Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

AdaKVT-Mem: An Auditable Lifecycle-Aware Memory Architecture for Evolving Conversational Records

QQ F
Nantong University
Zuhui HU
hzh@ntu.edu.cn
Nantong University

Research Article

Keywords: Conversational information systems, lifecycle management, provenance, versioned memory, auditable AI

Posted Date: October 23rd, 2026

DOI: https://doi.org/10.21203/rs.3.rs-10875711/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.

International Journal of Data Science and Analytics manuscript No.
(will be inserted by the editor)

AdaKVT-Mem: An Auditable Lifecycle-Aware Memory Architecture for Evolving Conversational Records

QQ F · HU Zuhui

Abstract Conversational memory requires not only retrieved text but validity, versioning, provenance, access state, and auditable change. We present AdaKVT-Mem, an inspectable maintenance protocol for conversational records interpreted by a large language model (LLM). After an assertion is mapped to a relation, the protocol commits one cross-layer mutation of a Stateful Memory Cell, its lifecycle and version, keyed placements, typed relations, and operation trace. It retains predecessors for historical access, logs rejected and accepted operations, and checks reference-integrity and lifecycle assertions before installation. Finite-domain checks and stored snapshots evaluate these conditional properties after interpretation. Controlled read-path comparisons characterize budget-dependent quality–cost trade-offs rather than universal performance superiority. On double-annotated updates, high lifecycle accuracy coexists with 1.28% strict complete-update success, locating limitations in candidate recovery, predecessor ranking, and exact value reconstruction. AdaKVT-Mem is an auditable reference implementation for structured record maintenance, not a claim of reliable end-to-end updating or general answer-quality dominance.

Keywords Conversational information systems · lifecycle management · provenance · versioned memory · auditable AI

Nantong University, School of Artificial Intelligence and Computer Science, Nantong, Jiangsu, China
QQ F: E-mail: 2541320045@stmail.ntu.edu.cn; ORCID: 0009-0002-2577-0506
Corresponding author: HU Zuhui, E-mail: hzh@ntu.edu.cn; ORCID: 0000-0001-7973-4936

1 Introduction

Conversational memory sits at the boundary of artificial intelligence and data management. An agent must interpret natural-language assertions while maintaining evolving user information as queryable records with temporal validity, lineage, uncertainty, and version history. Memory is therefore information infrastructure rather than only a prompt component: its records must remain inspectable as interactions accumulate. Here, *auditable* denotes inspectable provenance, proposed and executed transitions, cross-structure references, and accepted or rejected operations—not answer entailment by retrieved evidence. Prior work has demonstrated memory streams, external archives, semantic segmentation, and hierarchical memory as useful building blocks [20; 32; 17; 19; 25].

Appending every interaction to a vector store, however, does not produce reliable memory. It fragments facts, preserves outdated preferences, obscures the relation between old and new assertions, and makes it difficult to trace a generated answer to source dialogue. Temporal and conditional-memory studies show that dialogue time and fact-validity time must be distinguished [31; 24; 26]. Graph-based systems improve multi-hop retrieval but can introduce extraction, maintenance, and traversal costs as interactions accumulate [29; 21; 3]. Conversely, a pure hierarchy may localize retrieval efficiently while failing to express cross-view relations and state transitions.

We address this information-management problem with AdaKVT-Mem, an engineering protocol that connects representation, update, indexing, and retrieval through one operational contract. KVT denotes *Keyed-View Transition*: keyed multi-view access combined with explicit lifecycle transitions. It fixes the interface between LLM interpretation and a checked state

change: a proposed relation is rejected or becomes a logged mutation with a retained predecessor, synchronized projections, and separate current/historical eligibility; LLM inference alone cannot authorize destructive state changes. The contribution is not any individual temporal, versioning, provenance, graph, or indexing mechanism, but their declared synchronization for one accepted conversational update.

The contributions are as follows:

1. We define a managed conversational-memory record that separates assertion validity, lifecycle state, storage disposition, access eligibility, version metadata, and source provenance.
2. We specify an executable maintenance contract for interpreted assertions, including rejection, target validation, uncertainty preservation, predecessor retention, and administrative archival, with five cross-layer invariants and an explicit post-interpretation verification boundary.
3. We organize query access through seven keyed projections, vector candidates, lifecycle-aware admission, and a bounded typed-relation graph over one versioned cell collection.
4. We release a reference implementation and its available records, and evaluate it through finite-domain protocol checks, stored-snapshot audits, strict budget-controlled read-path comparisons, double-annotated update-semantics evaluation, configuration-level end-to-end characterization, provenance-normalized retrieval, and resource accounting.

The study is organized around three questions. *RQ1* asks how evolving conversational assertions can be represented and maintained as auditable information objects. *RQ2* asks whether accepted staged updates preserve the declared lifecycle, version, index, relation, and provenance invariants. *RQ3* characterizes—rather than presumes superiority in—the answer quality, evidence access, and operating cost of the resulting integrated pipeline. Together they pair an executable information-management contract with a transparent empirical systems profile.

2 Related Work

2.1 Long-term memory for conversational agents

Generative Agents, MemoryBank, and MemGPT established external and persistent memory for agentic systems [20; 32; 17]. More recent systems improve memory construction with semantic segmentation and metadata augmentation [19]. Dynamic memory is especially important for changing preferences and facts: conditional memory explicitly models evolving user information [31]. Systems such as Mem0 and reflective-memory frameworks further expose the practical need to retain, recall, and revise memory over extended interaction [5; 13].

We avoid assigning categorical completeness grades to systems with different abstractions and instead compare their operational scenarios. APEX-MEM exposes schema, entity, graph-query, and search tools over temporally grounded semi-structured memory; Temporal Semantic Memory constructs semantic timelines and durative memories for time-intent retrieval [3; 24]. Zep dynamically maintains historical relationships in a temporal knowledge graph, while AdaMem routes questions over working, episodic, persona, and graph memories [21; 30]. KVT asks a narrower systems question: once an assertion has a target and relation, how are lifecycle, predecessor, keyed placement, typed dependency, provenance, and query eligibility changed and checked together? A bitemporal or event-sourced backend can persist records but does not itself supply this AI-facing decision and projection-synchronization contract (Online Resource 1, Table S6).

2.2 Memory organization and evolution

Hierarchical memory and recursive tree retrieval can reduce the search space through coarse-to-fine access [25; 22]. Graph memory provides explicit relational reasoning through associative retrieval, agent memory graphs, and temporal knowledge graphs [9; 29; 21]. AdaMem combines working, episodic, persona, and graph memories with question-conditioned routing [30]. AdaKVT-Mem instead centers stateful representation and relation-driven lifecycle transitions, while avoiding graphification of ordinary topical similarity.

2.3 Conversational memory retrieval

Retrieval-augmented generation provides a general mechanism for conditioning answers on external evidence [14]. Dense Passage Retrieval and Contriever provide supervised and unsupervised dense-retrieval reference points for evidence selection [12; 11]. Memory-augmented transformers and long-context benchmarks show that retention and access quality cannot be inferred from context length alone [28; 2]. In long-horizon memory, semantic relevance alone is insufficient: state validity, temporal applicability, source provenance, and evidence completeness must affect selection before generation [15]. Prompt compression can reduce evidence-

context cost, but it must preserve temporal and negation cues [18]. This requirement motivates the state- and time-aware evidence reranking used by AdaKVT-Mem.

2.4 Temporal, versioned, and provenance-aware data management

The framework builds on established information-management ideas. Temporal databases distinguish observation time from validity time [23]; provenance research attaches source lineage to derived records [4]; and entity-resolution work treats duplicate detection as a decision problem [7]. Truth-maintenance and belief-revision research motivate explicit dependency and revision policies [6; 1]. Incremental view maintenance shows that multiple indexes require explicit synchronization and cost accounting [8]. None of temporal intervals, provenance, version chains, vector retrieval, typed graphs, or prefix indexes is claimed as individually novel. The contribution is their adaptation to LLM-extracted conversational assertions through one inspectable update contract: a relation decision is mapped to a state transition, and the resulting version, keyed placements, typed dependencies, provenance, and retrieval eligibility are synchronized as one declared mutation.

3 Problem Definition and System Overview

3.1 Problem definition

Let $D = \{u_1, \dots, u_T\}$ be a time-ordered dialogue, where each interaction $u_t = (speaker_t, text_t, time_t)$. The system maintains a memory set M. A memory cell is

$$m_i = \big(id_i, c_i, s_i, \rho_i, \eta_i, \tau_i, E_i, R_i, V_i, P_i, z_i^{\text{ext}}, z_i^{\text{rel}}, v_i, i_i, b_i, n_i\big). \tag{1}$$

Here id_i is a unique identifier; c_i is normalized content; and $s_i \in \mathcal{S}$ is semantic lifecycle state. Storage disposition is ρ_i, the stored access flag is $\eta_i \in \{0,1\}$, and $\tau_i = [t_i^-, t_i^+)$ is a half-open semantic-validity interval, with $t_i^+ = +\infty$ permitted. E_i, R_i, and V_i are entity identifiers, typed relation edges, and keyed view paths; P_i is an ordered set of provenance records. Scores $z_i^{\text{ext}}, z_i^{\text{rel}} \in [0,1]$ record extraction and relation decisions; $v_i \in \mathbb{N}^+$ is the version; and $i_i, b_i \in [0,1]$ and $n_i \in \mathbb{N}$ are importance, stability, and reinforcement count. The implementation retains the legacy field name `retrieval_eligible` for η_i.

The query mode $\mu(q)$ is CURRENT, HISTORICAL, CHANGE, or OPEN; let $\mathcal{H}$ denote the historical and change modes. Temporal normalization maps a resolvable query to a half-open interval $\tau_q = [t_q^-, t_q^+)$. A point reference is represented by the smallest supported calendar interval containing that point, and

$$\text{TimeMatch}(\tau_i, q) = \mathbf{1}[\tau_i \cap \tau_q \neq \varnothing]\,. \tag{2}$$

The reported implementation resolves day/week-relative expressions against the source interaction timestamp and materializes year, month, and day intervals. For a current query without an explicit temporal expression, the query evaluation time supplies the point interval. It does not infer a missing year or timezone when more than one interpretation remains possible; such a value is retained as unresolved/uncertain rather than used for a destructive transition. Write $T_i(q) = \text{TimeMatch}(\tau_i, q)$. Semantic admissibility is

$$\text{SemAdm}(m_i, q) = \begin{cases} \mathbf{1}[s_i \in \{\text{ACTIVE}, \text{UNCERTAIN}\}]T_i(q), & \text{CURRENT}, \\ \mathbf{1}[s_i \in \mathcal{S}]T_i(q), & \mu(q) \in \mathcal{H}, \\ \mathbf{1}[s_i = \text{ACTIVE}], & \mu(q) = \text{OPEN}. \end{cases} \tag{3}$$

The final cell-admission predicate is

$$\text{Admit}(m_i, q) = \mathbf{1}[\rho_i = \text{ONLINE}]\,\eta_i\,\text{SemAdm}(m_i, q). \tag{4}$$

The OPEN branch is atemporal: it denotes a question with no personal-time constraint after routing. It remains memory-optional rather than memory-prohibited, because an open-form question may still be answerable from personal history. Explicit dates are not required for historical routing: lexical cues such as "previously", "used to", "before", or an explicit request for change also select HISTORICAL/CHANGE. Only questions without such cues remain OPEN, for which superseded and expired cells are excluded by Eq. 3. An UNCERTAIN cell may be admitted for current queries but is down-weighted, and the generator is instructed to preserve the corresponding verbal qualification. Separating the two confidence values prevents an extraction score from being reused as evidence for a relation transition. Given a query q, retrieval chooses views, returns evidence $\mathcal{E}(q)$, and constrains generation to that evidence:

$$\begin{aligned} a &= \text{Generate}(q, \mathcal{E}(q)), \\ \mathcal{E}(q) &= \text{ReRank}(\text{Retrieve}(q, M)). \end{aligned} \tag{5}$$

The objective is not unrestricted similarity recall. It is to maximize coverage of currently valid evidence, minimize obsolete or unsupported evidence, preserve provenance, and control operational cost.

The formulation distinguishes interaction time from the time at which a fact is valid. It also distinguishes semantic similarity from update compatibility: two similar statements may be duplicates, complementary descriptions, successive versions, or conflicting assertions. The system therefore decides both *where* a candidate is stored and *how* it changes the existing memory lifecycle.

Define the mutable system state as $\Omega = (M, T, X, G, L)$, where T is the family of keyed MHI view maps, X is the online-cell vector index, G is the typed relation graph, and L is the append-only decision trace. A KVT protocol instance is

$$\mathcal{K} = (\Omega, \delta, \pi), \tag{6}$$

The transition function δ maps a candidate and a prior state to a finite cell-mutation set, while π maps a query and Ω to an ordered evidence list. Vector, view, graph, and log mutations are deterministic projections of accepted cell mutations (Online Resource 1, Algorithm S1). The transition table plus default rejection rule makes δ total over its declared input domain. A conformant execution targets five invariants: (I1) $\rho_i = \text{ARCHIVED} \Rightarrow \eta_i = 0$ and id_i is absent from cell candidate sets; (I2) a current query admits neither superseded nor expired cells, while historical/change queries require interval compatibility; (I3) every successor names its predecessor and creating operation; (I4) every online access-enabled cell is reachable from every path declared in V_i, and no path or graph edge names a missing cell; and (I5) every returned cell identifier resolves to its stored provenance and write event. Sect. 4.5 specifies δ, and Sect. 6.1 evaluates the executable protocol checks.

3.2 Design objectives

The design targets are: (i) explicit state semantics for current and historical questions; (ii) inspectable updates; (iii) query-conditioned retrieval; (iv) evidence traceability; and (v) measurable cost and latency. Each write records the candidate, selected target, relation, operation, state change, and source, while the MHI and relation graph provide complementary access paths over the same stateful cell collection.

3.3 System overview

Figure 1 summarizes the write and read paths. A dialogue turn becomes normalized candidates, relation decisions, and validated state transitions whose cell references are synchronized with the MHI and typed graph. The Boston-to-Seattle residence update used below closes a prior interval, creates a retained successor, and changes current versus historical access without duplicating the record. A question is routed to applicable views; path and vector candidates are merged, relation-sensitive queries trigger bounded expansion, and lifecycle/temporal scores rank provenance-bearing evidence.

4 AdaKVT-Mem

4.1 Candidate extraction and normalization

At time t, the vector index first returns N_t, the ten online memories nearest to the incoming interaction. Let $M_X = \{m \in M : \rho_m = \text{ONLINE}\}$ denote the write-comparison population exposed by the vector index. The candidate extractor receives the interaction and these compactly identified neighbors and returns a set of prospective memories,

$$\begin{aligned} C_t &= \text{Extract}(u_t, N_t), \\ N_t &= \text{Top10}_{m \in M_X}\, \text{sim}(u_t, m). \end{aligned} \tag{7}$$

It retains only atomic, verifiable, and updateable facts, preferences, commitments, task events, or time-bounded events. Greetings, repeated confirmations, stylistic fillers, and instructions whose validity ends with the turn are not written. This gate prevents the index from becoming a transcript archive and makes subsequent state decisions meaningful.

For each retained candidate, normalization resolves aliases and coreference, canonicalizes temporal expressions, and binds the source turn. It preserves source text for audit while using the canonical form for matching. Extraction only proposes a candidate; the relation and transition policies separately decide whether to insert, merge, revise, or withhold it.

4.2 Stateful Memory Cell: unified representation

The Stateful Memory Cell is the persistent unit of AdaKVT-Mem. Rather than treating a memory as a vector plus a snippet, it associates a normalized assertion with semantic validity, source provenance, state, version history, view placements, and explicit relations. Figure 2 shows the representation and Online Resource 1, Table S7 gives the field-level schema. The fields jointly support three questions that ordinary vector memory leaves implicit: what was asserted, when was it valid, and why is this the appropriate version to retrieve now?

The record distinguishes the turn timestamp in provenance from the interval in which a fact

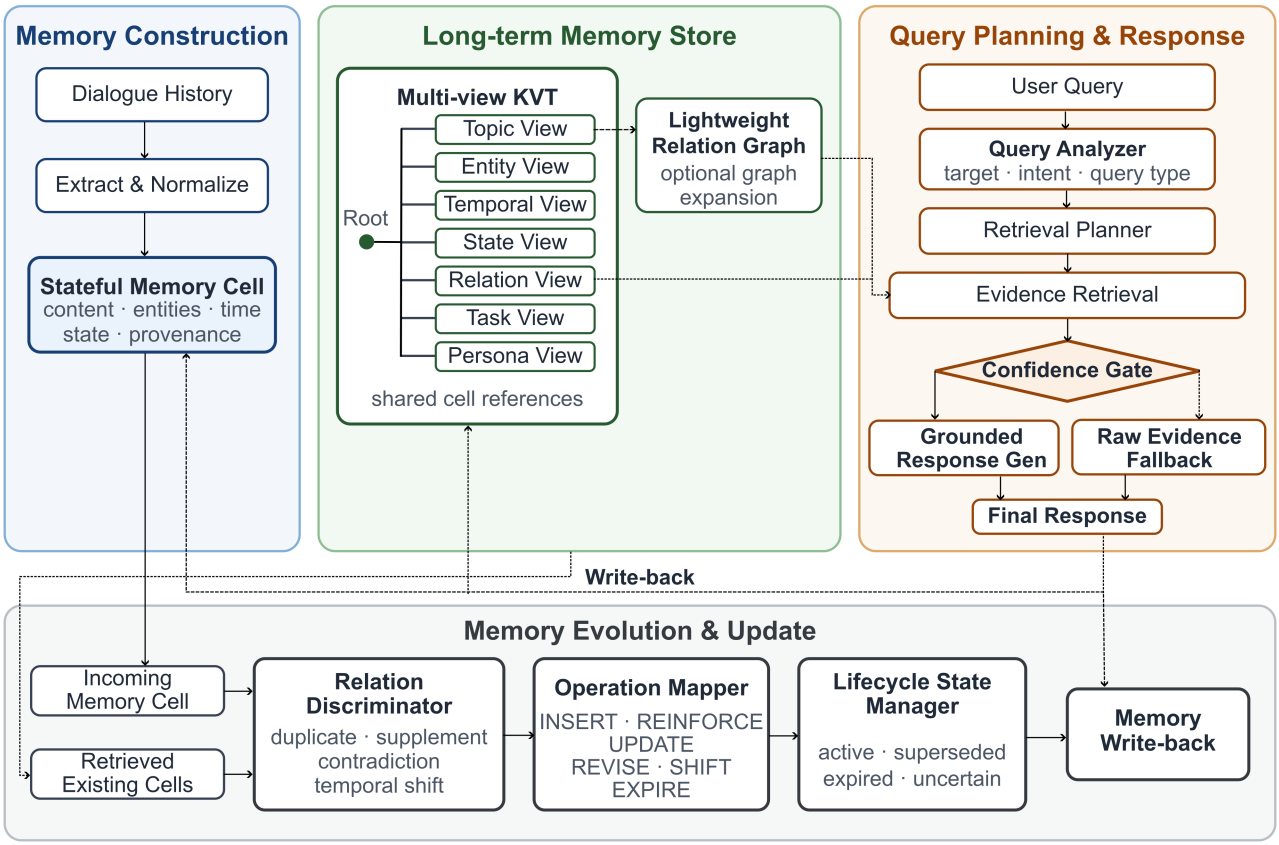

Fig. 1 Overall AdaKVT-Mem framework. The five prefix-hierarchical views (topic, entity, temporal, task, and persona) are accompanied by one-level state and relation indexes, yielding seven shared-reference keyed access structures. The relation graph is a separate typed-edge structure; state and relation are not graph-only or scoring-only metadata.

is claimed to hold. A cell may aggregate multiple source fragments while retaining one normalized expression. Importance represents retrieval utility, whereas stability represents persistence of evidential support. Importance starts at 0.50 and stability at 0.55. Accepted reinforcement adds 0.03 and 0.06 respectively (clipped to $[0, 1]$); revision, contradiction, temporal shift, or expiry adds 0.04 to importance and subtracts 0.08 from stability. These increments are fixed heuristics selected during development, not learned or probability-calibrated quantities. Importance keeps revised facts visible for inspection-sensitive retrieval, while stability discounts claims whose support has changed. The normative write-event tuple is $(candidate, F(c), m^*, r, o, \Delta, t, source, config, model)$; its first eight fields form the core transition trace checked in Sect. 6.1.

4.3 Multi-view hierarchical index and relation graph

Multi-view hierarchical index. A cell is attached by reference to seven keyed access structures: five prefix-hierarchical views (topic, entity, time, task, and persona) plus one-level categorical state and relation indexes. State and relation are first-class maintained indexes rather than undocumented scoring metadata; Figs. 1 and 2 illustrate the shared-reference organization. The same cell identifier is shared across placements, so the indexes organize access to one object rather than create independent copies. Hierarchical nodes contain a path label, child map, and identifiers reachable below that path; embedding similarity remains in a separate vector index. Let $\mathcal{P}(q)$ be the view–path pairs emitted by the query planner and $I_v(p)$ the identifiers below path p in view v. The implemented path score is

$$T(m, q) = \min\left(1, \sum_{(v,p)\in\mathcal{P}(q)} w_v \, \mathbf{1}[m \in I_v(p)]\right). \tag{8}$$

The fixed path weights are 0.35 for topic, 0.25 for entity, state, and relation, 0.10 for time, and 0.20 for task and persona. They encode view priority and are independently applied before clipping the sum to one. Insertion adds a cell identifier to every node on each derived

path; an update removes old references and rebuilds them from current metadata. The resulting MHI is a deterministic multi-key access structure, while vector similarity remains a complementary candidate source. The intervention experiment compares this complete structured-access bundle with vector-only retrieval.

The hierarchy is declared rather than learned. Topic, task, and persona use normalized slash-delimited paths with fixed lexical fallback; entities use canonical type/value paths; and time materializes year, month, and day prefixes. State and relation remain one-level categorical indexes. Retrieval traces list the five routed prefix views and record lifecycle admission and relation expansion separately. Thus the structure is a prefix hierarchy over declared keys, not an induced topic taxonomy.

For a set of paths $\mathcal{P}_m$, insertion touches $O(\sum_{p\in\mathcal{P}_m}|p|)$ tree nodes and lookup costs $O(|p| + |I_v(p)|)$ before downstream scoring. Updating a cell removes its references from the user's view trees and reinserts rebuilt paths, with worst-case maintenance cost linear in stored tree size plus new path lengths. Sect. 6.7 reports the measured endpoint effect of structured access, and Online Resource 1 reports the storage timing envelope.

Bounded typed relation graph. The graph stores cross-view entity links, temporal succession, causality, supplement, contradiction, replacement, and explicit multi-hop bridges. Ordinary same-topic similarity does not create an edge. Given a seed set, breadth-first expansion visits each node at most once for $H \le 2$ hops and returns at most $K_G = 20$ candidates. If h_m is the shortest discovered hop and e_m is the number of matched edges, the executable scoring specification computes

$$G(m,q) = \min\left(1, \frac{0.35}{\max(h_m,1)} + 0.08\min(e_m,5)\right). \tag{9}$$

Cycles are blocked by a visited set, duplicate destinations retain the maximum score, and the candidate cap is applied after scoring. Thus $G(m,q) \in [0,1]$. Relation type determines whether expansion is permitted; it does not introduce an unreported additive score.

Online Resource 1, Table S8 specifies relation direction, creation, retention, and query triggers. Supersession and expiry preserve historical edges; archival removes an endpoint from online generation; physical deletion removes incident edges. Successors inherit eligible edges only after endpoint and key checks.

4.4 Relation-driven update semantics

For candidate $c \in C_t$, let $L(c)$ be its proposed explicit target identifiers and $k(c)$ its canonical entity–predicate key. The key is the tuple

$$k(c) = (\mathit{owner}(c),\ \mathit{entity}(c),\ \mathit{slot}(c),\ \mathit{event}(c)), \tag{10}$$

where aliases and coreference are resolved before forming the canonical entity; the normalized slot names the predicate/attribute role; and the event component is empty for persistent attributes. Attribute values are deliberately excluded. Thus Boston and Seattle are values of the same residence slot rather than different keys. Event memories include their event type and participant role so that unrelated events involving one entity are not forced into one chain. If entity or slot resolution is ambiguous, $k(c) = \perp$ and no destructive predecessor update is allowed.

Slot cardinality. The canonical key identifies a comparison domain; it does not by itself imply that all values in that domain are mutually exclusive. A slot catalogue assigns $\chi(\mathit{slot}) \in \{\textsc{single}, \textsc{set}, \textsc{ordered}, \textsc{event}\}$. For a SINGLE slot, a validated correction or replacement creates a successor chain. For a SET slot, distinct values coexist as separate cells under the same key and additions map to SUPPLEMENT; an explicit removal targets and closes only the named item. Ordered slots retain item identity and order metadata, while event slots use the event discriminator, participants, and time to separate parallel occurrences. When cardinality or target identity is unresolved, the safe operation is insertion as uncertain/supplementary evidence rather than destructive replacement. This typed interpretation covers multi-valued preferences, skills, projects, medications, residences, and goals while preserving the shared-key lookup benefit.

A proposed link is accepted only by

$$\begin{aligned}\mathrm{LinkOK}(c,m) = {} & \mathbf{1}[id_m \in L(c)]\mathbf{1}[id_m \neq id_c] \\ & \mathbf{1}[k(m) = k(c)] \\ & \mathbf{1}[\sigma(c,m) \ge \theta_l \vee h(m) = h(c)],\end{aligned} \tag{11}$$

where $h(\cdot)$ is the normalized-content hash. The feasible comparison set is $F(c) = \mathrm{Dedupe}(N_t \cup \{m \in M : \mathrm{LinkOK}(c,m) = 1\})$. Let $\ell_j = \mathrm{LinkOK}(c,m_j)$ and $\sigma_j = \sigma(c,m_j)$. For nonempty $F(c)$, the implementation selects

$$m^* = \operatorname*{arg\,max}^{\mathrm{lex}}_{m_j \in F(c)} (\ell_j, \sigma_j). \tag{12}$$

Thus a link that has passed Eq. 11 has priority over an unlinked vector neighbor; similarity orders candidates within the same link-validity class. This declared

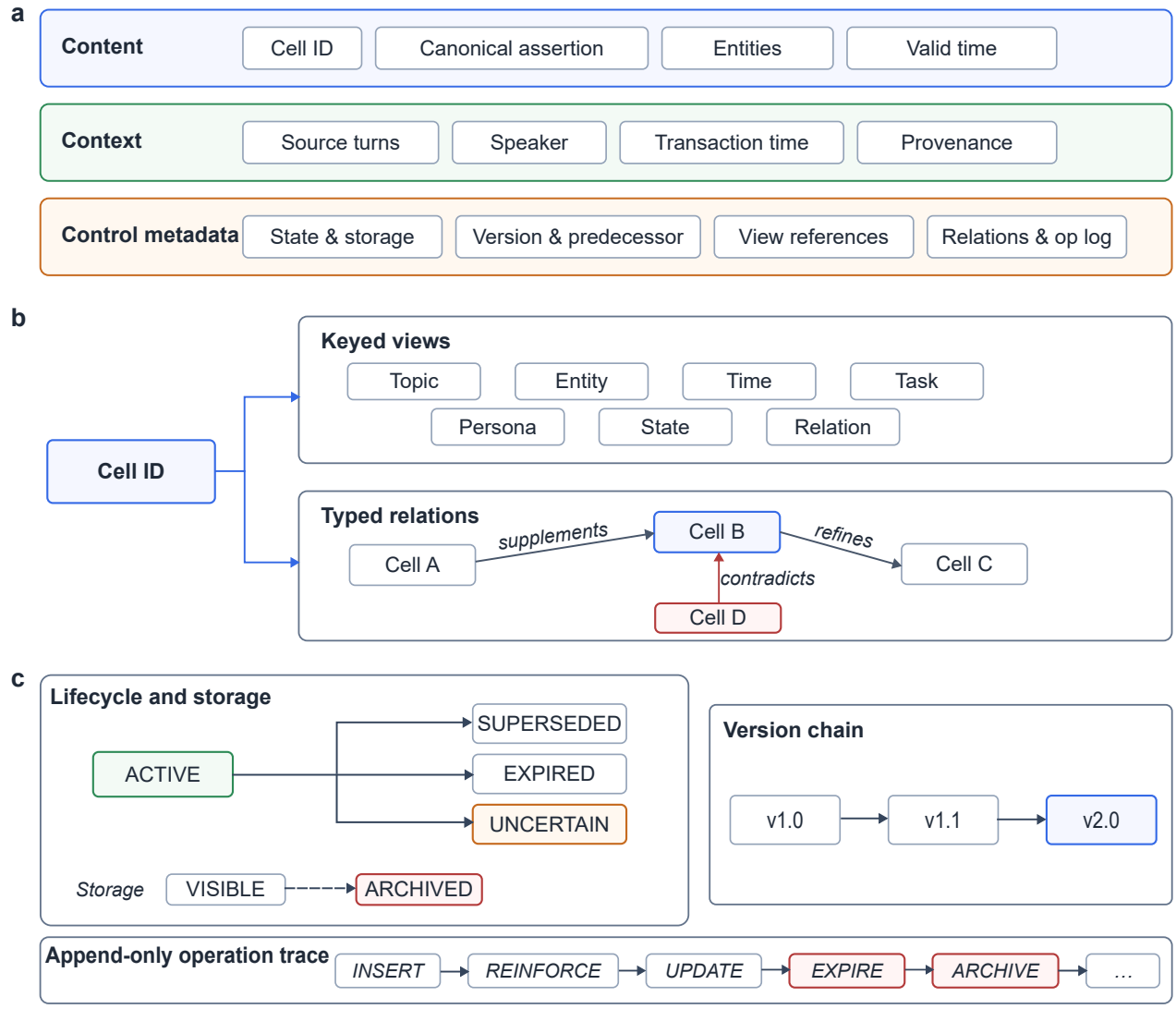

Fig. 2 Stateful Memory Cell representation and maintenance. (a) The cell schema separates semantic content, source context, and protocol-managed metadata. (b) A single cell identifier is referenced by keyed views and a typed relation graph, avoiding duplication of the underlying record. (c) Accepted operations update lifecycle or storage status, preserve the version chain, and append an auditable operation event. Solid arrows denote state, version, or dependency transitions; dashed arrows denote contradiction or storage-only transitions.

priority rule makes target selection explicit and auditable. The rule decision r_0 is the first satisfied branch of the ordered policy. The similarity thresholds are $\theta_d = 0.88$ (reinforcement), $\theta_u = 0.72$ (update), and $\theta_l = 0.65$ (linked update). Same-interval contradiction uses $\theta_c = 0.72$ on z^{rel}, while expiry uses 0.45 only when the canonical key matches. The extractor first splits a multi-fact turn into atomic candidates; one primary content-changing operation is committed per candidate. Consequently, one utterance may update several facts without assigning several competing predecessors to the same atomic candidate. Compatible neighbors may additionally receive non-mutating supplement or reinforcement edges.

The canonical key serves as a high-precision comparison and validation key rather than an unconditional destructive-update address. It becomes available after owner, entity, slot, cardinality, and event normalization; set-valued and event slots can legitimately retain several cells under one key, while unresolved aliases yield $k(c) = \perp$. Directly treating every key match as the predecessor would therefore collapse valid coexisting values. The implementation instead combines exposed semantic candidates with validated explicit links, then uses key equality and slot cardinality to authorize a content-changing operation. This design favors safe update semantics; the observed Recall@1–20 profile quantifies the associated candidate-ranking trade-off.

The reported configuration enables a bounded JSON-only LLM review only when the rule gate marks a decision ambiguous: an UPDATE/SHIFT below 0.68, an add/update/shift below 0.78 with reference or change cues, or a contradiction/temporal-shift decision that also contains reference cues. The review can override r_0 only with a valid relation–operation pair and confidence at least 0.72; otherwise $r = r_0$. Consequently, the deterministic rules define the normal path and the language model does not score every write. The complete cue patterns, review prompt, and validation code are included in the accompanying artifact.

The policy distinguishes three levels: a *relation* labels the semantic link between an incoming assertion and an existing cell; an *operation* is the executable command; and a lifecycle *state* is part of the transition result. The task-layer labels used to evaluate natural language are a fourth, measurement-only level. In order, exact duplicates reinforce; explicit expiry closes validity; uncertain or supplementary assertions are inserted separately; temporal shifts, contradictions, refinements, and updates create retained successor chains; otherwise a new active cell is inserted. Thus relation REFINEMENT→operation UPDATE and relations UPDATE/CONTRADICTION→operation REVISE are mappings, not synonyms. Table 1 gives the compact task-to-protocol crosswalk; Online Resource 1 gives the complete guard/effect table.

Low-value text is removed by the extraction gate before relation discrimination and therefore does not create a lifecycle transition.

4.5 State-dependent lifecycle and online maintenance

The semantic lifecycle state space from Eq. 1 and the separate storage-disposition space are

$$\begin{aligned}\mathcal{S} &= \{\text{ACTIVE}, \text{SUPERSEDED}, \text{EXPIRED}, \text{UNCERTAIN}\},\\ \mathcal{D}_\rho &= \{\text{ONLINE}, \text{ARCHIVED}\}.\end{aligned} \tag{13}$$

The complete operation vocabulary is partitioned into relation-emitted content operations and authenticated administrative operations:

$$\begin{aligned}\mathcal{O}_{rel} &= \{\text{INSERT}, \text{REINFORCE}, \text{UPDATE},\\ &\qquad \text{REVISE}, \text{SHIFT}, \text{EXPIRE}\},\\ \mathcal{O}_{adm} &= \{\text{ARCHIVE}, \text{RESTORE}\}.\end{aligned} \tag{14}$$

The ordered relation policy emits $\mathcal{O}_{rel}$; retention policy and authenticated restoration emit $\mathcal{O}_{adm}$. Authorized conflict resolution is represented by REVISE with $\kappa = 1$. For a same-interval high-confidence conflict, define the source-priority vector

$$\omega(x) = \big(e_x, u_x, t_x, z_x\big), \qquad w = \operatorname*{arg\,max}^{\text{lex}}_{x \in \{m^*, \hat{c}\}} \omega(x), \tag{15}$$

where e_x indicates an explicit correction, u_x a direct user assertion rather than an assistant inference, t_x the source timestamp, and z_x the stored or candidate relation score; an exact tie is resolved by the lexicographically smaller provenance identifier. Speaker identity and timestamp come from provenance; explicit-correction status comes from deterministic correction cues linked to the selected target; and z_x comes from the rule/review decision. The latter is a bounded decision score used for deterministic ordering. The transition function is

$$\delta(m^*, \hat{c}, o, r, z^{\text{rel}}, \kappa, w) \to (\Delta, a), \tag{16}$$

where $m^* = \perp$ denotes no predecessor, $a \in \{\text{ACCEPT}, \text{REJECT}\}$ records legality, and Δ is the finite set of predecessor/successor cell mutations. Online Resource 1, Table S9 defines Δ for every supported transition family. Inputs not matched by a row return $(\varnothing, \text{REJECT})$, so δ is total and rejection is side-effect free except for its trace event. Content-changing operations create a successor and retain the predecessor as the historical copy.

Equation 15 makes the conflict winner an explicit transition input. An assistant inference cannot supersede an explicit user assertion solely through recency or score. The authorization flag κ is set only by an explicit user correction/confirmation linked to the conflict or by an authenticated user/administrator resolution command; ambiguity review may propose a relation but cannot set κ. When several neighbors are compatible, exactly one primary target receives the content-changing operation. Secondary targets may receive validated non-mutating edges, and a failed secondary-edge validation leaves all cell lifecycle states unchanged.

For example, "I no longer live in Boston; I moved to Seattle last year" is split into a Boston-ending candidate and a Seattle-starting candidate. The first closes the Boston validity interval; the second creates an active Seattle successor linked to the prior residence record. A current query returns Seattle, a historical query can recover Boston, and a change query can return both intervals and their transition. The executable cross-layer contract makes this behavior consistent across storage, indexing, retrieval, and provenance.

The current planner admits predecessor and successor records for change queries but does not enforce that every returned answer contains a complete before/after

Table 1 Task labels measure natural-language decisions; native relations express semantic links; operations are executable commands; lifecycle is the resulting state change.

Task label	Native relation	Operation	Lifecycle result
REPLACEMENT	UPDATE	REVISE	predecessor superseded; successor active
ACCUMULATION	REFINEMENT	UPDATE	predecessor superseded; successor active
REVERSAL	CONTRADICTION	REVISE	predecessor superseded; successor active
RELOCATION	temporal shift	SHIFT	predecessor superseded; successor active
CORRECTION	CONTRADICTION	REVISE	predecessor superseded; successor active
NO_UPDATE	rejection	none	existing active cell retained

pair. LongMemEval Complete Evidence measures annotated source coverage across all eligible questions; it is not a dedicated change-pair metric. Pair completeness is therefore an explicit target for the update-semantics evaluation rather than a claimed guarantee.

`Active` cells receive the highest score for ordinary current-memory retrieval. `Superseded` cells retain prior versions for change/history questions. `Expired` cells fall outside their validity interval. `Uncertain` cells represent unresolved evidence and are down-weighted. Archival is an administrative storage disposition, not a fifth semantic state. Both $\rho = \text{ONLINE}$ and $\eta = 1$ are checked before scoring. An archived cell is never returned; authorized restoration sets ρ to ONLINE and restores the eligibility value saved at archival while preserving the cell's semantic lifecycle state.

For current questions, the stateful score favors active cells and the temporal paths favor the requested period. Historical or change questions can retrieve superseded cells through state, relation, and time paths. State transitions prefer explicit temporal boundaries and revision signals; uncertain assertions are stored separately instead of overwriting a stronger active cell. All transitions are recorded in the append-only log.

The implementation groups accepted cell, MHI, and graph changes under one process-local operation identifier and checks that every online, access-enabled cell remains reachable from its declared paths and that no path or edge names a missing cell. Replacement and conflict edges connect predecessor and successor; other inherited edges are retained after endpoint and validity checks. This identifier delimits the staged mutation and audit trace. The same interface supports a transactional backend with durable write-ahead logging, rollback, concurrent-writer isolation, and replay recovery.

Online Resource 1, Algorithm S1 gives the online write pseudocode. The incoming turn retrieves ten vector neighbors before extraction. The extractor can propose only the temporary identifiers exposed for those neighbors; the full identifier map subsequently resolves and validates a proposed target through Eq. 11, but it does not recover a predecessor omitted from the ten-neighbor candidate set. Thus predecessor-link recall is bounded by write-time candidate recall. Cell, MHI, graph, and history mutations are staged on copies, checked against I1–I5, and then installed together under one process-local operation identifier.

KVT guarantee boundary. KVT is defined over logical cell, keyed-view, vector, relation, and event-log interfaces; JSON and Qdrant are replaceable realizations. Given an interpreted target, relation, operation, and validity interval, rejection leaves the installed state unchanged except for its trace event; an accepted staged tuple is installed only after checking predecessor retention, reference integrity, lifecycle access rules, and I1–I5. These are conditional implementation-consistency guarantees, not guarantees of candidate extraction, predecessor recall, value reconstruction, source coverage, or answer entailment. A durable backend can additionally provide atomic commit and recovery.

4.6 Question-conditioned retrieval and evidence reranking

The rule-based router predicts a question type and emits only the applicable view paths from the natural-language query, recognized entities, temporal expressions, and task or persona cues. It does not read LoCoMo/LongMemEval category labels, gold answers, or gold evidence at test time. Persona questions enable persona paths; entity and project questions enable entity paths; temporal phrases add time paths; and task-progress questions add task paths. The view weights themselves are fixed as specified in Eq. 8. A low-confidence JSON review may amend the path set, while the memory-optional path handles questions that do not concern the user's history. This prevents a personalized memory system from injecting irrelevant personal evidence into open-form answers.

Retrieval resolves the planned MHI paths and complements those candidates with normalized vector similarity. If the query is relation-sensitive, the graph expands from the seed cells according to Eq. 9. For each

cell, the stateful score is

$$C(m,q) = 0.40 i_m + 0.30 b_m + 0.10 \min(n_m/5, 1) + 0.20 s_q(m), \tag{17}$$

where i_m, b_m, and n_m are the fields in Eq. 1. For current/change routing, the active, superseded, uncertain, and expired state values are 1.0, 0.45, 0.35, and 0.15; inadmissible combinations are removed before scoring. For historical/open routing, the executable specification uses 0.80, 0.40, 0.35, and 0.20, with the open admission rule retaining only active cells. Schema-invalid cells are rejected before scoring. Hence $C(m,q) \in [0,1]$. After applying Eq. 4, the final score is

$$R(m,q) = 0.45 T(m,q) + 0.30 V(m,q) + 0.15 C(m,q) + 0.10 G(m,q), \tag{18}$$

where $V \in [-1,1]$ is the native cosine-similarity score returned by Qdrant and G is Eq. 9 (zero when graph expansion is disabled). No candidate-set min–max normalization is applied. Since $T, C, G \in [0,1]$, $R(m,q) \in [-0.30, 1]$; R is a ranking utility, not a probability. Entity, temporal, and view compatibility affect T, while lifecycle metadata affects C. The 0.45/0.30/0.15/0.10 coefficients are the values in the executable artifact and were frozen after development-split selection. The returned record retains every component score, matched path/edge, state, timestamp, and source identifier.

Consequently, a negative native cosine score, if returned, lowers R rather than being remapped or clipped. This native-scale choice is calibrated for the reported embedding/index configuration; a replacement similarity backend uses the same development-split calibration procedure before evaluation.

Evidence-sufficiency gate and fallback. The architecture does not switch between two answer models. After initial retrieval, the deterministic gate computes

$$H(q) = \min(0.25, 0.12 n_e) + \min(0.25, 0.08 n_k) + 0.35 I_{type} + 0.15 I_{raw}, \tag{19}$$

where n_e is the number of distinct planned entities matched by evidence identifiers or canonical aliases, n_k is the number of distinct non-stopword query terms matched in normalized content, $I_{type} = 1$ only when at least one retrieved span contains a deterministic signal compatible with the requested answer type, and $I_{raw} = 1$ only when a provenance-linked raw window contains a matched entity or key term. Thus $H(q) \in [0,1]$. The threshold is 0.45 for single-hop/open-form routes and 0.55 otherwise. Time/duration, list, shared-entity, and open-form questions also have deterministic type guards. The reported configuration permits at most three retrieval-query formulations in total, including the original question. If the gate fails and capacity remains, typed formulations are added within that same cap; raw fallback retains at most eight windows of at most four turns each. The same answer generator is used afterward. A JSON-only planner review is invoked only when no rule path is available. Every expansion, fallback, and review decision is recorded in the retrieval trace.

Read-path complexity. Let $P = |\mathcal{P}(q)|$, let U_0 be the number of distinct path/vector candidates before truncation, let $K_0 = \min(U_0, 15)$ be the retained seed count, and let d_H be the number of graph edges visited within two hops before the $K_G = 20$ output cap. Excluding backend embedding/vector-query cost C_{vec}, candidate collection costs $O(\sum_{(v,p)\in\mathcal{P}(q)}(|p| + |I_v(p)|) + d_H)$, initial seed ordering costs $O(U_0 \log U_0)$, and final graph-augmented ordering costs $O((K_0 + K_G)\log(K_0 + K_G))$. The limits $K_0 \le 15$ and $K_G \le 20$ bound each configured ranking stage; the reported cap of three total formulations produces at most three memory-retrieval passes, including the initial query, and raw fallback is bounded by eight four-turn windows. The resource profile in Sect. 6.8 measures the remaining model and orchestration cost.

4.7 Evidence-constrained answer generation

The final evidence block contains each selected memory identifier, normalized content, state, validity interval, provenance, and retrieval score. The generator is instructed to answer from this block and to preserve temporal and state qualifications. A lightweight consistency check compares answer entities, times, and claimed states against the evidence. When they cannot be aligned, the system performs at most one bounded regeneration using the inconsistent fields as feedback, thereby guarding against unsupported claims after evidence selection.

The resulting read path exposes an end-to-end inspection trace: the answer points to evidence cells, while the cell and write trace retain source, relation, operation, lifecycle transition, and version metadata. Sect. 6.1 evaluates the corresponding endpoint and reference-consistency checks.

5 Evaluation Design

5.1 Evaluation questions and data

The evaluation follows the three stated questions. *RQ1* is answered by the record model, lifecycle contract,

and running example in Sects. 3–4. *RQ2* uses finite-domain checks, snapshots, and natural-language update diagnosis to separate declared invariant preservation from upstream interpretation. *RQ3* uses LoCoMo [16], strict pairs, LongMemEval-S [27], interventions, diagnostics, and cost records to characterize budget-dependent quality–cost trade-offs. These are distinct evidence layers; Sect. 7 defines their transfer envelope.

5.2 Experimental protocol

The reported baselines are BM25 RAG, Dense RAG, Hybrid RAG, MemoryBank, A-MEM, Mem0, Mem0g, and Zep. For the retrieval-only experiment, Contriever, BGE-M3, Stella, and BM25+BGE-M3 are additionally reported. LoCoMo is partitioned by complete conversation, not by individual question, into one setup/development conversation, two validation conversations, and seven held-out test conversations. No benchmark conversation is used to fit model weights. The setup partition contains `conv-26` (152 non-adversarial questions), validation contains `conv-43` and `conv-48` (358), and test contains `conv-30`, `conv-41`, `conv-42`, `conv-44`, `conv-47`, `conv-49`, and `conv-50` (1,019). All questions, turns, and evidence belonging to a conversation remain in the same split. The test split comprises 198 single-hop, 216 multi-hop, 546 temporal, and 59 questions in LoCoMo's original *open-domain* category (called *open-form* here); no test-question subsampling is used. The retrieval experiment uses the benchmark-distributed file named `longmemeval_s_cleaned.json`; every method is run on the same hashed file.

The dialogue-level split script uses seed 42 for the setup, validation, and seven-conversation test allocation listed above; exact identifiers and hashes are retained in the split manifest. The path-intervention archive contains the full configuration and eight one-switch variants under seeds 42, 43, and 44, each evaluated on the same 1,019 questions. These seeds initialize controllable adapter ordering and tie handling, while temperature-zero generation minimizes decoding variation. For intervention uncertainty, we average seeds within question, aggregate paired Full-minus-variant F1 by conversation, resample all seven conversations, enumerate the 2^7 paired sign patterns, and apply Holm correction across eight contrasts. Archived run arguments fix the caps at 15 candidates, three retrieval queries, eight raw windows, four turns per window, and 64 answer tokens.

All methods use the same `gpt-4o-mini` answer generator, temperature-zero decoding, 64-token maximum answer length, conversation split, and matched reading caps: at most 15 candidates, three retrieval queries, eight raw-source windows, and four turns per raw window. The dense backend uses `text-embedding-3-large` with 3,072 dimensions, cosine similarity, and Qdrant storage. BM25, Dense, and Hybrid RAG are raw-dialogue controls; MemoryBank, A-MEM, Mem0, Mem0g, and Zep are memory-system controls; and AdaKVT-Mem combines stateful cells with provenance-linked raw evidence. The baseline registry distinguishes official components from local evaluation adapters and records their configurations. These caps equalize the exposed read budget, while each system retains its native construction and internal retrieval path; calls and processed tokens are measured rather than forced equal. Test answers, labels, and gold evidence are excluded from memory construction, routing, retrieval, reranking, and response generation. Online Resource 1 provides the split manifest, configurations, result manifests, and run-to-table mapping.

Parameter governance. Decision gates, scoring vectors, and resource caps are selected on the two-conversation validation partition and frozen before held-out evaluation. Online Resource 1, Table S2 records every reported value, its validation/design/cap origin, role, and archived-argument digest. Fixed heuristic increments encode monotone priorities rather than fitted probabilities. Held-out test answers and labels are excluded from parameter selection, providing a clear separation between configuration and evaluation evidence.

For AdaKVT-Mem, the retriever returns up to 15 memory candidates, uses at most three retrieval-query formulations, and maintains an eight-item raw-evidence shortlist with four-turn windows. Graph-expanded candidates compete within the same final evidence budget rather than enlarging the answer context.

LongMemEval-S reports pre-generation Recall@1/5/10, MRR, graded evidence-span nDCG@10, and Complete Evidence@10; LoCoMo evaluates answers. Of 500 LongMemEval-S questions, 30 abstention items lack annotated gold evidence and are excluded uniformly, leaving 470. Methods share raw sessions and gold identifiers, with answer regeneration and raw fallback disabled. Native units are expanded through provenance to source-turn, unique-session, and approximate token budgets. Other diagnostics cover semantic and temporal answers, candidate admissibility, latency, tokens, and calls.

Strict budget-controlled read-path comparison. We evaluate the two memory backends under two paired protocols on the same 1,019 held-out LoCoMo questions and seeds 42, 43, and 44. The equal-token pair shares

routing, retrieval, prompting, decoding, and priority truncation; raw fallback is disabled. Its 3,057 stored paired rows have identical serialized-input counts (620–696; mean 658.19). The equal-call pair shares the raw-evidence policy and 2,048-token rule, and fixes three retrieval calls per question; the paired rows also have the same observed raw-window counts. The memory backend is the only allowed configuration difference. Online Resource 1 provides the configurations and question-level validators.

Human-annotated update-semantics evaluation. To test the write path before answer generation, we construct a double-annotated set of 78 natural knowledge-update examples from LongMemEval-S. Each example contains the candidate universe, predecessor and successor identifiers, relation, operation, lifecycle transition, valid-time interval, provenance links, and a before/after value pair. Two independent annotators label the records and a third-party adjudication produces the gold data; relation, operation, and lifecycle agreement are all reported with Cohen's κ. The main evaluation pools the 78 original examples over five reporting partitions without fold-specific training or tuning. Controlled augmentations (54 examples) and 660 derived hard negatives are diagnostic-only and are never mixed with the natural main score. We compare AdaKVT-Mem with ablations that remove relation-driven operation mapping, lifecycle/valid-time handling, or the state machine, and with matched Mem0g and raw-context baselines. The operational predecessor resolver retains at most 10 candidates; Recall@20 is reported only as an offline ranking diagnostic and is not an online system setting.

5.3 Metric definitions and level of inference

For LoCoMo, per-question token F1 lower-cases and tokenizes alphanumeric/CJK spans, with equivalent absolute English date surfaces canonicalized without inventing missing fields; BLEU-1 uses the same tokenizer. Category-macro weights the four task types equally, whereas question-weighted F1 is the mean over the observed 198/216/546/59 single-hop, multi-hop, temporal, and open-form questions. LLM-Judge is a method-blind, temperature-zero semantic diagnostic using `Deep-Seek-V4-Pro`, separate from the `gpt-4o-mini` answer generator; F1 and BLEU-1 remain primary.

Retrieval Recall@K counts any provenance-identifier hit, Complete Evidence@K requires all annotated identifiers, and MRR uses the first matched normalized unit. nDCG@10 uses five distinct evidence spans with gains $(3, 2, 2, 1, 1)$. Turn/session duplicates retain their first raw rank and unioned provenance, while token budgets preserve raw order. Re-evaluation found zero discrepancies across 16,500 stored traces and 15,510 eligible metric records after abstention exclusion. Invalid Evidence Rate is measured on the post-retrieval, pre-generation trace; zero final-context invalid evidence follows from the admission filter and is not treated as a quality gain. Temporal Answer Accuracy applies guarded exact containment only to temporal questions. Online Resource 1 gives the complete formulas, normalization rules, provenance budgets, prompts, and boundary cases.

Level of inference. The seven held-out conversations are independent clusters and the 1,019 questions are repeated measurements. We average repeated seeds within question, retain conversation-level scores, use 20,000 whole-cluster bootstrap replicates, and enumerate all $2^7 = 128$ paired sign patterns. Effect, interval, cluster wins, exact test, and Holm family are reported together; with seven clusters, they characterize this held-out set rather than establish population-level superiority. Online Resource 1, Tables S12–S14 provide the complete records.

Online Resource 1, Table S1 consolidates the comparison controls and reproducibility records, while the main text foregrounds the measured outcomes.

6 Results Organized by Research Question

6.1 RQ2: Protocol correctness after interpretation

We test the lifecycle policy at two levels without LLM or embedding calls. First, a dependency-free implementation of Eq. 16 exhaustively enumerates the declared finite input domain, $17 \times 8 \times 4 \times 2 \times 3 \times 2 = 6{,}528$ tuples: 17 predecessor configurations (including no predecessor), eight operations, four relation cases, two authorization values, three conflict-winner values, and two saved-eligibility values. It produces 1,960 accepted and 4,568 rejected tuples in 23 deterministic outcome classes and zero violations of six transition-level assertions: side-effect-free rejection, archive access suppression, restore/state orthogonality, exactly one active high-conflict winner, no reinforcement reactivation, and predecessor retention for content-changing operations. The executable specification, assertion mapping, and complete result record are included in Online Resource 1.

Second, we evaluate a stored snapshot containing five controlled update behaviors and nine cells.

The cases cover insertion, temporal replacement, same-interval contradiction, uncertainty preservation, and expiry. All five endpoint conditions hold. Recursive traversal reports zero orphan MHI references, missing declared paths, unresolved cell/version links, or graph edges with missing endpoints, and all nine cells retain the core transition tuple and source identifier. These results establish structural conformance for the declared post-interpretation inputs; the following human-annotated evaluation examines the upstream decisions that propose them.

6.2 RQ2: Natural-language update diagnosis

The two annotators agree strongly on the 78 original examples: relation $\kappa = 0.943$, operation $\kappa = 0.944$, and lifecycle $\kappa = 0.934$; candidate-set Jaccard agreement is 0.983. About 30.8% of examples require adjudication, yielding an independently reconciled gold set. Table 2 reports the prespecified task-layer subtype Macro-F1: 78.06 for relation and operation, 98.72% lifecycle accuracy, and 70.51% valid-time exact match. Online Resource 1, Table S17 separately reports 91.86/92.05 after a deterministic projection of these same predictions to the native protocol relation/operation labels; it is neither a second trial nor an alternative primary result. The stage-wise profile is 62.91 candidate F1, 36.11% predecessor Recall@1, 88.89% Recall@5, 7.69% value-exact completeness, and 1.28% Complete (95% CI [0.00, 3.85]). Lifecycle scores one field; Complete jointly requires candidate, predecessor, relation, operation, time, provenance, values, and final state. The two values therefore identify upstream extraction, target ranking, and value reconstruction—not lifecycle transition alone—as the natural-language bottlenecks.

A post-evaluation annotation and label-schema audit retained the original snapshot and regenerated the current tables. Because prediction-blind status was not contemporaneously recorded, an independent reviewer subsequently re-reviewed the two changed core-record value pairs using only source evidence and a frozen guideline; both were supported as high-confidence replacements (3→4 and \$350,000→\$400,000), and the blind-review package passed its complete hash check. Both snapshots remain disclosed: relation Macro-F1 changes from 77.51 to 78.06, lifecycle accuracy from 92.31% to 98.72%, value-exact completeness from 5/78 to 6/78, and complete success from 0/78 to 1/78; temporal exact match remains 70.51%. Online Resource 1 machine-checks both records and preserves the review protocol and outcome.

An ordered per-sample cascade records 48 candidate-set misses, 12 rank-1 predecessor misses, 17 exact-value-pair misses after all other recorded checks, and one complete record. Given an exact candidate set and rank-1 predecessor, all 18 retained records have correct relation, operation, lifecycle, time, and provenance; only 1/18 has an exact value pair. Online Resource 1 gives the full conditional account.

The ablations locate dependencies within the fixed task-layer evaluation. Replacing relation-driven operation mapping with a generic operation policy lowers operation Macro-F1 from 78.06 to 30.67 while leaving the relation decision unchanged. Removing valid-time-aware state handling lowers valid-time exact match to 6.41% while retaining the separately predicted lifecycle label. Removing finite-state transition enforcement reduces operation Macro-F1 to 13.85 and lifecycle accuracy to zero. The matched Mem0g and raw-context baselines obtain 59.56 relation/operation Macro-F1 and 92.31% lifecycle accuracy, with zero complete updates. The pattern localizes configuration-level differences to operation mapping, state transitions, and temporal normalization.

Conditional protocol result. Among the 18 examples with both an exact candidate set and rank-1 predecessor, relation, operation, lifecycle, valid time, and provenance are all correct; only 1/18 has an exact before/after value pair. Complete-update success is therefore 1/78 under its joint criterion. Online Resource 1, Table S19 gives class support and outcomes; the singleton CORRECTION class supports no category-level conclusion.

6.3 RQ3: Retrieval and answer behavior

Online Resource 1, Table S20 reports a configuration-level engineering comparison of complete pipelines on held-out LoCoMo records. AdaKVT-Mem's observed category-macro and question-weighted F1 are 50.00 and 50.09, respectively; Mem0g records 45.13 and 46.85, while Hybrid RAG records 29.20 and 30.42 as a raw-dialogue control under the same split, answer model, and final evidence allowance. Because each system retains its native construction and internal retrieval, these are integrated operating points rather than isolated causal estimates of the KVT protocol; the strict controls below provide the mechanism-focused two-backend comparison.

To avoid implying conventional external-knowledge QA, we refer to the benchmark's *open-domain* label as *open-form*: it denotes a question without an explicit personal-time constraint, yet it can still require evidence from the user's dialogue history. For example, "Which café do I usually visit?" is open-form: it has no

Table 2 Human-annotated update-semantics evaluation on 78 natural LongMemEval-S examples. Relation and operation are Macro-F1; lifecycle and valid time are single-field exact accuracies; Complete is a strict joint accuracy requiring all decisions, links, values, time, provenance, and final state to be exact. All non-Complete scores are percentages. Complete reports successful records/78 (thus 1/78=1.28%); 0/78 denotes an evaluated zero rather than an unavailable metric.

System	Candidate F1	Relation	Operation	Lifecycle	Valid time	Complete (n/78)
AdaKVT-Mem	62.91	**78.06**	**78.06**	**98.72**	**70.51**	1/78
w/o relation→operation mapping	62.91	78.06	30.67	98.72	70.51	0/78
w/o valid-time handling	64.24	78.06	78.06	98.72	6.41	0/78
w/o FSM enforcement	65.38	59.56	13.85	0.00	7.69	0/78
Mem0g matched	66.03	59.56	59.56	92.31	3.85	0/78
Raw context matched	16.03	59.56	59.56	92.31	0.00	0/78

date but can depend entirely on personal history. The descriptive category values differ across configurations (Online Resource 1, Table S20).

The observed category profile is relatively even within this configuration, with particularly strong single-hop and multi-hop values. Temporal and open-form results motivate duration-sensitive extraction and memory-optional routing; released traces expose expansion depth, raw-window use, and answer utilization. Conversation-cluster inference in the next subsection quantifies the uncertainty of the observed aggregate ordering.

6.4 RQ3: Budget-dependent trade-off

We recomputed overall per-question F1 separately for each held-out conversation. AdaKVT-Mem has the higher observed score in five of seven clusters, with a conversation-mean descriptive difference of 3.72 points. The question-count-weighted difference is 3.24 points, with a 95% conversation-cluster bootstrap interval of -1.03 to 7.21; the exact sign-flip value is 0.15625. In a deletion sensitivity analysis–not leave-one-out training or cross-validation–removing each test conversation in turn preserves rank 1, with six-conversation F1 ranging from 49.69 to 50.35. Online Resource 1, Tables S12–S13 report all eight F1 contrasts and every omission result.

6.5 Strict paired read-path comparison

Table 3 isolates the two memory backends under identical exposed read paths. Under the 696-token protocol, the mean F1 values are nearly identical: 51.74 for AdaKVT-Mem and 51.69 for Mem0g. Averaging three seeds within question and then forming seven dialogue-cluster pairs gives +0.35 F1 points (95% cluster-bootstrap CI [-3.39, 3.70]; exact two-sided sign-flip $p = 0.859375$; 4/7 cluster wins). Within a seed, paired rows have equal input-token counts, but serialization can differ across seeds. Seed-level question-mean differences are +4.93, -2.16, and -2.65 (42, 43, 44); leave-one-seed means are -2.40, +1.14, and +1.39. The mixed signs support parity, not a stable directional advantage. Candidates/ranks are fixed, but missing payload hashes and provider IDs leave payload construction versus provider execution—including external model revisions despite temperature zero—unresolved (Table S21). Under the equal-call protocol, both systems make three retrieval calls per question and receive the same raw-window fallback and 2,048-token truncation policy; the paired difference is -2.88 points ([-6.59, 0.37]; $p = 0.21875$; 2/7 wins), with Mem0g's run mean higher (55.09 versus 51.90). The controls reveal parity under a tight token budget and a Mem0g advantage with matched raw fallback.

6.6 RQ3: Execution and input diagnostics

The full configuration in the three-seed intervention archive yields question-weighted F1 values of 50.09, 49.81, and 50.37, or 50.09 ± 0.28 (mean ± sample standard deviation). This run-level spread is reported separately from the preceding conversation-cluster interval, which addresses variation across independent dialogues. The archive contains all 3,057 Full-run question records and validates their aggregate F1, BLEU-1, and Contains values against the run matrix.

The diagnostic protocol reports four conditions separately with 2,000-replicate conversation-cluster bootstrap intervals. AdaKVT-Mem records 48.68 F1 [47.82, 49.86] for deterministic query paraphrasing, 49.52 [48.65, 50.74] under cross-conversation memory noise, 47.44 [46.34, 48.33] on 448 questions containing explicit temporal expressions, and 47.09 [44.74, 49.26] on the 216 native multi-hop questions with bridge distractors. These records are diagnostic stress tests rather than a general robustness claim: semantic preservation has deterministic proxy checks, and cross-conversation noise

Table 3 Strict paired LoCoMo read-path comparison on 1,019 held-out questions in seven dialogue clusters. F1 is three-seed mean ± sample SD. Δ is AdaKVT-Mem minus Mem0g after averaging seeds within question and then pairing whole conversations; its interval is a 20,000-replicate cluster bootstrap and p is the exact two-sided sign-flip result. Only the memory backend differs within each protocol.

Protocol	Shared constraint	AdaKVT-Mem	Mem0g	ΔF1 [95% CI]; p
Equal token	696 input tokens; no raw fallback	51.74 ± 0.04	51.69 ± 4.20	+0.35 [-3.39, 3.70]; .859375
Equal call	3 retrieval calls; raw window 4	51.90 ± 1.57	55.09 ± 0.24	−2.88 [-6.59, 0.37]; .218750

establishes a controlled baseline for future same-entity and stale-version hard negatives. Formula-extrapolated long-context scores are excluded from manuscript performance evidence. Online Resource 1, Tables S5 and S10 specify the included records and boundaries.

6.7 RQ3: Retrieval interventions and evidence access

The intervention study uses the same answer generator, 1,019-question split, matched read caps, and evaluation code for all variants. It comprises 27 runs: the Full configuration and eight one-switch variants under seeds 42, 43, and 44. The Full mean is 50.09 F1. Disabling lightweight-graph retrieval, relation-driven update, or the generation evidence constraint produces the three largest descriptive Full-minus-variant differences: 6.26 [6.05, 6.48], 6.12 [5.87, 6.33], and 5.89 [5.62, 6.17] points. After averaging seeds within question, every removal is lower than Full in all seven conversations. Online Resource 1, Table S3 reports all variants, conversation-cluster intervals, and tests; all eight raw exact sign-flip values are 0.015625, and the family-wise Holm-adjusted value is 0.125. The ordering is therefore an effect ranking for this configuration, not eight confirmed population-level causal effects.

On the 470 evidence-bearing LongMemEval-S questions, AdaKVT-Mem reaches native-unit values of 55.87 Recall@1, 62.58 MRR, and 59.09 Complete Evidence@10. Under common source-turn budgets it records 55.74 Recall@1, 62.15 MRR, and 58.51 Complete Evidence@10; source-session values are 56.60, 63.36, and 59.36, and source-token values are 56.17, 62.78, and 58.94. These establish gold-source coverage, not provenance precision or answer-to-evidence entailment; here *auditable* means inspectable source-linked traces and cross-layer consistency. Online Resource 1, Table S4 reports every method, unit, budget, and metric.

6.8 RQ3: Prototype operating cost

On the common 159-question resource subset, AdaKVT-Mem records 124.4 s mean and 164.2 s P95 latency, 10.4 calls, approximately 84.3k tokens, and a dated estimate of $0.0131 per question. Relative to Mem0g, this is 30.5% lower mean latency and 86.5% fewer calls, but 56.0% more tokens and 36.6% higher recorded cost. Totals cover provider input/output calls for candidate extraction/normalization, relation review, query planning/rewrite, raw fallback, and answer/consistency generation. Because the stored profile has no stage-tagged token or latency fields, it cannot divide the 84.3k total between these stages or separate architecture from prompt/orchestration cost. Online Resource 1, Table S11 uses $C = (T_{\text{in}}p_{\text{in}} + T_{\text{out}}p_{\text{out}})/10^6$, where T is provider-token count and p is the dated USD price per million input (in) or output (out) tokens. These measurements characterize the JSON reference backend and identify evidence compression and indexed persistence as optimization targets.

7 Architecture Scope and Empirical Trade-offs

7.1 Evidence scope and claim boundary

The evidence has a deliberate hierarchy: finite-domain transitions and snapshots test the post-interpretation lifecycle/reference contract; double-annotated updates test upstream structured decisions and joint reconstruction; LoCoMo, strict pairs, interventions, and LongMemEval characterize retrieval and answer behaviour under stated budgets; and cost records locate the JSON implementation's operating point. Thus protocol checks do not validate language interpretation, complete-pipeline comparisons do not isolate causal components, stress tests do not establish general robustness, and gold-source coverage does not establish citation entailment. Throughout, a *configuration-level difference* is an observed integrated-system contrast, a *descriptive difference* makes no population claim, and an *effect ranking* orders one-switch configurations without asserting independent causal effects.

Five successful traces cover relative-time resolution, lifecycle-filtered preference retrieval, conflict retention, two-hop traversal, and open-form evidence use. Among 233 questions with F1 below 0.20, the exclusive fail-

ure census assigns 136 to retrieval/extraction miss, 35 to structured-memory miss despite raw evidence, 39 to partial evidence, and 23 to answer utilization. Online Resource 1 provides definitions, three traced examples, and a seeded 100-case reliability sample. The resulting priorities are duration-sensitive retrieval, temporal grouping, and validation of entities, times, and numbers.

7.2 Privacy and user control

Long-term memory creates privacy and user-control obligations. Users should be able to inspect, correct, archive, and remove stored memories [10]. AdaKVT-Mem implements logical archival and tests reference suppression in controlled snapshots; a production hard-deletion cascade over payloads, embeddings, indexes, edges, raw sources, and caches remains a deployment requirement rather than an evaluated result. The write path treats dialogue and retrieved text as untrusted: extraction cannot authorize archival, restoration, destructive conflict resolution, or source-identity changes. Production deployment additionally requires authenticated ownership, tenant isolation, sensitive-attribute policy gates, and prompt-injection quarantine.

7.3 Reproducibility and comparison boundary

All Primary rows use the declared split, answer generator, 64-token output limit, and matched read caps. Online Resource 1, Table S1 identifies the compact table-input package and separate full-record archive; manifests and validators check every distributed path. The strict-pair artifact retains the 12 three-seed records, shared configurations, and question-level call/token compliance; the update artifact retains annotations, adjudicated gold, predictions, write traces, prompts, and hashes. Its README distinguishes aggregate consistency, stored-prediction recomputation, and end-to-end execution with external providers. Table S20 characterizes integrated pipelines, while Table 3 supplies the paired backend control. Test answers and labels are excluded from construction, routing, threshold selection, and prompts.

7.4 Evaluation scope and transfer considerations

The evaluation uses seven held-out dialogue clusters, one shared generator, two strictly paired backends, and a 159-question resource profile; transfer is limited to the tested models, dialogue distribution, and JSON backend. It establishes equal-token parity with Mem0g and a matched-raw-fallback trade-off, while identifying evidence compression and indexed persistence as cost targets.

Finite-domain checks cover 6,528 staged inputs and five snapshots; natural updates identify candidate access and exact before/after value recovery as the main reconstruction bottlenecks. Concurrent writers, recovery, hard deletion, evidence precision, citation entailment, and live scale growth remain production-service requirements.

8 Conclusion

We presented AdaKVT-Mem as an inspectable maintenance protocol for LLM-interpreted conversational records. A relation decision synchronizes a cell's validity, lifecycle/storage state, version, provenance, keyed placements, typed relations, and operation trace through one executable cross-layer contract. Finite-domain and snapshot checks verify the contract for interpreted inputs; update-semantics evaluation measures the separate reconstruction bottleneck; and RQ3 characterizes budget-dependent quality–cost trade-offs. The result is a reference implementation for manageable and checkable records, not a claim of reliable end-to-end reconstruction or universal system superiority.

Statements and Declarations

Funding. The authors did not receive support from any organization for the submitted work.

Competing interests. The authors have no relevant financial or non-financial interests to disclose.

Author contributions and AI-assisted tools. Both authors contributed to study conception and design, read and approved the final manuscript, and take responsibility for it. AI-assisted tools were used for language/formatting edits and code analysis/debugging; the authors reviewed and verified all resulting text and code.

Data and code availability. Benchmark datasets are available from their original publications. Online Resource 1 contains Tables S1–S20, Algorithm S1, table inputs, derived statistics, evaluation records, validation code, and a package manifest. A separate full-record archive contains the available code and per-question

records; provider credentials, unavailable provider snapshots, and redistributable benchmark source copies are excluded.

References

1. Alchourrón CE, Gärdenfors P, Makinson D (1985) On the logic of theory change: Partial meet contraction and revision functions. The Journal of Symbolic Logic 50(2):510–530, https://doi.org/10.2307/2274239
2. Bai Y, Lv X, Zhang J, Lyu H, Tang J, Huang Z, Du Z, Liu X, Zeng A, Hou L, Dong Y, Tang J, Li J (2024) Longbench: A bilingual, multitask benchmark for long context understanding. In: Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics
3. Banerjee P, Moshtaghi M, Subramanian S, Misra A, Chadha A (2026) Apex-mem: Agentic semi-structured memory with temporal reasoning for long-term conversational ai. In: Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics, pp 16470–16489, https://doi.org/10.18653/v1/2026.acl-long.749
4. Buneman P, Khanna S, Tan WC (2001) Why and where: A characterization of data provenance. In: Database Theory—ICDT 2001, Springer, Lecture Notes in Computer Science, vol 1973, pp 316–330, https://doi.org/10.1007/3-540-44503-X_20
5. Chhikara P, Khant D, Aryan S, Singh T, Yadav D (2025) Mem0: Building production-ready ai agents with scalable long-term memory. arXiv preprint arXiv:250419413
6. Doyle J (1979) A truth maintenance system. Artificial Intelligence 12(3):231–272, https://doi.org/10.1016/0004-3702(79)90008-0
7. Elmagarmid AK, Ipeirotis PG, Verykios VS (2007) Duplicate record detection: A survey. IEEE Transactions on Knowledge and Data Engineering 19(1):1–16, https://doi.org/10.1109/TKDE.2007.250581
8. Gupta A, Mumick IS (1995) Maintenance of materialized views: Problems, techniques, and applications. IEEE Data Engineering Bulletin 18(2):3–18
9. Gutiérrez BJ, Shu Y, Gu Y, Yasunaga M, Su Y (2024) Hipporag: Neurobiologically inspired long-term memory for large language models. In: Advances in Neural Information Processing Systems, vol 37
10. Huang Z, Gutierrez S, Kamana H, MacNeil S (2023) Memory sandbox: Transparent and interactive memory management for conversational agents. In: Companion Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology, pp 97:1–97:3
11. Izacard G, Caron M, Hosseini L, Riedel S, Bojanowski P, Joulin A, Grave E (2022) Unsupervised dense information retrieval with contrastive learning. Transactions on Machine Learning Research
12. Karpukhin V, Oguz B, Min S, Lewis P, Wu L, Edunov S, Chen D, Yih Wt (2020) Dense passage retrieval for open-domain question answering. In: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, pp 6769–6781
13. Latimer C, Boschi N, Neeser A, Bartholomew C, Srivastava G, Wang X, Ramakrishnan N (2025) Hindsight is 20/20: Building agent memory that retains, recalls, and reflects. arXiv preprint arXiv:251212818
14. Lewis P, Perez E, Piktus A, et al. (2020) Retrieval-augmented generation for knowledge-intensive nlp tasks. In: NeurIPS
15. Liu NF, Lin K, Hewitt J, et al. (2024) Lost in the middle: How language models use long contexts. Transactions of the Association for Computational Linguistics 12:157–173
16. Maharana A, Lee DH, Tulyakov S, Bansal M, Barbieri F, Fang Y (2024) Evaluating very long-term conversational memory of llm agents. In: ACL
17. Packer C, Wooders S, Lin K, et al. (2023) Memgpt: Towards llms as operating systems. arXiv preprint arXiv:231008560
18. Pan Z, Wu Q, Jiang H, Xia M, Luo X, Zhang J, Lin Q, Rühle V, Yang Y, Lin CY, Zhao HV, Qiu L, Zhang D (2024) Llmlingua-2: Data distillation for efficient and faithful task-agnostic prompt compression. In: Findings of the Association for Computational Linguistics: ACL 2024, pp 963–981
19. Pan Z, Wu Q, Jiang H, et al. (2025) Secom: On memory construction and retrieval for personalized conversational agents. In: ICLR
20. Park JS, O'Brien JC, Cai CJ, Morris MR, Liang P, Bernstein MS (2023) Generative agents: Interactive simulacra of human behavior. In: Proceedings of UIST
21. Rasmussen P, Paliychuk P, Beauvais T, et al. (2025) Zep: A temporal knowledge graph architecture for agent memory. arXiv preprint arXiv:250113956
22. Sarthi P, Abdullah S, Tuli A, et al. (2024) Raptor: Recursive abstractive processing for tree-organized retrieval. In: ICLR
23. Snodgrass RT (1999) Developing Time-Oriented Database Applications in SQL. Morgan Kaufmann
24. Su M, Guo Y, Hou Z, Bai L, Li Z, Zhang Y, Yin G, Lin W, Jin X, Guo J, Cheng X (2026) Beyond dialogue time: Temporal semantic memory for personalized llm agents. In: Findings of the Association for Computational Linguistics: ACL 2026, pp 29935–29951, https://doi.org/10.18653/v1/2026.findings-acl.1496
25. Sun H, Zeng S, Zhang B (2026) H-mem: Hierarchical memory for high-efficiency long-term reasoning in llm agents. In: EACL
26. Sun H, Zhang Z, Zeng S (2026) Preference-aware memory update for long-term llm agents. In: Findings of ACL
27. Wu D, Wang H, Yu W, Zhang Y, Chang KW, Yu D (2025) Longmemeval: Benchmarking chat assistants on long-term interactive memory. In: ICLR
28. Wu Y, Rabe MN, Hutchins D, Szegedy C (2022) Memorizing transformers. In: The Tenth International Conference on Learning Representations
29. Xu W, Liang Z, Mei K, et al. (2025) A-mem: Agentic memory for llm agents. arXiv preprint arXiv:250212110
30. Yan S, Ni J, Zheng L, Zhang J, Wu P, Yin D, Lyu J, Yuan C, Rao F (2026) Adamem: Adaptive user-centric memory for long-horizon dialogue agents. arXiv preprint arXiv:260316496 URL `https://arxiv.org/abs/2603.16496`
31. Yuan R, Sun S, Li Y, et al. (2025) Personalized large language model assistant with evolving conditional memory. In: COLING
32. Zhong W, Guo L, Gao Q, Ye H, Wang Y (2024) Memorybank: Enhancing large language models with long-term memory. In: Proceedings of AAAI

Supplementary Files

This is a list of supplementary files associated with this preprint. Click to download.

- ESMAdaKVTMem.pdf