $A = n^{-1} \sum_i q_i$; LiC first averages its three frozen outcomes per instance, whereas BFCL uses official success, with paired changes computed on common IDs. LiC cost sums logged calls and normalizes input by full history, so it measures context rather than latency. The mechanism suite uses exact-JSON match and reports prompt and completion tokens for its single model call per attempt. Intervals use paired task-level bootstrap resampling; binary conversation outcomes use exact McNemar tests where indicated, while BFCL and mechanism comparisons use cluster sign-flips. Reported p values are unadjusted unless stated otherwise. Failed answers score zero; unreconstructable evaluator failures remain unscored and are excluded rather than imputed as failures. Appendices A.1 and D.2 give the exact measures, resampling counts, and renderings.

4.2 Response interference and transfer

LiC places early model responses before the complete task evidence, making it a direct test of whether those responses should remain in the final context. Across three frozen rollouts, ACTF improves the instance mean over full history; compatibility merging raises it further (Table 1). The gain appears in each rollout rather than arising from one favorable seed.

The locally fitted Huang-style selector improves on full history by 0.070, while merged ACTF exceeds it by 0.098. ERGO is statistically indistinguishable from full history and Context-Agent trails it; merged ACTF exceeds these two methods by 0.171 and 0.242, respectively. All differences refer to the common LiC cohort, not the methods' original tasks. The adapted methods also incur distinct helper-model costs, so selected-context length alone is not a total inference-cost measure. Appendix A.3 reports paired ID-clustered intervals, per-task outcomes, and logged target usage.

The full policy ordering in Table 6 further narrows the explanation. Marking, summarization, and placeholder replacement still expose the model to earlier answer-like content and recover less performance than omission, while final-turn removal likewise leaves earlier decisions in context. However,

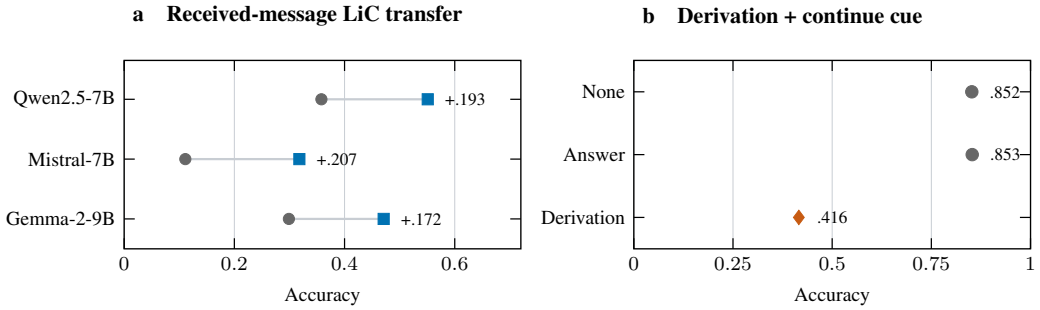

Figure 3: **Transfer and a prompted-interference check.** (a) Received-message LiC accuracy for three additional open models ($n = 308$ each); connected points pair full history (gray) with ACTF (blue). (b) Archived wrong-response injection: the displayed derivation condition includes a *continue* cue, whereas the answer-only condition uses a neutral cue. The factorial contrast and policy cells are in Table 18 and Appendix B.1.

these rendering policies do not constitute a causal comparison. For controlled injection, an incorrect derivation is harmful when the final prompt explicitly asks the model to continue from it. Under a neutral cue, an incorrect derivation and an incorrect answer yield comparable performance (Figure 3(b); Table 18). These results suggest that retained reasoning can act as an interference channel, but its impact depends on how the subsequent turn conditions on that reasoning.

The timing of deletion matters for the interpretation. Final-turn removal cannot undo intermediate responses produced while stale assistant text was still present; the every-turn policy tests that pathway, while the unmerged and merged ACTF rows expose the effect of chat-template compatibility. Since LiC declares no call identifiers or state references, role-based omission and ACTF select the same types of records here. LiC can establish the reasoning benefit but cannot, by itself, establish that the policy is safe for agents that execute tools.

The effect is not confined to the primary Qwen checkpoint. Under received-message delivery, ACTF improves all three additional open models in Figure 3(a); the varying full-history baselines make the paired within-model change more informative than an absolute-score ranking. Table 16 tests different, restated-instruction settings: the much larger gains there and on synthetic corrections should not be pooled with received-message LiC. The per-policy and per-task cells, including the 29 positive closed-model cells out of 30, appear in Appendices B.1 and C.2.

Evidence quality alone does not explain the primary effect. With verbatim shards, merged ACTF outperforms both full history and the single-turn reference, while the one-final-answer control preserves this ordering after removing repeated answer opportunities. Appendix Figure 6(a) further stratifies performance by the amount of evidence delivered to the model. Together, these controls support a response-interference interpretation, while allowing evidence-delivery quality to modulate the magnitude of the effect. Appendix D.1 reports the corresponding paired controls.

4.3 Generalization and Dependency Boundary

Forgetting helps only when removed records are not later prerequisites. Tables 2 and 3 separate this boundary from LiC: MultiChallenge improves overall, LongBench is neutral, and tool results test preservation of executable call–result state rather than an accuracy gain.

MultiChallenge's overall gain repairs 56 full-history failures but damages 30 successes (paired 95% interval $[0.034, 0.169]$; exact McNemar $p = 0.0067$). Its version-editing axis reverses direction because the earlier text is the object to revise (Appendix Figure 6b). On HotpotQA, the F1 change is negligible and its paired interval $[-0.0265, 0.0258]$ includes both small gains and losses. These complete runs support task-dependent selection, not universal deletion.

The three fully scored BFCL subsets test whether selection preserves tool interactions across turns (Table 3). Across 200 dialogues per subset, ACTF changes success relative to full history by -0.010, 0.000, and 0.000, while deleting prior calls and results lowers base success by 0.235. The contrast shows why those interactions cannot be treated as disposable assistant text. Full-history scores on

Table 2: Complete text-benchmark runs and MultiChallenge axes. Scores retain each benchmark's native metric, not a pooled accuracy: HotpotQA token F1 and MultiChallenge conversation pass rate (both ↑). Selected history is role-only omission on HotpotQA and omission with adjacent-user merging on MultiChallenge. Indented axes partition its 266 IDs.

Benchmark	N	Metric ↑	Full	Selected	Δ
LongBench HotpotQA	200	Token F1	0.6395	0.6394	-0.0001
MultiChallenge	266	Pass rate	0.3421	**0.4398**	+0.0977
Inference memory	109	Pass rate	0.3670	**0.4312**	+0.0642
Instruction retention	69	Pass rate	0.2609	**0.5072**	+0.2464
Version editing	40	Pass rate	**0.4500**	0.3750	-0.0750
Self-coherence	48	Pass rate	0.3125	**0.4167**	+0.1042

Table 3: Fully scored BFCL v4 multi-turn subsets, Qwen2.5-14B. Each cell is official success rate over the same 200 IDs; Δ compares ACTF with full history. Other attempted tool populations with missing official scores are excluded (Appendix C.3.1).

Benchmark	N	Full ↑	ACTF ↑	Δ ↑
BFCL base	200	**0.355**	0.345	-0.010
BFCL missing function	200	0.040	0.040	0.000
BFCL missing parameter	200	0.150	0.150	0.000

the missing-function and missing-parameter subsets are themselves low, so these runs primarily test preservation of interaction state. The attempted long-context BFCL and τ^2-Bench populations have incomplete official scoring and are audited separately in Appendix C.3.1.

ACTF retains complete tool interactions, but this does not yield a uniform token reduction. Relative to full history, its mean BFCL input tokens rise by 4.8% on base and fall by 1.9% and 1.6% on the two missing-tool subsets; the equal-size, fully scored three-subset mean is effectively unchanged (Appendix 28). The constructed mechanism tests distinguish retention from state use: ACTF keeps the declared inputs for partial updates, but exact-JSON accuracy is 0.317 because the model often fails to compose them (Appendix A.5).

Additional controls. Observed gains cannot be explained solely by shorter contexts or repeated answer opportunities. On a 100-item mathematics subset matched to ACTF's token budget, role-based selection achieves an accuracy of 0.270, compared with 0.090 for attention-based ranking, 0.100 for random selection, and 0.230 for full history. Moreover, under a stricter setting with verbatim evidence and a single final answer per task, ACTF improves accuracy from 0.567 to 0.690 ($+0.123$; 95% paired interval $[0.049, 0.197]$). Complete control results are provided in Appendix D.2.

5 Conclusion

We introduced ACTF, a training-free method for selecting the interaction records carried into an agent's next decision. ACTF first constructs an active basis containing records that remain current, and then restores any additional prerequisites required by those records through dependency closure. This allows stale or superseded records to be omitted without breaking the interaction structure required for subsequent decisions. On the frozen LiC suite, merged ACTF improves accuracy from 0.442 to 0.610. Selected history also improves MultiChallenge, while HotpotQA remains comparable to full history. The BFCL controls show why preserving call–result records matters: dropping them reduces success, whereas ACTF stays near full-history performance on the fully scored subsets. Together, these results support selective removal of stale model outputs without discarding the interaction state needed for subsequent actions.

Limitations and future work. Our experiments focus on text-based conversations and tool traces in which relevant state is represented in the interaction record. Longer-lived agents may also rely on implicit references or changing external artifacts; identifying these dependencies is a natural extension of ACTF. Combining dependency-aware selection with summarization could further support workflows whose retained state grows beyond a single conversation.

AI USE STATEMENT

During preparation of this manuscript, Codex (OpenAI) was used to assist with language editing and LaTeX formatting; literature discovery and bibliography checks; discussions of research ideas, experimental design, and result interpretation; and drafting and revising portions of the manuscript. It also assisted with experiment organization and consistency checks across reported results. The authors reviewed and verified the AI-assisted content and take full responsibility for the methods, experiments, numerical results, claims, citations, and conclusions. LLMs used as components of the reported method are described separately in the main text and appendix.

REFERENCES

Lakshya A Agrawal, Shangyin Tan, Dilara Soylu, Noah Ziems, Rishi Khare, Krista Opsahl-Ong, Arnav Singhvi, Herumb Shandilya, Michael J Ryan, Meng Jiang, Christopher Potts, Koushik Sen, Alexandros G. Dimakis, Ion Stoica, Dan Klein, Matei Zaharia, and Omar Khattab. Gepa: Reflective prompt evolution can outperform reinforcement learning. In *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/0e9e708b6f48e14fd0ac29e167413f76-Abstract-Conference.html. arXiv:2507.19457.

Anthropic. Managing context on the claude developer platform. https://www.anthropic.com/news/context-management, 2025.

Anthropic. Extended thinking. https://docs.anthropic.com/en/docs/build-with-claude/extended-thinking, 2026.

Yushi Bai, Xin Lv, Jiajie Zhang, Hongchang Lyu, Jiankai Tang, Zhidian Huang, Zhengxiao Du, Xiao Liu, Aohan Zeng, Lei Hou, Yuxiao Dong, Jie Tang, and Juanzi Li. LongBench: A bilingual, multitask benchmark for long context understanding. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3119–3137, 2024. doi: 10.18653/v1/2024.acl-long.172. URL https://aclanthology.org/2024.acl-long.172/.

Payman Behnam, Yaosheng Fu, Ritchie Zhao, Po-An Tsai, Zhiding Yu, and Alexey Tumanov. RocketKV: Accelerating long-context LLM inference via two-stage KV cache compression. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 3358–3392, 2025. URL https://proceedings.mlr.press/v267/behnam25a.html.

Maximillian Chen, Ruoxi Sun, Tomas Pfister, and Sercan Arik. Learning to clarify: Multi-turn conversations with action-based contrastive self-training. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/4ffd05ca3cf3985f4572af015b4cfc1e-Abstract-Conference.html.

Tianlang Chen, Shirley Wu, and Jure Leskovec. Found in conversation: LLMs teach themselves to close the multi-turn gap. *arXiv preprint arXiv:2605.24432*, 2026a.

Xingwu Chen, Zhanqiu Zhang, Steven Y. Guo, and Difan Zou. Breaking contextual inertia: Reinforcement learning with single-turn anchors for stable multi-turn interaction. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 6302–6318, 2026b. doi: 10.18653/v1/2026.findings-acl.313. URL https://aclanthology.org/2026.findings-acl.313/.

Yihua Cheng, Yuhan Liu, Jiayi Yao, Yuwei An, Xiaokun Chen, Shaoting Feng, Yuyang Huang, Samuel Shen, Kuntai Du, and Junchen Jiang. LMCache: An efficient KV cache layer for enterprise-scale LLM inference. *arXiv preprint arXiv:2510.09665*, 2025. URL https://arxiv.org/abs/2510.09665.

DeepSeek. Thinking mode: Api guide. https://api-docs.deepseek.com/guides/thinking_mode/, 2026.

Kaustubh Deshpande, Ved Sirdeshmukh, Johannes Baptist Mols, Lifeng Jin, Ed-Yeremai Hernandez-Cardona, Dean Lee, Jeremy Kritz, Willow E. Primack, Summer Yue, and Chen Xing. MultiChallenge: A realistic multi-turn conversation evaluation benchmark challenging to frontier LLMs. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 18632–18702, 2025. doi: 10.18653/v1/2025.findings-acl.958. arXiv:2501.17399.

Junan Hu, Shudan Guo, Wenqi Liu, Jianhua Yin, and Yinwei Wei. Context-Agent: Dynamic discourse trees for non-linear dialogue. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 29449–29462, 2026. doi: 10.18653/v1/2026.findings-acl.1472. URL https://aclanthology.org/2026.findings-acl.1472/.

Mengkang Hu, Tianxing Chen, Qiguang Chen, Yao Mu, Wenqi Shao, and Ping Luo. Hiagent: Hierarchical working memory management for solving long-horizon agent tasks with large language model. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 32779–32798, 2025. doi: 10.18653/v1/2025.acl-long.1575. URL https://aclanthology.org/2025.acl-long.1575/.

Jenny Y. Huang, Leshem Choshen, Ramon Astudillo, Tamara Broderick, and Jacob Andreas. Do LLMs benefit from their own words? *arXiv preprint arXiv:2602.24287*, 2026. URL https://arxiv.org/abs/2602.24287.

Minki Kang, Wei-Ning Chen, Dongge Han, Huseyin A. Inan, Lukas Wutschitz, Yanzhi Chen, Robert Sim, and Saravan Rajmohan. ACON: Optimizing context compression for long-horizon LLM agents. In *International Conference on Machine Learning*, 2026. URL https://www.microsoft.com/en-us/research/publication/acon-optimizing-context-compression-for-long-horizon-llm-agents/. arXiv:2510.00615.

Jang-Hyun Kim, Jinuk Kim, Sangwoo Kwon, Jae W. Lee, Sangdoo Yun, and Hyun Oh Song. Kvzip: Query-agnostic kv cache compression with context reconstruction. In *Advances in Neural Information Processing Systems*, 2025. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/f4eaa4b8f2d08edb3f0af990d56134ea-Abstract-Conference.html. Oral; arXiv:2505.23416.

Wai-Chung Kwan, Xingshan Zeng, Yuxin Jiang, Yufei Wang, Liangyou Li, Lifeng Shang, Xin Jiang, Qun Liu, and Kam-Fai Wong. Mt-eval: A multi-turn capabilities evaluation benchmark for large language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 20153–20177, 2024.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *SOSP*, 2023.

Philippe Laban, Hiroaki Hayashi, Yingbo Zhou, and Jennifer Neville. Llms get lost in multi-turn conversation. In *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/59f6421e64707225fdf5b28840679a07-Abstract-Conference.html. arXiv:2505.06120.

Kuang-Huei Lee, Xinyun Chen, Hiroki Furuta, John Canny, and Ian Fischer. A human-inspired reading agent with gist memory of very long contexts. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 26396–26415, 2024. URL https://proceedings.mlr.press/v235/lee24c.html.

Yoonho Lee, Roshen Nair, Qizheng Zhang, Kangwook Lee, Omar Khattab, and Chelsea Finn. Meta-harness: End-to-end optimization of model harnesses. *arXiv preprint arXiv:2603.28052*, 2026.

Ming Li, Pei Chen, Zhenhao Zhang, Tao Yang, Xinyang Zhang, Han Li, Tianyu Cao, Ming Zeng, Zhuofeng Wu, Meng Jiang, Huasheng Li, Lihong Li, and Bing Yin. Mitigating lost in multi-turn conversation via curriculum RL with verifiable accuracy and abstention rewards. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 33364–33381, 2026a. doi: 10.18653/v1/2026.acl-long.1540. URL https://aclanthology.org/2026.acl-long.1540/.

Tianjian Li, Jingyu Zhang, William Jurayj, Xi Wang, Chuanyang Jin, Mehrdad Farajtabar, Eric Nalisnick, and Daniel Khashabi. Self-compacting language model agents. *arXiv preprint arXiv:2606.23525*, 2026b.

Yuhong Li, Yingbing Huang, Bowen Yang, Bharat Venkitesh, Acyr Locatelli, Hanchen Ye, Tianle Cai, Patrick Lewis, and Deming Chen. SnapKV: LLM knows what you are looking for before generation. In *NeurIPS*, 2024.

Zhiyu Li et al. MemOS: An operating system for memory-augmented generation (MAG) in large language models. *arXiv preprint arXiv:2505.22101*, 2025.

Tobias Lindenbauer, Igor Slinko, Ludwig Felder, Egor Bogomolov, and Yaroslav Zharov. The complexity trap: Simple observation masking is as efficient as LLM summarization for agent context management. *arXiv preprint arXiv:2508.21433*, 2025.

Geng Liu, Fei Zhu, Rong Feng, Changyi Ma, Shiqi Wang, and Gaofeng Meng. Intent mismatch causes LLMs to get lost in multi-turn conversation. *arXiv preprint arXiv:2602.07338*, 2026.

Shu Tong Luo, Wenqin Liu, Rui Liu, Mingming Gong, and Jiaxian Guo. Multi-turn reasoning when context arrives in pieces: Scalable sharding and memory-augmented RL. *arXiv preprint arXiv:2606.12941*, 2026.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. Self-Refine: Iterative refinement with self-feedback. *arXiv preprint arXiv:2303.17651*, 2023. URL https://arxiv.org/abs/2303.17651.

Haziq Mohammad Khalid, Athikash Jeyaganthan, Timothy Do, Yicheng Fu, Vasu Sharma, Sean O'Brien, and Kevin Zhu. ERGO: Entropy-guided resetting for generation optimization in multi-turn language models. In *Proceedings of the 2nd Workshop on Uncertainty-Aware NLP (UncertaiNLP 2025)*, pp. 273–286, 2025. doi: 10.18653/v1/2025.uncertainlp-main.23. URL https://aclanthology.org/2025.uncertainlp-main.23/.

OpenAI. Reasoning models. https://platform.openai.com/docs/guides/reasoning, 2026.

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

Joon Sung Park, Joseph C. O'Brien, Carrie J. Cai, Meredith Ringel Morris, Percy Liang, and Michael S. Bernstein. Generative agents: Interactive simulacra of human behavior. In *UIST*, 2023. arXiv:2304.03442.

Shishir G. Patil, Huanzhi Mao, Fanjia Yan, Charlie Cheng-Jie Ji, Vishnu Suresh, Ion Stoica, and Joseph E. Gonzalez. The Berkeley function calling leaderboard (BFCL): From tool use to agentic evaluation of large language models. In *Proceedings of the 42nd International Conference on Machine Learning*, pp. 48371–48392, 2025.

Jiawei Shen, Jia Zhu, Hanghui Guo, Weijie Shi, Yue Cui, Qingyu Niu, Guoqing Ma, Jingjiang Liu, Yidan Liang, Yilin Wang, Shimin Di, and Jiajie Xu. ACR: Adaptive context refactoring via context refactoring operators for multi-turn dialogue. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 3149–3167, 2026. doi: 10.18653/v1/2026.findings-acl.155. URL https://aclanthology.org/2026.findings-acl.155/.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. In *NeurIPS*, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/1b44b878bb782e6954cd888628510e90-Abstract-Conference.html.

Jyotika Singh, Fang Tu, Miguel Ballesteros, Weiyi Sun, Sandip Ghoshal, Michelle Yuan, Yassine Benajiba, Sujith Ravi, and Dan Roth. MT-OSC: Path for LLMs that get lost in multi-turn conversation. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 27137–27160, 2026. doi: 10.18653/v1/2026.findings-acl.1354. URL https://aclanthology.org/2026.findings-acl.1354/.

Weiwei Sun, Miao Lu, Zhan Ling, Kang Liu, Xuesong Yao, Yiming Yang, and Jiecao Chen. Scaling long-horizon LLM agent via context-folding. *arXiv preprint arXiv:2510.11967*, 2025.

Yang Wan, Zheng Cao, Zhenhao Zhang, Zhengwen Zeng, Shuheng Shen, Changhua Meng, and Linchao Zhu. Mitigating conversational inertia in multi-turn agents. *arXiv preprint arXiv:2602.03664*, 2026. ICML 2026.

Qingyue Wang, Yanhe Fu, Yanan Cao, Shuai Wang, Zhiliang Tian, and Liang Ding. Recursively summarizing enables long-term dialogue memory in large language models. *Neurocomputing*, 639:130193, 2025. doi: 10.1016/j.neucom.2025.130193. URL https://doi.org/10.1016/j.neucom.2025.130193.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *ICLR*, 2023. arXiv:2203.11171.

Chien-Sheng Wu, Andrea Madotto, Ehsan Hosseini-Asl, Caiming Xiong, Richard Socher, and Pascale Fung. Transferable multi-domain state generator for task-oriented dialogue systems. In *ACL*, 2019.

Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. Efficient streaming language models with attention sinks. In *ICLR*, 2024.

Ying Xie. Learning to forget: Sleep-inspired memory consolidation for resolving proactive interference in large language models. *arXiv preprint arXiv:2603.14517*, 2026.

Buqiang Xu, Zirui Xue, Dianmou Chen, Chenyang Fu, Chiyu Wu, Caiying Huang, Chen Jiang, Jizhan Fang, Xinle Deng, Yijun Chen, Yunzhi Yao, Xuehai Wang, Jin Shang, Gong Yu, and Ningyu Zhang. TokenPilot: Cache-efficient context management for LLM agents. *arXiv preprint arXiv:2606.17016*, 2026.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations*, 2023. URL https://arxiv.org/abs/2210.03629.

Rui Ye, Zhongwang Zhang, Kuan Li, Huifeng Yin, Zhengwei Tao, Yida Zhao, Liangcai Su, Liwen Zhang, Zile Qiao, Xinyu Wang, Pengjun Xie, Fei Huang, Siheng Chen, Jingren Zhou, and Yong Jiang. AgentFold: Long-horizon web agents with proactive context management. *arXiv preprint arXiv:2510.24699*, 2025.

Hongli Yu, Tinghong Chen, Jiangtao Feng, Jiangjie Chen, Weinan Dai, Qiying Yu, Ya-Qin Zhang, Wei-Ying Ma, Jingjing Liu, Mingxuan Wang, and Hao Zhou. MemAgent: Reshaping long-context LLM with multi-conv RL-based memory agent. In *International Conference on Learning Representations*, 2026a. arXiv:2507.02259.

Yi Yu, Liuyi Yao, Yuexiang Xie, Qingquan Tan, Jiaqi Feng, Yaliang Li, and Libing Wu. Agentic memory: Learning unified long-term and short-term memory management for large language model agents. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics*, pp. 21457–21483, 2026b. doi: 10.18653/v1/2026.acl-long.981. URL https://aclanthology.org/2026.acl-long.981/.

Jiayi Zhang, Jinyu Xiang, Zhaoyang Yu, Fengwei Teng, Xiong-Hui Chen, Jiaqi Chen, Mingchen Zhuge, Xin Cheng, Sirui Hong, Jinlin Wang, Bingnan Zheng, Bang Liu, Yuyu Luo, and Chenglin Wu. Aflow: Automating agentic workflow generation. In *ICLR*, 2025. Oral.

Yuxiang Zhang, Jiangming Shu, Ye Ma, Xueyuan Lin, Shangxi Wu, and Jitao Sang. Memory as action: Autonomous context curation for long-horizon agentic tasks. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 19149–19164, 2026. doi: 10.18653/v1/2026.findings-acl.956. URL https://aclanthology.org/2026.findings-acl.956/.

Zhenyu Zhang, Ying Sheng, Tianyi Zhou, Tianlong Chen, Lianmin Zheng, Ruisi Cai, Zhao Song, Yuandong Tian, Christopher Ré, Clark Barrett, Zhangyang Wang, and Beidi Chen. H2O: Heavy-hitter oracle for efficient generative inference of large language models. In *NeurIPS*, 2023.

Haoyu Zheng, Yun Zhu, Shu Yuan, Shangming Chen, Qing Wang, Wenqiao Zhang, Jun Xiao, and Yueting Zhuang. MAIGO: Mitigating lost-in-conversation with history-cleaned on-policy self-distillation. *arXiv preprint arXiv:2605.27186*, 2026.

Lianmin Zheng, Liangsheng Yin, Zhiqiang Xie, Chuyue Sun, Jeff Huang, Cody Hao Yu, Shiyi Cao, Christos Kozyrakis, Ion Stoica, Joseph E. Gonzalez, Clark Barrett, and Ying Sheng. SGLang: Efficient execution of structured language model programs. In *Advances in Neural Information Processing Systems*, volume 37, 2024. doi: 10.52202/079017-2000. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/724be4472168f31ba1c9ac630f15dec8-Abstract-Conference.html.

Appendix

Contents

Part A Method and evaluation details
A.1 Formal guarantees and protocol fields 15
A.2 Rendering and comparison provenance 16
A.3 Recent cross-task comparisons 18
A.4 Airline rendering diagnostic 19
A.5 Mechanism diagnostics 20
A.6 Context policies 22
Part B Reasoning transfer and mechanism analyses
B.1 Open- and closed-weight model transfer 22
B.2 Causal wrong-response intervention 23
B.3 Per-task closed-weight results 24
B.4 Removal dose and response add-back 24
B.5 Marking versus removal 25
B.6 Evidence fidelity, dynamics, and retention cases 25
Part C Cross-benchmark and systems results
C.1 Long-context document QA: complete LongBench HotpotQA 27
C.2 Synthetic benchmark results (F1, F2, F3) 28
C.3 Scored BFCL and MultiChallenge results; exclusion audit 29
C.4 Removal at the cache level 31
C.5 Tool continuation and completion checks 31
Part D Statistical robustness
D.1 Evidence delivery and final-answer evaluation 33
D.2 Budget controls and paired tests 34

A METHOD AND EVALUATION DETAILS

Part A records the formal guarantees, protocol fields, rendering provenance, matched tool-use audit, mechanism construction, and exact policy definitions needed to reproduce or interpret result tables.

A.1 FORMAL GUARANTEES AND PROTOCOL FIELDS

Algorithm 1 computes the set in Equation 4. The settings have different tasks and scorers; Table 4 isolates which protocol fields are available to the selector before each decision.

Table 4: Application of our method to each evaluation setting.

Evaluation	Call–result link	State key	Message reference	Selected history
Lost-in-Conversation	no	no	no	user evidence without earlier assistant responses
BFCL scored archive	turn tags	no	no	prior call–result records without unrelated assistant text; native-ID validation is separate
τ^2 airline diagnostic	native	adapter	no	linked interactions and the latest eligible state; rewards excluded
State-update constructions	declared	declared	declared	current values and unfinished tool interactions

Recorded case: current account balance

Benchmark: Constructed mechanism suite, complete-refresh test case 000 (seed 17; Qwen2.5-14B).

Msg. #	Speaker	Message or observation	ACTF status
1	User	"Store the supplied records for a later request."	**Basis**
2	Assistant	`{"tool": "read", "object": "account_A"}`	Omitted
3	Tool	`{"balance": 131}` (reply to 2)	Superseded
4	Assistant	`{"tool": "read", "object": "account_A"}`	**Restored**
5	Tool	`{"balance": 235}` (reply to 4)	**Basis**
6–12	Assistant	Seven unrelated ticket memos (grouped here).	Omitted
13	User	"Return the current balance of `account_A` as `{"balance":integer}`."	**Basis**
Next	Assistant	`{"balance":235}` (recorded ACTF answer; correct)	**Answer**

Blue marks direct retention ($\mathcal{B}_t$); **orange** marks the call restored by dependency closure; gray marks omissions; **green** marks the generated answer. Message 5 points to its producing call in message 4, so the model sees messages 1, 4, 5, and 13. The seven memos are grouped; displayed messages and the answer are verbatim. Table 11 reports aggregate results.

Proposition 1 (Minimality under declared dependencies)**.** *Relative to the supplied $\mathcal{B}_t$ and $\mathcal{R}_t$, $\mathcal{K}_t$ is the unique minimum-cost context that contains $\mathcal{B}_t$ and preserves $\mathcal{R}_t$ for any positive additive message cost.*

Proposition 2 (Special cases)**.** *If no call is pending and no retained message depends on an assistant response, the assistant history is removed. If all external messages are in $\mathcal{B}_t$ and every assistant response is a prerequisite, our method keeps the full history.*

Proposition 3 (Order and reuse)**.** *Selection preserves message order. For two consecutive serialized contexts, a prefix cache is reusable through their longest identical token prefix. State replacement, restored references, and template merging determine where that prefix ends.*

Proof of Proposition 1. Starting from $\mathcal{B}_t$, Algorithm 1 adds only prerequisites of retained messages and stops after at most t additions. Any set that contains $\mathcal{B}_t$ and preserves $\mathcal{R}_t$ contains, by induction on dependency depth, every message added by the algorithm. It therefore contains $\mathcal{K}_t$. Since every

message has positive cost, every strict superset costs more, so $\mathcal{K}_t$ is the unique minimum relative to the supplied state and dependencies.

Proof of Proposition 2. If no call is pending and no retained message depends on an assistant response, $\mathcal{K}_t$ contains no assistant history. If every external message is in the basis and every assistant response is recursively referenced, $\mathcal{K}_t$ contains the full history.

Proof of Proposition 3. The selected indices are sorted in original message order. For a deterministic tokenizer and fixed template, an identical token prefix has identical causal hidden states and KV entries. Reuse ends at the first changed token; preserving message order alone places no lower bound on the length of that prefix.

Protocol fields and implementation. Call identifiers associate each tool result with the assistant request that produced it. Explicit references connect a current message to earlier records it uses. Validated state identifiers mark successive observations of the same object, while an environment-issued completion signal can release a finished transaction. The implementation checks that state updates and completion signals come from authorized sources. A retained reference can still restore an earlier record through dependency closure. The mechanism cases supply these links and state annotations before selection; LiC supplies only message roles.

A.2 Rendering and comparison provenance

Accuracy against rendered input. Figure 4 reports the separate archived rendering audit used to measure context ratios. Its merged run reaches 0.617 at 0.18 of full-history input; the complete three-rollout accuracy estimates are reported independently in Table 1. The two removal implementations use the same role-only selection on LiC, so differences between separate rollouts are not evidence of an additional dependency mechanism.

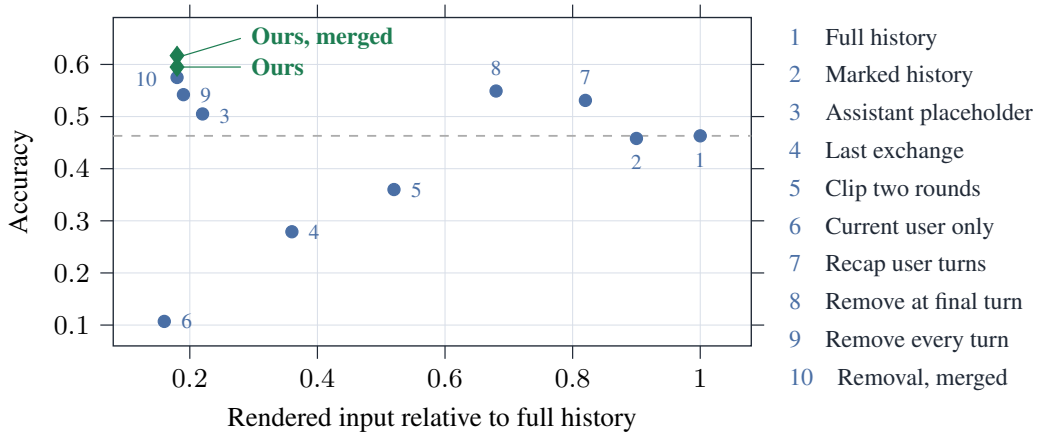

Figure 4: Archived accuracy–context audit for Qwen2.5-14B on the 308-item LiC suite. Input is summed over the conversation and divided by full history. These single-run points are separate from the complete three-rollout estimates in Table 1.

LiC rendering and result provenance. The archived primary ACTF and remove-every-turn arms are separate rollouts of the same system/user-only selection on LiC. Replaying the archived renderer on all 1,556 decision prefixes of the 306 primary ACTF conversations retains zero assistant messages and gives identical selections in the two modes. Their score difference therefore measures rollout variation, not an artifact-reference benefit. Adjacent user messages are merged only for designated merged runs and Mistral/Gemma template compatibility. The simulator sees the accumulated interaction when generating the next user message, so fixing the task and settings does not fix the realized user text. Verbatim delivery supplies each revealed shard as written; final-instruction restatement separately supplies the canonical task prompt.

Tool-message handling. In BFCL archive, messages are tagged by user turn and by whether they contain a call, a result, or assistant text. Selection changes only prior turns. It preserves call records

and result records, clears accompanying prose on retained calls, and keeps all steps of the current turn. The results-only control drops calls but retains returned text. The complete-deletion control removes both, concatenates user messages, and preserves the current exchange. Thus this archive evaluates retention of tool records. The reference implementation additionally validates native call identifiers. A message containing several tool calls is retained with every available result from that batch. Calls without a result remain pending, and an error result remains associated with its attempted call. In the reference implementation, results that precede their call or use a duplicate identifier are rejected. These cases are covered by unit tests; the archived benchmark scores predate these implementation checks.

State replacement and executed actions. A replacement key denotes a complete observation of the same object, not merely a repeated tool name. For example, a new read of reservation A may replace an earlier read of A; it does not replace a read of reservation B. A payment, update, or cancellation result also records an executed action and remains until the environment explicitly releases that operation. Failed reads do not replace successful observations. The airline adapter implements these distinctions through a fixed mapping of read tools and their object arguments. The tool-task accuracy table reports the archived call-result selection policy; state replacement is a separate extension.

Equal-budget attention diagnostic. The archived sample contains 100 mathematics conversations with two to eight assistant turns, shuffled with seed 42. Attention scores average the final query position over all layers and heads. Prior tokens are selected at the role-only budget; the current user turn is retained in full. Selected token IDs are re-prefilled with consecutive positions and decoded greedily for at most 300 tokens. The common grader accepts a matching boxed answer, a matching answer after `####`, or an occurrence of the target number. Table 37(b) reports this numeric-match diagnostic. It compares retention criteria, rather than reproducing the complete H2O serving algorithm.

Table 5: Archived mechanism diagnostics. Arrows indicate the preferred direction. Conditions appear in column order within each row; separate audits are not pooled.

Setting (reference / control / target)	n	Metric	Reference	Control(s)	Target
Wrong-response injection (none / answer+neutral / derivation+continue)	6	Accuracy ↑	0.852	0.853	0.416
LiC removal dose (0 / 25,50,75 / 100%)	308	Accuracy ↑	0.463	0.446/0.429/ 0.455	**0.542**
Reasoning selector (full / attention,random / role)	100	Match rate ↑	0.230	0.090/0.100	**0.270**
BFCL selector (full / length,recency / role)	200	Accuracy ↑	**0.300**	0.195/0.205	0.140

Table 6: All nine frozen LiC policy arms on Qwen2.5-14B. Each policy has 308 IDs and 924 scored attempts; SD is across three rollout means. The method-comparison rows in Table 1 additionally use the same IDs and scorer.

Policy	Rollout 1	Rollout 2	Rollout 3	Mean	Δ vs. full	SD
Full history	0.435	0.435	0.455	0.442	0.000	0.011
Clip two rounds	0.367	0.370	0.406	0.381	−0.061	0.022
Marked history	0.519	0.448	0.513	0.494	+0.052	0.039
One-sentence summary	0.519	0.481	0.477	0.492	+0.051	0.023
Assistant placeholder	0.532	0.516	0.555	0.535	+0.093	0.020
Recap user turns	0.542	0.503	0.549	0.531	+0.090	0.025
Remove at final turn	0.536	0.568	0.532	0.545	+0.104	0.020
ACTF	0.578	0.620	0.565	0.588	+0.146	0.029
ACTF, merged	**0.601**	**0.627**	**0.604**	**0.610**	**+0.169**	0.014

Table 7: Complete, single-rollout local fixed-policy adaptations on the 308-item LiC suite. Scores are measured locally rather than copied from the cited papers; neither represents a learned selector.

Prior method	Local realization	Evaluation	Metric	Value (↑)
Huang: summary	Local summary adapter; different generation settings	LiC	Accuracy	0.487
Huang: last exchange	Retain the latest exchange and current user turn	LiC	Accuracy	0.279

The archived Huang placeholder adaptation has only 307/308 attempts and is excluded from this complete-result table. The fixed rows above and our locally fitted selector below are not the released selector or published scores of Huang et al. (2026); the trained MAIGO policy was not reproduced. The full-cohort cross-task adaptations are evaluated separately below.

A.3 Recent cross-task comparisons on the frozen LiC cohort

ERGO (Mohammad Khalid et al., 2025) resets generation after an entropy-triggered context reconstruction; Context-Agent (Hu et al., 2026) selects a path through a discourse tree. We adapted both training-free procedures to the 308 incrementally revealed LiC conversations. A third adaptation follows the full-versus-omitted-history selection structure of Huang et al. (2026), fitting a local selector rather than using the authors' unavailable training labels or weights. All three use the same Qwen2.5-14B target assistant, 7B simulated user and answer verifier, frozen evaluation IDs, three rollouts, and task scorer as Table 1. These are cross-task adaptations, not exact replications of the papers' original evaluations.

The Huang-style selector is trained on 100 calibration IDs disjoint from evaluation: 25 each from math, database, summary, and data-to-text tasks, with a frozen 80/20 ID split. At each usable calibration turn, the same history yields full-context and assistant-omitted candidates; a local 7B pairwise judge supplies preference labels. A class-balanced sparse logistic model uses observed-turn features and local embeddings, with preprocessing fitted only on training IDs; 77 training and 19 validation IDs yield 315 and 79 usable turns. The cost weight and decision threshold are chosen on validation IDs and frozen before the 924 evaluation attempts. ERGO separately calibrates its threshold on 80 disjoint IDs. Context-Agent retains separate decision, summary, and embedding models. The source revisions, substitutions, selector hash, and run configurations are preserved with the aggregate provenance.

Table 8: Frozen LiC successes by task type. Every cell gives correct attempts / all attempts; no scored failure is removed from the denominator. Each task ID appears in three rollouts.

Policy	Math (103 IDs)	Code (100 IDs)	Actions (105 IDs)	Total (308 IDs)
Full history	166 / 309	119 / 300	123 / 315	408 / 924
Huang adaptive	185 / 309	154 / 300	134 / 315	473 / 924
ERGO (adapted)	173 / 309	123 / 300	110 / 315	406 / 924
Context-Agent (adapted)	144 / 309	111 / 300	85 / 315	340 / 924
ACTF, merged	**212 / 309**	**174 / 300**	**178 / 315**	**564 / 924**

Huang adaptive improves over full history in math and code but less so in actions. ERGO recovers seven more mathematics and four more code attempts than full history but loses thirteen action attempts; Context-Agent is lower on all three task types. Merged ACTF's gain is present in each type, not just the largest subset. These are task-specific transfer results for LiC's progressively revealed evidence.

All five arms have exactly 924 unique (task ID, rollout) keys over the same 308 IDs, with no duplicate or out-of-manifest key. The respective counts of null `score` values are 22, 20, 16, 20, and 18 for full history, Huang adaptive, ERGO, Context-Agent, and merged ACTF; every such row is explicitly graded incorrect by `is_correct=false`. Thus these are failed answers, not missing

Table 9: Matched LiC differences in accuracy (left minus right). Percentile 95% intervals use 8,000 bootstrap resamples of 308 task IDs, keeping each ID's three attempts together. Better/worse/tied counts are at the ID level.

Comparison	Δ	95% interval	Better / worse / tied IDs
Merged ACTF − full history	+0.169	[0.128, 0.210]	139 / 51 / 118
Merged ACTF − Huang adaptive	+0.098	[0.061, 0.137]	111 / 58 / 139
Merged ACTF − ERGO	+0.171	[0.128, 0.214]	143 / 55 / 110
Merged ACTF − Context-Agent	+0.242	[0.201, 0.286]	157 / 33 / 118
Huang adaptive − full history	+0.070	[0.034, 0.109]	104 / 61 / 143
ERGO − full history	−0.002	[−0.041, 0.035]	81 / 73 / 154
Context-Agent − full history	−0.074	[−0.111, −0.036]	56 / 101 / 151

evaluator outcomes. Numeric-zero mathematics rows whose `is_correct` field is null are likewise graded by the numeric score. None of the five arms has a genuinely unscored attempt.

Table 10: Logged inference usage over all 924 LiC attempts for the three adapted methods.

Policy	Asst. calls	Asst. in	Asst. out	Helper calls	Helper in	Helper out
Huang adaptive	4,985	1,525,685	740,675	9,046	4,305,077	56,950
ERGO (adapted)	5,495	4,090,581	1,321,674	—	—	—
Context-Agent (adapted)	5,093	3,544,811	1,226,144	19,010	9,369,315	342,331

Usage fields are present on every adapted-method attempt; the shared LiC simulated user and answer verifier are outside these selector/target totals. Huang's separate 100-ID calibration used 888 target calls and 1,282 helper calls, excluded from the inference table; ERGO records 244 resets. The earlier LiC context-ratio audit in Figure 4 does not supply comparable all-model call totals, so we do not infer a wall-time or total-cost ranking from these numbers.

Paired comparisons. Figure 9(a) compares the archived primary ACTF run with the prompting baselines on common instances using 8,000 paired bootstrap resamples. It improves more conversations than it worsens in each comparison and improves over full history, marking, placeholders, summaries, and clipping at $p \leq 0.011$. The smaller gain over user recap is 0.069 ($p = 0.048$). Across the role-only replications, complete removal can also score higher: 0.483 versus 0.468 under simulated evidence and final-answer scoring. Tool and mechanism experiments evaluate the additional dependency rules.

Summary-adapter provenance. The archived renderer defaults to Qwen2.5-7B for summaries (overridable by an environment variable), truncates each source response to 6,000 characters, caps summary generation at 80 tokens, and substitutes the omission placeholder if generation fails. It caches summaries but does not propagate summarizer token metadata into the returned history. The effective summarizer override and complete summarization cost have not been reconstructed from these archives. This fixed summary adapter and our locally fitted cross-task selector remain distinct from Huang et al.'s original pipelines.

A.4 Airline rendering diagnostic

The matched 30-task airline pilot attempted full history, call-result retention, state replacement, and observation masking (120 task-policy records). The assistant and simulated user used Qwen2.5-14B-Instruct at temperature zero and seed 17. The masking adapter replaced observations after ten completed interactions while keeping calls and reasoning. All attempts finished, but only 118 received an official score: task 12 is unscored under both retention policies after replay encountered the unavailable `send_email_confirmation` tool and discarded the trajectory. Consequently, the pilot is excluded from success-rate tables, paired performance tests, and performance claims. Its saved trajectories still permit the following *offline* rendering diagnostic; these measurements do not require task rewards.

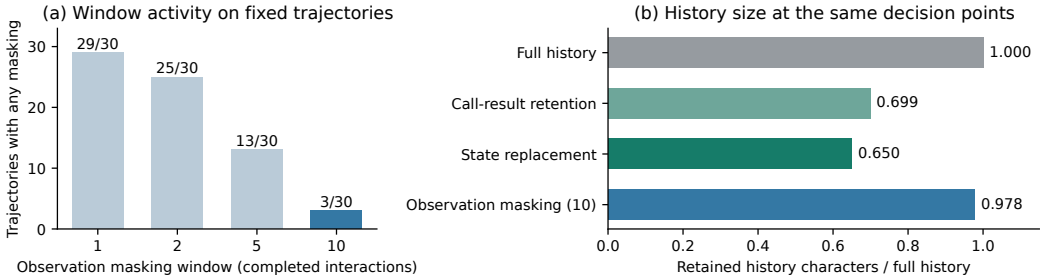

Figure 5: **Offline activity on 30 fixed full-history airline trajectories.** All policies are rendered at the same 393 decision points, with no new model calls. (a) Masking is active in 29, 25, 13, and 3 trajectories for windows of 1, 2, 5, and 10 complete interactions. The model comparison uses the preselected window of 10. (b) The fraction of history characters retained, summed over these decision points; this excludes the system prompt and tool schema and is not an input-token ratio. State replacement changes six trajectories beyond call-result retention. Every rendered prefix passes the native call-result history validator. These measurements describe mechanism activity, not task-success effects.

Aggregation and reproducibility. For each LiC instance, repeated logged outcomes are averaged before computing the instance mean; headline means weight the 103 math, 100 code, and 105 actions instances equally. Failed answers count as zero, while missing official tool rewards are not imputed. The supplement includes per-instance scores, source-file hashes, and the audit script. The airline pilot's lost trajectories also prevent full-cohort inference-cost estimates for its retention arms, so the pilot supplies neither a performance nor a cost comparison.

A.5 Mechanism diagnostics

Table 11 uses the complete frozen mechanism suite. It has six families, 40 held-out test cases per family, and three seeds (17, 29, 43). An additional ten development cases per family are not included in the table. Each case has a fixed history, expected JSON response, declared call identifiers and, where applicable, state keys or direct references. Five policies render that same history before a single Qwen2.5-14B call at temperature 0.7 with a 192-token output cap. For uniform input across policies, historical tool observations are serialized as user messages rather than native tool-role messages; this suite tests selection with supplied metadata, not native tool-protocol execution. The scorer applies `json.loads` to the complete response and requires exact equality to the expected object; malformed responses score zero. There are 3,600 unique case–policy–seed records and no unscored outcomes. The frozen generator, cases, and raw records are supplied with the results.

The families isolate distinct decisions. Artifact lookup and confirmation require retaining an earlier assistant message named by the final user request. A complete refresh should replace the old observation for the same state key; a different object or failed read should not. The partial-update case requires combining a prior balance with a later delta that is explicitly incomplete. The selected context retains both records, but ACTF's 0.317 accuracy in that family shows that retaining the necessary inputs does not ensure correct composition by the model. These synthetic, client-annotated histories test selection given metadata; they do not measure automatic reference inference or autonomous tool execution.

In the four ceiling-scored families (proposed action, complete refresh, different object, and failed read), both policies succeed on all 120 attempts per family, while ACTF uses 53.9–67.7% fewer prompt tokens. This saving is specific to the constructed, single-call mechanism histories; it does not compensate for the partial-update failures. BFCL contains call–result interactions without these explicit snapshot and delta semantics, so its task success and full-dialogue token counts remain separate in Tables 29 and 28.

Table 11: Mechanism exact-match successes out of 120 attempts per policy and family (40 held-out cases × 3 seeds). All attempts are scored. These constructed cases test declared-dependency handling, not end-to-end benchmark success.

Family	Full	Role-only	Pair-only	State-closed	ACTF
Referenced artifact	74	0	0	0	85
Proposed action	120	0	0	0	120
Complete refresh	120	0	120	120	120
Different object	120	0	120	120	120
Failed read	120	0	120	120	120
Partial update	101	0	38	38	38

Table 12: Mechanism successes by frozen seed, out of 40 test cases per cell. Full history and ACTF use the same IDs and scorer.

	Full history			ACTF		
Family	17	29	43	17	29	43
Referenced artifact	26	24	24	28	28	29
Proposed action	40	40	40	40	40	40
Complete refresh	40	40	40	40	40	40
Different object	40	40	40	40	40	40
Failed read	40	40	40	40	40	40
Partial update	35	34	32	16	12	10

Table 13: Mean logged tokens per mechanism attempt, averaged over all 120 test attempts in each family. Each attempt makes one model call; these are model-call tokens, not wall-clock or cache-adjusted compute.

	Prompt tokens ↓		Completion tokens ↓	
Family	Full	ACTF	Full	ACTF
Referenced artifact	453.1	147.8	7.4	7.4
Proposed action	451.5	146.1	22.1	22.1
Complete refresh	563.2	182.0	8.0	8.0
Different object	563.1	259.0	8.0	8.0
Failed read	561.6	259.0	8.0	8.0
Partial update	564.7	259.0	8.0	8.1

Partial-update failure audit. For all 120 ACTF attempts in this family, the selected history contains both the earlier complete balance and the later incomplete delta. Of the 82 incorrect responses, each returns the old balance rather than its updated value. In one saved case, a balance of 541 followed by a delta of 49 requires 590: full history returns 590, while ACTF returns 541. The error therefore cannot be explained by omission of either input record; it is a failure to compose retained state. This limits what dependency closure alone can guarantee about the model's final answer.

Task-level uncertainty. The referenced-artifact family shows 85/120 successes for ACTF and 74/120 for full history, but its 40-case paired difference is $+0.092$ with a cluster-bootstrap 95% interval of $[-0.050, 0.233]$. The partial-update difference is -0.525 with interval $[-0.667, -0.367]$ (Holm-adjusted $p = 0.0023$ across the planned core contrasts). The remaining four families are at or near ceiling for both full history and ACTF. Thus the suite illustrates specific retention behavior and a composition failure, not a uniform accuracy advantage across stateful tasks.

A.6 CONTEXT POLICIES

Table 14: Context policies used in the experiments. SHARDED, CONCAT, and RECAP are the LiC settings of Laban et al. (2026); other rows are local controls or the noted adaptation.

Policy	History retained	Comparison role
Full history	Every previous message	LiC SHARDED baseline
Single-turn reference	Complete user evidence in one turn	LiC CONCAT reference
Evidence recap	Full history, then the original instruction restated	LiC RECAP remedy
Recap from user turns	Full history, then user messages restated verbatim	Local recap control
Marked history	Full history, with earlier assistant messages labeled outdated	Local marking control
Assistant placeholder	Earlier assistant text replaced by a fixed placeholder	Adapted from Huang et al. (2026)
Last exchange only	Previous user–assistant exchange and current user message	Local positional control
Remove at final turn	LiC: omit assistant history; BFCL: omit prior calls and results at the final user turn	Local timing control
Remove every turn	LiC: omit assistant history; BFCL: omit prior calls and results, retaining the current exchange	Local deletion control
Our method	Current task messages and explicitly referenced earlier records	Proposed selector

B REASONING TRANSFER AND MECHANISM ANALYSES

Part B tests model transfer, causal response interference, removal dose, and the conditions under which assistant history should remain.

B.1 OPEN- AND CLOSED-WEIGHT MODEL TRANSFER

Table 15 supplies the full policy cells behind the received-message comparisons in Figure 3(a). The restated-instruction comparisons use a different delivery protocol and appear separately in Table 16; they are not pooled with these cells.

Table 15: Open-weight transfer on received-message LiC ($n = 308$ per model). The primary Qwen2.5-14B result appears in Table 1.

Model	Full history	Marked history	Remove every turn	ACTF
Qwen2.5-7B	0.358	0.349	0.547	**0.551**
Mistral-7B-v0.2	0.111	0.111	0.289	**0.318**
Gemma-2-9B	0.299	0.286	0.448	**0.471**

Table 16: Transfer with the instruction restated at the final turn. LiC rows evaluate ACTF; synthetic rows report the measured assistant-omission arm, which selects the same message types when no dependencies are declared. These rows use a different delivery protocol from Figure 3(a).

Setting and model	Full history	Selected context	Δ
LiC, Qwen2.5-32B	0.370	**0.952**	+0.582
LiC, Llama-3.1-8B	0.139	**0.777**	+0.638
Synthetic F1–F3, Qwen2.5-14B	0.603	**1.000**	+0.397
Synthetic F1–F3, Llama-3.1-8B	0.393	**1.000**	+0.607
Synthetic F1–F3, Mistral-7B-v0.2	0.251	**0.806**	+0.555
Synthetic F1–F3, Gemma-2-9B	0.400	**0.949**	+0.549
Synthetic F2–F3, Qwen1.5-MoE-A2.7B	0.275	**0.934**	+0.659

Six closed-weight models spanning two capability tiers and a family trained for reasoning test whether the drop is specific to small or open models, over the 600-instance LiC superset. The single-turn gap is positive in 29 of 30 model-task cells, from $+0.010$ on GPT-5 summarization to $+0.500$ on GPT-5 code, and the remaining cell is -0.002. The magnitude tracks the task rather than raw capability: tool actions and code carry the largest gaps, and the gap does not shrink consistently with model capability. Appendix B.3 reports all five gradable task types.

Comparison on the full test set. Table 17 compares the context policies on GPT-4o-mini over all 323 mathematics, code, and data-to-text instances. Removing earlier assistant responses improves average accuracy by 0.205 when applied only before the final answer and by 0.227 when applied throughout the conversation. Appendix D.1 grades one final answer from every policy and obtains the same ordering.

Table 17: GPT-4o-mini accuracy on all 323 Lost-in-Conversation instances and transitions on the 203 binary graded cases. Removal policies restate the instruction at the final turn.

Policy	Math Acc. ↑	Code Acc. ↑	Data-to-text Acc. ↑	Mean Acc. ↑	Δ vs. full (↑)	Repaired	Damaged
Full history	0.631	0.530	0.192	0.451	0	0	0
Single-turn reference	0.893	0.650	0.293	0.612	+0.161	56	17
Remove at final turn	0.903	0.750	0.313	0.655	+0.205	56	**6**
Remove every turn	0.913	0.810	0.312	**0.678**	+0.227	**66**	9
Our method	**0.932**	0.750	0.286	0.656	+0.205	61	8

The changes per instance show a broad repair rather than a trade between failure modes. On the 203 binary graded cases, our method repairs 61 and damages 8, compared with 66 and 9 for complete removal. Their Lost-in-Conversation averages differ by 0.022, while the complete tool-use results in Table 29 show why our method cannot be replaced by removal at every turn.

B.2 Causal wrong-response intervention

We insert a fabricated assistant response before the final query while keeping the complete problem statement unchanged. Table 18 varies whether the inserted response contains only a wrong answer or a full wrong derivation, and whether the final prompt asks the model to reconsider or continue. GPT-4o-mini remains at 0.77 after a wrong answer or a wrong derivation with a neutral prompt, but falls to 0.23 when a wrong derivation is followed by a request to continue. The factorial contrast identifies an interaction between a retained derivation and the request to continue from it.

Table 18: Controlled injection on the single-turn base, GPT-4o-mini math, accuracy. Baseline without injection is 0.900.

Injected prior content	Neutral cue Acc. ↑	Continue cue Acc. ↑
Bare wrong answer	0.770	0.730
Elaborate wrong derivation	0.770	0.230

Table 19 repeats two selected conditions on six models. The bare-answer plus neutral-cue condition changes accuracy little on average; the derivation plus continue-cue condition yields 0.23–0.35 on the four GPT-4 models, 0.50 on GPT-5-mini, and 0.75 on GPT-5. Because both the injected content and final cue change between these columns, the factorial test in Table 18, rather than this cross-model contrast alone, identifies their interaction.

Table 19: Controlled wrong-response injection across six models on the same mathematics set. The bare-answer condition uses a neutral cue; the wrong-derivation condition asks the model to continue. Table 18 separates these two factors.

Model	Baseline Acc. ↑	Bare answer Acc. ↑	Wrong derivation Acc. ↑
GPT-4o-mini	0.900	0.767	0.233
GPT-4.1-mini	0.947	0.947	0.316
GPT-4o	0.865	0.919	0.351
GPT-4.1	0.783	0.870	0.348
GPT-5-mini	0.867	0.800	0.500
GPT-5	0.750	0.812	0.750

B.3 Per-task closed-weight results

Table 20: Closed-weight LiC accuracy (↑) and the single-turn minus full-history gap.

	Math			Code			Data-to-text		
Model	Single turn	Full history	Δ (↑)	Single turn	Full history	Δ (↑)	Single turn	Full history	Δ (↑)
GPT-4o-mini	0.845	0.684	0.161	0.650	0.500	0.150	0.303	0.205	0.098
GPT-4.1-mini	0.932	0.804	0.128	0.830	0.610	0.220	0.405	0.307	0.098
GPT-4o	0.874	0.637	0.237	0.800	0.610	0.190	0.453	0.253	0.200
GPT-4.1	0.903	0.777	0.126	0.890	0.720	0.170	0.547	0.291	0.256
GPT-5-mini	0.874	0.684	0.190	0.850	0.510	0.340	0.481	0.285	0.196
GPT-5	0.939	0.837	0.102	0.900	0.400	0.500	0.404	0.235	0.169

Table 21 reports all five Lost-in-Conversation task types across the six closed-weight models. The gap is positive in 29 of 30 cells. It is largest for tool actions, from 0.29 to 0.47, and code, up to 0.50, and smallest for summarization.

Table 21: Accuracy change (↑), single turn minus full history, on the LiC superset. Database is omitted because its grader requires the Spider execution databases.

Model	Math	Code	Data-to-text	Actions	Summary
GPT-4o-mini	+0.161	+0.150	+0.098	+0.467	+0.048
GPT-4.1-mini	+0.128	+0.220	+0.098	+0.429	+0.166
GPT-4o	+0.237	+0.190	+0.200	+0.294	+0.108
GPT-4.1	+0.126	+0.170	+0.256	+0.441	+0.130
GPT-5-mini	+0.190	+0.340	+0.196	+0.315	+0.010
GPT-5	+0.102	+0.500	+0.169	+0.371	−0.002

B.4 Removal dose and response add-back

Table 22 reports the aggregate comparison on Qwen2.5-14B and Llama-3.1-8B, and Table 23 gives the task-level results on Qwen2.5-14B. Partial removal changes the aggregate Qwen2.5-14B score by at most 0.034, whereas complete removal gains 0.157 on tool actions and 0.075 on code.

Partial removal yields no improvement on Qwen2.5-14B and limited improvement on Llama-3.1-8B, while complete removal produces gains of 0.080 and 0.512. Adding back only the longest or most recent assistant response significantly reduces accuracy, showing that the result does not depend on selecting a particular position in the history.

Table 22: Accuracy under partial removal and after adding assistant responses back to the selected context. Qwen2.5-14B uses 308 conversations without instruction restatement; Llama-3.1-8B uses the earlier restated run. The final panel uses 146 Llama mathematics conversations.

(a) Qwen2.5-14B, earliest responses removed first

Fraction removed	0.00	0.25	0.50	0.75	1.00
Accuracy ↑	0.463	0.446	0.429	0.455	**0.542**

(b) Matched removal direction at 50%

Model	Earliest first Acc. ↑	Latest first Acc. ↑	Δ latest − earliest (↑)
Qwen2.5-14B	0.429	0.464	+0.035
Llama-3.1-8B	**0.185**	0.145	−0.040

(c) Responses retained after complete removal

History retained	Accuracy ↑	Δ (↑)	Sign-test p	n
None	**0.651**	0.000	Reference	146
Longest assistant response	0.486	-0.165	2.7×10^{-4}	146
Most recent assistant response	0.418	-0.233	6.2×10^{-7}	146
All assistant responses	0.390	-0.261	7.6×10^{-7}	146

On data-to-text generation, retaining earlier prose is more useful because it conveys style and format. This contrast motivates conditioning removal on whether a later task message uses the response, rather than applying a fixed truncation rate.

Table 23: Accuracy by task under partial and complete removal on Qwen2.5-14B, 308 conversations, without restating the instruction.

Policy	Math Acc. ↑	Code Acc. ↑	Actions Acc. ↑
Full history (K=0)	0.617	0.405	0.367
Remove earliest K=0.25	0.544	0.475	0.324
Remove earliest K=0.50	0.534	0.430	0.324
Remove earliest K=0.75	0.553	0.440	0.371
Remove latest K=0.50	0.534	0.450	0.410
Remove at final turn (K=1.0)	**0.670**	0.440	**0.533**
Remove every turn	0.621	**0.480**	0.524

B.5 Marking versus removal

Marked history keeps every earlier assistant response and labels it as outdated. On the three received-message transfer models in Table 15, marking changes accuracy by -0.009, 0.000, and -0.013, whereas ACTF gains 0.193, 0.207, and 0.172 over full history. The primary Qwen2.5-14B rollouts differ: marking gains 0.052 in Table 1, still below the unmerged ACTF gain of 0.146. A label therefore does not reliably reproduce the effect of removing earlier responses.

B.6 Evidence fidelity, dynamics, and retention cases

The improvement from removing earlier assistant responses is expected when those responses were made before the task's evidence was complete and are not required by the task. It need not extend to tasks that explicitly ask the model to revise an earlier artifact, remain consistent with a previous statement, or continue a tool interaction. This distinction explains why removal helps sharded reasoning but harms fully scored BFCL base tool use. Reflexion (Shinn et al., 2023) also falls outside the first setting because earlier failures are explicitly presented as feedback, while self-consistency (Wang et al., 2023) aggregates independent answers rather than carrying one answer through later turns.



Figure 6: **Evidence delivery and deletion boundary.** (a) Archived LiC fidelity strata (45/77/235 full-history cases, distinct from the frozen cohort); policy-specific strata are descriptive rather than matched-ID contrasts. (b) MultiChallenge axes (109/40/69/48 cases): three improve, while version editing requires earlier text.

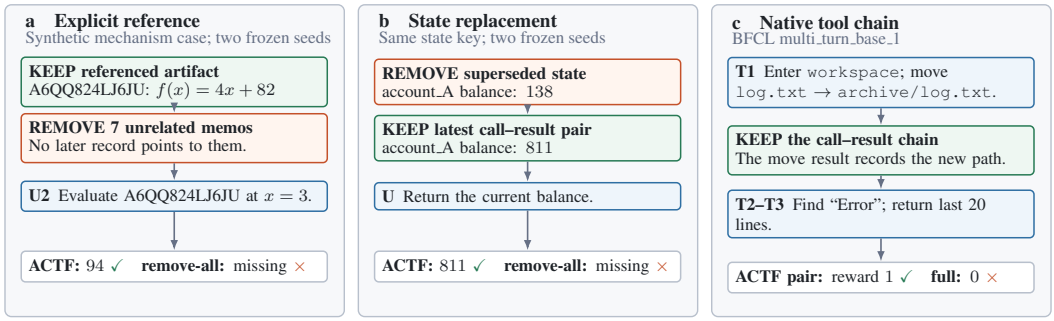

Figure 7: **Complementary retention cases.** ACTF does not delete assistant history indiscriminately. It retains an explicitly referenced artifact (a), replaces stale state while preserving the latest call–result pair (b), and closes the interaction chain needed by a native four-turn BFCL task (c). Panels (a)–(b) illustrate the constructed mechanism cases; panel (c) is a scored BFCL case.

The evidence-fidelity stratification is an archived diagnostic, not a partition of the frozen 308-ID LiC run. Its three reported strata total 357 cases, so their rates must not be pooled with Table 1.

Table 24: Archived evidence-fidelity diagnostic, Qwen2.5-14B (357 full-history cases across three strata). Conversations are grouped separately for each policy by the lowest word recall of any shard in their own user messages. Counts are from full history, not the frozen 308-ID run; columns within a stratum need not contain identical task IDs.

Lowest shard recall	n	Full history Acc. ↑	Remove every turn Acc. ↑	Remove at final Acc. ↑	ACTF Acc. ↑	ACTF + restatement Acc. ↑
At least 0.90	45	0.467	0.759	0.775	**0.806**	0.829
0.70 to 0.90	77	0.416	0.509	0.519	0.511	**0.841**
Below 0.70	235	0.485	0.522	0.514	0.583	**0.919**

These policy-specific strata describe delivered evidence and do not estimate a paired treatment effect within any one row. The matched-ID evidence-delivery controls are reported in Appendix D.1.

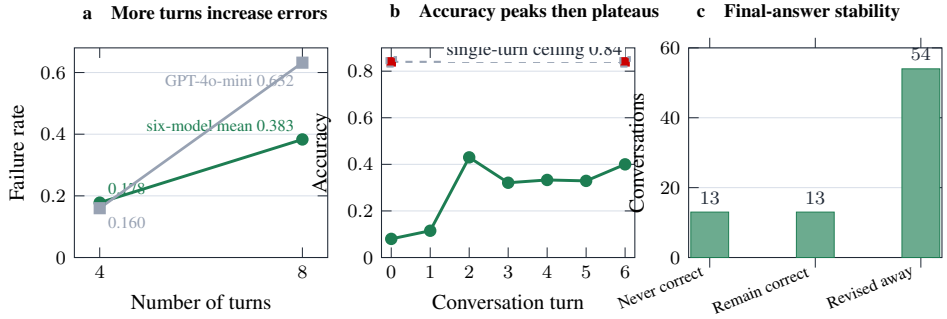

Figure 8: Effect of conversation length. Failure increases when the same evidence is divided over more turns. Accuracy peaks after the informative turn and then declines, and 54 of 80 conversations revise a correct intermediate answer before the final turn.

C Cross-benchmark and systems results

Part C reports complete-scoring LongBench, synthetic-task, BFCL, and MultiChallenge results summarized in the main text. An audit records tool populations omitted from comparative results because official rewards were missing.

C.1 Long-context document QA: complete LongBench HotpotQA

We evaluate all 200 LongBench HotpotQA instances (Bai et al., 2024). Each document is divided into four paragraph-level segments and revealed over four turns, and the assistant summarizes each segment before answering the original question. We compare full history, a single-turn presentation of the full document, and removal of the intermediate summaries before the final answer.

Table 25: LongBench HotpotQA, n=200, Qwen2.5-14B, four-chunk delivery. All values use the complete frozen population.

Policy	F1 ↑	Substring ↑	Exact ↑	Input tok. (↓)	Output tok. (↓)
Full history	**0.640**	0.570	0.480	49,292	558
Remove intermediate summaries	0.639	0.565	**0.505**	47,970	508
Single-turn reference	0.491	**0.585**	0.320	13,420	18

Removing the intermediate summaries changes F1 by only -0.0001 (paired bootstrap 95% CI $[-0.026, 0.026]$) and reduces mean input by 1,322 tokens. Exact match rises by 0.025, but substring accuracy falls by 0.005; these metric-specific movements do not support a general quality gain. The neutral result is consistent with summaries that restate evidence rather than commit to an answer. The paired records clarify this aggregate: token F1 is unchanged on 177 of 200 questions, improves on 13, and declines on 10. Exact match repairs eight answers and damages three, explaining its small positive change without a corresponding F1 gain. Removing summaries lowers input tokens on all 200 questions; the mean quality effect remains unresolved within the reported interval. These counts describe the complete cohort, not a selected subset.

C.2 Synthetic benchmark results (F1, F2, F3)

The three archived synthetic tasks test additive constraints, explicit corrections, and a numerical value required by a later turn. Each task has 30 input instances for Qwen2.5-14B and Llama-3.1-8B; all three tasks were also evaluated on Mistral-7B and Gemma-2-9B, and the correction and value-reuse tasks on Qwen1.5-MoE-A2.7B. These restated-instruction diagnostics use a different protocol from received-message LiC and are not pooled with the frozen benchmark comparisons.

Table 26: Archived synthetic-constructor accuracy (↑) for Qwen2.5-14B and Llama-3.1-8B. F1–F3 name tasks, not the F1 metric; removal policies restate the final instruction. Aggregate cells are descriptive because trial-level scoring records are unavailable in the current archive.

Model	Policy	F1 additive	F2 corrective	F3 value reuse	Mean ↑
Qwen2.5-14B	Single-turn reference	0.977	1.000	1.000	0.992
	Full history	0.609	0.767	0.433	0.603
	Remove at final turn	1.000	1.000	1.000	1.000
	Remove every turn	1.000	1.000	1.000	1.000
	Our method	**1.000**	**1.000**	**1.000**	**1.000**
Llama-3.1-8B	Single-turn reference	0.617	0.867	1.000	0.828
	Full history	0.328	0.733	0.117	0.393
	Remove at final turn	**0.810**	**0.967**	**1.000**	**0.926**
	Remove every turn	**1.000**	**1.000**	**1.000**	**1.000**

Table 27: Archived synthetic-constructor transfer under the same restated-instruction protocol. Qwen1.5-MoE-A2.7B covers F2–F3 only, so its mean is over those two tasks.

Model	Policy	F1 additive	F2 corrective	F3 value reuse	Mean ↑
Mistral-7B	Single-turn reference	0.183	0.383	0.533	0.366
	Full history	0.300	0.383	0.069	0.251
	Remove at final turn	0.433	0.667	0.667	0.589
	Remove every turn	**0.617**	**0.933**	**0.867**	**0.806**
Gemma-2-9B	Single-turn reference	0.700	0.917	1.000	0.872
	Full history	0.683	0.467	0.050	0.400
	Remove at final turn	**0.867**	0.933	**1.000**	0.933
	Remove every turn	**0.867**	**0.981**	**1.000**	**0.949**

Model	Policy	F2 corrective	F3 value reuse	Mean ↑
Qwen1.5-MoE-A2.7B	Single-turn reference	0.250	0.750	0.500
Qwen1.5-MoE-A2.7B	Full history	0.400	0.150	0.275
Qwen1.5-MoE-A2.7B	**Remove at final turn**	0.650	0.883	0.767
Qwen1.5-MoE-A2.7B	**Remove every turn**	**0.867**	**1.000**	**0.934**

The archived aggregates show lower full-history scores than removal on these five models. The largest reported differences occur on F3: full history scores 0.069 on Mistral-7B, 0.050 on Gemma-2-9B, and 0.150 on Qwen1.5-MoE, compared with 0.867, 1.000, and 1.000 under every-turn removal. These constructors contain no later message that refers to an assistant response, so ACTF and complete assistant-history removal select the same record types. Without recoverable trial-level scoring, these cells remain a descriptive transfer diagnostic rather than a separate inferential test of the dependency rule.

C.3 Fully scored BFCL and MultiChallenge results

This section compares only populations for which every frozen ID received an official score. MultiChallenge tests when assistant-history omission is beneficial or harmful in ordinary conversation; the three BFCL subsets test whether ACTF preserves executable interaction state. Attempts without official scores appear only in the exclusion audit below.

Table 28: BFCL success and mean logged tokens per dialogue, Qwen2.5-14B. Every subset–policy cell has all 200 frozen IDs and sums tokens over all model steps. Input Δ is relative to full history within each subset. The pooled rows average token use over 600 dialogues per policy; no cross-subset success rate is reported.

Subset	Retained history	Success ↑	Input tok. (↓)	Output tok. (↓)	Input Δ (↓)
Base	Full history	**0.355**	52,594	825	reference
	ACTF	0.345	55,093	873	$+4.75\%$
	Mask observations after three turns	0.350	53,686	827	$+2.08\%$
	Remove calls and results	0.120	62,343	1,829	$+18.54\%$
Missing function	Full history	0.040	74,614	1,477	reference
	ACTF	0.040	73,222	1,454	-1.87%
Missing parameter	Full history	0.150	65,694	1,056	reference
	ACTF	0.150	64,644	1,042	-1.60%
Three-subset mean	Full history	—	64,301	1,119	reference
	ACTF	—	64,319	1,123	$+0.03\%$

The three included BFCL subsets contain 600 distinct dialogues, with 200 fully scored IDs per subset for each reported policy. ACTF differs from full history by -0.010, 0.000, and 0.000 on base, missing-function, and missing-parameter success. Removing prior calls and results loses 0.235 on base and increases input tokens by 18.54%. On the other two subsets ACTF uses modestly fewer input and output tokens at the same success point estimate. Across all three, mean token use is nearly unchanged: the measurable tool-use benefit is retaining the call–result state without the large base-subset loss from deleting it, not a pooled compression gain. Table 29 gives the policy-level accuracy breakdown.

Table 29: Fully scored BFCL v4 multi-turn subsets, Qwen2.5-14B. Each accuracy uses 200 frozen IDs in its subset. The base-only factorial ablation uses a separate complete 200-ID comparison.

Policy	Base ↑ n=200	Miss. func. ↑ n=200	Miss. param. ↑ n=200
Full history	**0.355**	0.040	**0.150**
ACTF	0.345	0.040	**0.150**
Mask observations after three turns	0.350	**0.045**	0.140

Base-only factorial policy	Full	Delete both	Calls only	Results only	Keep both
Accuracy ↑ (n=200 each)	**0.355**	0.120	0.155	0.320	0.315

Tool calls against tool results. On BFCL base, keeping only results reaches 0.320 and keeping only calls reaches 0.155, compared with 0.355 for full history, 0.345 for ACTF, and 0.120 when both are removed. Removing both records is harmful, and results carry more predictive information than calls alone in this subset. The factorial comparison does not establish an accuracy advantage from keeping both over results alone; call–result closure is a protocol constraint, not a measured positive interaction in these scores.

Table 30: Complete MultiChallenge pass rates with Qwen2.5-14B. Omit+merge is the compatibility rendering used by merged ACTF.

Axis	n	Full pass ↑	Omit pass ↑	Omit+merge pass ↑	Merged Δ (↑)
Inference memory	109	0.367	**0.459**	0.431	+0.064
Instruction retention	69	0.261	0.493	**0.507**	+0.246
Reliable version editing	40	**0.450**	0.375	0.375	-0.075
Self-coherence	48	0.312	0.354	**0.417**	+0.104
Overall	266	0.342	0.436	**0.440**	**+0.098**

The merged policy repairs 56 full-history failures and damages 30 successes (exact McNemar $p = 0.007$; paired bootstrap 95% CI for the mean change $[0.034, 0.169]$). The 0.075 loss on version editing is the counterexample: those prompts explicitly require revising earlier assistant content, so omission removes useful state.

C.3.1 Exclusion audit for tool populations without complete scoring

The remaining frozen tool populations were attempted at their planned sizes but are not part of the performance tables or paired significance tests. A missing official reward is not a failure and is not imputed. Table 31 is a coverage audit only; it contains no success rate or policy ranking. Long-context BFCL failures arise primarily when prompts exceed the fixed 32,768-token window. The τ^2 evaluator can discard trajectories after malformed tool JSON or unknown-tool replay. These mechanisms may depend on policy, so restricting to jointly scored IDs would change the evaluated population. The six other airline controls, each with 150 attempts, have unscored counts of 1 (`c0r1`), 0 (`c1r0`), 2 (`c1r1`), 2 (`last_exchange`), 2 (`mask_m10`), and 0 (`placeholder`); across all eleven airline policies, 20 of 1,650 attempts lack an official score.

Table 31: Attempted but excluded tool populations. All frozen attempts have raw records; official reward coverage is incomplete. The counts are per policy, not performance outcomes.

Population	Policy	Attempts	Unscored
BFCL long context	Full history	200	39
	ACTF	200	38
	Mask after three turns	200	39
τ^2 airline (50 tasks × 3 seeds)	Full history	150	1
	ACTF	150	2
	State-closed ACTF	150	2
	Remove calls/results	150	7
	Mask after three turns	150	1
τ^2 retail	Full history	114	38
	ACTF	114	32
	State-closed ACTF	114	32
	Remove calls/results	114	11
	Mask after three turns	114	38
Matched airline pilot (30 tasks)	Call-result retention	30	1
	State replacement	30	1

C.4 REMOVAL AT THE CACHE LEVEL

Our method edits messages before the forward pass; cache-row slicing is a separate systems intervention. We test their agreement on 48 seeded synthetic instances with Qwen2.5-7B and greedy decoding. Each instance contains an unreferenced wrong estimate and a referenced correct table. Slicing only the estimate reproduces text removal and re-prefilling on all 48 instances; slicing the table as well reduces agreement to 2 of 48. Retained KV rows may still encode removed text through earlier attention, so this diagnostic does not establish general equivalence. On 40 failed sharded mathematics conversations, evicting, masking, or position-scrambling earlier assistant rows scores 0.250 versus 0.275 for text removal; zeroing keys scores 0.000 because the entries remain in the attention softmax.

Table 32: Cache-row reuse after full prefill on 48 seeded synthetic instances, Qwen2.5-7B, greedy decoding.

Condition	Accuracy ↑	Estimate-echo rate ↓
Text, everything kept	1.000	0.000
Text, estimate removed	1.000	0.000
Text, estimate and table removed	0.062	0.021
Cache, estimate rows sliced after full prefill	1.000	0.000
Cache, estimate and table rows sliced	0.083	0.021
Re-prefill survivors without the estimate	1.000	0.000

C.5 TOOL CONTINUATION AND COMPLETION CHECKS

We test whether history selection can also improve the execution of the next user request. Each case starts from the same successfully executed prefix of an archived BFCL base dialogue. All policies receive the same current request, available tools, and initial tool state. We evaluate completion of this one user turn using the official state and returned-output checks. This measures performance conditional on the supplied prefix, separately from the whole-dialogue accuracy in Table 29.

Case selection and inference. The archive contains 287 eligible prefix positions. We select eight exploratory cases and 24 validation cases from different dialogues, with equal numbers involving one and two tool classes. Selection uses SHA-256 ordering with seeds 20260908 and 20260909, respectively; the validation manifest excludes the exploratory dialogues and is fixed before choosing the final policies. Qwen2.5-14B-Instruct uses greedy decoding, seed 17, at most 768 output tokens per response, and at most eight responses per continuation. The four validation policies run on a dedicated GPU with one request at a time, prefix caching disabled, and a 32,768-token context limit. Every response contributes to input and output cost, including failed attempts and completion checks. Reference actions and evaluator outputs are unavailable to the policy.

History selection and checking. We compare full history and our BFCL selection, each with and without a completion check. When the model produces no complete tool call and still has a response available, the check repeats the current user request and asks whether the executed calls and returned values satisfy its actions and conditions. It asks the model to use actual returned values for dependent arguments and complete any missing operation. Each continuation receives at most one such request within the same eight-response budget. A length-truncated response without a complete call also triggers the check. The full-history check retains assistant prose; the selected-history check removes it, including prose from the ongoing turn, while keeping actual calls and results.

Self-Refine (Madaan et al., 2023) uses model-generated feedback for iterative revision, while Reflexion (Shinn et al., 2023) retains verbal feedback across trials. Our comparison tests a completion check alongside history selection, using the current request and observed tool execution within one continuation. The eight-case exploration also tested request repetition, more frequent prose removal, references to repeated outputs, retaining the latest assistant reply, and sequential call generation. The supplementary records retain all these trials, including unsuccessful variants and a partially completed check run on a shared service. Final validation uses the same dedicated service for all four policies.

Table 33: Completion of one user turn from a shared successful prefix, 24 validation dialogues. Costs include every response in the continuation.

Policy	Completed (↑)	Mean input tok. (↓)	Mean output tok. (↓)
Full history	22/24	13,352	181
Calls and results retained	23/24	11,615	142
Full history with completion check	22/24	22,043	356
Calls and results with completion check	24/24	18,434	280

Table 33 shows that selection completes one more case than full history with 13.0% fewer input tokens and 21.8% fewer output tokens. Adding a check to selected history repairs one case and damages none, but increases input by 58.7% and output by 97.5%. The paired accuracy change is 0.042 (95% bootstrap interval $[0, 0.125]$; exact McNemar $p = 1.0$). Before the added check, all 24 trajectories in each checked policy match the corresponding baseline's prompts, responses, calls, and results exactly.

The repaired case attempts to fill a fuel tank beyond its capacity. After the tool returns an error, the model incorrectly describes the tank as full. The check leads it to query the fuel level and issue a corrected fill amount. Checking full history repairs the same case but damages another by purchasing travel insurance again when the user only requested an invoice. These outcomes motivate restricting checks to observed execution errors.

A separate test of checks after tool errors. After inspecting these results, we fix a new rule: issue the same check only if the current turn has produced an explicit tool error. An error is a nonempty top-level JSON `error` field or the executor's exception prefix. The trigger uses the returned error directly, so it may also check after an error that the model has already corrected. We select another 24 dialogues with seed 20260910, excluding all 32 dialogues used above, and compare full history, selection, and selection with this conditional check on the same dedicated server. All other decoding, step limits, and scoring rules remain fixed. This second sample evaluates the rule chosen after the first comparison.

We also run the universal check on these same 24 cases to compare its accuracy and cost directly with checking after errors. This additional control is fixed before examining the second sample's aggregate results.

Table 34: A separate 24-dialogue sample for checking after tool errors. All policies use the same cases and inference settings.

Policy	Completed (↑)	Checks	Mean input tok. (↓)	Mean output tok. (↓)
Full history	18/24	0	11,562	134
Calls and results retained	15/24	0	10,860	139
With universal completion check	16/24	24	17,129	260
With check after tool errors	16/24	3	11,678	154

Checking after errors matches the universal check's success/failure outcome on every case while using 31.8% fewer input tokens and 40.9% fewer output tokens (Table 34). Relative to selection without checking, it repairs one case and damages none, with 7.5% more input and 10.8% more output. The repaired case uses an incorrect departure airport and reverses the requested route; after the tool reports that the route is unavailable, the check uses the returned airport code and obtains the requested fare. The accuracy change is again 0.042 (95% bootstrap interval $[0, 0.125]$; exact McNemar $p = 1.0$).

Full history still completes 18 cases against 16 for either checked selection. The remaining differences include claiming to log out without calling the logout tool and performing additional state-changing actions, such as re-authenticating an unrelated service. Those actions need not produce

a tool error. Thus explicit errors provide a useful, inexpensive trigger for correction, but do not identify every incomplete or unnecessary action.

D STATISTICAL ROBUSTNESS

Part D separates evidence-delivery fidelity from final-answer scoring, then reports paired tests from the complete frozen cross-benchmark runs.

D.1 EVIDENCE DELIVERY AND FINAL-ANSWER EVALUATION

The two controls in this subsection answer different questions and use different denominators. Table 35 preserves the standard LiC scoring protocol but delivers every evidence shard verbatim on all 308 conversations. Table 36 removes repeated answer opportunities: it runs the 203 binary-graded mathematics and code conversations to the final shard and scores exactly one response per policy. We report them separately and do not pool their outcomes.

Table 35: LiC with verbatim evidence, Qwen2.5-14B ($n = 308$); the single-turn reference is 0.830.

	Full history	Remove every turn	Remove every turn, merged	ACTF	ACTF, merged
Accuracy ↑	0.629	0.864	0.844	0.825	**0.877**
Δ vs. full ↑	0.000	+0.235	+0.215	+0.195	**+0.248**

The standard LiC protocol can accept an answer before all evidence has arrived, giving multi-turn policies more than one opportunity to succeed. In the final-answer control, a response not recognized as an answer receives zero. Auditing the final-turn records reproduces every reported outcome: under verbatim delivery, full history has six such nonanswers and the two removal runs have none.

Table 36: One final answer per policy under rephrased and verbatim delivery, Qwen2.5-14B, 203 math and code instances. Changes use the full-history run of the same delivery condition.

Policy	Mathematics Acc. ↑	Code Acc. ↑	Mean Acc. ↑	Δ vs. full (↑)
Simulator-rephrased delivery				
Full history	0.534	0.320	0.429	0.000
Remove every turn	0.612	0.350	0.483	+0.054
Our method	0.563	0.370	0.468	+0.039
Our method, instruction restated	**0.893**	**0.670**	**0.783**	**+0.355**
Verbatim delivery; change against its full-history run				
Full history	0.709	0.420	0.567	0.000
Remove every turn	0.893	0.470	0.685	+0.118
Our method	**0.903**	**0.470**	**0.690**	**+0.123**

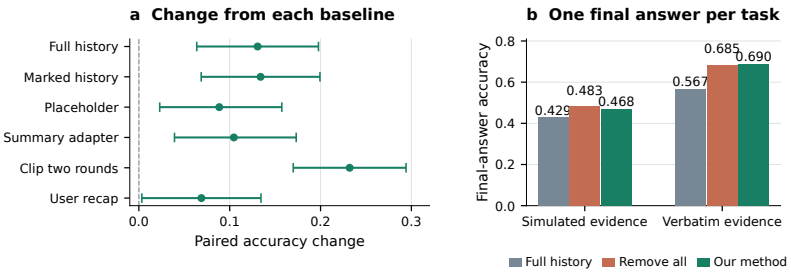

Figure 9: **Controlled scoring.** (a) Paired LiC changes with 95% bootstrap intervals. (b) With one final answer per task and verbatim evidence, ACTF gains 0.123 over full history, with a 95% interval of $[0.049, 0.197]$.

With one answer per policy, removing earlier responses gains 0.039 to 0.054 over full history on mathematics and code, with p above 0.18, and restating the instruction gains 0.355 at $p < 0.001$. With verbatim delivery, the gains with final-answer scoring are 0.118 and 0.123 at $p \leq 0.002$. Answer opportunities and evidence delivery both affect the measured gain, and removal remains effective with one answer after all shards arrive.

D.2 Budget controls and paired tests

The two equal-budget diagnostics in Table 37 use different tasks and metrics. Panel (a) is an archived BFCL run separate from the fully scored BFCL cohort in Table 29: full history is the uncapped reference, and the three selection controls are limited to 60% of its history budget. Each archived BFCL arm was evaluated in a separate process to avoid evaluator state carry-over. Panel (b) matches a 100-item mathematics sample to ACTF's token budget. These are within-cohort selection diagnostics, not scores that can be compared across panels or with the current BFCL run.

Table 37: Equal-budget controls. Changes are within each panel relative to full history; the BFCL panel reports paired tests.

(a) Archived BFCL, $n = 200$; selection controls capped at 0.60 of full history			
Criterion	Accuracy ↑	Δ	Paired p
Full history	**0.300**	0.000	–
Message length	0.195	-0.105	0.001
Recency	0.205	-0.095	0.002
Message role	0.140	-0.160	< 0.001

(b) Reasoning, $n = 100$, token budget matched to ACTF		
Criterion	Match rate ↑	Δ
Full history	0.230	0.000
Attention	0.090	-0.140
Random	0.100	-0.130
Message role	**0.270**	+0.040

All tests pair common task instances. The controlled LiC final-answer analysis above uses 8,000 paired bootstrap resamples (seed 20260908) and exact two-sided McNemar tests on binary outcomes: 43 repairs and 18 damages for our method ($p = 0.00187$), and 38 repairs and 14 damages for remove-every-turn ($p = 0.00120$). Table 38 includes only complete-scoring frozen populations. Appendix C.3.1 lists attempted populations omitted from these tests.

Table 38: Paired tests on fully scored frozen populations. LongBench reports a paired bootstrap interval, BFCL uses cluster sign-flips, and MultiChallenge uses an exact McNemar test. Listed p values are unadjusted; Holm-adjusted $p = 1$ for each BFCL row across the 47 planned core contrasts. No row uses a scored-only subset of an incompletely evaluated population.

Evaluation	Comparison	Paired n	Δ (↑)	Paired test
LongBench HotpotQA	Omit summaries versus full	200	0.000	CI $[-0.026, 0.026]$
MultiChallenge	Omit+merge versus full	266	+0.098	$p = 0.007$
BFCL base	ACTF versus full	200	-0.010	$p = 0.852$
BFCL missing function	ACTF versus full	200	0.000	$p = 1.000$
BFCL missing parameter	ACTF versus full	200	0.000	$p = 1.000$

Table 39: Paired-inference units and procedures. Repeated rollouts or seeds are clustered by task; archived diagnostics and incomplete tool populations are excluded.

Comparison	Unit	Interval / test
LiC primary and adaptations	308 IDs $\times$ 3	8,000 task-bootstrap resamples
LongBench HotpotQA	200 documents	8,000 paired bootstrap resamples
MultiChallenge	266 conversations	8,000 bootstrap resamples; exact McNemar
BFCL scored subsets	200 dialogues / subset	5,000 cluster bootstrap; sign-flip; Holm
Constructed mechanisms	40 cases / family $\times$ 3	5,000 cluster bootstrap; sign-flip; Holm

The paired results separate three regimes: omitting assistant history improves MultiChallenge overall, leaves LongBench neutral, and does not establish a change on any of the three fully scored BFCL subsets. This does not imply equivalence in tool use; the complete base ablation shows a large loss when both calls and results are removed. No inference is drawn from the omitted long-context or τ^2 reward subsets.