# COGNIFOLD: ALWAYS-ON PROACTIVE MEMORY VIA COGNITIVE FOLDING

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Agents receive fragmented observations over time, creating a need to organize experience into reusable memory. Existing agent memory remains predominantly reactive and retrieval-based, lacking the capacity to autonomously organize events into persistent cognitive structure. Toward genuinely autonomous agents, we introduce CogniFold, a proactive agent memory that continuously folds fragmented event streams into persistent structures, bootstrapping progressively higher-level cognition from incoming events and accumulated knowledge. CogniFold organizes memory into source events, reusable concepts, and candidate intents. As the stream arrives, cognitive structures are proactively assembled, interpreted through accumulated structure, merged when semantically similar, and reweighted by recency, while candidate intents surface from converging concept and event evidence. We evaluate structure formation with CogEval-Bench, a benchmark whose ground truth is established by construction, and show that CogniFold uniquely produces event-grounded concepts that match the reference structure together with supported candidate intents. CogniFold also performs robustly on conventional long-term memory benchmarks (LoCoMo, LongMemEval).

## 1 INTRODUCTION

Memory-Augmented Agents (Packer et al., 2023; Sumers et al., 2024) have empowered Large Language Models (LLMs) to transcend finite context constraints, enabling long-horizon reasoning (Shinn et al., 2023), context-grounded personalization (Chhikara et al., 2025), and experience-driven continual learning (Majumder et al., 2023). However, as agents evolve from on-demand systems into always-on assistants, their input paradigm shifts from bounded, prompt-driven inputs to continuously arriving, fragmented event streams (Zacks, 2020; Kurby & Zacks, 2008). This raises an increasing demand for **proactive** behavior: an assistant that self-organizes structure and surfaces intents before the user issues a query.

Yet, existing memory architectures share a common limit: their topology is fixed once formed. Whether leveraging static knowledge graphs (Gutiérrez et al., 2024; 2025), text-level rewrites (Chhikara et al., 2025), hybrid decoupling (Jiang et al., 2026), or temporal tracking (Rasmussen et al., 2025), memory remains a graph-as-product—a finished artifact to retrieve from, never a substrate that metabolizes under the stream. Consequently, agents are forced to graft proactivity on top as application-layer machinery, such as scheduled triggers, planning loops (Wang et al., 2023; Yang et al., 2024), or periodic reflection (Shinn et al., 2023; Xu et al., 2025). This separation creates a structural ceiling: intents can only arise from sources the application layer was explicitly designed to handle. We argue that proactivity must instead be a property of the memory substrate—intents should emerge from the topology accumulating the conditions for them.

We introduce CogniFold, a proactive always-on agent memory that folds continuously arriving event streams into self-emerging cognitive structure. CogniFold organizes memory into a base graph with three layers of nodes: *events*, which retain source observations; *concepts*, which organize recurring facts and patterns; and *intents*, which record prospective proposals linked to evidence. The graph bootstraps in a strict sense: current state is the interpretive context for the next event, which in turn modifies the state for future events—a self-referential loop in which the system organizes input entirely through its accumulated structure. To enable proactivity, the **intent layer** is part of the same graph: rather than being hardcoded (Bratman, 1987), candidate intents emerge from



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.

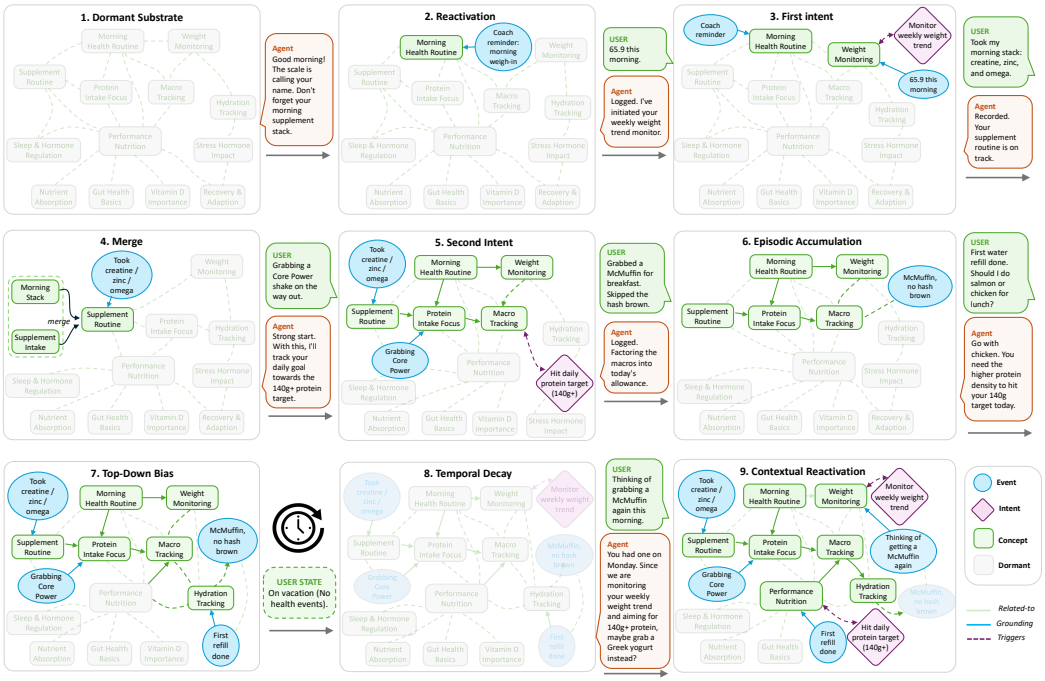


Figure 1: Continuous cognitive folding. Under an asynchronous event stream, the memory substrate dynamically self-organizes. The graph autonomously consolidates episodic events (Panel 3), merges associated schemata (Panel 4), and crystallizes goal-directed intents from converging concept density (Panel 5). This living topology natively supports top-down cognitive bias (Panel 7), natural temporal decay (Panel 8), and structure-driven proactive intervention (Panel 9).

converging concept and event evidence during ingestion. The graph recursively scaffolds higher-level cognition from accumulated structure—a transparent, auditable form of test-time learning, distinct from both surface-level text rewriting (e.g., A-Mem (Xu et al., 2025)) and opaque gradient updates (e.g., Titans (Behrouz et al., 2024)).

We conduct a two-layer evaluation. At the **conventional long-term memory** layer, we evaluate on LoCoMo and LongMemEval-S, confirming that CogniFold simultaneously performs competitively on conventional memory tasks. At the **structural** layer, we introduce CogEval-Bench, a first-principles evaluation framework that directly measures whether the topology formed under continuous event streams matches cognitive expectations, demonstrating that CogniFold uniquely produces event-grounded concepts, coherent conceptual structure, and proactive intent emergence.

Our contributions are summarized as follows:

- **Always-On Proactive Memory Paradigm.** We recast agent memory from a reactive retrieval target into an always-on cognitive substrate (Fig. 1), natively supporting continuous organizing and proactive anticipation before retrieval, enabling self-emerging intents from accumulated concepts.

- **Continuous Topological Self-Organization.** We fold streaming events into a typed event–concept–intent graph through retrieval-guided, update-first graph operations with explicit provenance, yielding a transparent and auditable form of test-time learning.

- **CogEval-Bench Evaluation Framework.** We introduce a structural benchmark, built by top-down generation with ground truth by construction, that isolates proactive emergence from retrieval accuracy. Alongside established long-term memory benchmarks, we jointly validate CogniFold's effectiveness in both high-level cognitive emergence and conventional memory robustness.

## 2 Related Work

**Graph-based agent memory.** HippoRAG (Gutiérrez et al., 2024; 2025) retrieves from a static knowledge graph via Personalized PageRank, Mem0 (Chhikara et al., 2025) rewrites memory text turn by turn, MAGMA (Jiang et al., 2026) decomposes memory into orthogonal graphs refined by an LLM slow path, and Zep (Rasmussen et al., 2025) tracks bi-temporal fact validity. Their topology grows by accumulation, rewriting, or invalidation, but does not fold redundant fragments or reorganize between events.

**Non-topological test-time learning.** Generative Agents (Park et al., 2023) and MemGPT (Packer et al., 2023) page or search natural-language memories, and A-Mem (Xu et al., 2025) rewrites Zettelkasten-style notes; these remain inspectable but structurally static. Titans (Behrouz et al., 2024) instead updates a neural memory by surprise-driven gradient descent, storing knowledge in opaque weights that cannot be audited or selectively deleted (McCloskey & Cohen, 1989).

**Proactive agent memory.** BDI architectures (Bratman, 1987) derive intentions from hardcoded rules and explicit goals, and prospective-memory and cognitive-agent work (Einstein & McDaniel, 2005; Sumers et al., 2024; Shinn et al., 2023) acknowledges goal-directed organization without anchoring intent formation in the evolving memory structure. CogniFold forms candidate intents in the same graph that organizes events and concepts.

## 3 Continuous Cognitive Folding

An always-on agent requires a fundamentally different memory substrate. Continuously arriving event streams demand an architecture capable of incremental, online integration. CogniFold interprets each new observation using the current memory and incorporates it through an explicit graph update, continuously folding to eliminate redundancy and bootstrapping to climb levels of abstraction, sustaining cognitive agility under an always-on event stream.

### 3.1 Three-Layer Graph Representation

CogniFold represents accumulated experience as a directed multigraph $G_t = (V_t, E_t)$ with three semantic layers. The *event layer* preserves observations with source identifiers, timestamps, and text. The *concept layer* represents recurring facts and patterns, linking them to the events that support them. The *candidate-intent layer* records prospective goals suggested by those observations and concepts. An auxiliary *Time* node type provides temporal anchors. Thus the implementation has four node types, with Time supporting the three representational layers.

CogniFold makes continually updated concepts an explicit, standalone memory layer, giving distinct representations to retaining details (events) and summarizing regularities (concepts). For example, suppose a user says in succession: "Recent releases keep getting blocked by security approval," "Waiting on the security team's sign-off again today," and "For the next release, I want to schedule the security review in advance." The system can retain all three events while maintaining a single concept: *the recurring bottleneck in the release process is security approval, which should be scheduled early*. This concept is not a raw message but an induction over multiple observations. A memory that stores only events must re-retrieve and re-induce to answer "What is the main bottleneck in our release process?"; one that stores only concepts may lose which release it was, who said it, and what actually happened at the time.

Three layers separate evidence, abstraction, and prospective use. A concept can accumulate support and change its description while its source events remain independently retrievable. A candidate intent links to the observations or concepts that motivate it. Typed relations distinguish evidential support, semantic association, and temporal constraints (Appendix A.1). The schema fixes the available node and relation types; the concept identities, evidence assignments, and connections are formed from the incoming stream.

**Algorithm 1** Continuous cognitive folding

**Require:** Event stream $(e_t)_{t=1}^{T}$, writer $f_\theta$, selector $S$, budget $k$, maintenance $\mathcal{M}$, and schedule $b_t$

$G \leftarrow \varnothing$
**for** $t = 1, \ldots, T$ **do**
  Commit $e_t$ verbatim as an immutable Event node in $G$
  $C_t \leftarrow S(G, e_t; k)$ by event-conditioned retrieval or graph-state scoring
  $\Delta_t \leftarrow f_\theta(e_t, C_t)$ with reasoning and provenance
  Resolve identifiers, deduplicate proposed concepts, validate $\Delta_t$, and snapshot $G$
  Apply $\Delta_t$ atomically; restore the snapshot on execution failure
  Preserve source events and repair links from explicit provenance
  **if** $b_t$ is a maintenance boundary **then**
    $G \leftarrow \mathcal{M}(G)$: reinforce, merge, decay, and complete
  **end if**
  Record candidate intents proposed in $\Delta_t$ with links to their supporting evidence
  Update category preferences from new user feedback via EMA
  Expose $G$ for subsequent events and surface newly formed intents
**end for**
**return** $G$

### 3.2 Recurrent Update

Given new event $e_t$, CogniFold interprets each new observation using the current memory and incorporates it through an explicit graph update:

$$C_t = \mathcal{S}(G_{t-1}, e_t; k), \tag{1}$$

$$\Delta_t = f_\theta(e_t, C_t), \tag{2}$$

$$G_t = \mathcal{M}_{b_t}(\mathcal{A}(G_{t-1}, \Delta_t)). \tag{3}$$

Here $\mathcal{S}$ selects at most $k$ records for the write context $C_t$, and the fixed LLM $f_\theta$ proposes a typed `UpdatePlan` $\Delta_t$. The executor $\mathcal{A}$ validates and applies the plan. Enabled maintenance runs through $\mathcal{M}_{b_t}$ at configured session or ingestion boundaries; between boundaries it is the identity.

In the memory evaluations, this recurrence is triggered by an arriving event or batch and takes only that observation and the current graph as input; downstream evaluation questions and answers are withheld. The writer can assemble concepts, attach evidence, and revise existing descriptions. Self-organization refers to the evolving concepts and assignments within the typed schema. Algorithm 1 specifies this instance of memory formation.

### 3.3 Proactive Context Assembly

Context assembly maintains a focused working view of the persistent graph. The event-conditioned selector brings records relevant to the arriving observation into the bounded context $C_t$. The graph-state selector available in the engine instead combines structural importance, access recency, frequency, and deadline urgency; its hierarchical variant separates immediate, working, and background records. Recomputing this view as events and graph state change lets the system shift attention across the accumulated memory. The selectors are alternative implementations of $\mathcal{S}$, with their configuration and scoring rules given in Appendix A.2.

The Full LoCoMo writer uses event-conditioned concept retrieval. For each event batch, it retrieves the 15 most similar existing concepts by embedding similarity and exposes their identifiers and descriptions. Update-first instructions ask the writer to check these candidates before creating another concept. It uses `UPDATE_NODE` to revise a match or reinforces it with additional support; distinct information can introduce a new concept. This retrieval-and-instruction scheme provides the evaluated mechanism for directing new evidence toward existing abstractions.

An `UpdatePlan` records the proposed node and edge operations, explanatory text, and explicit provenance references. The executor resolves identifiers and validates operation schemas and edge constraints. It applies the plan atomically, restoring a snapshot on execution failure. A REINFORCES relation attaches further support to an existing concept, while GROUNDS links preserve the source

evidence for derived records. Duplicate-add suppression can redirect redundant proposals to existing nodes. Full also repairs provenance links and resolves relative dates against session timestamps, allowing descriptions to retain both evidential and temporal context.

For example, a new observation about joining a running group can bring an existing race-preparation concept into context. A fold can update that concept and attach the new event, so later observations encounter the accumulated preparation history. A subsequent event about an unrelated project can select a different part of the graph. This illustrative sequence distinguishes the persistent representation from the working context that changes with current input.

Folding also exposes configurable maintenance operations: semantic merging consolidates redundant concepts, association completion links related records, and recency or relative edge-age weighting affects selection priority. Appendix A.2 specifies their schedules and parameters. LoCoMo ingestion groups consecutive dialogue turns into event batches. Full includes concept retrieval, update-first instructions, duplicate suppression, and grounding/date repair; an earlier configuration C, reported in Appendix D.1, omits these repairs. Semantic merge is disabled in these matched runs. The CogEval reference protocol uses event-conditioned retrieval of up to ten records and maintenance after ingestion. These experiments use their stated event-conditioned selectors rather than the engine's graph-state ranking.

### 3.4 Emergent Structure and Memory Use

The reuse scheme folds repeated observations into shared concept nodes while retaining the underlying event records. We measure compactness by the number of concepts representing the same event stream. A smaller concept layer can organize repeated evidence more economically, but its usefulness also depends on what a reader can recover. We therefore first evaluate memory utility through downstream answers, then examine the concepts, connections, and candidate intents formed during ingestion. Total memory includes source events, relations, and indexes in addition to concepts.

The third layer provides a place for prospective information. The writer can propose an Intent node when visible observations and concepts suggest an unmet goal, connecting it to supporting records through relations such as TRIGGERS. Candidate states include pending, resolved, rejected, and deferred. CogEval-Bench evaluates this layer through its Proactivity metric (Section 4.4.2); the LoCoMo evaluation concerns events and concepts. The user-feedback interface and intent measurements are detailed in Appendices A.3 and B.14.

At query time, the reader retrieves from a saved graph and receives source events, concepts, or a mixture under a specified context budget. The mixed policy used in the LoCoMo comparisons allocates a 30% share to concepts alongside retrieved events. Holding the reader fixed allows us to compare the memories formed by different writers.

## 4 Experiments and Results

We evaluate CogniFold along two axes: whether the memory formed during ingestion supports subsequent use (long-term memory), and what concepts, connections, and candidate intents it forms from an event stream (proactive evaluation).

### 4.1 Datasets

**Long-term Memory.** We use all ten LoCoMo conversations (Maharana et al., 2024), which evaluate recall and reasoning over conversational history. Accuracy J is the LLM-as-a-judge score on the 1,540 questions in categories 1–4; the all-question Strict score, including 446 adversarial questions, appears in Appendix C.1. LongMemEval-S (Wu et al., 2025) evaluates 500 questions involving single-session recall, multi-session reasoning, knowledge updates, and temporal reasoning; its results are reported in Appendix C.3.

**Proactive Evaluation.** QA accuracy alone cannot validate the central claim that cognitive structures emerge from event-stream folding—a flat RAG system with strong BM25 can score well on factual QA without forming any concepts, and a verbatim event store can pass multi-hop retrieval without any compression. We therefore introduce **CogEval-Bench**, a structural bench-

mark. CogEval-Bench uses **top-down generation**: for each scenario a gold concept graph $\mathcal{G}^* = (\mathcal{C}^*, \mathcal{R}^*, \mathcal{H}^*, \mathcal{I}^*)$ is specified first (concepts, inter-concept relations, hierarchy parents, expected intents, and planted multi-hop chains), then grounded first-person events are generated from it, followed by distractor injection (10–15%) and temporal shuffling. The benchmark spans 6 scenarios (SoftEng, Health, Team, News, Academic, Support) across 4 domains; scale statistics are reported in Appendix B. Ground truth is established **by construction** rather than through post-hoc annotation. Three evaluation tracks are computed per system: **Concept Emergence** (Gold F1 via Hungarian-matched (Kuhn, 1955) soft-matching, LLM Quality, Harmony, Purity), **Relationship Topology** (Chain Discovery, Clustering, Modularity, Edge Type Entropy), and **Compression & Proactivity** (Compression Ratio, PageRank Gini, Proactivity). Full schemas, generation prompts, and per-scenario breakdowns are in Appendix B.

### 4.2 Baselines

**Long-term Memory.** On LoCoMo we report published results for Mem0 (Chhikara et al., 2025), Zep (Rasmussen et al., 2025), Supermemory (Supermemory Team, 2026), MemU (NevaMind AI, 2025), MemOS (Li et al., 2025), ENGRAM (Patel & Patel, 2025), and EverMemOS (Hu et al., 2026) with GPT-4o-mini, and for LangMem, Mem0, Zep, Nemori, PREMem, and EverMemOS with GPT-4.1-mini. Comparisons with published systems remain descriptive because their evaluation protocols differ. Under a shared reader and judge, we compare Full with two alternative write-time organization schemes: *feedback-blocked* writes a concept per atomic fact without seeing existing concepts, and *online clustering* assigns new evidence by a fixed similarity threshold and uses the same LLM to name or revise each group.

**Proactive Evaluation.** On CogEval-Bench, seven systems are compared under identical LLM and events: **OpenIE KG** (HippoRAG-style triples), **Cognee** (Topoteretes, 2026) (Extract–Cognify–Load pipeline), **HippoRAG 2** (Gutiérrez et al., 2025) (deep passage + synonym expansion), **GraphRAG** (Edge et al., 2024) (batch community detection + LLM summarization), **Mem0** (Chhikara et al., 2025) (text-rewrite memory cells), **Zep** (Rasmussen et al., 2025) (temporal knowledge graph), and CogniFold.

### 4.3 Implementation Details

On LoCoMo, GPT-4o-mini writes, answers, and judges. Questions are replayed from frozen graphs with fp16/256 reranking and four QA workers. Feedback-blocked and online clustering each use three independent builds, and all configurations share the same mixed reader; actual writer consumption is measured. Feedback-blocked retains source events, concept outputs, and the final reader, using historical events as write context, so it tests retrieved context and operation instructions jointly (Appendix D.1). On CogEval-Bench, every system uses `gpt-4o-mini` and `text-embedding-3-small`, so cross-system differences are attributable to architectural design rather than model capability. Events are ingested as a stream: each event is processed sequentially through context assembly, `UpdatePlan` generation, and atomic execution, so structure formed from earlier events enters the context of later ones, as in deployment. Parameters and schedules are given in Appendix A.2.

### 4.4 Results

#### 4.4.1 Long-term Memory

Table 1 reports published systems alongside CogniFold. With GPT-4o-mini, CogniFold reaches 86.0% overall, the best temporal (86.6) and single-hop (91.2) scores in its group, and is second overall to EverMemOS, whose numbers use a three-judge ensemble. With GPT-4.1-mini, it reaches 91.1% overall, ties EverMemOS on multi-hop (91.1) and temporal (89.7) questions, and exceeds all other baselines in every category.

**Write-time organization and compactness.** Full, feedback-blocked, and online clustering give three answers to where new evidence belongs: Full judges it against retrieved concepts and favors reuse, whereas the two alternatives do not consult existing concepts.

Table 1: **Detailed performance (LLM-as-a-Judge score) on LoCoMo by question type.** J-Score is the LLM-as-judge accuracy (Zheng et al., 2023) with `gpt-4o-mini` as the judge. Following common practice, Category 5 (adversarial) is excluded. Rows are grouped by backbone LLM. **Bold**: best per column within each group; underline: second-best.

| LLM | Methods | Multi-Hop | Temporal | Open-Domain | Single-Hop | **Overall** |
|---|---|---|---|---|---|---|
| | ♯Questions | 282 | 321 | 96 | 841 | 1540 |
| **GPT-4o-mini** | Mem0 (2025) | 51.2 | 55.5 | **72.9** | 67.1 | 66.9 |
| | Zep (2025) | 74.1 | 67.7 | 66.0 | 79.8 | 75.1 |
| | Supermemory (2026) | 51.1 | 31.8 | 42.7 | 67.3 | 55.3 |
| | MemU (2025) | 63.1 | 27.1 | 50.0 | 66.3 | 56.6 |
| | MemOS (2025) | 67.5 | 75.2 | 55.9 | 81.1 | 75.8 |
| | ENGRAM (2025) | 79.8 | 70.8 | **72.9** | 79.9 | 77.6 |
| | EverMemOS† (2026) | **86.2** | 81.9 | 66.7 | 91.1 | **86.8** |
| | **CogniFold** | 79.8 | **86.6** | 57.3 | **91.2** | 86.0 |
| **GPT-4.1-mini** | LangMem | 71.0 | 50.8 | 59.0 | 84.5 | 73.4 |
| | Mem0 (2025) | 68.2 | 56.9 | 47.9 | 71.4 | 66.3 |
| | Zep (2025) | 53.7 | 60.2 | 43.8 | 66.9 | 61.6 |
| | Nemori | 75.1 | 77.6 | 51.0 | 84.9 | 79.5 |
| | PREMem | 61.0 | 74.8 | 46.9 | 66.2 | 65.8 |
| | EverMemOS (2026) | **91.1** | **89.7** | **70.8** | **96.1** | **92.3** |
| | **CogniFold** | **91.1** | **89.7** | 66.7 | 94.4 | 91.1 |

†Reported under a 3-LLM-judge ensemble protocol; numbers as reported in Hu et al. (2026) Table 1.

Table 2: Concept memory and downstream utility on LoCoMo. Concept counts span three builds; J is frozen-replay accuracy (%), averaged over three builds for the two alternative schemes. Clustering forms 507 centroid clusters in each build; its count includes additional fact concepts. Full combines concept retrieval with update-first instructions; blocked uses historical-event context and add-first instructions.

| Writing scheme | Concept nodes | J |
|---|---|---|
| CogniFold (Full) | 3,179–3,275 | **86.04** |
| Feedback-blocked | 5,518–5,618 | 83.18 |
| Online clustering + naming | 997–1,006 | 82.64 |

Full forms 3,179–3,275 concepts, compared with 5,518–5,618 for feedback-blocked: 42.4% fewer concept nodes, computed as $100(1 - \overline{N}_{\text{Full}}/\overline{N}_{\text{blocked}})$, while preserving source events. Full minus feedback-blocked is $+2.86$ points. The reuse scheme thus supports a compact concept layer without loss of downstream accuracy.

Online clustering produces 507 clusters. Full's J is higher by $3.40$ points. The three configurations expose different organization granularities under a common reader. Appendix D.2 reports writer operation counts for the earlier B/C configurations; Appendix D.1 details the controls.

### 4.4.2 Proactive Evaluation

Table 3 reports CogEval-Bench averages across all six scenarios; per-scenario breakdowns are in Table 4 and full evaluation details in Appendix B. CogniFold achieves Harmony 0.476, substantially above GraphRAG (0.323, the strongest baseline) and far above entity-level systems (OpenIE KG 0.138; Cognee 0.094; HippoRAG 2 0.095). CogniFold is the only system producing non-zero Purity (0.361)—its concepts are coherently grounded in their constituent events, while all baselines lack event-level grounding. On topology, CogniFold's clustering coefficient (0.327) reflects genuine triadic closure among semantically related concepts, distinct from HippoRAG 2's high raw clustering (0.716) that arises from synonym-expansion cliques among name variants. On compression and

Table 3: **CogEval-Bench: structural evaluation across 7 systems.** Averages over 6 scenarios (small scale, $\sim$42 events each). Track A measures concept quality, Track B measures graph topology, Track C measures compression and proactivity. Arrows indicate preferred direction. All systems share GPT-4o-mini and `text-embedding-3-small`; differences are attributable to architecture. Only CogniFold achieves non-zero purity and proactivity—structural properties absent from entity-level, batch-processed, or text-rewrite representations. Bold: best; underline: second-best.

| Track | Metric | OpenIE KG | Cognee | HippoRAG 2 | GraphRAG | Mem0 | Zep | CogniFold |
|---|---|---|---|---|---|---|---|---|
| Track A | Harmony ↑ | 0.138 | 0.094 | 0.095 | <u>0.323</u> | 0.000 | 0.138 | **0.476** |
| | Gold F1 ↑ | 0.081 | 0.053 | 0.058 | <u>0.232</u> | 0.000 | 0.081 | **0.358** |
| | LLM Quality ↑ | 0.492 | 0.454 | 0.341 | <u>0.541</u> | 0.000 | 0.494 | **0.733** |
| | Purity ↑ | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 | **0.361** |
| Track B | Chain Disc. ↑ | 0.750 | **1.000** | <u>0.917</u> | 0.833 | **1.000**† | <u>0.917</u> | 0.833 |
| | Clustering ↑ | 0.005 | 0.200 | **0.716** | 0.002 | 0.382 | <u>0.704</u> | 0.327 |
| | Modularity | 0.848 | 0.714 | 0.759 | **0.873** | 0.434 | 0.784 | 0.546 |
| | Edge Entropy ↑ | **0.968** | 0.540 | 0.000 | <u>0.968</u> | 0.000 | 0.507 | 0.624 |
| Track C | Compression ↑ | 0.3× | 0.5× | 0.2× | <u>1.2×</u> | 1.0× | 0.5× | **4.6**× |
| | PR Gini | 0.323 | 0.352 | 0.310 | 0.317 | 0.189 | 0.242 | 0.274 |
| | Proactivity ↑ | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 | **0.614** |

†Mem0 has no native graph; we materialize each extracted memory as a node and induce edges via vector-similarity neighbours, which trivially saturates Chain Disc. but produces no typed-edge structure (Edge Entropy $= 0$).

proactivity, CogniFold achieves $4.6\times$ compression (38–47 events $\rightarrow$ 7–12 concepts) while OpenIE KG and HippoRAG 2 expand the representation; and CogniFold is the only system emitting intent nodes at all, reaching Proactivity 0.614 (61% of intents grounded by $\geq 2$ supporting connections).

The seven-system comparison reveals an ordered hierarchy in representational richness: **entity graphs** (OpenIE KG)—fragmented triples, high modularity from disconnection; **entity graphs with enrichment** (Cognee, HippoRAG 2)—shallow structure atop entity extraction, Harmony stuck at $\sim$0.09; **community graphs** (GraphRAG)—batch community detection yields the strongest baseline (Harmony 0.323) but with zero event-level grounding (Purity $= 0$) and negligible clustering (0.002); **cognitive graphs** (CogniFold)—online folding with merging yields event-grounded concepts, genuine triadic closure, substantial compression, and proactive goal identification simultaneously. The critical architectural distinction is **online, incremental processing with merging**: Cognee and HippoRAG 2 add machinery (ECL, synonym expansion, PPR) without cross-event integration and remain at the entity-enrichment tier, confirming that the bottleneck in concept emergence is not extraction depth but the ability to recognize that events $e_1, e_3, e_7$ ground the same underlying concept and merge them into a single abstraction. This hierarchy parallels a neuroscience progression (McClelland et al., 1995; Gilboa & Marlatte, 2017; Preston & Eichenbaum, 2013): episodic storage, pattern separation without consolidation, shallow categorization, schema extraction, and active consolidation with goal generation.

Table 4 provides per-scenario breakdowns on three key metrics. CogniFold's Harmony ranges from 0.383 (Team) to 0.572 (SoftEng), consistently above GraphRAG's best (0.390, Support). Compression is robust (3.4–5.5×), highest on Support and Team where repetitive event patterns benefit most from folding. Clustering ranges from 0.132 (Health) to 0.392 (Support), lowest on Health where longer temporal spans create sparser event overlap.

## 5 Discussion

**Memory organization and utility.** CogniFold instantiates proactive memory formation through an event–concept–intent graph and a concept-reuse writing scheme. LoCoMo and LongMemEval characterize its use in conversational answering. Full forms fewer concepts than the historical-event, add-first control with higher observed answer accuracy. CogEval describes the concepts, connections, and candidate intents formed during ingestion. These results characterize the complete

Table 4: **CogEval-Bench per-scenario Harmony scores.** Harmony = harmonic mean of gold F1 and LLM quality (Track A). CogniFold consistently leads across all 6 scenarios despite their diversity. Mem0's flat memory store (no concept abstraction) yields zero across the board; entity-level systems (OpenIE KG, Cognee, HippoRAG 2, Zep/Graphiti) cluster around 0.07–0.19; community-level GraphRAG reaches 0.23–0.39; CogniFold's folding achieves 0.38–0.57.

| Scenario | OpenIE KG | Cognee | HippoRAG 2 | GraphRAG | Mem0 | Zep | **CogniFold** |
|---|---|---|---|---|---|---|---|
| SOFTENG | .107 | .095 | .076 | .312 | .000 | .103 | **.572** |
| HEALTH | .138 | .090 | .101 | .309 | .000 | .081 | **.529** |
| TEAM | .157 | .086 | .107 | .226 | .000 | .181 | **.383** |
| NEWS | .127 | .089 | .069 | .376 | .000 | .116 | **.433** |
| ACADEMIC | .144 | .106 | .111 | .325 | .000 | .190 | **.423** |
| SUPPORT | .158 | .100 | .105 | .390 | .000 | .158 | **.515** |
| **Average** | .138 | .094 | .095 | .323 | .000 | .138 | **.476** |

| | Compression Ratio ↑ | | | | | | | Clustering Coefficient ↑ | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Scenario | KG | Cog | HR2 | GR | M0 | Zp | **CF** | KG | Cog | HR2 | GR | M0 | Zp | **CF** |
| SOFTENG | 0.3 | 0.4 | 0.2 | 1.1 | 1.0 | 0.4 | **3.4** | .013 | .202 | .721 | .000 | .400 | .679 | **.367** |
| HEALTH | 0.2 | 0.4 | 0.2 | 1.3 | 1.0 | 0.5 | **4.9** | .000 | .202 | .684 | .000 | .452 | .742 | **.132** |
| TEAM | 0.3 | 0.5 | 0.2 | 1.2 | 1.0 | 0.8 | **5.4** | .018 | .230 | .730 | .011 | .351 | .666 | **.316** |
| NEWS | 0.2 | 0.4 | 0.1 | 1.1 | 1.0 | 0.4 | **4.2** | .000 | .176 | .736 | .000 | .372 | .773 | **.372** |
| ACADEMIC | 0.2 | 0.5 | 0.2 | 1.1 | 1.0 | 0.4 | **4.3** | .000 | .193 | .706 | .000 | .371 | .638 | **.380** |
| SUPPORT | 0.3 | 0.5 | 0.2 | 1.3 | 1.0 | 0.6 | **5.5** | .000 | .200 | .720 | .000 | .347 | .726 | **.392** |
| **Average** | 0.3 | 0.5 | 0.2 | 1.2 | 1.0 | 0.5 | **4.6** | .005 | .200 | .716 | .002 | .382 | .704 | **.327** |

KG = OpenIE KG; Cog = Cognee; HR2 = HippoRAG 2; GR = GraphRAG; M0 = Mem0; Zp = Zep/Graphiti; CF = CogniFold. Compression = events/concepts (×). Clustering marked **bold** for CogniFold due to semantic meaningfulness; HippoRAG 2's high clustering arises from synonym edges, Zep's from dense entity co-mention edges, and Mem0's from undifferentiated similarity-induced edges (see text).

organization scheme and its deployment trade-offs, including the additional construction cost and the reader's access to derived concepts.

**Prospective use.** Candidate intents are formed from converging event and concept evidence during ingestion. The LoCoMo case analysis in Appendix B.15 illustrates the intended behavior: a plan or goal is recorded before its later realization, remains connected to supporting evidence, and is subsequently confirmed by an event in the same natural conversation stream. This complements CogEval's controlled formation measurements with an external, qualitative demonstration of prospective memory. Source storage continues to grow; sustained throughput, update latency, and concept stability remain deployment considerations.

## 6 CONCLUSION

We present CogniFold, an always-on proactive agent memory that folds fragmented events into persistent cognitive structure. Unlike reactive retrieval systems, CogniFold builds a living graph that continuously folds, merges, decays, and reconnects under the event stream. Because cognition grows recursively from the system's own products, goal-directed intents naturally emerge from converging evidence. We validated this design across two critical axes: proactive structural emergence on CogEval-Bench, and robust memory quality across conventional long-term memory benchmarks. As foundation-model capability grows, what a system computes in a single forward pass approaches what a human can reason about in a moment; what it accumulates, organizes, and bootstraps across time is where the value of an always-on agent will increasingly accrue. We release CogniFold as a foundation for research on real-time interaction, proactive collaboration, and agent cognition that bootstraps beyond memory.

## AI USE STATEMENT

Language models are used for synthetic event generation, memory updates, and model-based evaluation, as described in Sections 3–4 and Appendix B. Generative AI assistance was also used to polish the manuscript. The authors are responsible for the final text, claims, data, and artifacts.

## ETHICS STATEMENT

Persistent agent memory can contain sensitive personal information and can propagate incorrect inferences across interactions. Deployment requires consent, access controls, and deletion procedures covering source records and their derived representations. Candidate intents represent suggested goals; an application should obtain appropriate user authorization before taking consequential actions. The evaluations use public benchmark data and synthetic event streams.

## REPRODUCIBILITY STATEMENT

Equations 1–3 and Algorithm 1 define the update procedure. Appendix A provides implementation details, Appendix B describes the structural benchmark and comparison systems, and Appendix C details the downstream protocols. The tables derived from local result artifacts use a deterministic aggregation script (Appendix C.1). Appendix D specifies the controlled component comparisons.

## REFERENCES

Tyler Barnes. Observational memory: 95% on LongMemEval. `https://mastra.ai/research/observational-memory`, 2026.

Frederic Charles Bartlett. *Remembering: A study in experimental and social psychology*. Cambridge university press, 1995.

Ali Behrouz, Peilin Zhong, and Vahab Mirrokni. Titans: Learning to memorize at test time. *arXiv preprint arXiv:2501.00663*, 2024.

Michael Bratman. *Intention, plans, and practical reason*. Harvard University Press, 1987.

Prateek Chhikara, Dev Khant, Saket Aryan, Taranjeet Singh, and Deshraj Yadav. Mem0: Building production-ready ai agents with scalable long-term memory. *arXiv preprint arXiv:2504.19413*, 2025.

Leyang Cui, Yu Wu, Shujie Liu, Yue Zhang, and Ming Zhou. Mutual: A dataset for multi-turn dialogue reasoning. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 1406–1416, 2020.

Darren Edge, Ha Trinh, Newman Cheng, Joshua Bradley, Alex Chao, Apurva Mody, Steven Truitt, Dasha Metropolitansky, Robert Osazuwa Ness, and Jonathan Larson. From local to global: A graph rag approach to query-focused summarization. *arXiv preprint arXiv:2404.16130*, 2024.

Gilles O Einstein and Mark A McDaniel. Prospective memory: Multiple retrieval processes. *Current Directions in Psychological Science*, 14(6):286–290, 2005.

Jizhan Fang, Xinle Deng, Haoming Xu, Ziyan Jiang, Yuqi Tang, Ziwen Xu, Shumin Deng, Yunzhi Yao, Mengru Wang, Shuofei Qiao, Huajun Chen, and Ningyu Zhang. LightMem: Lightweight and efficient memory-augmented generation. *arXiv preprint arXiv:2510.18866*, 2025. URL `https://arxiv.org/abs/2510.18866`.

Asaf Gilboa and Hannah Marlatte. Neurobiology of schemas and schema-mediated memory. *Trends in cognitive sciences*, 21(8):618–631, 2017.

Bernal J Gutiérrez, Yiheng Shu, Yu Gu, Michihiro Yasunaga, and Yu Su. Hipporag: Neurobiologically inspired long-term memory for large language models. *Advances in neural information processing systems*, 37:59532–59569, 2024.

Bernal Jiménez Gutiérrez, Yiheng Shu, Weijian Qi, Sizhe Zhou, and Yu Su. From rag to memory: Non-parametric continual learning for large language models. *arXiv preprint arXiv:2502.14802*, 2025.

Chuanrui Hu, Xingze Gao, Zuyi Zhou, Dannong Xu, Yi Bai, Xintong Li, Hui Zhang, Tong Li, Chong Zhang, Lidong Bing, and Yafeng Deng. Evermemos: A self-organizing memory operating system for structured long-horizon reasoning. *arXiv preprint arXiv:2601.02163*, 2026.

Dongming Jiang, Yi Li, Guanpeng Li, and Bingzhe Li. Magma: A multi-graph based agentic memory architecture for ai agents. *arXiv preprint arXiv:2601.03236*, 2026.

Tomáš Kočiskỳ, Jonathan Schwarz, Phil Blunsom, Chris Dyer, Karl Moritz Hermann, Gábor Melis, and Edward Grefenstette. The narrativeqa reading comprehension challenge. *Transactions of the Association for Computational Linguistics*, 6:317–328, 2018.

Harold W Kuhn. The hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2):83–97, 1955.

Christopher A Kurby and Jeffrey M Zacks. Segmentation in the perception and memory of events. *Trends in cognitive sciences*, 12(2):72–79, 2008.

Matthew Le, Y-Lan Boureau, and Maximilian Nickel. Revisiting the evaluation of theory of mind through question answering. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 5872–5877, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1598. URL https://www.aclweb.org/anthology/D19-1598.

Zhiyu Li, Chenyang Xi, Chunyu Li, Ding Chen, Boyu Chen, Shichao Song, Simin Niu, Hanyu Wang, Jiawei Yang, Chen Tang, et al. Memos: A memory os for ai system. *arXiv preprint arXiv:2507.03724*, 2025.

Jiaqi Liu, Yaofeng Su, Peng Xia, Siwei Han, Zeyu Zheng, Cihang Xie, Mingyu Ding, and Huaxiu Yao. SimpleMem: Efficient lifelong memory for LLM agents. *arXiv preprint arXiv:2601.02553*, 2026. URL https://arxiv.org/abs/2601.02553.

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

Bodhisattwa Prasad Majumder, Bhavana Dalvi Mishra, Peter Jansen, Oyvind Tafjord, Niket Tandon, Li Zhang, Chris Callison-Burch, and Peter Clark. Clin: A continually learning language agent for rapid task adaptation and generalization. *arXiv preprint arXiv:2310.10134*, 2023.

James L McClelland, Bruce L McNaughton, and Randall C O'Reilly. Why there are complementary learning systems in the hippocampus and neocortex: insights from the successes and failures of connectionist models of learning and memory. *Psychological review*, 102(3):419, 1995.

Michael McCloskey and Neal J Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, volume 24, pp. 109–165. Elsevier, 1989.

NevaMind AI. memU: Memory infrastructure for LLMs and AI agents. https://github.com/NevaMind-AI/memU, 2025.

Mark EJ Newman. Modularity and community structure in networks. *Proceedings of the national academy of sciences*, 103(23):8577–8582, 2006.

Charles Packer, Vivian Fang, Shishir G. Patil, Kevin Lin, Sarah Wooders, and Joseph E. Gonzalez. Memgpt: towards llms as operating systems. *arXiv preprint arXiv:2310.08560*, 2023.

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

Daivik Patel and Shrenik Patel. Engram: Effective, lightweight memory orchestration for conversational agents. *arXiv preprint arXiv:2511.12960*, 2025.

Alison R Preston and Howard Eichenbaum. Interplay of hippocampus and prefrontal cortex in memory. *Current biology*, 23(17):R764–R773, 2013.

Preston Rasmussen, Pavlo Paliychuk, Travis Beauvais, Jack Ryan, and Daniel Chalef. Zep: a temporal knowledge graph architecture for agent memory. *arXiv preprint arXiv:2501.13956*, 2025.

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

Theodore Sumers, Shunyu Yao, Karthik R Narasimhan, and Thomas L Griffiths. Cognitive architectures for language agents. *Transactions on Machine Learning Research*, 2024.

Supermemory Team. Supermemory: State-of-the-art memory and context engine for ai. `https://github.com/supermemoryai/supermemory`, 2026.

Topoteretes. cognee: Memory control plane for ai agents. `https://github.com/topoteretes/cognee`, 2026.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. MuSiQue: Multi-hop questions via single-hop question composition. *Transactions of the Association for Computational Linguistics*, 10:539–554, 2022.

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

Di Wu, Hongwei Wang, Wenhao Yu, Yuwei Zhang, Kai-Wei Chang, and Dong Yu. LongMemEval: Benchmarking chat assistants on long-term interactive memory. In *International Conference on Learning Representations (ICLR)*, 2025. URL `https://arxiv.org/abs/2410.10813`.

Wujiang Xu, Zujie Liang, Kai Mei, Hang Gao, Juntao Tan, and Yongfeng Zhang. A-mem: Agentic memory for llm agents. *arXiv preprint arXiv:2502.12110*, 2025.

John Yang, Carlos E Jimenez, Alexander Wettig, Kilian Lieret, Shunyu Yao, Karthik Narasimhan, and Ofir Press. Swe-agent: Agent-computer interfaces enable automated software engineering. *Advances in Neural Information Processing Systems*, 37:50528–50652, 2024.

Jeffrey M Zacks. Event perception and memory. *Annual review of psychology*, 71(1):165–191, 2020.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in neural information processing systems*, 36:46595–46623, 2023.

## A Update Procedure and Implementation

Algorithm 1 specifies the event-driven update interface. The graph-selection policy and consolidation schedule are explicit inputs. Event-level plans and boundary-level maintenance can therefore be configured separately. In the CogEval reference runner, each event seeds retrieval from the current graph; consolidation follows ingestion. Thus the concepts and relations created by an event plan can enter later event contexts, while end-of-pass merging and association completion affect the exported graph and read path rather than later writes in that same pass. The conversational runner supports consolidation between sessions. A deployment can choose more frequent boundaries according to its latency budget.

**Execution and provenance.** The implementation uses Python, NetworkX for the multigraph, and Pydantic schemas for nodes, relations, and plans. An update carries operation arguments and natural-language reasoning; provenance fields identify the evidence referenced by derived nodes. Plan execution and graph-wide maintenance are distinct stages. Snapshot rollback applies to atomic plan execution. Provenance makes the basis of an abstraction inspectable; factual correctness still depends on the quality of the source evidence and the writer's interpretation.

**Resource scaling.** Let $N$ and $M$ denote graph nodes and edges and $C$ the number of concepts. A PageRank iteration costs $O(N + M)$; score sorting costs $O(N \log N)$. Exhaustive concept-pair merging uses $O(C^2)$ comparisons, and dense nearest-neighbor completion uses $O(C^2 d)$ arithmetic for embedding dimension $d$, with $O(C^2)$ storage for its similarity matrix. These maintenance costs are amortized over the configured boundary interval. The LLM write context is bounded by $k$, while source-record storage grows with the input stream. These complexity terms describe the reference implementation; measured throughput and latency also depend on API service times and concurrency.

### A.1 Principal Relations

Table 5: Relations used to organize evidence, abstractions, and prospective information. Association and provenance have separate meanings.

| Relation | Typical endpoints | Interpretation |
|---|---|---|
| GROUNDS | Event → Concept/Intent | Source evidence for a derived representation |
| REINFORCES | Event → Concept | Additional support for an existing concept |
| TRIGGERS | Event/Concept → Intent | Evidence motivating a candidate intent |
| PART_OF | Concept → Concept | Membership in a broader abstraction |
| DERIVED_FROM | Derived node → source | Derivation relationship |
| CAUSES | Event/Concept → Event/Concept | Proposed causal relationship |
| RELATED_TO | Concept ↔ Concept | Associative similarity or relatedness |
| DEADLINE_FOR | Time → Intent | Temporal constraint on a goal |
| USER_FEEDBACK | Feedback Event → Intent | User response to a proposed goal |

The graph is directed. The bidirectional symbol in the association row denotes symmetric semantic relatedness; the stored edge retains an explicit source and target. The undirected graph used for connectivity diagnostics is a separate evaluation projection.

### A.2 Scoring and Maintenance Parameters

The reference implementation provides configurable selectors and maintenance operations in addition to the concept-retrieval writer described in Section 3.3. The CogEval reference construction uses event-conditioned retrieval; Full on LoCoMo retrieves candidate concepts by embedding similarity. Neither uses the graph-state score below to select its reported write context. Semantic merge is disabled in the matched LoCoMo runs. The following components specify implementation options, without attributing an independent downstream gain to each operation.

**Graph-state selection.** The optional graph-state score is

$$S(v;\tau) = \left[\alpha p_\tau(v) + \beta e^{-\lambda \Delta\tau_v} + \gamma \frac{a(v)}{\max\{1, \max_{u \in V} a(u)\}}\right] U(v;\tau), \tag{4}$$

where $p_\tau$ is weighted PageRank normalized to sum to one, $a(v)$ is the access count, and $\Delta\tau_v$ is time since the last access in hours. $U$ is a bounded urgency multiplier for approaching deadlines. The default coefficients are $(\alpha, \beta, \gamma) = (0.4, 0.4, 0.2)$. Recency and normalized access lie in $[0, 1]$, while PageRank mass is distributed over the graph. The hierarchical selector uses these signals to construct immediate, working, and background contexts.

**Relative edge-age weighting.** The optional effective edge weight supplied to PageRank is

$$\widetilde{w}_{uv}(\tau) = w_{uv} \exp\left[-\lambda_e \max(0, \tau - \tau_{uv}^{\text{created}})\right], \tag{5}$$

with time measured in hours and $\lambda_e = 0.005$. Source records and base weights remain stored. On a fixed graph whose edges have all been created, advancing $\tau$ multiplies every weight by the same factor, which cancels in PageRank's outgoing normalization. This is therefore a relative edge-age input, not a standalone forgetting mechanism. Node-access recency contributes separately through Eq. 4; processing and access clocks are distinct from historical event timestamps.

**Merging and association completion.** When enabled, consolidation compares concepts at similarity threshold $0.85$, using embedding cosine similarity when vectors are available and normalized title similarity otherwise. Merging transfers incident relations to a surviving concept while source events retain their identities. A separate nearest-neighbor pass adds RELATED_TO associations among embedded concepts, using up to three neighbors with cosine similarity at least $0.35$ in the benchmark hook. These semantic associations are distinct from the GROUNDS links that connect derived records to explicit source evidence. End-of-pass maintenance changes the exported graph; only operations performed before a later write can affect that write's context.

Table 6: Reference implementation parameters. Engine-level selector defaults and benchmark overrides are listed separately.

| Component | Value | Interpretation |
|---|---|---|
| Graph-state weights | $0.4/0.4/0.2$ | PageRank / node recency / normalized access |
| PageRank damping | $0.85$ | Weighted graph-state PageRank |
| Node recency rate | 0.01/hour | Time since last access |
| Edge age rate | 0.005/hour | Effective weight for graph ranking |
| Urgency multiplier | $[1, 2]$ | Linear increase within 24 hours of a deadline |
| Engine context capacity | 45 nodes | Default graph-state context bound |
| Hierarchical caps | 5 / 15 / 25 | Immediate / working / background nodes |
| CogEval write context | 10 nodes | Event-conditioned retrieval results |
| Merge threshold | $0.85$ | Cosine similarity; title similarity fallback |
| Completion hook | $k = 3$, $s \geq 0.35$ | Concept association edges in benchmark runner |
| RRF constant | 60 | Rank-fusion smoothing |
| Traversal hop factor | $0.85$ | Graph-expansion attenuation |
| BFS / PPR blend | $0.6/0.4$ | Read-path graph expansion |

The hierarchical caps use all 45 positions in the default capacity; threshold filtering can further reduce the selected context. Nodes are selected without duplication across levels. Immediate selection emphasizes recency and urgency ($0.7/0.3$), working selection uses PageRank, recency, and node type ($0.5/0.3/0.2$), and background selection combines PageRank and diversity ($0.8/0.2$). The benchmark's event-conditioned selector uses the first 200 characters of the event description as a retrieval query and retains up to ten results.

For a connected deadline $d$, the urgency multiplier is $1 + (1 - (d - \tau)/24)$ when $0 \leq d - \tau < 24$ hours and 1 otherwise; the maximum over connected deadlines is used. Access counts are divided by the graph maximum, with zero returned when all counts are zero. PageRank uses probability normalization. The edge-age kernel alters the weights supplied to PageRank, preserving stored source content and the base relation weights.

### A.3 Candidate-Intent Feedback Interface

The writer receives the event, selected graph context, node and edge schemas, and instructions to identify patterns and evidence-supported goals. A candidate intent is emitted as a graph update with links to its supporting observations or concepts. The prompt-level decision is the operational emission rule in this implementation.

User feedback can update a category-preference profile. Starting from a weight of one, the profile uses

$$w_c^{(t)} = (1 - \eta) w_c^{(t-1)} + \eta s_t, \qquad \eta = 0.3, \tag{6}$$

where $s_t$ equals $1.5$ for accept, $1.1$ for modify, $0.8$ for defer, and $0.3$ for reject. Category preferences are translated into generation context and candidate-priority adjustments. This feedback interface is part of the system design; the reported benchmark evaluations contain no interactive user-feedback study.

### A.4 Writer Instructions

The writer prompt is assembled from node schemas, edge definitions, connectivity rules, and a domain profile. Its operational instructions are: preserve the incoming observation; inspect relevant existing records; use an existing concept when the event adds evidence for the same pattern; create a concept for a distinct recurring pattern; connect derived nodes to their source evidence; propose merges for redundant concepts; and create candidate intents when the available patterns suggest a goal. The output is a structured plan with explicit identifiers and operation arguments. The incoming event and selected context are provided as data to this update task. Domain profiles specify the input format and task vocabulary.

## B CogEval-Bench Details

### B.1 Gold Graph Schemas

For each scenario we manually define a gold concept graph $\mathcal{G}^* = (\mathcal{C}^*, \mathcal{R}^*, \mathcal{H}^*, \mathcal{I}^*)$ with four components. **Concepts** $\mathcal{C}^*$: 8–9 non-hierarchical concepts, each specified with label, natural-language description, representative keywords, and expected event count. Concepts are grounded in established domain knowledge (e.g., SoftEng includes Code Review, Sprint Planning, Deployment) and deliberately avoid CogniFold-specific abstractions to keep the gold standard system-agnostic. **Relationships** $\mathcal{R}^*$: 9–14 labeled inter-concept edges using four relationship types drawn from cognitive schema theory (Gilboa & Marlatte, 2017; Bartlett, 1995): `PART_OF` (compositional hierarchy), `TRIGGERS` (temporal causation), `REINFORCES` (feedback strengthening), `CAUSES` (interventional causation). **Hierarchy parents** $\mathcal{H}^*$: 1–3 superordinate concepts (e.g., Work Projects subsumes Coding Sessions and Code Review) that test hierarchical abstraction. **Expected intents** $\mathcal{I}^*$: 2 goal nodes per scenario, each grounded in 2–3 supporting concepts with a specified trigger pattern (e.g., "3+ exercise events within the week" $\rightarrow$ Maintain Regular Exercise). Each gold graph additionally contains 2 **planted multi-hop reasoning chains** (3–4 hops) following the compositional methodology of MuSiQue (Trivedi et al., 2022): chains are sequences of events connected through causal or temporal links across different concepts (e.g., deployment $\rightarrow$ staging bug $\rightarrow$ client demo failure), requiring cross-concept traversal to reconstruct. Complete gold concept graphs for all 6 scenarios are provided in the supplementary material.

### B.2 Event Generation Pipeline

From each gold graph, a grounded event stream is generated in four stages. (1) **Concept-grounded events**: for each concept $c \in \mathcal{C}^*$, GPT-4o-mini generates $n_c$ realistic first-person events conditioned on the concept's label, description, and keywords; each event receives the by-construction label gold_concept $= c$. (2) **Chain events**: for each planted chain, the pipeline generates sequential events following the chain's step descriptions, ensuring entity consistency across hops (e.g., the same "auth service v2.3" appears in the deployment, bug discovery, and client demo events). (3) **Distractor injection**: 10–15% of events come from unrelated topics (weather, unrelated news, etc.), labeled gold_concept $=$ `null`. (4) **Temporal shuffling**: all events receive timestamps spanning the scenario's temporal window (5–60 days), then are sorted chronologically with chain events interleaved among non-chain events to prevent trivial sequential pattern matching.

### B.3 Scenarios

Six scenarios span four domains (Table 7): professional work (SoftEng), medical recovery (Health), team coordination (Team), breaking news (News), academic research (Academic), and customer support (Support). These domains test distinct cognitive patterns: daily routines with gradual concept consolidation, crisis cascades requiring causal chain tracking, topic drift across independent threads, deep abstraction hierarchies, and repetitive pattern detection. The benchmark totals 251

Table 7: **CogEval-Bench scenarios.** Six scenarios across four domains, with controlled gold concept graphs. Each scenario defines gold concepts, planted multi-hop chains (3–4 hops), expected intents, and distractor events ($\sim$15%). Events generated by GPT-4o-mini from gold graphs, temporally shuffled.

| Scenario | Domain | Events | Gold Concepts | Chains | Intents |
|---|---|---|---|---|---|
| SOFTENG | Professional/tech work | 41 | 8 | 2 (3-hop) | 2 |
| HEALTH | Medical recovery | 39 | 8 | 2 (3–4-hop) | 2 |
| TEAM | Product launch | 38 | 8 | 2 (3–4-hop) | 2 |
| NEWS | Breaking news stream | 42 | 8 | 2 (3–4-hop) | 2 |
| ACADEMIC | Graduate research lab | 47 | 9 | 2 (3–4-hop) | 2 |
| SUPPORT | Customer support center | 44 | 8 | 2 (3–4-hop) | 2 |
| **Total** | 4 domains | 251 | 49 | 12 | 12 |

events, 49 gold concepts, 12 planted multi-hop chains, and 12 expected intents across the six scenarios.

### B.4 EVALUATION METRICS

Three tracks are computed per system and averaged across scenarios. **Track A: Concept Emergence**—(i) **Gold F1**: precision and recall of system concepts against gold concepts via embedding-based soft matching (`text-embedding-3-small`, cosine $\geq 0.75$) with optimal one-to-one assignment via the Hungarian algorithm (Kuhn, 1955); (ii) **LLM Quality**: GPT-4o-mini judge rates meaningfulness, groundedness, and abstraction level, each 0–1, then averaged; (iii) **Harmony**: harmonic mean of Gold F1 and LLM Quality; (iv) **Purity**: average pairwise embedding similarity among events grounding each concept. **Track B: Relationship Topology**—Chain Discovery Rate (fraction of planted chains recoverable via BFS between endpoints), Clustering Coefficient, Modularity (Newman Q) (Newman, 2006), Edge Type Entropy. **Track C: Compression & Proactivity**—Compression Ratio (input events / output concepts), PageRank Gini, Proactivity (fraction of intents with $\geq 2$ grounding connections).

### B.5 COMPARISON SYSTEMS

Seven systems under identical LLM and events. **OpenIE KG**: HippoRAG-style triples, flat entity graph, no concept folding. **Cognee** (Topoteretes, 2026): Extract–Cognify–Load pipeline building property graphs via LLM entity extraction and classification; no cross-event merging or temporal tracking. **HippoRAG 2** (Gutiérrez et al., 2025): deeper passage integration, synonym expansion edges, PPR retrieval; still entity-level with no concept abstraction. **GraphRAG** (Edge et al., 2024): LLM entity/relation extraction, Leiden community detection, LLM summarization per community; batch, no temporal folding or intent emergence. **Mem0** (Chhikara et al., 2025): text-rewrite memory cells with vector retrieval; no native graph (we materialize each cell as a node and induce vector-similarity edges). **Zep** (Rasmussen et al., 2025): temporal knowledge-graph memory with entity-centric edges and time stamps. **CogniFold**: full merge-fold pipeline (events $\rightarrow$ concepts $\rightarrow$ intents), 9 typed edges, online event-by-event processing, temporal decay, lifecycle management.

### B.6 WHY ENRICHMENT $\neq$ ABSTRACTION

Cognee and HippoRAG 2 occupy the same Harmony tier as vanilla OpenIE KG despite adding substantial computational machinery (ECL pipeline, synonym expansion, Personalized PageRank). This confirms that the bottleneck in concept emergence is not extraction depth but **cross-event integration**—the ability to recognize that events $e_1, e_3, e_7$ ground the same underlying concept and merge them into a single abstraction. Systems that process each event in isolation, however thoroughly, cannot produce this integration.

### B.7 Proactivity as Emergence Evidence

Proactivity captures a property no baseline can produce: identifying goals from converging evidence before being asked. An average of 22 intent nodes per scenario, with 61.4% well-grounded by multiple events, constitutes evidence of intent emergence beyond memory organization. While these intents are LLM-generated rather than autonomously emergent from prediction error, they represent a measurable step toward proactive intelligence that flat memory systems cannot take.

### B.8 Construct Validity

Two metrics that most sharply separate CogniFold from baselines—Purity (event-level grounding) and Proactivity (intents with $\geq 2$ groundings)—measure representational features that only CogniFold emits: `GROUNDS` edges for Purity, intent nodes for Proactivity. A reader may reasonably worry the metric is biased toward our representation. We address this risk with three choices: (i) the representational hierarchy holds not only on these two metrics but also on task-agnostic ones (Harmony, Gold F1, LLM Quality, Compression, Clustering), on which baselines are free to score well; (ii) CogEval is deliberately supplemented by the long-term memory results (Section 4.4.1), which use external benchmarks and metrics; (iii) we report raw intent counts and grounding densities, not normalized scores. Even so, a fully adversarial benchmark—one where our representation could lose—is future work.

### B.9 Scale and Scope Limitations

CogEval-Bench uses synthetic events generated from predefined gold graphs, which may not capture the full complexity of natural event streams. The scale is deliberately small ($\sim$42 events per scenario) to enable controlled structural evaluation; whether the structural hierarchy holds at larger scales remains to be validated. The LLM judge introduces potential evaluation bias; we mitigate this by per-concept independent scoring (not pairwise comparison) and report the full judge prompt above.

### B.10 Event Generation Prompt

For each concept $c$ with label $\ell$ and description $d$, events are generated using the following prompt template:

```
Generate {n} realistic first-person events for a
{scenario} scenario, grounded in the concept
"{label}": {description}.

Each event should:
- Be a specific, timestamped experience (not generic)
- Use first-person perspective
- Be self-contained (understandable without context)
- Relate clearly to the concept keywords: {keywords}

Output as JSON array with fields: title, description,
timestamp, event_type.
```

Chain events use a sequential prompt that references entities from prior chain steps to maintain cross-hop consistency. Distractor events are generated from unrelated topics (e.g., weather, unrelated hobbies) without concept grounding.

### B.11 LLM Judge Prompt

The concept quality judge evaluates each system concept independently:

```
Evaluate this concept extracted from an event stream.

Concept: "{concept_label}"
Grounding events (if any): {event_summaries}
Scenario context: {scenario_description}

Rate on three dimensions (0.0 to 1.0):

1. MEANINGFULNESS: Is this a semantically coherent
```

```
   concept? (1.0 = clearly defined theme/pattern,
   0.0 = incoherent or trivial)

2. GROUNDEDNESS: Is the concept well-supported by
   its grounding events? (1.0 = strong evidence,
   0.0 = no supporting evidence)

3. ABSTRACTION LEVEL: Is this the right level of
   abstraction? (1.0 = useful generalization,
   0.0 = too specific or too vague)

Output JSON: {"meaningfulness": X, "groundedness": X,
  "abstraction": X}
```

### B.12 Embedding Similarity Threshold Sensitivity

The Gold F1 computation uses cosine similarity $\geq 0.75$ for concept matching. Table 8 reports CogniFold's Harmony score averaged across 6 scenarios at varying thresholds, confirming that the ranking is robust to threshold choice.

Table 8: Sensitivity of Harmony to embedding similarity threshold (CogniFold, averaged over 6 scenarios).

| Threshold | 0.65 | 0.70 | **0.75** | 0.80 | 0.85 |
|---|---|---|---|---|---|
| Harmony | 0.501 | 0.489 | **0.476** | 0.452 | 0.418 |

Higher thresholds are more conservative (fewer matches, lower recall); lower thresholds are more permissive (more matches, risk of false positives). The relative ranking of systems is preserved across all tested thresholds.

### B.13 Event-Order Sensitivity

The update recurrence makes presentation order a relevant variable. To characterize it, we ingest each scenario in its original order and in two permutations (seeds 42 and 123), retaining the event content and timestamps. Table 9 reports means over these three independently constructed graphs. The study comprises 18 ingestion runs.

Table 9: Variation across three presentation orders per scenario. Concept count, events per concept, and clustering are means over the three orders. These independent ingestions include both order sensitivity and model-generation variability.

| Scenario | Events | Concept count | Events/concept | Clustering |
|---|---|---|---|---|
| SoftEng | 41 | 14.7 | 2.8 | 0.182 |
| Health | 39 | 12.0 | 3.3 | 0.363 |
| Team | 38 | 6.7 | 5.7 | 0.430 |
| News | 42 | 12.7 | 3.3 | 0.269 |
| Academic | 47 | 9.7 | 4.8 | 0.210 |
| Support | 44 | 9.0 | 4.9 | 0.410 |

The same event set can lead to different concept counts and connections. In Team, the three runs produce 6, 4, and 10 concepts; in Support they produce 4, 10, and 13. These independent constructions characterize structural variability under changed presentation orders and model sampling. The reference concept inventory is withheld from the writer, so the resulting identities and relations instantiate a stream-specific organization. Concept counts alone do not identify which trajectory preserves the most useful information.

Table 10: Concepts and candidate intents from the six reference-protocol runs in Table 4. Support incidence is the percentage of intents with at least two qualifying incident relations; it measures graph support, not goal-matching accuracy.

| | SoftEng | Health | Team | News | Academic | Support |
|---|---|---|---|---|---|---|
| Input events | 41 | 39 | 38 | 42 | 47 | 44 |
| Formed concepts | 12 | 8 | 7 | 10 | 11 | 8 |
| Candidate intents | 21 | 24 | 19 | 20 | 22 | 26 |
| Support incidence (%) | 66.7 | 50.0 | 52.6 | 60.0 | 77.3 | 61.5 |

### B.14 CANDIDATE-INTENT DIAGNOSTICS

The writer also proposes candidate intents when the visible events and concepts suggest a goal. These proposals arise during ingestion, without downstream evaluation questions or the reference-goal list in the writer input. Table 10 shows the resulting three-layer organization: each scenario's event stream produces a concept inventory and a set of candidate intents.

Support incidence measures the fraction of candidate intents with at least two incident relations of type TRIGGERS, GROUNDS, REINFORCES, or CAUSES. Its scenario mean is $61.4\%$; relations can share a supporting node, and semantic-association edges are excluded. These measurements show that the system generates prospective records and integrates them with its memory graph. A small natural-conversation case analysis showing intents that anticipate later events and are subsequently fulfilled is given in Appendix B.15.

The six recorded runs produce 21, 24, 19, 20, 22, and 26 candidate intents for SoftEng, Health, Team, News, Academic, and Support, respectively (22 per scenario on average). The support-incidence statistic is

$$\text{SupportIncidence}(G) = \frac{1}{|\mathcal{I}|} \sum_{i \in \mathcal{I}} \mathbf{1}\left[ \sum_{r \in E} \mathbf{1}[r \text{ incident to } i,\ \text{type}(r) \in \mathcal{T}] \geq 2 \right], \tag{7}$$

where $\mathcal{T} = \{\text{TRIGGERS}, \text{GROUNDS}, \text{REINFORCES}, \text{CAUSES}\}$. Both incoming and outgoing relations count. The reported scenario-mean value is $61.4\%$. The unit counted is a qualifying relation; distinct relations can share a supporting node.

This statistic describes the incidence of support relations around generated candidates. Similarity-completion edges of type RELATED_TO are outside $\mathcal{T}$.

### B.15 LOCOMO QUALITATIVE INTENT CASES

CogEval provides controlled evidence that the writer can form candidate intents from an event stream. To illustrate the same behavior in a natural conversation, we inspected a small set of LoCoMo streams for an intent that was formed before the later event that fulfilled it. The cases below are selected for a simple temporal pattern: converging observations establish a plausible goal, the graph records the goal with supporting links, and a subsequent conversation event confirms the anticipated direction.

In each case, the intent is created from evidence available before the confirming event; the confirming event is not needed to create the earlier record. The intent remains linked to its supporting concepts and source events, so the later event can be interpreted as fulfillment of an already represented prospective goal rather than as a goal reconstructed only after the fact. These examples provide a compact external illustration of proactive memory formation: the writer records a future-oriented interpretation while the stream is unfolding, and subsequent experience supplies an observable confirmation.

Table 11: Qualitative LoCoMo cases of prospective intent formation. Each case follows the event stream chronologically; the final column records the later event that confirms the earlier intent.

| Stream | Earlier evidence | Formed intent | Later confirmation |
|---|---|---|---|
| Caroline | Repeated discussion of adoption agencies and preparation | Pursue an adoption plan | Adoption meeting appears later |
| Caroline | Interest in psychology, counseling, and support for transgender people | Pursue counseling as a career | Counseling direction is chosen later |
| Caroline | Repeated participation in LGBTQ community activities | Continue community and mentoring work | Later parade, activist, and mentoring events |

# C Downstream Evaluation Details

## C.1 LoCoMo Aggregation and Uncertainty

All ten conversations are ingested independently. The dataset contains 1,986 questions, of which 1,540 belong to the four scored categories. Single-judge accuracy is the fraction of eligible answers receiving a correct judgment. The primary judgments use the binary semantic-answer evaluation in the LoCoMo runner. Its input contains the question, reference answer, and generated answer; the instructions allow semantic equivalence and sufficient key information. Table 12 reports replays from saved graphs under the frozen read protocol. Question pairs use dataset position so repeated question–answer text is not silently dropped.

The table's strict score is $100\, n_{\text{correct}}/1{,}986$, micro-averaged over all questions, including adversarial questions; partial labels receive no credit. The runner assigns these labels through normalized string matching against the answer or context, followed by three-way evaluation when no match is found. J instead uses a separate binary semantic-answer judgment on the 1,540 eligible answers. Strict is reported descriptively because both its evaluator and denominator differ from J; it does not isolate a benefit of the concept-reuse writing scheme.

**Paired replay differences.** The B/C/F replays use all 1,540 eligible question positions, retaining the 11 repeated question–answer texts. Table 13 reports independent build pairs under the same reader; per-build question intervals condition on the saved answers. All C–B intervals include zero. These intervals do not resample build identities. The bundled grounding/date repairs give F–C $+3.18$ points overall.

Table 12: Frozen LoCoMo replay with fp16/256 reranking and four QA workers. J uses 1,540 eligible questions; strict and usage use all 1,986 questions. Query cost excludes building and judging. F supplies the headline and matched controls; paired differences are in this appendix.

| Configuration | J (%) | Strict (%) | Ctx tok/q | Cost/q (mUSD) |
|---|---|---|---|---|
| B Basic concepts + mixed 30% | 82.47 | 64.87 | 4,036 | 0.727 |
| C Writer reuse + mixed 30% | 82.86 | 64.50 | 4,208 | 0.754 |
| F Full + mixed 30% | 86.04 | 65.63 | 4,499 | 0.798 |

Table 13: Paired LoCoMo replay differences. Delta is percentage points; CIs are nominal 95% question-level bootstrap intervals.

| Comparison | Replay | $\Delta$ (pp) | Question CI |
|---|---|---|---|
| C vs B | r1/r2/r3 | +0.26/+0.32/+0.58 | [-1.30,+1.75]/[-1.17,+1.82]/[-0.84,+2.01] |

Table 14: **LongMemEval-S comparison.** LLM-as-judge accuracy (%) (Zheng et al., 2023) on the 500-question split of Wu et al. (2025). *Build* and *Answer* are the models on the write (memory construction) and read (QA) paths. **Bold**/underline: best/second per column.

| Model | Build | Answer | Judge | SSU | SSA | SSP | MS | KU | TR | **Overall** |
|---|---|---|---|---|---|---|---|---|---|---|
| Mem0[†] (2025) | 4o-mini | 4o-mini | 4o-mini | 81.4 | 41.1 | 60.0 | 46.2 | 70.1 | 40.2 | 53.6 |
| LightMem (2025) | 4o-mini | 4o-mini | 4o-mini | 87.1 | 32.1 | 68.2 | 71.7 | 83.1 | 67.2 | 68.6 |
| Zep (2025) | 4o-mini | 4o | 4o | 92.9 | 80.4 | 56.7 | 57.9 | 83.3 | 62.4 | 71.2 |
| ENGRAM (2025) | 4o-mini | 4o-mini | 4o-mini | **97.1** | 87.5 | 93.3 | 60.2 | 74.4 | 55.6 | 71.4 |
| SimpleMem (2026) | 4.1-mini | 4.1-mini | 4.1-mini | 85.7 | 75.0 | 76.7 | 60.9 | 79.5 | 83.5 | 76.9 |
| EverMemOS[‡] (2026) | 4.1-mini | 4.1-mini | 3-ens. | **97.1** | 85.7 | 93.3 | 73.7 | 89.7 | 77.4 | 83.0 |
| Supermemory[§] (2026) | 4o | 5 | 4o | **97.1** | **100.0** | 76.7 | 75.2 | 87.2 | 81.2 | 84.6 |
| Mastra[§] (2026) | gemini-2.5-flash | 5-mini | 4o | 95.7 | 94.6 | **100.0** | 87.2 | **96.2** | **95.5** | **94.9** |
| **CogniFold** | 4o-mini | 5.4-mini | 4o | **97.1** | **100.0** | 93.3 | **91.0** | 94.9 | 88.7 | 93.0 |

[†]Mem0 has no first-party LongMemEval result; row from Fang et al. (2025). [‡]3-LLM-judge ensemble. [§]Vendor-reported, not peer-reviewed. Names drop `gpt-`/`gemini-` prefixes.

### C.2 Published LoCoMo Configurations

Table 1 in the main text reports published reference results alongside CogniFold. Table entries round only after aggregating the raw counts. Published rows use their respective prompts, readers, judges, and budgets and therefore do not support a controlled ranking. EverMemOS uses an ensemble of three judge models.

### C.3 Published LongMemEval Configurations

LongMemEval-S (Wu et al., 2025) evaluates 500 questions involving single-session recall, multi-session reasoning, knowledge updates, and temporal reasoning. The reported configuration constructs memory with GPT-4o-mini, answers with GPT-5.4-mini, and uses GPT-4o for judgment. It reaches 93.0% overall (Table 14), with 100.0% on single-session assistant recall and 88.7% on temporal reasoning. The latter requires resolving event dates and relative temporal expressions across the retrieved history.

The CogniFold configuration uses `text-embedding-3-small` for graph construction and retrieval embeddings in these downstream runs. The reported scores describe complete system configurations, including model and retrieval choices. The LongMemEval row is therefore a standalone configuration rather than a same-reader matched comparison; no same-reader LongMemEval comparison is reported.

### C.4 Broader Benchmark Results

The task-level measurements below use a GPT-4o-mini answer model together with the input adapters described in the caption and following paragraph.

Table 15: Additional downstream results for CogniFold under task interfaces that remain interpretable as memory use. NarrativeQA uses story summaries; MuTual and ToMi map dialogue or narrative observations to events.

| Benchmark | Metric | Questions | Score |
|---|---|---|---|
| NarrativeQA (Kočiskỳ et al., 2018) | Token F1 | 500 | 72.0 |
| MuTual (Cui et al., 2020) | Accuracy | 500 | 93.2 |
| ToMi (Le et al., 2019) | Exact match | 500 | 91.6 |

NarrativeQA's adapter uses the provided story summary, split into short groups of sentences; its score is therefore summary-based QA rather than full-story QA. MuTual maps dialogue turns to events, and ToMi represents the narrative observations used for belief tracking.

Table 16: Matched LoCoMo controls. Graph nodes count all stored nodes (source events, concepts, intents, and time anchors); B and C report their first build, the others span three builds; J (%) is frozen-replay accuracy, averaged over three independent builds for the B, C, blocked, and clustering configurations. Details and paired effects: Appendix D.1.

| Configuration | Write context and instruction | Graph nodes | J |
|---|---|---|---|
| B basic concepts + mixed 30% | Basic concept extraction | 6,419 | 82.47 |
| C writer reuse + mixed 30% | Concept reuse, no repairs | 5,145 | 82.86 |
| Full, matched protocol | 15 retrieved concepts, update-first | 5,217–5,326 | 86.04 |
| Feedback-blocked, events retained | 15 historical events, add-first | 7,810–7,922 | 83.18 |
| Online clustering + naming | Centroid assignment (cos $\geq 0.75$) | 2,568–2,577 | 82.64 |

# D Controlled Comparisons and Write Traces

The evidence package separates two questions: how the implemented concept-reuse writing scheme changes representation and utility (Full versus feedback-blocked), and whether a simpler online organization procedure obtains the same behavior (online clustering plus naming). The B/C deployment rows repeated in Table 16 are reference runs: B is a basic-concepts readout and C adds writer reuse, so C–B cannot substitute for the Full versus feedback-blocked intervention. LoCoMo accuracy is question-micro-averaged over the 1,540 eligible positions; replay pairs retain repeated question–answer text by dataset position.

## D.1 Concept-Reuse Writing Scheme and Resource Control

**Full graph and feedback-blocked graph.** For LoCoMo, both configurations retain source events, concept outputs, maintenance boundaries, and the final mixed reader. Full retrieves the 15 most similar existing concepts; feedback-blocked instead retrieves the 15 most similar historical raw events. Serialized blocked inputs exclude Concept/Intent records, identifiers, descriptions, neighbor summaries, and nested derived context. The writer has no alternative graph-query tool. Both arms use duplicate suppression, grounding/date fixes, and the same maintenance settings, with semantic merge disabled.

The implemented intervention also changes the instruction block. Full asks the writer to check candidates first, use `UPDATE_NODE` for the same fact, and prefer updating or reinforcing a match over adding a duplicate. Blocked asks it to `ADD_NODE` one concept per atomic fact and reinforce supporting historical messages. Both resolve relative dates from session timestamps. Thus, the comparison tests the implemented reuse procedure, including its operation instructions; it does not isolate context exposure alone. Blocked exposes zero derived identifiers and performs no updates, but readable identifier collisions still produce 51–118 edges per conversation touching existing concepts. Zero exposure therefore does not imply zero references to all prior concepts.

The replay manifests record run identifiers, maintenance flags, graph snapshots, and cost fields. Semantic merge is disabled in the canonical B/C replay; duplicate suppression, writer reuse, and reader policy are frozen by the manifests. This keeps the replay comparison separate from the build-stage QA values.

**Online clustering and naming.** LoCoMo event-batch embeddings are assigned to the nearest current centroid at cosine similarity $\geq 0.75$; otherwise a new cluster is opened. Centroids use only events observed so far. GPT-4o-mini names or revises the affected cluster once per batch using the observed evidence. Each cluster retains event membership; the export supplies cluster concepts and source links to the same mixed reader, without cross-concept writer operations. This is a competing online organization method, not a component ablation. Primary statistics use builds r2–r4;

For the matched protocol, Full is fixed to duplicate suppression, writer-reuse, and the mixed reader policy; semantic merge is disabled unless a separate merge condition is explicitly reported. This operational definition is distinct from the historical deployment label and prevents a node-count change from being attributed to an unlogged maintenance operation.

**Token and call accounting.** The LoCoMo matched protocol uses GPT-4o-mini for writing, cluster naming, answering, and judging, with `text-embedding-3-small` shared across retrieval, writer-side candidate ranking, and clustering. No common writer token cap is imposed; consumption is measured. Writing proceeds over event batches, each retained as a source node and passed to the writer or cluster namer. The Full r1 and blocked r3 records average 6.1k/6.7k input and 0.6k/0.9k output tokens per writer call. The reader generates at most 100 output tokens per answer; multi-hop questions add one decomposition call. The answer replays had no retries or repair calls; ingestion-plan rollbacks are accounted for separately below. Phase costs include observed input/output and embedding tokens, with actual provider billing when present.

**Results.** Full reaches $86.04\%$ J; the three blocked builds reach $83.51/82.40/83.64$ and the three clustering builds $82.40/83.51/82.01$. Full$-$blocked is $+2.86$ and Full$-$clustering is $+3.40$ points.

The saved graphs retain the same 1,571 source-event contents across all primary Full, blocked, and clustering builds. Full records 1,571 plans per build, with 1,568/1,562/1,561 successfully executed; blocked records 1,571 plans with zero exposed derived identifiers and 1,558/1,560/1,557 successfully executed. Remaining plans rolled back. Complete build meters record 1,571 writer calls for each Full build and blocked r3. Each primary clustering build records 1,571 naming calls and 1,571 successful cluster updates.

Full's concept counts are 3,192/3,275/3,179; blocked's are 5,618/5,605/5,518. The reduction is $100(1 - \overline{N}_{\text{Full}}/\overline{N}_{\text{blocked}}) = 42.4\%$, using the ratio of three-build means. This is not a total-graph reduction: representative Full r1 and blocked r3 have 5,223 and 7,810 total nodes (33.1% fewer). All three primary clustering builds form 507 centroid clusters; their final graphs contain 1,006/997/1,000 concept-typed nodes after additional fact extraction. Table 2 uses these total concept counts. Approximate build costs per ten-conversation set are clustering $0.29, Full $2.03, and blocked $2.40 (r3).

### D.2 Writer Traces

**Measured writer-use and organization diagnostics.** The canonical ingestion manifests record writer-side operation traces. Across the first canonical builds, the logged `UPDATE_NODE` operations on existing concepts increase from 28 for B to 1,048 for C; the trace records which exposed identifiers were later referenced, rather than treating a final repair edge as reuse. Across three independent canonical builds, C has 19.1–20.4% fewer graph nodes than B (the corresponding C–B replay differences are still only $+0.26$–$+0.58$ points with intervals containing zero). These counts establish that the writer path is active and changes organization, but they do not by themselves identify an accuracy effect.

The repairs used in F add an automatic provenance edge for newly created concepts and carry session dates into concept text. Across its three independent builds, structured A1 evaluation finds supported concepts at 97.1%, 97.1%, and 97.5%; a regular-expression date check finds absolute dates in 81.7%, 82.0%, and 82.1% of concepts. Separately, on the earlier canonical graphs, strict concept-only next-session coverage is higher for C than B by $+0.014$ with a 95% interval $[+0.009, +0.020]$. These measure support, date presence, and coverage, rather than answer correctness. F supplies the maintable Full result; its improvements are not attributed separately to individual repair operations.

The matched controls in Table 16 report the implemented writing schemes. Full minus feedback-blocked is $+2.86$ points, and Full minus online clustering is $+3.40$. The blocked writer exposed zero derived identifiers in each of its three build traces. These effects are descriptive paired comparisons.