STRATA: SPATIOTEMPORAL ALIGNMENT OF AGENTIC MEMORY FOR LONG-HORIZON PREFERENCE TRACKING

Anonymous authors
Paper under double-blind review

ABSTRACT

Large language model assistants increasingly maintain relationships with users across months or years, yet most memory systems still treat personalization as a retrieval problem. This framing fails when a user's preferences change over time, and the conversation turns most semantically similar to a query are no longer the correct value. We argue long-term memory should instead be framed as a *state inference* problem by computing the value of an attribute at a given time from its past states. We present STRATA, a memory architecture organized into three complementary layers (Spatial, Temporal, and Spatiotemporal) that create a single traversable graph via casual and relational edges. Preferences are stored as evolutionary chains, mapping the trajectory of its value over time. At inference time, an LLM agent traverses the memory graph with tool calls both within and across layers to intelligently retrieve relevant context. On PERSONAMEM, STRATA attains a new SOTA of 61.6% accuracy compared to six baselines, and a second-best on LONGMEMEVAL of 86.4% overall, driven by a 6.8-point margin on multi-session recall questions. Tool call analysis reflects the agent exhibiting emergent routing based on dataset and question type. Controlled ablations show the memory representation is crucial for preference tracking questions, and that agentic, query-conditioned traversal at inference time drives meaningful gains.

1 INTRODUCTION

Large language model (LLM) assistants are moving beyond single-turn interactions toward persistent relationships with users. Rather than answering isolated questions, these systems must remember the user's preferences over the span of days, months, or years. To support long-horizon personalization for conversational agents, recent work has explored reflective memory (Park et al., 2023; Shinn et al., 2023), reinforcement learning (RL) (Yan et al., 2025; Wang et al., 2025), and graph-based memory structures (Rasmussen et al., 2025; Chhikara et al., 2025). While these approaches differ in their architectures, they share a common assumption: personalization is treated as a retrieval problem, where success depends on recovering relevant information from past interactions.

This retrieval-centric framework works well for factual recall tasks. However, true personalization often centers on non-static values. In the real world, user preferences, interests, and goals evolve over time. A user might change jobs or move to a different city, creating new preferences or revising old ones. In these settings, the relevant information about a user's preferences is rarely contained in a single interaction. Instead, the agent must synthesize evidence across multiple sessions potentially months apart. This is especially difficult for memory systems that rely solely on semantic retrieval. Crucially, an interaction that most closely matches a query semantically may describe a preference that is outdated (Figure 1). Without some method of temporal organization or intelligent reasoning, a set of semantically similar turns on its own does not reveal what the user's latest preferences are.

We therefore argue that long-term memory should be formulated as a *state inference* problem rather than a *retrieval* problem. To address this gap, we introduce STRATA (Structured Temporal Retrieval and Attribute Tracking Agent). Instead of representing memories as independent observations, STRATA models user attributes as evolving states. Given a query, an LLM-based memory agent actively reconstructs the user's preferences by traversing temporal trajectories and typed

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.

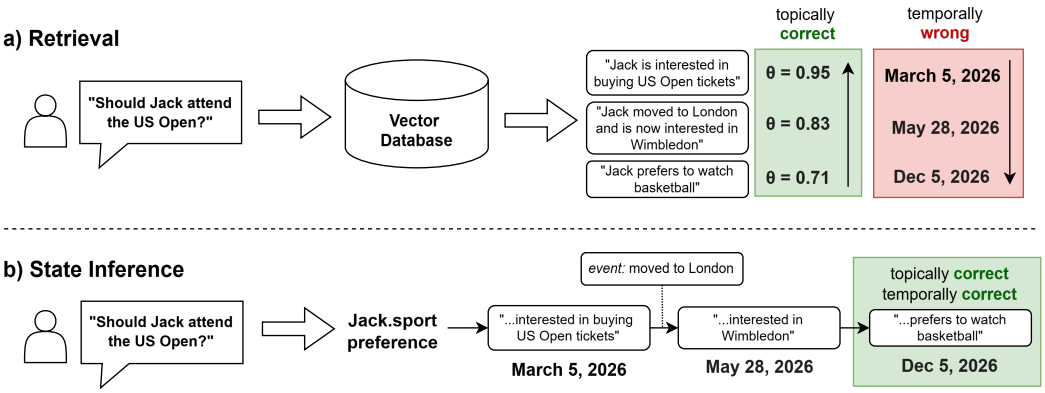


Figure 1: Retrieval is the wrong primitive for evolving preferences. A passage with the highest semantic similarity may be topically correct, but temporally wrong. State inference instead chains historical evidence together to provide the most up-to-date snapshot of a user's preferences.

edges, gathering evidence across several memory layers before producing an answer. This reasoning is done at inference time, which is notably different from previous methods that place reasoning in pre-processing or ingestion phases. Across two well-known benchmarks, LongMemEval and PersonaMem, STRATA consistently improves performance on more complex tasks requiring belief revision, temporal reasoning, and preference tracking. Our contributions are summarized below:

1. **STRATA Architecture.** We introduce STRATA, a novel memory architecture that represents user attributes as temporal evolution chains. Unlike prior graph-based memory systems with edges organized by semantic similarity, STRATA's graph is organized by relational links and causal triggers, enabling full provenance for preference evolution.
2. **State Inference Approach.** We reformulate long-horizon personalization as a problem of *state inference*. Instead of retrieving historical evidence in isolation, we model how user attributes evolve by applying an agent that reasons over past trajectories at inference time.
3. **Empirical Validation.** STRATA scores a new SOTA of 61.63% on PersonaMem, and a second-best of 86.40% on LongMemEval, with focused improvements on difficult categories like Multi-Session Synthesis (LongMemEval) and Revisit Reasons (PersonaMem).

2 Related Work

2.1 Structured Memory Representations

Recent memory systems have substantially improved long-horizon personalization through advances in the way memory is stored. Zep organizes memory as a bi-temporal knowledge graph (Rasmussen et al., 2025). A-MEM builds a self-organizing Zettelkasten-style note network with LLM-generated links (Xu et al., 2025), and HippoRAG runs Personalized PageRank over an extracted knowledge graph to approximate associative retrieval (Gutiérrez et al., 2024). More recently, Mnemis organizes memory into two graphs: a base graph of extracted entities and relations for similarity-based retrieval, and a hierarchical graph built by repeatedly abstracting entities into higher-level categories (Tang et al., 2026). Collectively, these systems show that structured memory representations outperform reading from raw conversational history or vanilla retrieval-augmented generation (RAG).

Despite their differences, existing approaches retain a retrieval-centric view of memory. A set of past turns are retrieved according to semantic relevance, but they do not explicitly model *how* or *why* user attributes evolve. A-MEM, for example, links notes through semantic association, but stores no temporal relationships between related observations. Consequently, these systems struggle to model preference changes over long time horizons. STRATA instead stores memory as evolution chains that record each state transition and its cause, allowing an attribute's history to be read directly at inference time rather than inferred from a bundle of retrieved, undifferentiated observations.

2.2 Agentic Memory Management

Agent-based systems exist in which LLMs actively participate in memory construction, retrieval, consolidation, and forgetting. MemGPT/Letta treats the context window as an OS-managed resource, with the model itself issuing function calls to page facts between main context and external storage (Packer et al., 2023). Memory-R1 jointly optimizes a memory-management agent with an answer agent via PPO/GRPO (Yan et al., 2025), and MEM1 learns to consolidate internal state turn-by-turn under an RL objective (Zhou et al., 2025). In each case, intelligence is applied at *write* time (i.e. deciding what memories to keep, merge, or discard) while the resulting memory store read at inference time remains as a fixed structure accessed via deterministic retrieval methods (e.g. RAG).

However, only optimizing write-time operations limits the system's ability to respond to complex queries. This is particularly costly in long-horizon settings, where a single query can require synthesizing information about multiple people, events, or preferences across hundreds of stored sessions. STRATA instead uses cheap LLMs for memory construction and places intelligence (an agent) at *inference* time. Using a novel retrieval mechanism, the agent first anchors on specific entities/preferences, then traverses typed edges to related nodes within a single unified graph to reconstruct how preferences change over time. This guided expansion is what lets the agent accurately retrieve and interpret the cross-session evidence needed to answer multi-hop queries.

2.3 Long-Horizon Personalization

Personalization across long horizons is difficult because a belief formed early in conversation history must be actively revised as the user's circumstances change. Prior work shows that models often anchor on stale beliefs even when a contradicting statement is placed directly in context (Longpre et al., 2021; Xie et al., 2023). Personalization benchmarks reflect this limitation, with models scoring lower on questions related to dynamic preferences versus static ones (Jiang et al., 2025b). While these benchmarks confirm the presence of stale-preference anchoring, none offer an architecture for a deployed agent to overcome it. STRATA fills that gap. Rather than optimizing for better information retrieval, STRATA optimizes for better reasoning. Notably, STRATA's graph is temporally aligned and previous preference values are appended rather than overwritten, providing explicit signals designed to counteract persistent belief-update failures while maintaining robust provenance.

3 Methodology

3.1 STRATA Overview

STRATA converts a user's conversation history into a compact, time-indexed memory representation through three phases (Figure 2). **(1) Extraction** reads each past session once and extracts structured memory items: Entities, Facts, Events, and Preferences. **(2) Consolidation** assembles extracted items into a three-layer memory graph, with cross-layer edges linking related attributes. **(3) Retrieval** answers a query with an agent that reads the memory graph through custom tools, ultimately generating a response with an evidence brief to support its output.

3.2 Extraction

Four distinct attributes are extracted from raw turns via a cheap LLM call (gpt-5.4-mini). **Entities** are the durable subjects a user's life centers on: people, places, organizations, goals, objects, and relationships. **Facts** are time-varying propositions about an Entity. They are stored as *(entity, attribute, value)* triples, and are bounded by a validity interval of timestamps. For example, the Fact *"Jack's city of residence is Denver"* might be stored as *(Jack, home_city, Denver*). **Events** are dated occurrences (i.e. a move, a promotion, a diagnosis), each with a timestamp and the Entities involved. **Preferences** are the user's subjective attitudes and choices, such as their favorite movie(s), how often they exercise per week, or dietary restrictions. Crucially, Facts and Preferences are stored not as a single value but as an evolution chain. For example, when a Fact changes (i.e. the user moves from Denver to Seattle), the old Fact node is appended to the new Fact node through a typed edge. Entities and Events remain as single records. Table 4 (Appendix A.1) summarizes how each attribute is represented. Each item also carries pointers back to the raw turns it was mined from for ground-truth provenance.

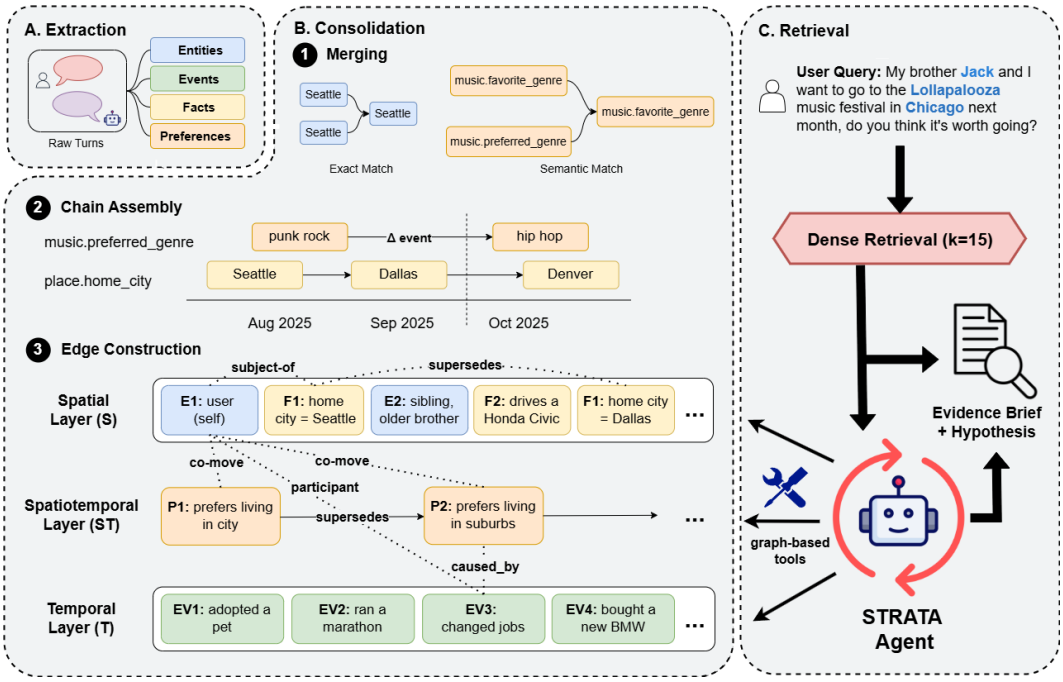


Figure 2: STRATA architecture. **(A) Extraction** reads raw turns and mines four nodes: Entities, Events, Facts, and Preferences. **(B) Consolidation** builds the unified graph in three steps: *(1) Merging* canonicalizes duplicate Entities by exact match and Preferences via semantic match; *(2) Chain Assembly* orders Facts/Preferences into time-stamped evolution chains; *(3) Edge Construction* generates cross-layer edges. **(C) Retrieval** answers a user query by combining *(1)* an evidence brief assembled by the STRATA agent, and *(2)* dense-retrieved turns as supplementary grounding.

3.3 Consolidation

Extracted nodes are consolidated into a three-layer memory system (Spatial, Temporal, and Spatiotemporal layers), with each layer named for the types of questions it resolves. First, Entities and Preferences are de-duplicated, and Fact and Preference chains are assembled. Then, cross-layer edges are drawn between nodes to turn the memory store into a single traversable graph.

Entity Resolution. Duplicate Entities are merged into a single Entity by exact keyword match. When a new Entity is detected, matching occurs against every Entity accumulated so far.

Preference Resolution. Preferences encoded by (*topic, attribute, value*) tuples are generated with an open vocabulary. To prevent over-fragmentation, we merge preference tuples with similar topics/attributes. Take an example Preference of (*music, preferred_genre, "punk rock"*). Only attributes under the same *topic* can merge. Within a topic (in this case, *music*), each (*topic, attribute, value*) tuple is embedded, and linked to its nearest neighbor, creating candidate clusters of music-related Preferences. For example, a mention filed under (*music, favorite_genre, "punk rock"*) might be mapped into the same cluster as *(music, preferred_genre, "punk rock"*). A cheap LLM call decides if the proposed cluster merge should be accepted, and what the resulting value(s) should be.

Evolution Chain Assembly. After deduplication, the two evolving attribute types (Facts and Preferences) are grouped into evolution chains by their key. A Fact keys on its *(Entity, attribute)* pair, while a Preference keys on its *(topic, attribute)* pair. Within each chain, observations are ordered by timestamps derived from the conversation history.

Memory Layers. Following chain assembly, all Entity, Preference, Event, and Fact nodes are stored in one of three layers. The **Spatial (S)** layer answers *"what is true about X?"* and holds both Entities and the Facts that describe them. The **Temporal (T)** layer answers *"what happened, and when?"* and holds Events sorted onto a timeline via temporal indices. The **Spatiotemporal (ST)** layer—the

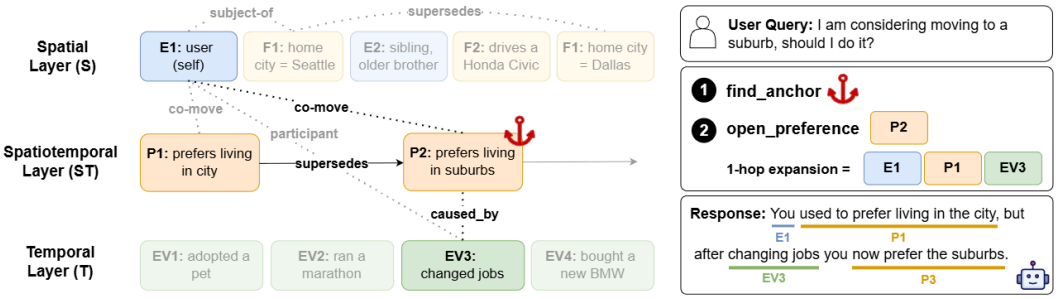


Figure 3: Example agentic trace. STRATA anchors on the relevant preference (P2) and opens it, surfacing its one-hop neighborhood in a single call, including the entity it's about (E1), the old preference (P1), and the trigger event (EV3). The resulting response is both temporally correct and rich in context, providing meaningful evidence that can't be obtained with flat semantic retrieval.

primary axis STRATA is built on—answers *"why a value changed, and who was affected?"*. It stores the evolution chains that represent an auditable history of a Preference's trajectory. Preference nodes are connected to Events/Entities/Facts via cross-layer edges (Figure 2).

Cross-Layer Edges. With all nodes in place, a deterministic step draws six types of cross-layer edges. `subject-of` links each Fact to the Entity it is about, while `participant` links each Event to every Entity in its participant list. The densest edge type, `co-move`, is generated by drawing an undirected link between any two nodes mined from the same raw turn but stored in different layers (e.g. a Preference in ST and a Fact in S, both extracted from turn #34). The `caused-by` edge maps a Preference to its trigger Event, and its construction is unique. During extraction, Preference nodes are optionally tagged with a free-text phrase describing an event, which we embed and link to the nearest prior-dated Event node (cosine $\geq \tau$=0.15). `supersedes` edges order the current/previous values within a Fact or Preference evolution chain, and `provenance` edges keeps the ids of the raw turns a node was mined from. Together these edges transform the S, T, and ST layers into a rich, traversable graph, letting the agent perform complex actions (like identifying the Entities tied to a Preference, or the Event that triggered a Preference change) in a single hop.

3.4 Retrieval

Dense Pre-retrieval. The user's query is first used to obtain the top-15 raw turns via dense similarity.

ReAct-loop Agent. The pre-retrieved turns are provided as seed context to a ReAct-based agent—the central component of our system. The agent acts as an orchestrator, navigating the memory graph through tool calls. `find_anchor` is called to locate a relevant node via dense semantic search; this search can optionally be limited to a specific layer (i.e. `find_anchor(spatial)` only searches Entity/Fact nodes). After receiving a set of nodes, the agent can choose one to read with node-specific tools: `open_preference`, `open_event`, `open_entity`, or `open_fact`.

Each `open` returns the node (for a Preference or Fact, its entire evolution chain) with a one-hop, cross-layer snapshot. This anchor-then-hop pattern is deliberately layer-conditioned. Rather than embedding multi-entity queries into a single vector to use for similarity search against a memory store, the agent first issues `find_anchor` calls to limit its scope to a specific layer/attribute it judges relevant, then reaches connected nodes that provide more topical context about said attribute (Figure 3). This ensures the agent can sufficiently explore each distinct person, object, event, or preference within a query (i.e. through multiple anchor-then-hop expansions). The tool-calling loop continues until the agent determines it has enough evidence to answer the query.

Answer Composition. The final output is an evidence brief containing the agent's reasoning trajectory and accessed nodes. The brief is combined with the pre-retrieved turns, providing grounding for open-ended recall tasks (i.e. retrieving exact names, numbers, or quotes).

4 EXPERIMENTAL SETUP

Datasets. We evaluate STRATA on two long-horizon personalization benchmarks. **PERSONAMEM** (Jiang et al., 2025a) is built on ∼180 synthetic personas and tests whether an agent tracks their evolving preferences. We evaluate at the 32K context-length setting. **LONGMEMEVAL-S** (Wu et al., 2024) is a free-text question-answering (QA) benchmark scored by an LLM judge, spanning single- and multi-session recall, temporal reasoning, and knowledge updates. We include it as a control to check that preference tracking does not come at the cost of ordinary long-term recall.

Baselines. On PERSONAMEM we compare STRATA against three established memory systems: LangMem (Langchain, 2025), Mem0 (Chhikara et al., 2025), and A-Mem (Xu et al., 2025). On LONGMEMEVAL we include the Full Context, RAG, Mem0, Zep, Nemori (Ma et al., 2025), EMem-G (Zhou & Han, 2025), and Mnemis baselines as reported by Tang et al. (2026). All methods are evaluated under the same backbone (`gpt-4o-mini`) so that accuracy gains are solely attributable to the memory architecture itself. For strict comparability, all embeddings use the same `all-MiniLM-L6-v2` model, pre-retrieval top-k is fixed, and temperature is 0. Due to cost and time limitations, ablations were run on a random 10% subset of PERSONAMEM, with a proportional number of questions per category in our subset compared to the full benchmark.

5 RESULTS

5.1 PERSONAMEM

Table 1: PERSONAMEM-32K accuracy by question type. Per-column best in **bold**.

Backbone	Method	Memory Recall			Reasoning & Adaptation			Temporal	
		Recall Facts *n=129*	Suggest Ideas *n=93*	Revisit Reasons *n=99*	Track Evol. *n=139*	Aligned Rec. *n=55*	General. Scen. *n=57*	Latest Pref. *n=17*	Avg. *n=589*
GPT-4o-mini	Full Context	45.74	11.83	82.83	53.96	60.00	43.86	**76.47**	50.59
	RAG	62.02	27.96	78.79	53.96	63.64	66.67	70.59	58.40
	LangMem	43.41	23.66	77.78	49.64	60.00	**68.42**	52.94	51.78
	Mem0	49.61	22.58	79.80	47.48	56.36	56.14	58.82	51.44
	A-Mem	**63.57**	**38.71**	78.79	53.24	65.45	**68.42**	64.71	60.44
	STRATA (ours)	57.36	36.56	**87.88**	**56.12**	**74.55**	**68.42**	58.82	**61.63**

STRATA attains the best overall accuracy (61.6%), a 1.2-point margin over the next-best baseline, A-Mem, at 60.4% (Table 1). Notably, STRATA achieves the best performance in the **Track Evol.** category, which is composed of questions asking for a preference's evolution trajectory, scoring 56.1% on the benchmark's largest category (n=139 of 589 items). STRATA also outperforms on **Revisit Reasons** (87.9%), the category that calls for explaining why a preference changed, and on **Aligned Rec.** (74.6%), which asks whether a candidate recommendation is consistent with the user's current interests. It ties for best on **General. Scen.** (68.4%), an open-ended domain evaluating whether a model can transfer stored preferences to a new scenario not covered in the conversation's history. These four categories are common in that they require synthesizing evidence across multiple turns/sessions as well as past preference states, which STRATA is uniquely positioned to excel at.

The categories in which STRATA underperforms are all recall-related. A-Mem excels on **Recall Facts** (63.6% vs. 57.4%), and the Full Context baseline wins **Latest Pref.** (76.5% vs. 58.8%). Per-category accuracies therefore suggest that STRATA's advantage is concentrated in reasoning over resolved preference states (i.e. explaining why a preference changed and how it affects the user) rather than in verbatim recall of facts or utterances.

5.2 LONGMEMEVAL

Table 2 provides results for LONGMEMEVAL, which tests if STRATA can serve as a competitive general-purpose long-term memory architecture that is not necessarily focused on personalization. STRATA reaches 86.4%, second only to Mnemis (87.2%) among compared systems and ahead of

Table 2: LONGMEMEVAL-S accuracy categorized by question type. Underlined values are the second-best per column; bold values are the best.

Backbone	Methods	SSU *n=70*	MS *n=133*	SSP *n=30*	TR *n=133*	KU *n=78*	SSA *n=56*	Overall *n=500*
GPT-4o-mini	Full Context†	78.6	38.3	6.7	42.1	78.2	89.3	55.0
	RAG†	88.6	47.4	<u>70.0</u>	63.2	70.5	91.1	67.2
	Mem0†	91.4	66.2	34.0	63.9	74.4	<u>96.4</u>	71.1
	Zep†	<u>92.9</u>	47.4	53.3	54.1	74.4	75.0	63.2
	Nemori†	88.6	51.1	46.7	61.7	61.5	83.9	64.2
	EMem-G†	87.0	73.6	32.2	74.8	**94.4**	87.5	77.9
	Mnemis†	**97.1**	<u>76.7</u>	**90.0**	<u>83.5</u>	<u>92.3</u>	**100.0**	**87.2**
	STRATA (ours)	88.6	**83.5**	63.3	**85.7**	89.7	**100.0**	<u>86.4</u>

† Reported by Tang et al. (2026)

every other baseline by at least 8.5 points. STRATA's advantage over Mnemis is concentrated in two categories. The larger is **multi-session (MS)** recall, where STRATA reaches 83.5% against Mnemis's 76.7%, a 6.8-point gap on the benchmark's joint-largest category (n=133). Multi-session questions require synthesizing information scattered across many conversations, which is the closest analogue LONGMEMEVAL offers to the kind of cross-session integration our graph is built to support. The second is **temporal reasoning (TR)**, where STRATA leads at 85.7% against Mnemis's 83.5% on an equally large slice (n=133); this is explained by STRATA's temporal organization. STRATA also ties Mnemis for best on the **single-session-assistant (SSA)** category (100.0%).

On **single-session-preference (SSP)**, STRATA's 63.3% trails both Mnemis (90.0%) and RAG (70.0%). LONGMEMEVAL's SSP questions ask whether a preference is followed within a single session, not if a preference changed or what caused it. Consequently, STRATA's rich cross-layer links are not suited for this task, as there is no temporal reasoning or preference trajectory to trace. STRATA is also behind in **knowledge update (KU)** (89.7% vs EMem-G's 94.4%) and **single-session-user (SSU)** (88.6% vs Mnemis's 97.1%). Both categories rely on reading historical values, which is the same recall-related gap the PERSONAMEM results allude to.

Read together, Tables 1 and 2 support STRATA's effective reframing of memory as a state inference problem rather than retrieval. On PERSONAMEM, where evaluation is built around preference modeling, STRATA's advantage concentrates on categories that require recovering a full evolution trajectory while maintaining the best overall performance among several strong baselines. On LONGMEMEVAL, where preference change is largely absent from evaluation, STRATA remains incredibly competitive—only 0.8 points behind Mnemis—with the advantage concentrated on cross-session synthesis and temporal reasoning, the two categories where an answer must be intelligently assembled rather than located. The two results are complementary evidence for a central claim: a graph organized on temporal/causal axes that encodes *when* and *why* a preference changed is more useful than memory organized around semantic retrieval without temporal/causal structure.

5.3 TOOL CALL ANALYSIS

Emergent tool routing. A per-dataset, per-category breakdown of the agent's tool calls (Figure 4) shows the STRATA agent routing itself efficiently. STRATA opens the spatiotemporal preference tier 2.28 times/item on PERSONAMEM but only 0.10 on LONGMEMEVAL (a factor of $\approx 23\times$), shifting to Events and Facts on the latter. Furthermore, it re-weights tiers by question type even within a single dataset, stressing `open_preference` tool calls on categories requiring preference chain evolution (Track Preference Evolution in Fig. 4a), while firing many `open_event` calls for questions demanding temporal recall (Temporal Reasoning in Fig. 4b).

Within-dataset tool call use varies by question semantics. On PERSONAMEM (Fig. 4a), `suggest_new_ideas` is especially telling: it pairs the heaviest preference usage (3.53 `open_preference`/item) with substantial retrieval effort (4.24 `find_anchor`/item). This is the one category whose answers do not appear in the conversation history. These questions are open-ended and require cross-session synthesis to make a personalized inference, so the agent

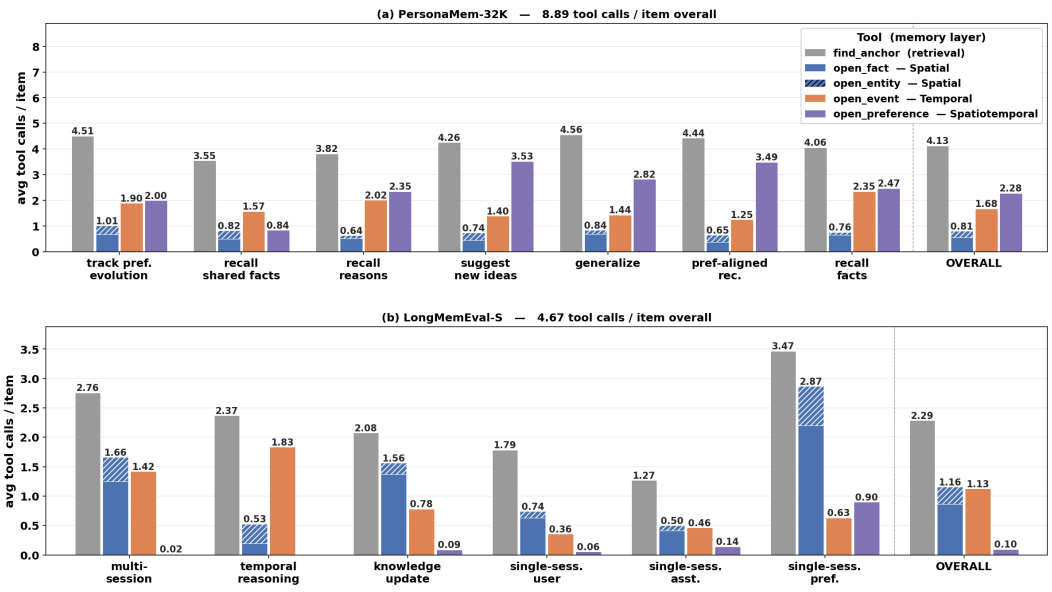


Figure 4: **Tool-call usage** by question category on (a) PERSONAMEM and (b) LONGMEMEVAL. Within each category, one bar per tool gives average calls per item, colored by the memory layer each tool accesses: *Spatial* (blue) is a single bar stacking `open_fact` (solid) and `open_entity` (hatched, same color to mark the shared spatial layer); *Temporal* (orange) is `open_event`; *Spatiotemporal* (purple) is `open_preference`; the neutral gray bar is `find_anchor`.

cannot simply look up a specific Entity or Fact. Instead, it must cast a wide anchor net, and leans hardest on the preference tier. The lone factual recall category is the visible exception: `recall_user_shared_facts` has the lightest `open_preference` usage of any PERSONAMEM category (0.84/item against a 2.27 dataset mean), leaning on Facts instead. LONGMEMEVAL (Fig. 4b) shows the same sensitivity across question types (e.g. `temporal-reasoning` is overwhelmingly Temporal with 1.83 `open_event` calls/item against just 0.20 `open_fact`).

The agent also spends more effort on the preference-heavy benchmark overall, with 8.89 tool calls/item on PERSONAMEM against 4.67 on LONGMEMEVAL. The layer traversals it performs are overwhelmingly cross-layer in both settings (97% and 94% of all edges traversed, respectively), confirming that the agent is genuinely hopping between layers via cross-layer edges. This points to the tool set and the specific Spatial/Temporal/Spatiotemporal memory representation of STRATA as being uniquely conducive to allowing for selective and efficient retrieval of information from relevant axes (temporal, factual, and causal). The results affirm STRATA's generalizability as a robust, long-term memory architecture that can potentially be used for tasks beyond preference-tracking.

5.4 ABLATION

Every tier carries weight. Table 3 isolates each component of the adopted graph system on PERSONAMEM. Removing any single layer degrades accuracy below the full system's 61.0%, and severing the Spatiotemporal preference layer is by far the most damaging (-16.9 points to 44.1% overall accuracy, and -40.0 points to 20% in Track Evol.). This can largely be explained by two reasons: a) the Spatiotemporal layer encodes Preference chains, which are central to a preference-related benchmark, and b) the Spatiotemporal layer serves as the main connective tissue between the Spatial and Temporal layers, with the most edges being formed between S and ST, and T and ST. When only one layer and two layers are reachable, the accuracy never exceeds 54.2%, showing that the tiers are complementary to one another and all provide unique value.

Mechanism controls. Disabling one-hop expansion falls to 52.5% (-8.5), with the Track Evol category lowering significantly to 26.7% (-33.3 points). This deficit is consistent with relevant

Table 3: Controlled ablation of STRATA on PERSONAMEM-32K. **Overall** is aggregate accuracy; **Track Evol.** isolates the benchmark's largest and most thesis-relevant category.

Condition	Tiers reachable	Overall	Track Evol.
Full system (S + T + ST)	S + T + ST	**61.0**	**60.0**
Drop-one (two layers reachable)			
− Temporal (events)	S + ST	54.2	46.7
− Spatial (facts / entities)	T + ST	50.8	46.7
− Spatiotemporal (preference)	S + T	44.1	20.0
Isolated (one later reachable)			
Spatial only	S	54.2	46.7
Temporal only	T	50.8	33.3
Spatiotemporal only	ST	52.5	20.0
Mechanism controls			
− Edges (opens node, but no one-hop)	S + T + ST	52.5	26.7
− Seed (pre-retrieval seed off)	S + T + ST	47.5	40.0
Fixed policy (non-agentic, full graph)	S + T + ST	45.8	40.0

evolutionary context being surfaced by cross-layer expansion. Removing the dense pre-retrieval seed costs 13.5 points (47.5%), marking it as a load-bearing step in steering the agent's orchestration.

Is the agent earning its keep? The "Fixed policy" row is a direct test of whether agentic tool choice adds value beyond the graph representation itself. It holds every variable constant and replaces *only* the LLM ReAct orchestrator with a deterministic retrieve-and-expand traversal. We employ a fixed BFS-style algorithm where we retrieve the top k=6 anchor nodes across all layers, open each through the corresponding `open` tool call (accessing the node, evolution chain if applicable, and one-hop cross-layer neighbors), enqueue the top 3 neighbors per node, and stop after 10 opens.

This fixed policy reaches only 45.8%, 15.2 points below the agent's 61.0%. This deficit is likely explained by the fact that the fixed policy commits to a single set of anchor nodes and can only expand outward, whereas the agent may abandon an unpromising node mid-trajectory and issue a fresh `find_anchor` call. This capacity to *re-anchor* matters when relevant evidence sits several hops away, which is precisely in LONGMEMEVAL's multi-session category where STRATA's margin over prior work is largest (Table 2). Consistent with this, the agent calls `find_anchor` multiple times per item on average (4.13/item on PERSONAMEM, 2.29/item on LONGMEMEVAL), proving it routinely redirects its search rather than committing to one initial node neighborhood. Notably, the deterministic algorithm opens *more* nodes per item than the agent (≈7–8 in deterministic vs. the agent's ≈3.1). Thus, the issue is not an evidence volume problem. Rather, the agent's ability to intelligently *choose* which nodes to anchor on and what edges to follow is a meaningful contribution.

Orchestrator ablation. To further isolate the effect of STRATA's graph representation, we tested the effect of replacing the orchestrator model. Substituting `claude-opus-4.6` for `gpt-4.1-mini` lowered accuracy by 1.70% (59.93% vs 61.63%) but cut cost and latency by more than $10\times$, and still beat Full Context, Mem0, and LangMem by over 8 points (Appendix A.3). Thus, STRATA's advantage can be attributed to its unique memory structure, not orchestrator choice.

6 CONCLUSION

We reframe long-term memory from a static retrieval problem to a dynamic state inference problem. STRATA creates a temporal graph linked by typed causal edges that an agent reasons over at inference time. Evaluations on two long-horizon benchmarks shows that STRATA outperforms on most baselines, with focused gains on difficult categories requiring full preference trajectories and multi-session synthesis. STRATA's novel combination of a layered graph combined with agentic navigation offers a uniquely efficient, accurate, and robust system for long-term preference tracking without sacrificing factual recall. Overall, the results support that future conversational agents may benefit by shifting from retrieval-based optimization to state-inference based reasoning over memory representations structured around causal and temporal axes.

AI USE STATEMENT

In this work, we did not use generative AI tools for any tasks that require disclosure. We have not used generative AI tools for proposing/refining hypotheses, providing feedback on experiments, implementing methods, cleaning datasets, or interpreting results. The use of AI for generating synthetic datasets, developing theoretical models, and formulating mathematical claims/proofs are not applicable to this work. Additionally, we have used generative AI tools for summarizing/analyzing existing literature and editing sections of this paper to improve grammar and readability. We have reviewed all AI-assisted work. In particular, all claims made in AI-generated summaries for synthesizing past literature were corroborated and cross-referenced with the original article(s), and no information was cited in this paper without consulting the primary source. For AI-generated grammar/readability fixes, all authors reviewed the suggestions and selectively applied fixes based on correctness. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REFERENCES

Prateek Chhikara, Dev Khant, Saket Aryan, Taranjeet Singh, and Deshraj Yadav. Mem0: Building Production-Ready AI Agents with Scalable Long-Term Memory, 2025. URL https://arxiv.org/abs/2504.19413. Version Number: 1.

Bernal Jiménez Gutiérrez, Yiheng Shu, Yu Gu, Michihiro Yasunaga, and Yu Su. HippoRAG: Neurobiologically Inspired Long-Term Memory for Large Language Models, 2024. URL https://arxiv.org/abs/2405.14831. Version Number: 3.

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, 2025a. URL https://arxiv.org/abs/2504.14225. Version Number: 2.

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, 2025b. URL https://arxiv.org/abs/2512.06688. Version Number: 1.

Langchain. LangMem: Long-Term Memory SDK for LLM Agents, 2025. URL https://langchain-ai.github.io/langmem/.

Shayne Longpre, Kartik Perisetla, Anthony Chen, Nikhil Ramesh, Chris DuBois, and Sameer Singh. Entity-Based Knowledge Conflicts in Question Answering. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 7052–7063, Online and Punta Cana, Dominican Republic, 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.565. URL https://aclanthology.org/2021.emnlp-main.565.

Wenquan Ma, Jiayan Nan, Wenlong Wu, and Yize Chen. What Deserves Memory: Adaptive Memory Distillation for LLM Agents, 2025. URL https://arxiv.org/abs/2508.03341. Version Number: 4.

Charles Packer, Sarah Wooders, Kevin Lin, Vivian Fang, Shishir G. Patil, Ion Stoica, and Joseph E. Gonzalez. MemGPT: Towards LLMs as Operating Systems, 2023. URL https://arxiv.org/abs/2310.08560. Version Number: 2.

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, 2023. URL https://arxiv.org/abs/2304.03442. Version Number: 2.

Preston Rasmussen, Pavlo Paliychuk, Travis Beauvais, Jack Ryan, and Daniel Chalef. Zep: A Temporal Knowledge Graph Architecture for Agent Memory, 2025. URL https://arxiv.org/abs/2501.13956. Version Number: 1.

Noah Shinn, Federico Cassano, Edward Berman, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language Agents with Verbal Reinforcement Learning, 2023. URL https://arxiv.org/abs/2303.11366. Version Number: 4.

Zihao Tang, Xin Yu, Ziyu Xiao, Zengxuan Wen, Zelin Li, Jiaxi Zhou, Hualei Wang, Haohua Wang, Haizhen Huang, Weiwei Deng, Feng Sun, and Qi Zhang. Mnemis: Dual-Route Retrieval on Hierarchical Graphs for Long-Term LLM Memory. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 23914–23928, San Diego, California, United States, 2026. Association for Computational Linguistics. doi: 10.18653/v1/2026.acl-long.1096. URL https://aclanthology.org/2026.acl-long.1096.

Yu Wang, Ryuichi Takanobu, Zhiqi Liang, Yuzhen Mao, Yuanzhe Hu, Julian McAuley, and Xiaojian Wu. Mem-: Learning Memory Construction via Reinforcement Learning, 2025. URL https://arxiv.org/abs/2509.25911. Version Number: 1.

Di Wu, Hongwei Wang, Wenhao Yu, Yuwei Zhang, Kai-Wei Chang, and Dong Yu. LongMemEval: Benchmarking Chat Assistants on Long-Term Interactive Memory, 2024. URL https://arxiv.org/abs/2410.10813. Version Number: 2.

Jian Xie, Kai Zhang, Jiangjie Chen, Renze Lou, and Yu Su. Adaptive Chameleon or Stubborn Sloth: Revealing the Behavior of Large Language Models in Knowledge Conflicts, 2023. URL https://arxiv.org/abs/2305.13300. Version Number: 4.

Wujiang Xu, Zujie Liang, Kai Mei, Hang Gao, Juntao Tan, and Yongfeng Zhang. A-MEM: Agentic Memory for LLM Agents, 2025. URL https://arxiv.org/abs/2502.12110. Version Number: 11.

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, 2025. URL https://arxiv.org/abs/2508.19828. Version Number: 5.

Sizhe Zhou and Jiawei Han. A Simple Yet Strong Baseline for Long-Term Conversational Memory of LLM Agents, 2025. URL https://arxiv.org/abs/2511.17208. Version Number: 2.

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, 2025. URL https://arxiv.org/abs/2506.15841. Version Number: 2.

A Appendix

A.1 Methods: Node Schemas

Table 4: The four node types stored in STRATA's graph.

Node	Layer	Fields	Evolves
Entity	**Spatial**	`(name, kind)` kind ∈ {person, place, org, goal, object, self, relationship}	✗
Fact	**Spatial**	`(Entity, attribute, value)` chain	✓
Event	**Temporal**	`(name, time / time-range, participants)`	✗
Preference	**Spatiotemporal**	`(topic, attribute, value, trigger-event)` chain	✓

Table 4 describes each node type in the STRATA graph, what fields they store, and whether the node is represented as a chain or not. All information in the Fields column is extracted by an LLM during the Extraction phase.

A.2 Cross-system Cost Analysis

Table 5: **End-to-end Computational Cost per item on PersonaMem-32K**.

System	Acc. (%)	LLM calls	In tok	Out tok	Total tok	Latency (s)	$
Full Context	50.59	1.01	25,824	48	25,872	3.5	0.0029
Mem0	51.44	13.97	12,080	3,360	15,440	49.1	0.0038
LangMem	51.78	14.65	39,106	30,186	69,292	359.8	0.0218
RAG (k=8)	58.40	1.00	2,098	40	2,138	1.4	0.0003
A-Mem	60.44	19.63	27,352	4,666	32,018	58.8	0.0068
STRATA (ours)	61.63	8.07	123,170	2,636	125,807	56.7	0.3117

STRATA consumes the most tokens of any system (125.8k per item against LangMem's 69.3k) (Table 5). This is expected with a ReAct loop, as every round of reasoning the agent makes re-sends the whole growing context with the tool descriptions, task instructions, and accumulated evidence. To compensate, it makes the *fewest* LLM calls per item of any memory system (8.07 per item against A-Mem's 19.63, LangMem's 14.65 and Mem0's 13.97). This saving is structural - STRATA's unique cross-link layers generated during ingestion allows for efficient information retrieval due to the 1-hop expansion. STRATA's latency is 57 s per item, between Mem0's 49 s and A-Mem's 59 s, and 6× faster than LangMem's 360 s. Thus, what separates STRATA's cost from every baseline is the repetitive prompt prefixes and high price per token of the claude-opus-4.6 orchestrator at inference time. §A.3 swaps out the orchestrator to show that similar accuracy can be achieved while lowering the token consumption and cost.

A.3 Orchestrator Ablation

To isolate the contribution of the orchestrator, we re-ran the full benchmark twice with two smaller orchestrator models: `gpt-4.1-mini` and `gpt-4o-mini`. Everything else is held fixed, including the extracted nodes, edges, tool set, and dense pre-retrieval seed.

Table 6: **Orchestrator ablation** on PersonaMem-32K.

Orchestrator	Acc. (%)	Rounds/item	In tok/item	$/item	s/item
`claude-opus-4.6` (STRATA)	61.63	5.55	101,785	0.3117	50.6
`A-Mem` (Comparator)	60.44	-	27,532	0.0068	58.8
`gpt-4.1-mini`	59.93	4.86	65,251	0.0196	5.9
`gpt-4o-mini`	58.23	3.25	39,780	0.0048	6.2

Most of the architecture's value is independent of the orchestrator. `gpt-4.1-mini` reaches 59.93% against Opus's 61.63%. It loses 1.70 points but costs 16× less per item ($0.0196 against $0.3117) and takes 8.6× less latency (5.9 s against 50.6 s). Despite the slight accuracy cut, STRATA with `gpt-4.1-mini` still beats Full Context by 9.3 points, Mem0 by 8.5 and LangMem by 8.2, and is below the best of A-Mem by only 0.51% (59.93% vs 60.44%) at a fraction of the latency (58.8s for A-Mem versus 5.9s for `gpt-4.1-mini`). `gpt-4o-mini` is even cheaper ($0.0048) but gives up 3.40 points. Thus, while a more powerful reasoning model (`claude-opus-4.6`) provides the few-point boost needed to reach the accuracy ceiling on PersonaMem, STRATA's graph carries value on its own and offers a compelling alternative to existing baselines at a fraction of the cost and latency with comparable token use.