IDENTIFY THEN REINFORCE: ACTION-LEVEL CREDIT FOR GRAPH-STRUCTURED MEMORY AGENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Personal memories are interdependent: new experiences can update earlier facts, introduce conflicting claims, or connect to related information from distant interactions. Existing learnable memory systems primarily operate on flat textual entries, leaving these relationships implicit and making it difficult to reconcile evolving information and retrieve related evidence across time. We introduce **PERSONAMAP**, a relational memory framework that *learns how personal memories should be extracted, organized, and accessed.* To learn memory construction and retrieval jointly, we propose a *shared policy strategy* that extracts atomic claims from interaction histories, constructs semantic relations, and navigates the Personal Memory Graph through tool interactions. To distinguish the contributions of individual memory actions, we infer action-level credit from outcome differences across natural rollouts and invoke counterfactual rollouts *only when natural evidence is insufficient to identify an action's effect*. PERSONAMAP further supports structure-aware forgetting that prunes weakly connected claims while preserving strongly connected memories within the graph. Across two personalized long-term memory benchmarks, PERSONAMAP improves overall accuracy over the strongest baselines by 7.68 % and 4.29 %, respectively. Controlled comparisons support the benefit of action-level credit. With 50% fewer stored claims, structure-aware forgetting retains complete mapped relational evidence for 80%.

1 INTRODUCTION

Large language models (LLMs) have demonstrated strong capabilities in reasoning (Wei et al., 2022; Kojima et al., 2022), planning (Huang et al., 2022; Ichter et al., 2023), and tool use (Yao et al., 2023; Schick et al., 2023), supporting them as interactive agents across diverse applications (Park et al., 2023; Ichter et al., 2023; Zhou et al., 2024). Sustained interaction requires agents to retain and use information from an expanding history across sessions, making long-term memory essential for reliable interaction (Park et al., 2023; Packer et al., 2023; Zhong et al., 2024). Personalized interactions introduce a distinct challenge: agents must maintain an evolving understanding of the user rather than merely recall previous exchanges. User information is often implicit, changes over time, and is revealed across multiple sessions (Maharana et al., 2024; Wu et al., 2025; Jiang et al., 2025b). Later interactions may update earlier observations or introduce complementary and conflicting evidence (Wu et al., 2025; Wang et al., 2026a). Effective personal memory therefore requires organizing not only what has been observed, but also how observations relate over time.

Existing work has approached long-term memory along two directions. One direction increases the amount of history a model can use by expanding its context window (Dai et al., 2019; Wu et al., 2022; Wang et al., 2023; Peng et al., 2024). These methods improve long-context processing but do not store history as explicit records that can be independently organized and updated. Another direction maintains external memory banks that store interaction history and support retrieval of relevant records (Lewis et al., 2020; Park et al., 2023; Packer et al., 2023; Zhong et al., 2024). This makes memory scalable, but the stored records are usually independent text chunks, summaries, or facts. Recent methods connect these records with links, hierarchies, or multiple levels of abstraction (Xu et al., 2025; Chhikara et al., 2025; Xu et al., 2026b). These structures make relationships explicit, but are often built with predefined schemas and pipelines. Reinforcement learning also makes memory management learnable. Recent work trains memory policies through interaction rather than prescribing each memory rule (Schulman et al., 2017; Shao et al., 2024; Yu et al., 2025). These

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.

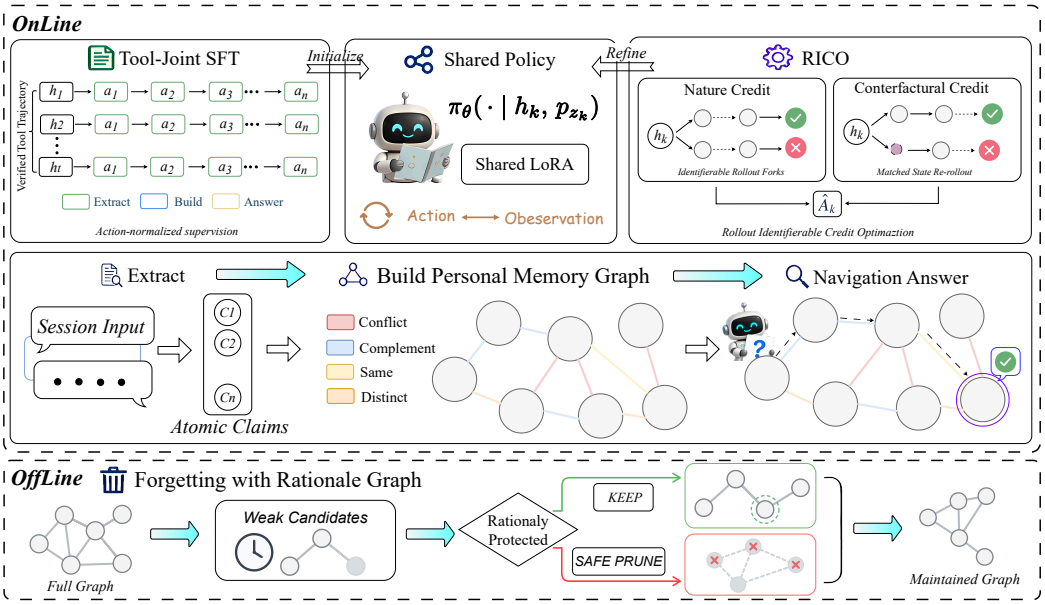


Figure 1: **Overview of PERSONAMAP. Online:** Tool-Joint SFT initializes policy for extracting atomic claims, constructing typed relations, and navigating the Personal Memory Graph. RICO then refines the policy using action-level credit obtained from natural rollout forks and selectively invoked counterfactual continuations. **Offline:** structure-aware forgetting identifies weak claims and prunes only those not protected by the graph, producing a compact maintained memory graph.

policies learn when to write, update, delete, or retrieve memories, using signals such as final task outcomes, process-level feedback, and comparisons among sampled trajectories (Yu et al., 2026a; Yan et al., 2026b; Wang et al., 2025c; Yu et al., 2026b; Xu et al., 2026a; Wang et al., 2026b). They treat memory management as a sequential decision problem: each action changes the memory state used by later interactions.

However, these advances leave two gaps. First, *structured memory systems typically construct relations using predefined rules, whereas learned policies often operate on flat entries*. Jointly learning which claims to connect and how to use those connections remains underexplored. Second, *trajectory-level feedback does not distinguish a faulty memory action from correct decisions in the same trajectory*. For example, an agent may correctly extract a new preference but misclassify its relation to an earlier claim, causing later queries to miss necessary evidence. Penalizing the entire trajectory may then penalize the correct extraction alongside the wrong relation decision. Then, the challenge is to IDENTIFY which memory action made the difference to the downstream outcome. In this paper, we address these gaps by proposing **PERSONAMAP**, a novel relational memory framework, as illustrated in Fig. 1 that learns the construction and access of a Personal Memory Graph. We further introduce ROLLOUT IDENTIFIABLE CREDIT OPTIMIZATION (**RICO**), which assigns credit at the action level when naturally sampled continuations distinguish downstream outcomes and invokes counterfactual rollouts when they leave an individual action's effect unresolved.

Our contributions are summarized as follows: **(1)** We introduce **PERSONAMAP**, which learns to construct and access personal memory as a graph of atomic claims and semantic relations, achieving strong performance on two long-term memory benchmarks. **(2)** We propose **RICO** to distinguish the effects of individual memory actions using natural rollout forks, with counterfactual continuations when natural evidence is insufficient. **(3)** We introduce structure-aware forgetting, which retains relational evidence for approximately 80% at a matched 50% claim-compression budget.

2 RELATED WORK

Long-Term Memory for LLM Agents. Long-term memory enables LLM agents to preserve historical information across sessions beyond a finite context window. Existing systems typically main-

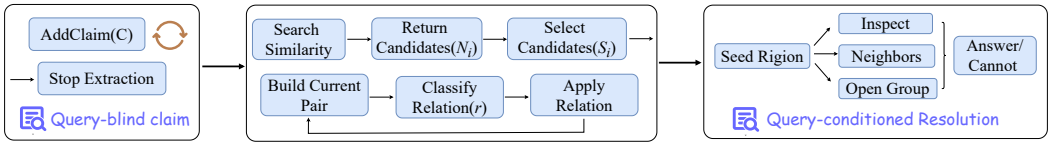


Figure 2: **PersonaMap tool-use pipeline.** The shared policy extracts claims, selects candidate claims and relation labels, and navigates the graph; the executor retrieves candidate pools, binds selected claim pairs, and applies graph updates.

tain external memory banks by adding, updating, and deleting memory entries (Park et al., 2023; Packer et al., 2023; Zhong et al., 2024). To organize growing interaction histories, recent methods introduce structured facts, dynamic graphs, and multi-granularity associations for relational retrieval (Rasmussen et al., 2025; Chhikara et al., 2025; Xu et al., 2025; Jiménez Gutiérrez et al., 2024; Xu et al., 2026b; Guo et al., 2026; Feng et al., 2026). More recent work replaces handcrafted rules with policies learned through supervision or reinforcement learning for memory construction. (Yu et al., 2026a; Yan et al., 2026b; Wang et al., 2025c; Yu et al., 2026b; Zhang et al., 2026; Xu et al., 2026a; Wang et al., 2026b). In personalized settings, recent studies introduce benchmarks and methods for tracking evolving user states and preferences, and integrate temporally distributed, implicit, or conflicting evidence (Maharana et al., 2024; Wu et al., 2025; Jiang et al., 2025a;b; Wang et al., 2026a; Xiao et al., 2026; Jung et al., 2026). However, learned policies still primarily operate on individual textual entries, while graph-based systems (Hamilton et al., 2017; Bordes et al., 2013) construct and update relations using predefined rules. *We thus represent personal histories as dynamic relational graphs, enabling to learn what to remember and how memories should be related.*

Credit Assignment in Agentic RL. Reinforcement learning builds on PPO-style optimization and critic-free group-relative variants such as GRPO and DAPO (Schulman et al., 2017; Shao et al., 2024; Yu et al., 2025). Credit assignment under delayed rewards has been studied through reward redistribution, hindsight-based attribution, and counterfactual contribution estimation (Arjona-Medina et al., 2019; Harutyunyan et al., 2019; Meulemans et al., 2023). For LLM agents, recent methods further decompose interaction trajectories into step-level training transitions or identify influential tokens and decisions (Luo et al., 2025; Lin et al., 2025; Wang et al., 2025b;a; Feng et al., 2025). However, token-level uncertainty does not measure the causal effect of an environment action, and natural trajectories reveal such effects only through comparable branches from shared states. Memory agents learn sequential writing, updating, deletion, and retrieval policies through end-task rewards, multi-conversation rollouts, curricula, or process supervision (Yu et al., 2026a; Yan et al., 2026b; Tan et al., 2025; Wang et al., 2025c; Zhang et al., 2026; Yu et al., 2026b; Xu et al., 2026a; Wang et al., 2026b). Despite expanding the memory-action space and training recipes, these methods generally do not test whether replacing an individual memory action changes the outcome under the same preceding memory state. *RICO first reuses comparable natural branches and invokes counterfactual continuations only when natural trajectories cannot identify an action's effect.*

3 METHOD

Given a sequence of interaction sessions, PersonaMap incrementally builds a persistent Personal Memory Graph and accesses it to answer later questions. We describe its graph representation, shared policy, supervised initialization, RICO optimization, and structure-aware forgetting below.

3.1 PERSONAMAP: POLICY-DRIVEN RELATIONAL MEMORY

PersonaMap manages personal memory through a learnable lifecycle. A shared policy extracts atomic claims from each interaction, organizes them through typed semantic relations, and accesses the resulting graph when responding to later requests. The graph schema specifies the valid memory structure, while its contents, relations, and use are determined by the learned policy.

Relational memory state. PersonaMap represents the persistent memory after session t as

$$\mathcal{G}_t = \left(\mathcal{V}_t^{\text{ent}}, \mathcal{V}_t^{\text{clm}}, \mathcal{V}_t^{\text{grp}}, \mathcal{E}_t^{\text{has}}, \mathcal{E}_t^{\text{sem}}\right), \tag{1}$$

where entity nodes organize personal subjects, claim nodes store atomic assertions, and conflict-group nodes collect unresolved alternatives. Structural edges in $\mathcal{E}_t^{\text{has}}$ associate entities with claims, while typed edges in $\mathcal{E}_t^{\text{sem}}$ relate claims across interactions. Each claim c_i is represented as

$$c_i = \langle e_i, \alpha_i, v_i, p_i, \text{ts}_i, \text{ctx}_i, \text{src}_i, \text{evid}_i, \kappa_i \rangle \,, \tag{2}$$

where e_i is the entity, α_i the attribute, v_i the value, p_i the polarity, ts_i the temporal scope, ctx_i the contextual qualifiers, src_i the source role, evid_i the supporting evidence, and κ_i the canonical key used for candidate retrieval.

Claim pairs may be connected by one of four semantic relations: SAMECLAIM identifies equivalent assertions; COMPLEMENTS connects compatible claims that provide additional information; CONFLICTSUNRESOLVED links incompatible claims under overlapping conditions; and DISTINCTBYQUALIFIER separates claims that differ because of temporal or contextual qualifiers. The decision UNRELATED creates no edge. A conflict relation additionally creates or extends a conflict group, allowing unresolved alternatives to remain jointly accessible rather than being collapsed by a latest-wins update.

Stage-conditioned memory policy. A fixed controller advances the agent through claim extraction, graph construction, and query-conditioned navigation. Within each stage, the same autoregressive policy generates tool actions:

$$a_k^z \sim \pi_\theta(\cdot \mid h_k^z, p_z), \qquad (\mathcal{S}_{k+1}^z, o_k^z) = \mathcal{T}_z(\mathcal{S}_k^z, a_k^z), \qquad z \in \{\text{ext}, \text{gph}, \text{nav}\}, \tag{3}$$

where p_z is the stage instruction, h_k^z contains the visible input and preceding action–observation history, and $\mathcal{S}_k^z$ contains the persistent graph and execution state. The resulting observation o_k^z is appended to the policy context for the next decision.

The policy determines the semantic content of memory operations, including what to extract, which claims to compare, how they relate, and which graph regions to access. The executor instead validates actions, binds graph endpoints, commits graph updates, and advances the interaction protocol deterministically.

Query-blind graph construction. Given a session x_t, the policy processes its dialogue without access to the future request q. It repeatedly emits $\text{AddClaim}(c)$ to create normalized claims and terminates extraction with StopExtraction. The policy determines each claim's content, the number of claims to store, and when extraction is complete. The executor assigns identifiers and inserts the claims with their entity associations.

For each new claim c_i, the policy invokes SearchSimilar, followed by $\text{SelectCandidates}(S_i)$ and pairwise $\text{ClassifyRelation}(r)$ decisions. The search tool combines canonical-key matching, entity–attribute lexical matching, and semantic retrieval. It fuses their rankings by reciprocal-rank fusion and returns the top 10 historical claims as $\mathcal{N}_i$, from which the policy selects $S_i \subseteq \mathcal{N}_i$ for relational comparison. The executor then binds each selected candidate c_j to the new claim and exposes the pair to the policy. For the environment-bound pair (c_i, c_j), the graph transition is

$$\mathcal{T}_{\text{rel}}(\mathcal{G}, c_i, c_j, r) = \begin{cases} \mathcal{G} \oplus (c_i, r, c_j), & r \in \mathcal{R}_{\text{sem}}, \\ \mathcal{G}, & r = \text{UNRELATED}, \end{cases} \tag{4}$$

where $\mathcal{R}_{\text{sem}} = \{\text{SAMECLAIM}, \text{COMPLEMENTS}, \text{CONFLICTSUNRESOLVED}, \text{DISTINCTBYQUALIFIER}\}$ and $\oplus$ commits the typed edge and updates conflict-group membership when necessary. Candidate selection and relation classification are policy decisions; endpoint binding and graph mutation are deterministic executor operations.

Graph navigation and response. When a subsequent request q arrives, a retrieval component first returns a bounded set of query-relevant seed claims. Starting from these seeds, the policy can issue INSPECTCLAIM, TRAVERSENEIGHBORHOOD, or OPENCONFLICTGROUP actions, and then produce an ANSWER or CANNOTANSWER response within a fixed interaction budget, as illustrated in Figure 2. Let $O_q = (o_1, \ldots, o_{T_q})$ denote the returned observations and $Z_q = \text{Compose}(O_q)$ the evidence claims used for terminal answering. The navigation interaction budget is B_q, so $T_q \leq B_q$.

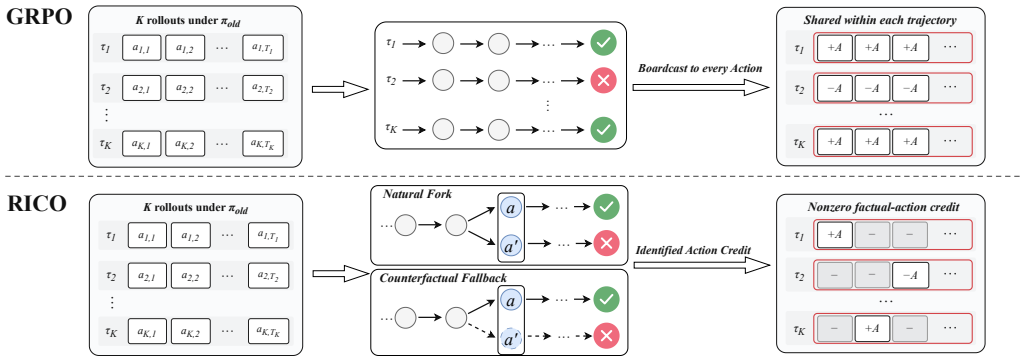


Figure 3: **Trajectory-wide credit in GRPO versus selective action credit in RICO.** GRPO assigns a trajectory-level advantage to every action, whereas RICO credits selected actions using natural forks and counterfactual fallback.

3.2 Tool-Joint Supervised Initialization

To obtain supervision for the learnable policy, we first build an offline structured memory-management pipeline. The pipeline uses GPT-5.1 (OpenAI, 2025) to produce schema-constrained memory decisions and deterministic procedures to execute them. We then translate the resulting execution traces into the tool protocol defined above, yielding trajectories for tool-joint supervised initialization. Appendix A describes the teacher pipeline and its conversion into executable tool demonstrations.

Tool actions differ substantially in length: an $\mathrm{AddClaim}$ action typically contains more tokens than a relation-classification or navigation action. To prevent longer actions from receiving greater weight solely because of their length, we normalize supervision within each action. For an action $a_j^\star = (a_{j,1}^\star, \ldots, a_{j,N_j}^\star)$, let $\mathcal{I}_j$ denote its supervised response-token positions and let $\mathcal{B}_+ = \{j : |\mathcal{I}_j| > 0\}$. We optimize

$$\mathcal{L}_{\mathrm{SFT}}(\theta) = -\frac{1}{|\mathcal{B}_+|} \sum_{j \in \mathcal{B}_+} \frac{1}{|\mathcal{I}_j|} \sum_{\ell \in \mathcal{I}_j} \log \pi_\theta \left(a_{j,\ell}^\star \mid h_j, p_{z_j}, a_{j,<\ell}^\star \right). \tag{5}$$

The loss first averages over supervised tokens within each action and then across actions in the minibatch. During supervised initialization, a single shared LoRA adapter (Hu et al., 2022) is trained on extraction, graph-construction, and navigation–response examples using the mixture detailed in Appendix B.1. This tool-joint initialization provides a shared starting point for all stages; RICO then refines selected actions according to their downstream consequences.

3.3 RICO: Refining Group-Relative Credit Assignment

RICO (*Rollout Identifiable Credit Optimization*) refines the credit assignment of Group Relative Policy Optimization (GRPO; Shao et al., 2024). Instead of assigning one outcome-derived advantage to every decision in a trajectory, it estimates selective action-level credit from comparable natural branches or counterfactual continuations. It retains critic-free group sampling and the clipped policy surrogate used by GRPO.

Rollout collection and Fork matching. For each training instance $(\mathcal{G}_{t-1}, x_t, q, y^\star)$, we sample K trajectories $\mathcal{D} = \{\tau^{(i)}\}_{i=1}^K$ under a fixed rollout policy π_{old}. Each trajectory extracts claims and builds the graph from x_t before seeing q, then navigates the graph to answer it; $y^\star$ is available only to the evaluator.

Let $E_i = 1$ if trajectory $\tau^{(i)}$ completes all required stages and terminates with Answer; a terminal CannotAnswer or invalid completion gives $E_i = 0$. A frozen Qwen3.8-27B judge (Qwen Team, 2026) J checks whether the question and observed claims Z_i support the reference answer $y^\star$. The terminal reward is

$$R_i = E_i \, \mathbf{1}[J(q, y^\star, Z_i) = 1]. \tag{6}$$

The training reward request supplies q, $y^\star$, and Z_i, but no benchmark reference facts; the judge does not score the generated answer.

RICO groups decisions by matched context s: the preceding action prefix, executor state, and policy-visible context must agree. Let $\mathcal{I}(s,a)$ be the eligible branches taking normalized action a at s, and $N(s,a) = |\mathcal{I}(s,a)|$. Eligibility requires a parsed trajectory with a binary terminal reward; trainable decisions additionally require recorded response tokens and must not be executor-forced. Natural and counterfactual branches compare terminal rewards under the same continuation policy. Further matching details appear in Appendix B.3.

Natural credit from rollout forks. As illustrated in Fig. 3, RICO first searches for naturally occurring rollout forks: trajectories that share the same decision prefix but take different actions. Different empirical continuation rewards provide candidate action-level training signals at the fork. To reduce sampling noise, we retain actions observed in at least two trajectories and average their continuation rewards:

$$\begin{aligned} \widehat{Q}_{\mathrm{nat}}(s,a) &= \frac{1}{N(s,a)} \sum_{i \in \mathcal{I}(s,a)} R_i, \\ \widehat{A}_{\mathrm{nat}}(s,a) &= \widehat{Q}_{\mathrm{nat}}(s,a) - \frac{\sum_{b \in \mathcal{A}_2(s)} N(s,b) \widehat{Q}_{\mathrm{nat}}(s,b)}{\sum_{b \in \mathcal{A}_2(s)} N(s,b)}, \end{aligned} \tag{7}$$

where $\mathcal{A}_2(s) = \{a : N(s,a) \geq 2\}$. A fork provides usable natural credit when at least two supported actions have different estimated continuation values. These empirical differences need not establish that the underlying action values differ. The resulting credit is assigned only to the original actions at the fork and requires no additional rollouts.

Counterfactual fallback. When a rollout group provides no usable natural credit, RICO selects at most B_{CF} unresolved decision contexts for counterfactual evaluation, as illustrated in Figure 3. The fallback is gated by the entire rollout group: once natural credit is available in that group, no counterfactual evaluation is invoked for its other unresolved contexts. For each selected context s, a denotes the action taken in the original trajectory and a' denotes a valid alternative. RICO reconstructs the preceding graph and replays the recorded prefix to restore s, forces either a or a', and then samples M suffix continuations per action using π_{old}. The continuation values and counterfactual credit are

$$\begin{aligned} \widehat{Q}_{\mathrm{CF}}(s,b) &= \frac{1}{M} \sum_{m=1}^{M} R^{(m)}(s,b), \qquad b \in \{a, a'\}, \\ \widehat{A}_{\mathrm{CF}}(s,a) &= \frac{1}{2} \left[\widehat{Q}_{\mathrm{CF}}(s,a) - \widehat{Q}_{\mathrm{CF}}(s,a') \right]. \end{aligned} \tag{8}$$

The factor $1/2$ centers the factual action against the mean of the two estimated branch values. Only the original action a receives this credit; the forced actions and sampled suffixes are used only for evaluation. If the two branches have equal estimated values, the context provides no training credit.

Selective action optimization. For each rollout group, RICO retains the original actions with nonzero natural credit; counterfactual credit is used only when the group provides no such actions. Let $\mathcal{F}$ denote the resulting credited action records, and let $\widehat{A}_j$ be the credit assigned to action j.

Because each action is generated auto-regressively, we broadcast its credit to its response tokens. For token ℓ of action j, the importance ratio is

$$r_{j,\ell}(\theta) = \frac{\pi_\theta\left(a_{j,\ell} \mid h_j, a_{j,<\ell}\right)}{\pi_{\mathrm{old}}\left(a_{j,\ell} \mid h_j, a_{j,<\ell}\right)}, \tag{9}$$

where h_j contains the stage instruction and preceding action–observation history.

Let $m_{j,\ell}$ indicate the tokens included in optimization and $D_j = \sum_\ell m_{j,\ell}$. The implementation masks prompts and executor observations, includes generated response tokens, and assigns zero advantage to uncredited actions retained in full-trajectory training records. RICO minimizes

$$\mathcal{L}_{\mathrm{RICO}}(\theta) = -\frac{1}{|\mathcal{F}|} \sum_{j \in \mathcal{F}} \frac{1}{D_j} \sum_{\ell} m_{j,\ell} \min\Big(r_{j,\ell}(\theta) \widehat{A}_j, \quad \mathrm{clip}\big(r_{j,\ell}(\theta), 1-\epsilon, 1+\epsilon\big) \widehat{A}_j \Big). \tag{10}$$

The objective first averages over tokens within each action and then over credited actions, preventing tool calls from receiving greater weight solely because of their length. Credit assignment is therefore action-level, while the clipped update is applied at the token level. Credits and old-policy probabilities are held fixed during optimization; if $\mathcal{F}$ is empty, the update is skipped. Uncredited actions receive no direct loss term, although updates to the shared policy can change their probabilities.

3.4 Structure-Aware Forgetting

PersonaMap applies a deterministic, query-independent forgetting operator after memory construction. The operator removes claims that are old and structurally weak, while protecting claims that participate in unresolved conflicts or remain connected to recent experience. It is a graph transformation applied after memory construction; it does not update the learned policy parameters.

Let $\mathcal{C}_t$ be the claim nodes in $\mathcal{G}_t$. For a claim c, let $\mathrm{age}_t(c) = t - t_{\mathrm{src}}(c)$ denote its age in sessions and $d_t(c) = |\mathcal{N}_t^{\mathrm{rel}}(c)|$ its number of distinct neighbors in the undirected claim-relation graph. An isolated claim is eligible for pruning once its age reaches τ_{stale}. A claim of degree at most $d_{\max}$ becomes eligible once its age reaches τ_{sparse}, provided it is disconnected from recent memory. Let $\mathcal{C}_{t,w}^{\mathrm{rec}}$ contain claims introduced in the most recent w sessions. We write $\mathrm{Reach}_t(c; \mathcal{C}_{t,w}^{\mathrm{rec}})$ if c is connected to this recent frontier by a claim-relation path of any length, and $\mathrm{Prot}_t(c)$ if c belongs to an unresolved conflict group. PersonaMap deletes c only when

$$\mathrm{Forget}_t(c) \iff \neg\,\mathrm{Prot}_t(c) \wedge \neg\,\mathrm{Reach}_t(c; \mathcal{C}_{t,w}^{\mathrm{rec}}) \wedge \Big[\big(\mathrm{age}_t(c) \geq \tau_{\mathrm{stale}} \wedge d_t(c) = 0\big) \quad \vee \big(\mathrm{age}_t(c) \geq \tau_{\mathrm{sparse}} \wedge d_t(c) \leq d_{\max}\big)\Big]. \tag{11}$$

The pruned graph removes these claims together with their incident edges, deletes any resulting orphan entities, and rebuilds conflict groups. The reported configuration and proxy evaluation are detailed in Appendix C. Age and sparse connectivity propose deletion candidates, whereas recent-memory reachability and conflict membership protect claims from deletion. These structural safeguards do not guarantee that every deleted claim will be irrelevant to future questions.

4 Experiment

We evaluate PersonaMap on two personalized long-term memory benchmarks: SubtleMemory (Wang et al., 2026a) and PersonaMem (Jiang et al., 2025a). We use SubtleMemory personas 0–6 for training and personas 7–9 for in-domain evaluation. To assess zero-shot cross-benchmark transferability, we directly evaluate the trained model on PersonaMem. Qwen2.5-7B-Instruct (Qwen Team, 2024) is our backbone. All experiments are conducted on 8 × NVIDIA A100 GPUs (80G). We use the same data splits and evaluation protocols for all compared methods.

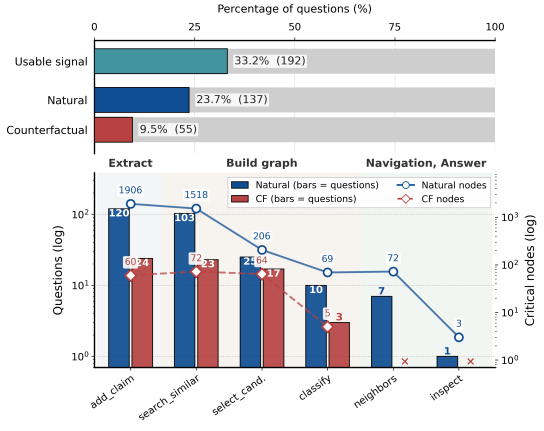


Figure 4: **RICO credit coverage.** Top: percentage and number of questions with usable credit from natural forks and counterfactual continuations. Bottom: question counts (bars) and credited decision-node counts (lines) for each tool action, grouped by memory stage.

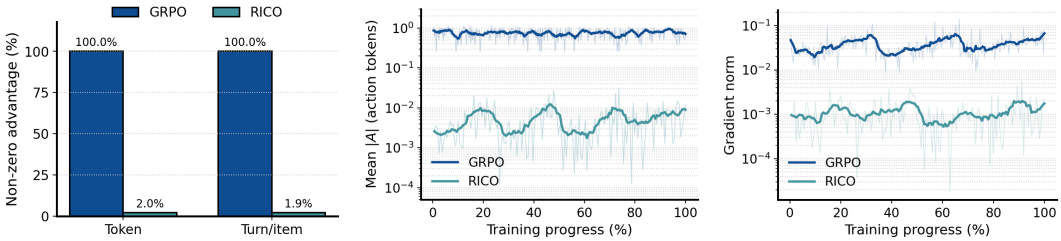
4.1 Overall Performance

Table 1 compares PersonaMap with retrieval-based, structured, and learned memory systems. On SubtleMemory, PersonaMap improves overall accuracy by 7.68 points over the strongest baseline, reaching 50.81%. It leads on complementary, nuanced, and contradictory questions by 2.03, 3.76, and 3.83 points over the strongest baseline in each category, respectively. The gains on nuanced and contradictory questions are consistent with the value of relating claims across interactions and keeping incompatible claims accessible.

Table 1: **Personalized long-term memory results.** We report answer accuracy on SUBTLEMEMORY (Wang et al., 2026a) and multiple-choice accuracy on PERSONAMEM (Jiang et al., 2025a) with 32K interaction histories under the persona-full protocol. All scores are percentages (↑). Bold indicates the best reported result in each column.

Method	SubtleMemory				PersonaMem						
	Com	Nua	Con	All	Evo	Rsn	Rcl	Rec	Gen	Ide	All
Training-free methods											
Full Context	4.67	5.31	1.85	4.27	47.48	72.73	44.52	43.64	24.56	24.73	44.82
RAG (Lewis et al., 2020)	31.78	49.28	0.00	32.23	56.83	71.72	47.95	43.64	31.58	25.81	48.56
Mem0 (Chhikara et al., 2025)	29.91	31.88	3.70	24.17	48.20	70.71	51.37	43.64	47.37	32.26	49.75
A-Mem (Xu et al., 2025)	49.53	61.84	0.93	43.13	46.76	69.70	50.68	49.09	31.58	32.26	48.05
LightMem (Fang et al., 2026)	33.64	33.82	3.70	26.07	57.55	67.68	43.84	**58.18**	40.35	35.48	50.76
Mem0^g (Chhikara et al., 2025)	27.10	12.56	1.85	13.51	47.48	70.41	51.37	45.45	38.60	29.03	48.30
MemoryOS (Kang et al., 2025)	33.64	52.17	3.70	35.07	48.92	72.73	45.89	45.45	35.09	27.96	47.20
GAM (Xu et al., 2026b)	39.25	55.56	2.78	37.91	50.36	71.72	49.32	49.09	33.33	33.33	49.24
Trainable methods											
MemAgent (Yu et al., 2026a)	14.02	8.70	4.63	9.00	52.46	72.41	44.70	38.78	27.27	27.71	46.02
Mem-α (Wang et al., 2025c)	14.95	15.46	3.70	12.32	57.55	**85.86**	57.53	50.91	24.56	17.20	52.12
UMA (Zhang et al., 2026)	19.63	24.64	14.81	20.85	57.55	77.78	38.36	49.09	33.33	10.75	45.67
MEM1 (Zhou et al., 2026)	15.89	9.18	3.70	9.48	58.20	61.80	37.12	30.61	36.36	24.10	43.40
Memory-R2 (Yan et al., 2026a)	42.99	42.03	7.41	33.41	**63.31**	69.70	26.71	30.91	21.05	13.98	40.41
PERSONAMAP	**51.56**	**65.60**	**18.64**	**50.81**	44.60	77.89	**69.86**	47.27	**52.63**	**38.71**	**56.41**

Abbreviations. Com: complementary; Nua: nuanced; Con: contradictory; Evo: preference evolution; Rsn: update reason; Rcl: fact recall (merging `recall_user_shared_facts` and `recalling_facts_mentioned_by_the_user`); Rec: recommendation; Gen: generalization; Ide: idea suggestion.



(a) Advantage sparsity. (b) Mean advantage magnitude. (c) Gradient norm.

Figure 5: **Optimization behavior of RICO under equal data budgets.** **(a)** GRPO assigns non-zero advantages to 100% of tokens, whereas RICO updates only approximately 2% of critical action tokens. **(b)** RICO produces advantages that are 1–2 orders of magnitude smaller than those of GRPO. **(c)** RICO consequently exhibits a consistently smaller gradient norm.

Without target-benchmark adaptation, PersonaMap reaches 56.41% overall accuracy on PERSONAMEM, a 4.29-point improvement. It also leads in fact recall by 12.33 points and generalization by 5.26 points, and obtains the highest accuracy in idea suggestion. These results suggest that the learned graph can support both access to stored facts and their use in new requests.

4.2 MECHANISM ANALYSIS

Figure 4 examines whether RICO can recover sufficiently broad training signals despite assigning credit only to identifiable actions. Across the evaluated questions, RICO obtains usable credit for 33.2% . Naturally occurring rollout forks account for 23.7% , while counterfactual continuations recover an additional 9.5% for contexts that cannot be resolved from the sampled trajectories alone. Natural credit therefore provides the majority of supervision, with counterfactual evaluation serving as a targeted fallback rather than the primary source of training data. The identified decisions span claim extraction, graph construction, and navigation, showing that RICO can localize consequential actions throughout the memory lifecycle.

Optimization behavior. Figure 5 compares the two methods under the same rollout budget. GRPO assigns non-zero advantages to 100% of tokens and decision turns, whereas RICO assigns

them to only 2.0% of tokens and 1.9% of turns (Fig. 5a). Consistent with this selective assignment, RICO's measured mean absolute action-token advantage is one to two orders of magnitude lower (b), and its gradient norms are generally lower throughout training (c). These results show that RICO concentrates credit on a small subset of decisions and produces lower-magnitude policy updates under the same rollout budget.

4.3 Ablation Study

Table 2 compares overall answer accuracy under the same graph representation, tool interface, and evaluation protocol. Tool-Joint SFT reaches 45.88%, slightly below the structured graph baseline (46.99%), indicating that supervised initialization alone does not improve on the fixed construction pipeline. Starting from the same SFT checkpoint (45.88%), RICO reaches 50.81% accuracy, compared with 48.12% for GRPO. The latter comparison supports the benefit of selective action-level credit under this training setting. The prompted graph performs substantially worse (31.58%), further showing that access to the graph interface alone is insufficient to obtain the gains.

Table 2: **Ablation on SubtleMemory.**

Method	Accuracy (%)
Structured Graph	46.99
Prompted Graph	31.58
Tool-Joint SFT	45.88
SFT + GRPO	48.12
SFT + RICO	**50.81**

4.4 Forgetting with Safe Pruning

We evaluate structure-aware forgetting on the final claim graphs of SubtleMemory. Of the 340 Full questions with fully mapped gold claims, 220 have at least one semantic relation in their gold-and-seed neighborhoods and are included.

For each question q, let $\mathcal{C}_q$ denote its mapped gold claims and $\mathcal{E}_q$ its relevant semantic relations. Given a pruned graph $\mathcal{G}_t^- = (\mathcal{V}_t^-, \mathcal{E}_t^-)$, we define

$$\text{Retain}(q) = \mathbf{1}\left[\mathcal{C}_q \subseteq \mathcal{V}_t^- \ \wedge \ \mathcal{E}_q \subseteq \mathcal{E}_t^-\right]. \quad (12)$$

Evidence retention is the percentage of questions for which all mapped gold claims and relevant relations survive pruning.

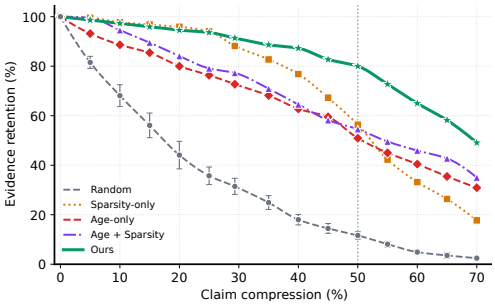


Figure 6: **Evidence retention under matched compression.**

We compare PersonaMap with random, age-only, sparsity-only, and age-plus-sparsity pruning. At each compression ratio, all methods remove the same number of claims according to their respective ranking rules.

Figure 6 shows that, at 50% claim compression, PersonaMap retains complete mapped relational evidence for approximately 80% of eligible questions, compared with 56% for the strongest alternative and 11% for random pruning. At 70% compression, PersonaMap retains approximately 49%, versus 18% for sparsity-only pruning. The results indicate that combining sparsity with temporal constraints, recent-memory reachability, and conflict protection helps preserve relational evidence under pruning.

5 Conclusion

We introduced PersonaMap, which learns to extract personal claims, organize their semantic relations, and navigate the resulting memory graph through a shared stage-conditioned policy. RICO refines this policy by assigning credit to consequential actions identified through natural rollout forks and selective counterfactual continuations. PersonaMap achieves 50.81% overall accuracy on SubtleMemory and 56.41% on PersonaMem, outperforming the strongest baseline on both benchmarks. In the controlled ablation, RICO reaches 50.81%, compared with 48.12% for GRPO. Structure-aware forgetting further removes 50% of stored claims while retaining complete mapped relational evidence for approximately 80% of eligible questions. These results show the value of learning not only what to remember, but also how memories should be related and which memory actions contribute to downstream outcomes.

AI USE STATEMENT

We used generative AI tools to assist with implementing research code and editing the manuscript for clarity. As part of the proposed method, GPT-5.1 generated structured memory-management decisions used to construct supervised demonstrations, and a frozen Qwen3.8-27B model evaluated evidence sufficiency during RICO training. The authors reviewed AI-assisted code and text, and validated the generated demonstrations by replaying them through the tool executor. We take responsibility for the final implementation, manuscript, and reported results.

REFERENCES

Jose A. Arjona-Medina, Michael Gillhofer, Michael Widrich, Thomas Unterthiner, Johannes Brandstetter, and Sepp Hochreiter. RUDDER: Return decomposition for delayed rewards. In *Advances in Neural Information Processing Systems*, volume 32, pp. 13544–13555. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper/2019/hash/16105fb9cc614fc29e1bda00dab60d41-Abstract.html.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Durán, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. In *Advances in Neural Information Processing Systems*, volume 26, pp. 2787–2795, 2013. URL https://proceedings.neurips.cc/paper/2013/hash/1cecc7a77928ca8133fa24680a88d2f9-Abstract.html.

Prateek Chhikara, Dev Khant, Saket Aryan, Taranjeet Singh, and Deshraj Yadav. Mem0: Building production-ready AI agents with scalable long-term memory. *arXiv preprint arXiv:2504.19413*, 2025. URL https://arxiv.org/abs/2504.19413.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc V. Le, and Ruslan Salakhutdinov. Transformer-xl: Attentive language models beyond a fixed-length context. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 2978–2988, 2019. doi: 10.18653/v1/P19-1285.

Jizhan Fang, Xinle Deng, Haoming Xu, Ziyan Jiang, Yuqi Tang, Ziwen Xu, Shumin Deng, Yunzhi Yao, Mengru Wang, Shuofei Qiao, Huajun Chen, and Ningyu Zhang. LightMem: Lightweight and efficient memory-augmented generation. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=dyJ0GWpjJB.

Lang Feng, Zhenghai Xue, Tingcong Liu, and Bo An. Group-in-group policy optimization for LLM agent training. In *Advances in Neural Information Processing Systems*, volume 38, 2025. doi: 10.52202/085713-1544. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/420c9f777c0b4f78d515e53cf74d58b2-Abstract-Conference.html.

Tao Feng, Chongrui Ye, Tianyang Luo, Jingjun Xu, Xueqiang Xu, Haozhen Zhang, Zhigang Hua, Yan Xie, Shuang Yang, Ge Liu, and Jiaxuan You. ExpGraph: Model-agnostic experience learning with graph-structured memory for LLM agents. *arXiv preprint arXiv:2605.30712*, 2026. URL https://arxiv.org/abs/2605.30712.

Weidong Guo, Dakai Wang, Zixuan Wang, Hui Liu, and Yu Xu. T-Mem: Memory that anticipates, not archives. *arXiv preprint arXiv:2606.15405*, 2026. URL https://arxiv.org/abs/2606.15405.

William L. Hamilton, Rex Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems*, volume 30, pp. 1024–1034, 2017. URL https://proceedings.neurips.cc/paper/2017/hash/5dd9db5e033da9c6fb5ba83c7a7ebea9-Abstract.html.

Anna Harutyunyan, Will Dabney, Thomas Mesnard, Mohammad Gheshlaghi Azar, Bilal Piot, Nicolas Heess, Hado P. van Hasselt, Gregory Wayne, Satinder Singh, Doina Precup, and Rémi Munos. Hindsight credit assignment. In *Advances in Neural Information Processing Systems*, volume 32, pp. 12467–12476. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper/2019/hash/195f15384c2a79cedf293e4a847ce85c-Abstract.html.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Wenlong Huang, Pieter Abbeel, Deepak Pathak, and Igor Mordatch. Language models as zero-shot planners: Extracting actionable knowledge for embodied agents. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 9118–9147. PMLR, 2022. URL https://proceedings.mlr.press/v162/huang22a.html.

Brian Ichter, Anthony Brohan, Yevgen Chebotar, Chelsea Finn, Karol Hausman, Alexander Herzog, Daniel Ho, Julian Ibarz, Alex Irpan, Eric Jang, Ryan Julian, Dmitry Kalashnikov, Sergey Levine, Yao Lu, Carolina Parada, Kanishka Rao, Pierre Sermanet, Alexander T. Toshev, Vincent Vanhoucke, Fei Xia, Ted Xiao, Peng Xu, Mengyuan Yan, Noah Brown, Michael Ahn, Omar Cortes, Nicolas Sievers, Clayton Tan, Sichun Xu, Diego Reyes, Jarek Rettinghouse, Jornell Quiambao, Peter Pastor, Linda Luu, Kuang-Huei Lee, Yuheng Kuang, Sally Jesmonth, Nikhil J. Joshi, Kyle Jeffrey, Rosario Jauregui Ruano, Jasmine Hsu, Keerthana Gopalakrishnan, Byron David, Andy Zeng, and Chuyuan Kelly Fu. Do as i can, not as i say: Grounding language in robotic affordances. In *Proceedings of the 6th Conference on Robot Learning*, volume 205 of *Proceedings of Machine Learning Research*, pp. 287–318. PMLR, 2023. URL https://proceedings.mlr.press/v205/ichter23a.html.

Bowen Jiang, Zhuoqun Hao, Young-Min Cho, Bryan Li, Yuan Yuan, Sihao Chen, Lyle Ungar, Camillo J. Taylor, and Dan Roth. Know me, respond to me: Benchmarking LLMs for dynamic user profiling and personalized responses at scale. In *Proceedings of the Conference on Language Modeling*, 2025a. URL https://openreview.net/forum?id=6ox8XZGOqP.

Bowen Jiang, Yuan Yuan, Maohao Shen, Zhuoqun Hao, Zhangchen Xu, Zichen Chen, Ziyi Liu, Anvesh Rao Vijjini, Jiashu He, Hanchao Yu, Radha Poovendran, Gregory Wornell, Lyle Ungar, Dan Roth, Sihao Chen, and Camillo Jose Taylor. PersonaMem-v2: Towards personalized intelligence via learning implicit user personas and agentic memory. *arXiv preprint arXiv:2512.06688*, 2025b. URL https://arxiv.org/abs/2512.06688.

Bernal Jiménez Gutiérrez, Yiheng Shu, Yu Gu, Michihiro Yasunaga, and Yu Su. HippoRAG: Neurobiologically inspired long-term memory for large language models. In *Advances in Neural Information Processing Systems*, volume 37, 2024. doi: 10.52202/079017-1902. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/6ddc001d07ca4f319af96a3024f6dbd1-Abstract-Conference.html.

Seokwon Jung, Alexander Rubinstein, Arnas Uselis, Sangdoo Yun, and Seong Joon Oh. MEME: Multi-entity and evolving memory evaluation. *arXiv preprint arXiv:2605.12477*, 2026. URL https://arxiv.org/abs/2605.12477.

Jiazheng Kang, Mingming Ji, Zhe Zhao, and Ting Bai. Memory OS of AI agent. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2025. URL https://aclanthology.org/2025.emnlp-main.1318/.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners. In *Advances in Neural Information Processing Systems*, volume 35, pp. 22199–22213, 2022. doi: 10.52202/068431-1613. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/8bb0d291acd4acf06ef112099c16f326-Abstract-Conference.html.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive nlp tasks. In *Advances in Neural Information Processing Systems*, volume 33, pp. 9459–9474, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/6b493230205f780e1bc26945df7481e5-Abstract.html.

Zicheng Lin, Tian Liang, Jiahao Xu, Qiuzhi Liu, Xing Wang, Ruilin Luo, Chufan Shi, Siheng Li, Yujiu Yang, and Zhaopeng Tu. Critical tokens matter: Token-level contrastive estimation enhances LLM's reasoning capability. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 37906–37918. PMLR, 2025. URL https://proceedings.mlr.press/v267/lin25j.html.

Xufang Luo, Yuge Zhang, Zhiyuan He, Zilong Wang, Siyun Zhao, Dongsheng Li, Luna K. Qiu, and Yuqing Yang. Agent lightning: Train any ai agents with reinforcement learning. *arXiv preprint arXiv:2508.03680*, 2025. URL https://arxiv.org/abs/2508.03680.

Adyasha Maharana, Dong-Ho Lee, Sergey Tulyakov, Mohit Bansal, Francesco Barbieri, and Yuwei Fang. Evaluating very long-term conversational memory of LLM agents. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 13851–13870. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.acl-long.747. URL https://aclanthology.org/2024.acl-long.747/.

Alexander Meulemans, Simon Schug, Seijin Kobayashi, Nathaniel D. Daw, and Gregory Wayne. Would I have gotten that reward? long-term credit assignment by counterfactual contribution analysis. In *Advances in Neural Information Processing Systems*, volume 36, pp. 68685–68735. Curran Associates, Inc., 2023. doi: 10.52202/075280-3006. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/d8bd445c2abe1343cce0e14b361b2fb3-Abstract-Conference.html.

OpenAI. GPT-5.1: A smarter, more conversational ChatGPT. https://openai.com/index/gpt-5-1/, November 2025. Accessed: 2026-09-22.

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 Jun Cai, Meredith Ringel Morris, Percy Liang, and Michael S. Bernstein. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*, 2023. doi: 10.1145/3586183.3606763. URL https://doi.org/10.1145/3586183.3606763.

Bowen Peng, Jeffrey Quesnelle, Honglu Fan, and Enrico Shippole. YaRN: Efficient context window extension of large language models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=wHBfxhZu1u.

Qwen Team. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024. doi: 10.48550/arXiv.2412.15115. URL https://arxiv.org/abs/2412.15115.

Qwen Team. Qwen3.8-Max: A new bar for coding and cowork, August 2026. URL https://qwen.ai/blog?id=qwen3.8.

Preston Rasmussen, Pavlo Paliychuk, Travis Beauvais, Jack Ryan, and Daniel Chalef. Zep: A temporal knowledge graph architecture for agent memory. *arXiv preprint arXiv:2501.13956*, 2025. URL https://arxiv.org/abs/2501.13956.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessi, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. In *Advances in Neural Information Processing Systems*, volume 36, pp. 68539–68551, 2023. doi: 10.52202/075280-2997. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/d842425e4bf79ba039352da0f658a906-Abstract-Conference.html.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017. URL https://arxiv.org/abs/1707.06347.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. DeepSeekMath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024. URL https://arxiv.org/abs/2402.03300.

Zhen Tan, Jun Yan, I-Hung Hsu, Rujun Han, Zifeng Wang, Long Le, Yiwen Song, Yanfei Chen, Hamid Palangi, George Lee, Anand Rajan Iyer, Tianlong Chen, Huan Liu, Chen-Yu Lee, and Tomas Pfister. In prospect and retrospect: Reflective memory management for long-term personalized dialogue agents. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 8416–8439. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.acl-long.413. URL https://aclanthology.org/2025.acl-long.413/.

Hanlin Wang, Jian Wang, Chak Tou Leong, and Wenjie Li. STeCa: Step-level trajectory calibration for LLM agent learning. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 11597–11614. Association for Computational Linguistics, 2025a. doi: 10.18653/v1/2025.findings-acl.604. URL https://aclanthology.org/2025.findings-acl.604/.

Shenzhi Wang, Le Yu, Chang Gao, Chujie Zheng, Shixuan Liu, Rui Lu, Kai Dang, Xiong-Hui Chen, Jianxin Yang, Zhenru Zhang, Yuqiong Liu, An Yang, Andrew Zhao, Yang Yue, Shiji Song, Bowen Yu, Gao Huang, and Junyang Lin. Beyond the 80/20 rule: High-entropy minority tokens drive effective reinforcement learning for LLM reasoning. In *Advances in Neural Information Processing Systems*, volume 38, 2025b. doi: 10.52202/085713-3850. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/a797c2d2e0c1fdabf4d1ab8cd0b465c6-Abstract-Conference.html.

Weizhi Wang, Li Dong, Hao Cheng, Xiaodong Liu, Xifeng Yan, Jianfeng Gao, and Furu Wei. Augmenting language models with long-term memory. In *Advances in Neural Information Processing Systems*, volume 36, pp. 74530–74543, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/ebd82705f44793b6f9ade5a669d0f0bf-Abstract-Conference.html.

Wenxuan Wang, Haoyu Sun, Fukuan Hou, Mingyang Song, Weinan Zhang, Yu Cheng, and Yang Yang. SubtleMemory: A benchmark for fine-grained relational memory discrimination in long-horizon AI agents. *arXiv preprint arXiv:2606.05761*, 2026a. URL https://arxiv.org/abs/2606.05761.

Xinyu Wang, Mingze Li, Peng Lu, Xiao-Wen Chang, Lifeng Shang, Jinpeng Li, Fei Mi, Prasanna Parthasarathi, and Yufei Cui. InfMem: Learning system-2 memory control for long-context agent. *arXiv preprint arXiv:2602.02704*, 2026b. URL https://arxiv.org/abs/2602.02704.

Yu Wang, Ryuichi Takanobu, Zhiqi Liang, Yuzhen Mao, Yuanzhe Hu, Julian McAuley, and Xiaojian Wu. Mem-α: Learning memory construction via reinforcement learning. *arXiv preprint arXiv:2509.25911*, 2025c. URL https://arxiv.org/abs/2509.25911.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc V. Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, volume 35, pp. 24824–24837, 2022. doi: 10.52202/068431-1800. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/9d5609613524ecf4f15af0f7b31abca4-Abstract-Conference.html.

Di Wu, Hongwei Wang, Wenhao Yu, Yuwei Zhang, Kai-Wei Chang, and Dong Yu. LongMemEval: Benchmarking chat assistants on long-term interactive memory. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/d813d324dbf0598bbdc9c8e79740ed01-Abstract-Conference.html.

Yuhuai Wu, Markus N. Rabe, DeLesley Hutchins, and Christian Szegedy. Memorizing transformers. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=TrjbxzRcnf-.

Jianfei Xiao, Xiang Yu, Chengbing Wang, Wuqiang Zheng, Xinyu Lin, Kaining Liu, Hongxun Ding, Yang Zhang, Wenjie Wang, Fuli Feng, and Xiangnan He. AlpsBench: An LLM personalization benchmark for real-dialogue memorization and preference alignment. In *Proceedings of the 49th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2026. doi: 10.1145/3805712.3808634. URL https://doi.org/10.1145/3805712.3808634.

Derong Xu, Shuochen Liu, Pengfei Luo, Pengyue Jia, Yingyi Zhang, Yi Wen, Yimin Deng, Wenlin Zhang, Enhong Chen, Xiangyu Zhao, and Tong Xu. Learning how and what to memorize: Cognition-inspired two-stage optimization for evolving memory. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 44987–45011. Association for Computational Linguistics, 2026a. URL https://aclanthology.org/2026.acl-long.2084/.

Derong Xu, Yi Wen, Pengyue Jia, Yingyi Zhang, Wenlin Zhang, Yichao Wang, Huifeng Guo, Ruiming Tang, Xiangyu Zhao, Enhong Chen, and Tong Xu. From single to multi-granularity: Toward long-term memory association and selection of conversational agents. In *The Fourteenth International Conference on Learning Representations*, 2026b. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/93e444b4a1a64815631c3c74977ee200-Paper-Conference.pdf.

Wujiang Xu, Zujie Liang, Kai Mei, Hang Gao, Juntao Tan, and Yongfeng Zhang. A-Mem: Agentic memory for LLM agents. In *Advances in Neural Information Processing Systems*, volume 38, 2025. doi: 10.52202/085713-0593. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/19909c36f51abc4856b4560aff3d36d6-Abstract-Conference.html.

Sikuan Yan, Ahmed Bahloul, Ercong Nie, Susanna Schwarzmann, Riccardo Trivisonno, Volker Tresp, and Yunpu Ma. Memory-R2: Fair credit assignment for long-horizon memory-augmented LLM agents. *arXiv preprint arXiv:2605.21768*, 2026a. URL https://arxiv.org/abs/2605.21768.

Sikuan Yan, Xiufeng Yang, Zuchao Huang, Ercong Nie, Zifeng Ding, Zonggen Li, Xiaowen Ma, Jinhe Bi, Kristian Kersting, Jeff Z. Pan, Hinrich Schütze, Volker Tresp, and Yunpu Ma. Memory-R1: Enhancing large language model agents to manage and utilize memories via reinforcement learning. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 12805–12825. Association for Computational Linguistics, 2026b. URL https://aclanthology.org/2026.acl-long.583/.

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

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 *The Fourteenth International Conference on Learning Representations*, 2026a. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/4264ee4376776907c0b87ed70b959585-Paper-Conference.pdf.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Juncai Liu, Lingjun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, Ru Zhang, Wang Zhang, Hang Zhu, Jinhua Zhu, Jiaze Chen, Jiangjie Chen, Chengyi Wang, Hongli Yu, Yuxuan Song, Xiangpeng Wei, Hao Zhou, Jingjing Liu, Wei-Ying Ma, Ya-Qin Zhang, Lin Yan, Yonghui Wu, and Mingxuan Wang. DAPO: An open-source LLM reinforcement learning system at scale. In *Advances in Neural Information Processing Systems*, volume 38, 2025. doi: 10.52202/085713-3775. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/a4277440d50f1f15d2cb4c14f7e0c0d2-Abstract-Conference.html.

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 (Volume 1: Long Papers)*, pp. 21457–21483. Association for Computational Linguistics, 2026b. URL https://aclanthology.org/2026.acl-long.981/.

Kehao Zhang, Shangtong Gui, Sheng Yang, Wei Chen, and Yang Feng. Learning to remember: End-to-end training of memory agents for long-context reasoning. *arXiv preprint arXiv:2602.18493*, 2026. URL https://arxiv.org/abs/2602.18493.

Wanjun Zhong, Lianghong Guo, Qiqi Gao, He Ye, and Yanlin Wang. MemoryBank: Enhancing large language models with long-term memory. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 19724–19731, 2024. doi: 10.1609/aaai.v38i17.29946. URL https://ojs.aaai.org/index.php/AAAI/article/view/29946.

Shuyan Zhou, Frank F. Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, Uri Alon, and Graham Neubig. WebArena: A realistic web environment for building autonomous agents. In *The Twelfth International Conference on Learning Representations*, pp. 15585–15606, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/4410c0711e9154a7a2d26f9b3816d1ef-Abstract-Conference.html.

Zijian Zhou, Ao Qu, Zhaoxuan Wu, Sunghwan Kim, Alok Prakash, Daniela Rus, Jinhua Zhao, Bryan Kian Hsiang Low, and Paul Pu Liang. MEM1: Learning to synergize memory and reasoning for efficient long-horizon agents. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=XY8AaxDSLb.

A Offline Teacher Pipeline and Trajectory Construction

This section describes the offline pipeline used to construct demonstrations for tool-joint supervised initialization. The pipeline first produces structured memory-management decisions with GPT-5.1 (OpenAI, 2025), executes these decisions using deterministic graph procedures, and then converts the resulting traces into the PersonaMap tool protocol defined in Section 3.1. The pipeline is used to construct the demonstration dataset; the learned policy subsequently generates the corresponding decisions directly.

A.1 Pipeline Overview

Given a session x_t and the preceding graph $\mathcal{G}_{t-1}$, the pipeline constructs an executed trace in three stages:

1. extracting and refining atomic claims from x_t;
2. retrieving historical candidates and constructing typed relations for the new claims; and
3. generating query-conditioned graph-navigation and response traces.

Claim extraction and graph construction are query-blind: neither the subsequent question q nor its reference answer $y^\star$ is available during these two stages. The question and reference answer are introduced only when constructing navigation–response demonstrations.

Each pipeline stage records the information exposed to the decision maker, the structured decision produced by the teacher, and the observation returned after executing that decision. These records are then translated into action–observation sequences under the same action schema used by PersonaMap during training and inference. Training and live prompts can differ in how executor observations are serialized; the action protocol is shared.

A.2 Atomic Claim Construction

For each session, GPT-5.1 first summarizes the entities and attributes discussed in the interaction into a session-level coverage schema. This schema is used only as a checklist for claim coverage and does not determine the relations in the final graph. We then obtain three schema-constrained claim extractions using temperatures 0.1, 0.4, and 0.7. Using multiple generations improves coverage of information that may be expressed indirectly or distributed across the session.

Each candidate is normalized into the claim representation in Equation 2. The refinement procedure checks that every retained claim: (i) expresses a single atomic assertion, (ii) identifies its entity, attribute, and value, (iii) preserves temporal and contextual qualifiers when present, and (iv) is supported by a span of the source interaction. Equivalent candidates are merged, while incompatible assertions are retained separately so that they can later be connected by a conflict relation. The coverage schema is used for a final completeness check, after which the resulting claims form the teacher targets for the extraction stage.

To construct extraction demonstrations, the normalized claims are ordered according to their source positions. Each claim is converted into an $\mathrm{AddClaim}(c)$ action, followed by the executor observation containing the assigned claim identifier. The sequence terminates with $\mathrm{StopExtraction}$. Consequently, the learned policy is supervised not only on claim content but also on how many claims to extract and when to stop.

A.3 Candidate Retrieval and Relation Construction

For every new claim c_i, the structured pipeline retrieves a bounded set of historical claims from $\mathcal{G}_{t-1}$. Candidate retrieval combines canonical-key matching, entity–attribute lexical matching, and semantic retrieval. The ranked lists are fused using reciprocal-rank fusion, and the top-K_{ret} historical claims are retained as the candidate set $\mathcal{N}_i$; we use $K_{\mathrm{ret}} = 10$.

Table 3: **Conversion of offline pipeline decisions into PersonaMap tool demonstrations.** Only policy actions are prediction targets; deterministic executor operations are included as observations.

Stage	Pipeline decision	PersonaMap action
Extraction	Emit a normalized atomic claim	$\mathrm{AddClaim}(c)$
Extraction	Finish claim extraction	$\mathrm{StopExtraction}$
Graph	Retrieve historical candidates	$\mathrm{SearchSimilar}$
Graph	Select candidates for comparison	$\mathrm{SelectCandidates}(S_i)$
Graph	Assign a relation to the bound pair	$\mathrm{ClassifyRelation}(r)$
Navigation	Inspect a retrieved claim	$\mathrm{InspectClaim}$
Navigation	Traverse typed relations	$\mathrm{TraverseNeighborhood}$
Navigation	Expose unresolved alternatives	$\mathrm{OpenConflictGroup}$
Response	Produce a terminal response	Answer or $\mathrm{CannotAnswer}$

GPT-5.1 examines the new claim and its retrieved candidates to select the claims that require relational comparison. For every selected pair (c_i, c_j), it predicts one of

$$\{\textsc{SameClaim},\ \textsc{Complements},\ \textsc{ConflictsUnresolved},\ \textsc{DistinctByQualifier},\ \textsc{Unrelated}\}.$$

The first four decisions create the corresponding semantic edge. UNRELATED creates no edge. A CONFLICTSUNRESOLVED decision additionally creates or extends a conflict group containing the incompatible alternatives.

These pipeline operations are converted into the graph-construction protocol as follows. Candidate retrieval produces a $\mathrm{SearchSimilar}$ action and a returned candidate list. The teacher-selected subset becomes $\mathrm{SelectCandidates}(S_i)$. The executor then binds each selected historical claim to c_i and places the resulting pairs in a deterministic queue. For every exposed pair, the teacher relation becomes a $\mathrm{ClassifyRelation}(r)$ action. The policy therefore predicts candidate selection and relation semantics, whereas claim identifiers, pair binding, queue advancement, and graph mutation remain executor operations.

A.4 NAVIGATION AND RESPONSE TRACES

After graph construction is complete, the subsequent question q is used to retrieve a bounded set of seed claims from the constructed graph. Starting from these seeds, the teacher can inspect individual claims, traverse typed neighborhoods, and open unresolved conflict groups. The resulting trace terminates with either an Answer or $\mathrm{CannotAnswer}$ action.

The complete graph is not inserted into the teacher context. At each step, the teacher receives only the question, the currently available graph entries, and observations returned by preceding tool actions. Navigation demonstrations therefore follow the same bounded-access setting used by the learned policy rather than teaching responses from an oracle view of the graph.

A.5 CONVERSION TO TOOL DEMONSTRATIONS

Table 3 summarizes the conversion from structured pipeline decisions to PersonaMap actions. The converter preserves the order of executed decisions and their observations, producing stage-conditioned contexts under the same action schema used by the learned policy.

Each converted assistant action constitutes one supervised example. Its conditioning context contains the stage instruction, the visible input, and all preceding action–observation pairs. Only the generated action tokens receive the supervision mask. Executor-generated content, including assigned identifiers, retrieved candidate lists, bound relation endpoints, queue positions, committed graph updates, and completion signals, remains visible in the context but is not treated as a prediction target.

Table 4: **Configuration for tool-joint supervised initialization.**

Parameter	Setting
Backbone	Qwen2.5-7B-Instruct
Adapter initialization	Fresh LoRA; no prior adapter
LoRA rank / scaling / dropout	16 / 32 / 0.05
LoRA target modules	`q_proj`, `k_proj`, `v_proj`, `o_proj`, `gate_proj`, `up_proj`, `down_proj`
Learning rate	10^{-4}
Learning-rate schedule	Cosine; warmup ratio 0.03
Optimizer steps	1,500
Effective batch size	24
Gradient accumulation	8 steps
Maximum sequence length	16,384 tokens
GPU model	NVIDIA A100
Checkpoint used	Final step-1,500 checkpoint

A.6 Execution Validation

We replay the converted trajectories through the PersonaMap executor before using them for supervised initialization. A trajectory is retained only if its actions are valid under the corresponding stage protocol and reproduce the intended graph transitions. The validation checks claim insertion, candidate availability, selection validity, relation-pair binding, edge creation, conflict-group updates, queue completion, and terminal navigation behavior. This replay step ensures that the demonstrations describe executable policy behavior rather than unverified serialized teacher outputs.

The validated trajectories form the three stage-specific training pools. Their episode counts, sampling mixture, and training configuration are reported in Section B.1. The action-normalized supervision objective is given in Section 3.2.

B Training Details

B.1 SFT Training

The teacher pool contains 1,700 distinct sessions and 1,100 QA instances from SubtleMemory personas 0–6. The final training set contains 6,642 tool episodes: 1,614 for extraction, 3,985 for graph construction, and 1,043 for navigation and response. We sample the graph episodes from a pool of 23,371 while retaining the extraction and navigation–response episodes, so graph construction accounts for approximately 60% of the final training set. Personas 7–9 are reserved for evaluation and excluded from training. Each episode provides executable tool actions and observations; only policy-generated action tokens receive supervision.

We use the final checkpoint for subsequent experiments without selecting an intermediate checkpoint by validation loss. The action-normalized supervision objective is given in Equation 5.

B.2 GRPO and RICO Training

GRPO and RICO start from the same final step-1,500 tool-joint SFT checkpoint and use the same graph representation, tool interface, and terminal evidence-sufficiency reward. The policy backbone is Qwen2.5-7B-Instruct with its trained LoRA adapter; RL updates the adapter parameters. In the reported RICO setup, each update samples $K = 16$ trajectories per question for a group of eight questions, computes action-level credit from natural forks or selective counterfactual continuations, and performs one policy update. The counterfactual budget is $B_{\mathrm{CF}} = 1$ per rollout group; for a selected context, each of the factual and alternative actions receives $M = 2$ suffix samples. The clipped policy surrogate uses $\epsilon = 0.2$. The frozen evidence-sufficiency judge used by RICO is Qwen3.8-27B. Section B.3 details its decision rule and the credit-assignment protocol.

GRPO uses the terminal outcome to form trajectory-level group-relative credit; RICO replaces this with selective action-level credit. The implementation stores the resulting A_{RICO} values before policy optimization. In the full-trajectory training pathway, uncredited decisions are represented by zero advantages. Prompt and executor tokens do not receive policy gradients.

B.3 RICO Credit Assignment and Judge Protocol

Frozen evidence-sufficiency judgment. RICO uses the frozen Qwen3.8-27B judge and the `evidence_sufficiency_v1` system prompt. Its training reward input contains the question ID, question, reference answer, observed claim records, and their IDs. The training path does not pass benchmark reference facts. The offline preflight scorer uses the same system prompt with an alternative payload containing `reference_facts`; there the observed-claims field is named `seen_claims`. Reference facts aid interpretation and are not a required evidence checklist.

The frozen judge system prompt is reproduced verbatim below:

```
You are a strict evidence-sufficiency verifier for a memory agent.

Your task is NOT to answer the question.
Your task is to decide whether the memory claims actually observed by the
agent contain enough evidence to support the reference answer.

You are given:
1. a question,
2. the reference answer,
3. reference facts supplied by the benchmark,
4. memory claims actually observed by the agent.

Judge evidence sufficiency under these rules:

- Use the reference facts only to understand what evidence is relevant to
  the specific question. Do NOT require every reference fact to be present;
  some reference facts may belong to the broader instance but be irrelevant
  to this particular question.

- Information explicitly provided in the question does not need to be
  repeated in the memory claims.

- Multiple memory claims may jointly provide the required evidence.

- The observed claims must contain the answer-bearing information needed
  to support the reference answer, including relevant entity, value,
  relation, time, qualifier, comparison, or preference when required.

- Related or topically similar claims are not sufficient if the necessary
  answer-bearing information is absent.

- If the observed claims contradict the reference answer or leave an
  essential ambiguity unresolved, return INSUFFICIENT.

- Do NOT use outside knowledge, memorized world knowledge, or assumptions
  to fill missing evidence. Pretend you know nothing beyond the question
  and the observed memory claims.

- The fact that you are shown the reference answer does NOT make the
  evidence sufficient. The reference answer is only the target whose
  support must be checked.

Return SUFFICIENT only if a reader given the question and observed claims
could reasonably derive the reference answer from them.

Respond with a single JSON object only (no markdown fences):
{
  "verdict": "SUFFICIENT" | "INSUFFICIENT",
  "support_claim_ids": ["claim_id_1", "claim_id_2"],
  "missing_evidence": "short description or empty string"
}

Rules for the JSON:
- support_claim_ids must be a subset of the provided seen claim_ids (or [] if none).
- If verdict is INSUFFICIENT, briefly state what is missing in missing_evidence.
- If verdict is SUFFICIENT, missing_evidence should be "".
```

Training reward input. The RICO reward call pairs the frozen system prompt above with the following user-message form (payload values omitted):

```
Return ONLY one JSON object with key verdict = SUFFICIENT or INSUFFICIENT.
Do not answer the question. Do not rewrite the reference answer.
Input payload:
{
  "question_id": "...",
  "question": "...",
  "reference_answer": "...",
  "observed_claims": [...],
  "seen_claim_ids": [...]
}
```

The judge returns JSON fields `verdict` (`SUFFICIENT` or `INSUFFICIENT`), `support_claim_ids`, and `missing_evidence`. Support claim IDs are restricted to actually observed claim IDs. Only a parsed `SUFFICIENT` yields evidence reward one; `INSUFFICIENT` and invalid or unparseable verdicts yield zero. A parsing failure is additionally marked `parse_ok=false` and is excluded from the valid judgment cache. The judge checks whether the observed evidence supports the reference answer; the policy's generated answer text is not part of the judgment.

Rollout eligibility and initial state. For each question, the rollout pipeline starts from the preceding session graph $\mathcal{G}_{t-1}$, executes the new session's memory actions, and then evaluates graph navigation. A trajectory is usable for credit assignment only if its reward is valid, its transcript parses successfully, and its terminal reward belongs to $\{0, 1\}$. A decision is trainable only if its response token IDs were recorded and the action was generated by the policy rather than forced by the executor. These checks precede natural-fork and counterfactual credit assignment.

Terminal response and reward. The CANNOTANSWER action is a valid terminal response in the tool interface and can appear in supervised demonstrations. In RICO, however, it does not count as a rewarded completion: it sets $E_i = 0$ in Equation 6, regardless of the evidence judge's output. A trajectory receives positive reward only after a valid ANSWER and a positive evidence-sufficiency judgment. The judge evaluates evidence with respect to the reference answer, not the correctness of the generated answer text.

Natural-fork matching and action equivalence. The matcher compares decisions under a shared preceding action prefix. Its fork key incorporates a fingerprint of the executor state and a fingerprint of the policy-visible context. Actions at the matched decision are compared using operation-specific normalized signatures: claim fields, relation labels, and selected candidate sets are normalized according to their respective action types. An action without a recognized operation has the signature `invalid`. The matcher also supports an optional semantic signature for $\mathrm{AddClaim}$; whether this optional matching mode is enabled is determined by the reported run configuration. Natural credit additionally requires at least two supported actions with unequal empirical mean terminal rewards.

Counterfactual replay and alternatives. When natural credit is unavailable for a rollout group, the counterfactual selector prioritizes unresolved decisions, including those at observed natural forks. It selects the original factual action and a distinct valid alternative; semantically equivalent paraphrases are rejected as alternatives. The counterfactual evaluator reconstructs the state by starting from $\mathcal{G}_{t-1}$ and replaying the recorded prefix followed by either the factual or the alternative action. Thus the fork state comes from execution replay, rather than an independently stored graph snapshot. Invalid parses, executor-forced actions, and empty responses cannot receive training credit. Only the original factual action is assigned the resulting A_{RICO}; forced alternatives and sampled continuations are evaluated but not directly optimized.

Optimization mask. The training pipeline consumes the precomputed A_{RICO} values. Policy updates are applied to generated response tokens; prompt tokens and executor observations are excluded. Within a response, the action mask includes the entire generated response rather than individual JSON fields. In full-trajectory training records, actions without credit carry zero advantage and therefore contribute no direct policy-gradient signal.

C STRUCTURE-AWARE FORGETTING: IMPLEMENTATION AND EVALUATION

Pruning configuration. We evaluate `personal_forget_v1` on SUBTLEMEMORY personal claim graphs after the preceding graph construction stage. The reported strong configuration uses $\tau_{\text{stale}} = 20$, $\tau_{\text{sparse}} = 8$, $d_{\max} = 1$, and $w = 8$, where age and the recent frontier are measured in sessions. An old claim with zero semantic degree is eligible for deletion; a sufficiently old claim of degree at most one is eligible only if graph search cannot connect it to the recent frontier. Claims in unresolved conflict groups are protected from deletion. The more conservative default configuration $(40, 20, 0, 15)$ is not used for the reported strong results.

Connectivity and candidate selection. The implementation forms an undirected adjacency graph among existing claim nodes: for each edge whose endpoints are both claims, it adds each endpoint to the other's neighbor set, except for the structural relations `has_claim` and `mention`. Degree counts distinct neighbors, and graph search checks whether a claim can reach any claim from the most recent w sessions by a path of arbitrary length. Any nonstructural claim-to-claim relation present in the graph contributes to degree and reachability. Conflict-group membership, adjacency, and recent claims are computed from the original graph. The implementation gathers all deletion candidates in one pass and applies pruning once, without recomputing degree or reachability after individual deletions. Claims lacking a session index are skipped. An optional minimum retained-claim fraction can further restrict deletion; its configured default is zero.

Graph updates. Pruning removes the selected claim nodes and their incident edges, cleans up orphan entities, and reconstructs conflict groups in the resulting graph. The transformation is evaluated on final graphs and ask-time graph snapshots; it does not overwrite the canonical `final_graph.json` or modify model weights. Graph reachability uses the undirected claim-relation graph and the recent-session frontier specified in Section 3.4.

Retention proxy and evaluation scope. The main evaluation uses personas 0–6. Among the 340 FULL questions with fully mapped gold claims, 220 have at least one relevant semantic relation and are eligible for the relational-evidence retention measure. At a matched budget of 50% fewer stored claims, approximately 80% of these questions retain all mapped gold claims and relevant relations. Each pruning method is evaluated at the same claim-compression budget. The thresholds above describe the forgetting operator; the 50% result comes from the matched-budget comparison. This graph-level retention proxy does not measure answer accuracy after re-running graph navigation and the answer judge on the pruned graph; that end-to-end evaluation remains pending.