LIFEFORGE: SYNTHETIC DIGITAL LIVES FOR MULTI-SOURCE AGENT MEMORY EVALUATION

Anonymous authors
Paper under double-blind review

ABSTRACT

AI assistants increasingly build memory from a user's personal data spread across many sources rather than from conversation alone. Evaluating them requires multi-source personal data with verifiable ground truth, which real users cannot provide for privacy reasons, and existing memory benchmarks remain almost exclusively conversational. We present **LIFEFORGE**, a two-stage framework that synthesizes a virtual user's multi-source data and compiles memory questions from it without human authoring. Stage one derives every record from one hidden event timeline, projecting each event into mutually consistent calendar entries, emails, office files and photos. Because every record traces back to a known event, Stage two can select the evidence for a question and keep only questions whose answer is unique. We release 11 digital lives in six countries and four languages, 29K records in all, and 1,034 questions covering four memory operations. Scoring five open-source memory systems under one harness, on answer quality, record-level retrieval and the cost of writing and reading memory, exposes two shared failures. They retrieve records one at a time, so most collapse once an answer has to gather many records from one stretch of time; giving the same reader the gold evidence closes most of that gap, which puts the fault in retrieval. They extract facts at ingestion and discard the details questions ask about, a loss a stronger reader widens rather than repairs.

1 INTRODUCTION

The memory of an AI assistant is no longer fed by conversation alone. Mainstream assistants now connect to a user's mailbox, calendar, cloud drive and photo library, and build user memory directly from these sources (OpenAI, 2026; Google, 2026; Anthropic, 2026; Microsoft, 2025; Apple, 2026). Such memory has to reconcile records written at different times by different people: one decision may appear in a mail thread, a calendar entry, a drive document and a photo. Evaluating it therefore requires a history that spans several sources over months, and questions whose answers draw on several of those sources at once. Existing benchmarks miss at least one of the two. Most are built on conversation transcripts (Maharana et al., 2024; Wu et al., 2025; Tavakoli et al., 2026; Hu et al., 2026); the multi-source ones (Xiu et al., 2026; Chai et al., 2026; Zeng et al., 2026) cover days rather than months; and LifeBench (Cheng et al., 2026), which does simulate a long horizon, renders the user as textual phone logs with no files or images.

Building such a benchmark runs into three coupled problems. First, real data is out of reach: months of a real user's mail, calendar and photos cannot be collected at scale, so the history has to be synthesized. Second, naive synthesis breaks consistency: when an LLM writes emails, events and files independently, the records contradict each other and silently corrupt the ground truth (Zeng et al., 2026). Third, and most fundamentally, the ground truth lacks traceable provenance. The evidence for a multi-hop question is scattered across mail, calendar, files and photos among tens of thousands of records, and without reading the entire history no one can confirm that the evidence set is complete, that the answer is unique, or that no other record overturns it. This is why hand-written benchmarks stay at a few hundred items (Wu et al., 2025; 2026a; Rein et al., 2024), and why LLM-drafted questions still need a human to verify and revise each one (Tavakoli et al., 2026).

Our key idea is to generate the provenance first and derive both the records and the questions from it. From a short free-text brief, LIFEFORGE writes a hidden *provenance structure*, consisting of a

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.

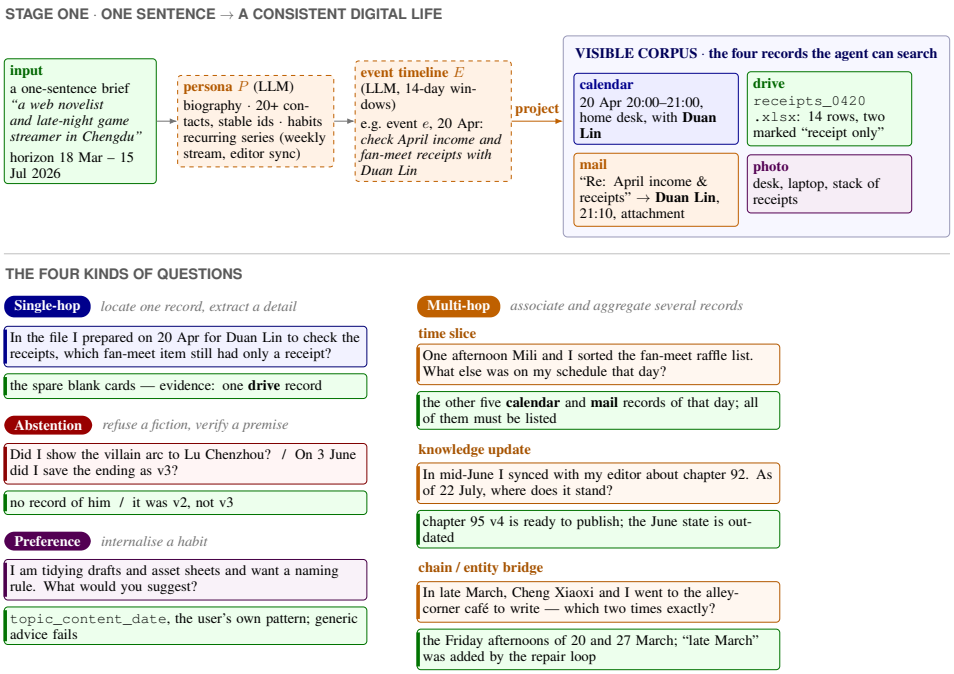

Figure 1: What LIFEFORGE-BENCH asks. Top: from a short free-text brief, Stage one generates a persona and an event timeline (hidden) and projects each event onto the platforms it would realistically touch (visible); the example event touches all four, and its records agree because they come from one event. Bottom: the four kinds of questions (examples translated from Chinese).

persona and an event timeline (Figure 1). Stage one projects each event onto the platforms where such an event would realistically leave traces: a dinner may leave only a photo, while a project meeting may produce an invitation, a follow-up mail and a shared spreadsheet. Records derived from the same event agree by construction, and these records are all a memory system sees. Stage two compiles questions from the hidden structure: the evidence is selected first, an LLM phrases the question from the visible records, and an LLM-based gate filters the candidates. Each question carries a layered ground truth, and those built from specific records also carry a record-level evidence set. Each question covers one of four memory operations: single-hop, multi-hop, abstention and preference. Abstention questions go beyond asking about entities that never existed: many alter one detail of a real record, testing whether a memory system corrects the false premise or answers along with it, a failure mode existing benchmarks rarely probe. Since the evidence is selected before the question is phrased, humans only audit questions and never write or rewrite them.

We evaluate five open-source memory systems (verbatim RAG, Mem0, Letta, EverOS and Graphiti) under one harness, with the same records, reader, judge and retrieval budget, so that score differences come from ingestion and retrieval alone. The evidence sets also let us report retrieval recall and separate the cost of building memory from the cost of using it. Our experiments show that verbatim systems handle single-record questions well and systems that replace records by extracted facts lose document contents; yet most drop sharply once an answer requires aggregating records over time. We summarize our contributions as follows.

- **A pipeline and a benchmark.** LIFEFORGE turns a short free-text brief into months of mutually consistent multi-source data and compiles questions from the same hidden provenance, with no human authoring or rewriting at either stage. Using it, we build LIFEFORGE-BENCH: 11 digital lives (29K records rendered to the systems under test, 3.6K openable office files, 3.6K photos, six countries, records in four languages) whose mail and calendar records follow schemas that real accounts can import, and 1,034 questions spanning four memory operations, eleven subtypes and six preference dimensions, including questions grounded in photo content and premise-perturbation questions built from real records. The compiler needs no human authoring, so the question set grows with the lives. Two lives were also audited item by item by human annotators, 192 questions in all. We report those two separately.

- **A controlled diagnosis of memory systems.** We evaluate five open-source systems (verbatim RAG, Mem0, Letta, EverOS, Graphiti) under the single harness described above. Alongside the score, we report record-level retrieval metrics, ingestion and answering costs, and run-to-run variance. The results show that current systems still fall short on questions that aggregate records over time, and that extraction-based systems lose detailed information at ingestion.

2 RELATED WORK

Memory benchmarks. Most memory benchmarks are built on conversation. LoCoMo (Maharana et al., 2024) expands persona-conditioned event graphs into dialogues with hand-written questions. LongMemEval (Wu et al., 2025) organises questions by what a memory system must do: extraction, multi-session reasoning, temporal reasoning, knowledge update and abstention. BEAM (Tavakoli et al., 2026) extends a dialogue to ten million tokens with LLM-drafted, human-verified questions. MemSim (Zhang et al., 2025) shares our labelling philosophy, since answers are inputs to generation and no verifier is needed, but its histories are isolated statements rather than a life. Later benchmarks add capacity, agentic use, test-time learning and streaming evaluation (Tan et al., 2025; Jiang et al., 2025; Hu et al., 2026; He et al., 2026; Wu et al., 2026a; Wei et al., 2025; Liu et al., 2026; Zhang et al., 2026b; Zhao et al., 2026). All of them take dialogue or plain text as the sole data channel. A second group scatters evidence over several sources. SMMBench (Chai et al., 2026) spreads evidence over dialogue, tables, images and documents; Setoka (Zeng et al., 2026) tests user understanding over heterogeneous records; other benchmarks cover file systems, companion assistants and office workspaces (Yang et al., 2026; Wu et al., 2026b; Tang et al., 2026; Li et al., 2026). LifeBench (Cheng et al., 2026) is the closest to our goal: it simulates one year of phone traces for ten users under a partonomic event hierarchy and writes questions during the simulation so that evidence is grounded. Its traces are text logs, with photos appearing only as entries, and its data are checked post hoc by LLM filtering and manual review. LIFEFORGE emits records that follow real platform schemas (threaded mail, calendar masters with exceptions, openable office files, photos with visual content) and validates every LLM call against the hidden provenance, so questions can ask about document bodies and image content and need only a proofreading audit rather than rewriting.

Multi-source data synthesis. The user data behind such benchmarks comes from a few recipes. Self-Instruct and PersonaHub (Wang et al., 2023; Ge et al., 2024) draw persona-conditioned samples that are mutually independent. Generative Agents (Park et al., 2023) obtains coherent behaviour from step-by-step simulation, at a cost that grows with the horizon. Environment benchmarks such as WebArena, AppWorld and τ-bench (Zhou et al., 2024; Trivedi et al., 2024; Yao et al., 2024) ship mock data as task props. Enron (Klimt & Yang, 2004) remains the only widely used real multi-month mailbox, and it has neither ground truth nor other modalities. None of these yields a self-consistent, multi-platform life with provenance labels. The closest ingredient is event projection. ASTRA-bench (Xiu et al., 2026), a tool-use benchmark, projects a storyline into mail, messages and calendar over about two weeks with human-written queries, and Setoka (Zeng et al., 2026) makes an event tree the unique source of truth for its records. We build on this idea and extend it to span months rather than days, and to emit records that follow real platform schemas (threaded mail and `RRULE` calendars in schemas that real accounts can import, openable office files, face-consistent photos). On the question side, our gates borrow the round-trip filter of synthetic QA (Lewis et al., 2021; Dai et al., 2023), in which an independent model re-answers each question from only what the system under test can see, and add explicit *visibility* and *identifiability* gates on every question, so that no human writes or rewrites a question at any stage.

Memory systems. Long-term memory systems for agents have developed along three lines. The first builds operating-system-like hierarchies of stores that page between context and external memory (MemGPT, MemoryBank, MemoryOS (Packer et al., 2023; Zhong et al., 2024; Kang et al., 2025)). The second compresses raw input into structured memory such as extracted facts, bi-temporal knowledge graphs, linked notes or multi-agent memory kinds (Mem0, Zep with its Graphiti engine, A-MEM, MIRIX (Chhikara et al., 2025; Rasmussen et al., 2025; Xu et al., 2025; Wang & Chen, 2025)). The third learns memory policies (Mem-α, Memory-R1, MemSkill (Wang et al., 2025; Yan et al., 2025; Zhang et al., 2026a)). Their evaluation corpora are almost always dialogue (Zhang et al., 2024; Huang et al., 2026; Du, 2026), so no benchmark could say how they behave on mail, calendar, files and photos. Comparisons between systems are also easy to con-

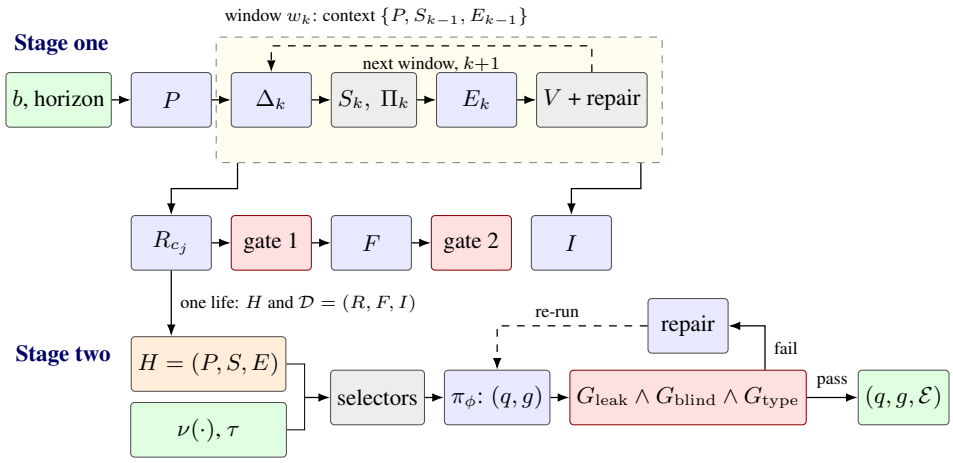

Figure 2: The two stages of LIFEFORGE (blue: LLM calls; grey: code; red: gates). Stage one: each 14-day window conditions only on the persona, the active series and the previous window; the hidden $H = (P, S, E)$ is projected onto four platforms, and two gates stop an inconsistent life. Stage two: selectors on the hidden structure choose the evidence, the question LLM sees only the visible view, and three gate groups keep, repair or drop the candidate.

found: MemDelta (Wang, 2026) shows that the memory method is routinely entangled with the language model, the embedding model and the retrieval pipeline, and that swapping only the embedding model in an otherwise identical pipeline shifts accuracy by 6.2 percentage points. It calls for a fixed embedding model, stratification by model family and reported write-path cost, and our comparison follows this recipe. We place five open-source implementations, namely verbatim RAG, Mem0, Letta, EverOS and Graphiti, on the benchmark under a single harness.

3 LIFEFORGE

LIFEFORGE turns a brief and a horizon into a digital life, and the digital life into a question set; Figure 2 shows the two stages. Stage one generates a hidden *event timeline* as the single source of truth and *projects* every event onto each platform, so that the calendar entry, mail thread, document and photos of one event are written from one object and checked against it. Stage two compiles questions from the hidden structure, which decides *what to ask*, while the fields a memory system can actually see decide *how to ask it*. In both stages every LLM output passes a validator computed in code before it is used, so that questions need no human writing or rewriting.

3.1 GENERATING A DIGITAL LIFE

Let $\Lambda = \{\text{cal}, \text{mail}, \text{drive}, \text{photo}\}$ denote the four platforms. A digital life is $\mathcal{D} = (R, F, I)$: platform records $R = \bigcup_{\lambda \in \Lambda} R_\lambda$, openable files F and images I. LIFEFORGE first samples a hidden *provenance structure* $H = (P, S, E)$—a persona P, recurring series S and an event timeline E—and then projects H onto the platforms. Every record stores the identifier of the event it was projected from, and the Stage-one validators drop a record whose stored revision is stale, so the map $\text{src} : R \to E$ is maintained in code and H serves as ground truth for any question about $\mathcal{D}$.

Let $\pi_\theta(y \mid x)$ be the distribution an LLM induces over structured outputs. The horizon is cut into 14-day windows $w_1, \ldots, w_K$. With S_k the series active after window k, Δ_k the series changes proposed in window k, Π_k the scheduled occurrences and E_k the events of window k,

$$p_\theta(H \mid b) = \pi_\theta(P \mid b) \prod_{k=1}^{K} \pi_\theta\big(\Delta_k \mid P, S_{k-1}, E_{k-1}, w_k\big)\, \pi_\theta\big(E_k \mid P, S_k, \Pi_k, E_{k-1}, w_k\big), \quad (1)$$

where $S_k = \text{Apply}(S_{k-1}, \Delta_k)$ and $\Pi_k = \text{Sched}(S_k, w_k)$ are computed in code. Given H, a second chain projects records chunk by chunk, renders files from a content specification and generates photos from reference portraits; Appendix A gives its factorisation.

Each factor of (1) conditions only on the persona, the compact series state S_{k-1} and the events of the previous window. The prompt therefore does not grow with k, K windows cost $O(K)$ calls, and the recursion can be resumed from any generated prefix, so the horizon is a freely settable parameter rather than a limit set by the context window. The price is that a window cannot see events older than the previous window. Two mechanisms carry information across that boundary: the series state S_k, which carries every recurring commitment, and the `triggered_by` link, by which an event names the event that caused it and which the projection step follows to fetch the records of the referenced event whatever its window. A second consequence of (1) is that the only free inputs are the brief b and the horizon. The brief is short free text, from one sentence to a paragraph; whatever it leaves unspecified (occupation, city, language, social circle, daily habits) is filled in from the LLM's prior, and all eleven released lives were seeded this way.

The chain is executed in four steps. (i) Write the persona: one call maps the brief to a biography, a career, a social network of at least twenty contacts with stable identifiers, a per-platform data profile and structured recurring templates. (ii) Generate the timeline window by window: one call proposes the series delta Δ_k, code expands each cadence into the schedule Π_k because LLMs readily miscompute dates such as "every second Monday", and a second call writes one occurrence per slot plus the ad hoc events of the window. (iii) Project the events in chunks of at most eight, each call seeing the already generated records of the events the chunk refers to. (iv) Materialise files and photos: drive records and attachments are rendered in code into real `.docx`, `.xlsx`, `.pptx` and `.pdf` files from a content specification and accepted only if the format's SDK opens them; photos are generated from one reference portrait per frequent contact and scene prompts conditioned on the participants' portraits. Figure 5 shows a portrait and three scene photos of one protagonist; Appendix C lists the eleven lives.

3.2 Compiling memory questions

A question set is compiled for a reference date τ: only records with event time $\leq \tau$ enter the pool $\mathcal{D}_{\leq\tau}$, every question carries τ, and the system under test answers with its clock set to τ. The part of a record that can be seen is its *visible view* $\nu(r)$, a whitelist computed in code. For mail it is sender, recipients, subject, body, time and attachment names; for calendar, title, time, location, attendees and recurrence rule; for drive, file name, type, creation time and full text. For photos it is the image and its capture time, but no location, file name or scene description. Event descriptions, causal links and series definitions stay hidden. Whatever is used to decide that a question is answerable must be a subset of what the system under test sees.

Which records form a question is decided by selectors on the hidden structure H, one per sub-type, and each of the four types tests one memory operation; Table 3 lists the sub-types with their selectors and ground-truth forms. Single-hop questions involve one record and test locating it and reading a detail; they are allocated to mail, calendar, drive and photo by a platform quota. Multi-hop questions test association and aggregation across records. Chains follow the `triggered_by` links back from an event and ask across three or four events of one causal line. Entity bridges take the projections of one event on two or more platforms, or the records of one contact within one month, and ask what connects them; the platforms carry no native link between these records, so relating them is left to the memory system. Time slices take all mail and calendar records of one day, anchor the day by one of them and ask for all the others, so the evidence set is the whole day minus the anchor rather than a sample. Knowledge updates take a causal line with a state change and ask for the latest state. The selector admits only lines that span at least 30 days and end at most 30 days before τ, code retrieves later records on the same topic from the whole pool, and the selecting LLM call decides whether one of them supersedes the line, in which case the ground truth moves forward. Abstention questions test the boundary and honesty of memory, and we deliberately include two kinds. Fictitious-person and fictitious-activity questions ask about someone or something that never occurred and test whether the system fabricates; a candidate entity is accepted only if a search computed in code finds zero hits in a superset corpus that includes spam mail, unpublished photos and contact-name variants. Memory-conflict questions take a real record, corrupt one detail (time, person, place, number or conclusion) and pose it as a premise; they test whether the system follows the wrong premise or checks it against memory and points out the mismatch, which we find in practice to be the more frequent situation. Preference questions test whether a habit has been internalised rather than whether a fact can be found. An LLM call mines candidate habits from batches of visible views, code keeps a habit only if at least three pool records support it, and the question asks for advice whose quality depends on

the habit. As a habit is an aggregate rather than a single record, these questions carry no evidence set and skip the uniqueness gates below.

A single LLM call then sees only the visible views of the selected records and writes the question q and a layered ground truth g. An evidence-bearing question ships with q, g, τ and an evidence set $\mathcal{E}$ whose entries resolve to online assets, so that retrieval metrics can be computed beside the judge's verdict. Grading follows only the rubric declared by the ground truth, the judge may not add rules. Appendix F gives the layered form of the ground truth and the judge protocol.

3.3 Validation as constrained sampling

Every LLM call in either stage is used through a validator V computed in code, with target

$$\tilde{\pi}(y \mid x) \;\propto\; \pi_\theta(y \mid x)\, \mathbf{1}\big[V(y; x) = 1\big]. \tag{2}$$

Eq. (2) specifies the support we want; we approximate it by validate-and-repair rather than by rejection sampling. A failing output goes to a repair call that sees only the failing items, under a bounded budget. In Stage one the need is practical. Current frontier models make a handful of errors over several hundred calls: miscomputed dates, schema violations, references to contacts or files that do not exist, two files sharing one name. In a chain whose contexts depend on each other these errors propagate: a wrong series change in window k shifts the whole schedule of window $k{+}1$. Checking by hand after generation would show downstream symptoms rather than upstream causes and force regenerating everything after the fix. Table 2 lists the rules applied at each step and the repair or fallback on failure. A chunk receives at most three repair calls, and items that cannot be repaired fall back to a conservative default such as a canceled occurrence. Two gates then re-check the persisted data as a whole. Gate 1 (`verify_core`) runs after projection and checks every record against the schema, the social network and the event revisions. Gate 2 (`verify_assets`) runs after files are rendered, as the last gate before a life is released, and checks that every file opens and agrees with its manifest. A gate that fails stops the pipeline and persists a report.

In Stage two, a candidate (q, g) written for an evidence structure s is kept only if it passes three gates:

$$\tilde{p}(q, g \mid s) \;\propto\; \pi_\phi\big(q, g \mid \nu(s), \mathrm{ctx}(s)\big)\, \mathbf{1}\big[G_{\mathrm{leak}}(q, g, s) \wedge G_{\mathrm{blind}}\big(q, g, \nu(s), \mathcal{K}(s)\big) \wedge G_{\mathrm{type}}(q, g, s)\big], \tag{3}$$

Here π_ϕ is the question-writing LLM call and $\mathcal{K}(s)$ are the five pool records lexically closest to q, which we call its *competitors*. The context $\mathrm{ctx}(s)$ contains a persona summary and three *identifiability facts* computed in code: how many pool items satisfy the predicate of s before subsampling, on how many dates each record's title recurs, and which records can serve as unique anchors.

G_{leak} is a code scan: neither q nor g may contain an internal identifier or a hidden photo field, and q may not contain a core-answer string. G_{blind} is a second LLM call that sees only $\nu(s)$ and $\mathcal{K}(s)$ and checks three properties: correctness (q is answerable from the evidence alone and its answer agrees with g), locatability (q singles out its evidence among the competitors and its scope words suffice to narrow the pool), and form (q does not reveal its answer and asks for recall rather than an action). G_{type} adds type-specific checks: for knowledge updates, code asserts $t_{\mathrm{old}} < t_{\mathrm{new}} \le \tau$ and the LLM confirms that the two states differ; for fictitious entities, code asserts zero hits in the superset corpus; for memory conflicts, the LLM confirms that the source record is locatable and that the premise contradicts it.

The requirement behind locatability is that q matches exactly its evidence set on the full pool, $|\{x \in \mathcal{D}_{\le\tau} : x \text{ satisfies } q\}| = |\mathcal{E}|$. This cannot be checked exactly, so the identifiability facts are given to the question writer as constraints and to G_{blind} as context, and G_{blind} makes the final judgement.

Candidates failing for non-uniqueness, missing scope or answer leakage are sent to a repair call that may add locating cues—approximate time, topic or people—but may not change the answer. The one exception is a photo question whose answer depends on a detail unreadable at upload resolution; its ground truth is coarsened instead. The repaired candidate re-enters all three gates, and is dropped after two unsuccessful rounds. Figure 4 illustrates the loop, and Appendix B gives the per-type selectors, thresholds and worked examples.

4 EXPERIMENTS

4.1 SETUP

Dataset. LIFEFORGE-BENCH consists of 11 digital lives and the 1,034 questions compiled from them, one set per life at that life's reference date; statistics, generation cost, photo examples and per-life details are in Appendix C. We score every system on all 1,034. The 192 questions of two of the three Chinese-language lives were additionally checked one by one against the raw records by native Chinese speakers on our team. The other nine sets, in languages our annotators do not read, are raw compiler output. Appendix D measures what that output is worth: record pass rates of 85%–99% by platform in outsourced review, 87.5% of raw compiler questions rated directly usable and a further 11.3% usable with minor flaws, mostly questions that read as machine-written, and 93.8% judge–human agreement.

Systems. We evaluate five open-source memory systems as store-and-retrieve layers. Verbatim RAG embeds every record whole and performs no extraction; Mem0 uses an LLM to extract facts and merges them with existing memories; Letta is the successor of MemGPT, and we use only its archival store; EverOS extracts memories through a cascade of episodes, facts, and profiles; Graphiti is the engine behind Zep and builds a temporal entity graph. The three systems that extract share one extraction LLM (GPT-4.1-mini, temperature 0), and all five share one embedding model. They ingest the same records, one record per message. We disable their conversational agents and agentic retrieval modes. Each system returns its $k = 30$ most relevant memories. A fixed *reader*, the LLM that writes the answer from the question and the retrieved memories alone (GPT-5.6-sol, with its clock frozen at τ), is shared by all systems. Score differences therefore arise only from ingestion and retrieval. A GPT-5-mini judge grades all arms in a single run. Appendix E gives the system versions, batching details and retrieval settings.

Metrics. The primary metric is the **weighted score**. It equals the percentage of questions judged PASS plus half the percentage judged PARTIAL. We also report the **PASS rate**. The systems store memories in different units, so we define retrieval metrics at the record level. The harness logs which records each memory was generated from, and a returned memory counts as a hit when one of those records belongs to the question's evidence set. On the 781 questions that carry an evidence set we report **Recall@k**, **Precision@k**, **full recall** (the share of questions whose evidence set is retrieved in full) and **MRR**, all computed over the memories that actually reach the reader. We also report **reader input tokens** per question (Table 8) and the **ingestion cost** for all 28,952 records. Mem0, EverOS and Graphiti attribute a memory to more records than it came from, which biases their retrieval metrics upward; Appendix G gives the mechanism for each.

4.2 MAIN RESULTS

Table 1 reports the weighted score of each system by sub-type, together with its retrieval metrics and costs. The two systems that keep records whole lead, Letta at 77.9 and verbatim RAG at 76.5. Graphiti follows at 71.8, and EverOS (68.2) and Mem0 (67.1) trail by about ten points. The Oracle column bounds what retrieval can still gain. It gives the same reader the gold evidence records instead of retrieved memories and changes nothing else, so the difference isolates retrieval. On the 781 questions that carry an evidence set it scores 93.5, against 74.8 for the best system on those same questions; on multi-hop questions it scores 91.3 against 59.3. The multi-hop gap therefore lies in retrieval, not in the reader. Photo questions stay at 59.5 even with perfect retrieval, because the reader sees a caption rather than the image.

Verbatim RAG keeps every record whole and has the highest Recall@30 (80.6). It scores 83.1–97.4 on mail, calendar and drive, but only 11.5 on time slices, where a question spanning a whole day exceeds what the reader can assemble from 30 unordered memories. Letta stores one archival passage per record and scores highest overall despite the lowest Recall@30 (62.4), because what it does retrieve is kept verbatim. Mem0 extracts facts from chunks of at most eight records. Its Recall@30 (78.2) is 2.4 points below RAG's, yet it scores nine points lower overall, because the extracted facts drop the details the questions ask about (48.7 on drive against RAG's 83.1, 78.8 on memory conflicts against 91.7); its ingestion is also the only one that is not reproducible, two runs of the same configuration differing by 7.3 points (Appendix H). EverOS merges the records of each

Table 1: Five open-source memory systems on the 1,034 questions of the eleven lives (weighted score, %; one harness, records up to each life's τ, $k = 30$, one reader, one judge run; best of the five systems in bold). Recall@30 is over the 781 questions with an evidence set; ingestion cost is for all 28,952 records.

		n	RAG	Mem0	Letta	EverOS	Graphiti	Oracle (n=781)
Overall weighted score		1,034	76.5	67.1	**77.9**	68.2	71.8	93.5
PASS rate		1,034	71.9	60.4	**72.9**	61.1	66.5	90.7
Single-hop	mail	119	95.4	78.6	**95.8**	79.4	91.2	100
	calendar	77	**97.4**	93.5	**97.4**	81.2	79.2	100
	drive	77	**83.1**	48.7	80.5	49.4	77.9	98.7
	photo	37	33.8	36.5	31.1	**37.8**	9.5	59.5
Multi-hop	chain	132	67.8	53.8	**71.2**	61.4	59.1	94.7
	entity bridge	99	64.6	55.1	**71.7**	49.0	63.1	97.0
	time slice	61	11.5	7.4	12.3	**41.0**	11.5	77.0
	knowledge update	47	52.1	51.1	**60.6**	54.3	57.4	88.3
Abstention	memory conflict	132	**91.7**	78.8	**91.7**	77.3	88.6	96.2
	fictitious entity	88	**100**	**100**	**100**	**100**	**100**	–
Preference	all dimensions	165	80.0	79.4	**80.3**	76.7	78.8	–
Recall@30 (781 questions, %)			**80.6**	78.2	62.4	67.1	68.9	–
reader input tokens / question (mean)			9.3K	2.9K	8.6K	9.4K	12.7K	1.6K
ingestion: LLM calls / tokens / hours			0 / 0 / 3.8	6.2K / 76M / 12.7	0 / 0 / 6.1	– / – / 6.9	186K / 693M / 104	–

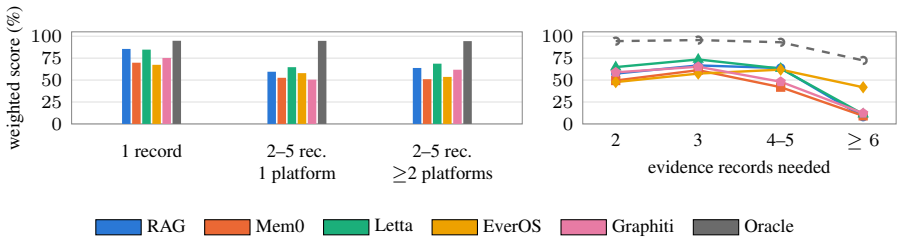

Figure 3: Scores against the spread of the evidence (weighted score, %, $k = 30$, reader GPT-5.6-sol; Oracle is the same reader given the gold evidence). Left: single-record questions, and multi-hop questions with 2–5 evidence records whose evidence lies on one platform or spans two or more. Right: multi-hop questions grouped by the number of evidence records they need.

period into episodes at ingestion. It is the only system above 13 on time slices (41.0) and the only one that does not collapse when six or more records are needed, and it pays for this with the same detail loss as Mem0 on drive (49.4). Graphiti builds a temporal entity graph from one episode per record, at 186K LLM calls and 693M tokens against none for the two verbatim stores, and buys third place with them while staying low on time slices (11.5) and photos (9.5).

Figure 3 shows what makes a multi-hop question hard. The number of evidence records an answer needs is decisive. Up to five records the systems sit between 41.9 and 73.5. At six or more every system except EverOS collapses to 8.2–12.2, while EverOS retains 41.8 and the Oracle reader still scores 72.4, so the questions remain answerable and it is the retrieval that fails. Whether those records sit on one platform or several barely moves the score, in either direction. Table 9 gives the grouped numbers.

4.3 Ablations

The main results fix the reader and $k = 30$. We vary each of them with the same systems and judge. The ablations run on the 192 questions that were audited item by item. Appendix H gives the tables and adds ablations on Mem0's extraction model, on Graphiti's episode granularity and on run-to-run variance.

Effect of the reader. Two further readers from the same series change every system by at most 3 points and leave the leading group unchanged. A smaller reader (GPT-4.1-mini) costs the verbatim stores, EverOS and Graphiti 9.6–13.1 points, but costs Mem0 only 0.6, because the details lost at extraction were never in its context. As the reader improves, the gap between Mem0 and verbatim RAG therefore widens from 3.9 points to more than 16. A reader can use what memory has kept,

but it cannot recover what ingestion has lost. The low multi-hop scores are also not an artefact of a weak reader: the strongest reader with the best system reaches only 56.6 on multi-hop questions.

Effect of k. Retrieving more memories helps every system up to $k = 20$, by 9.9–15.3 points over $k = 5$. Beyond $k = 20$ the picture is system-specific: every system except RAG keeps gaining on multi-hop questions, RAG loses two points there, and Graphiti gains ten on single-hop.

4.4 Analysis

Gaps in current evaluation. Existing memory benchmarks score answer accuracy, and a few add retrieval recall. The cost of building and using memory is rarely reported, and mostly only in system papers. Our harness also measures ingestion cost and reader input tokens, and Table 1 shows that neither moves in step with quality. Letta scores highest without a single LLM call at ingestion. Graphiti spends by far the most on ingestion and on reader tokens, yet scores no higher than verbatim RAG. Mem0 gives the reader the shortest context but scores lowest. Existing benchmarks also build memory from conversations alone, so their evidence never spans platforms and never accumulates. Such a benchmark can ask what a passage of dialogue said, but not what happened on one day across four accounts. Figure 3 shows that the second kind of question is where the systems separate.

Weaknesses of current memory systems. Three weaknesses emerge from the results above. *(i) Current systems handle records in isolation, not the relations between them.* Multi-hop questions are answered well only when the full evidence set reaches the reader. That happens for 20.1–42.8% of multi-hop questions and for 74.5–90.3% of single-hop ones (Appendix G). Figure 3 shows that once six or more records are required, every system but EverOS falls below 13, while the Oracle reader still scores 72.4 on the same questions. Time slices, which ask for everything that happened on a single day, are the hardest type of all: four of the five systems score below 13. The problem is the unit of retrieval. All five systems retrieve by semantic similarity, but the records of one day share a date rather than a topic, so no single query recovers them all. EverOS merges each period into an episode at ingestion, and it is the one exception. Memory should therefore be organised around how records relate in time and to people, not around topic alone. *(ii) Extraction discards the details that questions ask about.* Table 1 shows that systems which replace a record by extracted facts fall far behind the verbatim stores on questions about document contents. The reader ablation in §4.3 shows that a stronger reader widens this gap rather than closing it. What ingestion has discarded is unavailable to the reader however capable that reader is. The same loss undermines fact checking, because a corrupted date or number can be detected only against the original record. Every system solves fictitious-entity questions, but Mem0 scores 12.9 points below verbatim RAG on memory conflicts (78.8 against 91.7) even though its Recall@30 on those questions is within three points of RAG's. Extraction should therefore be layered over verbatim records rather than replacing them. *(iii) Photos remain outside memory.* No current system stores multimodal content, so in our evaluation photos could only be fed in as captions. How to represent, index and retrieve a user's multimodal content within a memory system remains an open question.

5 Conclusion and future work

We presented a method for building multi-source memory benchmarks. It expands a short brief into an event timeline, renders the events as mail, calendar entries, drive files, and photos, and tracks the source event of every record. From this provenance, it compiles questions with a unique evidence set and a grading rubric. We release eleven lives with 1,034 compiled questions.

We evaluated five open-source memory systems under one harness that measures cost as well as accuracy and retrieval. They retrieve single records well, which is what keeps the two verbatim stores in front. They also aggregate records over time poorly, lose details at extraction that a stronger reader cannot recover, and accept photos only as captions.

Building on our findings, we will also design memory systems that aggregate records by time, keep verbatim records beside extracted facts, and store photos as native memory.

AI USE STATEMENT

Generative AI is both the object of study and an engine of our framework, and we disclose its use in full. First, we used LLMs and image models to generate synthetic datasets: all lives, records, photos, and questions in LIFEFORGE-BENCH were produced under the deterministic constraints described in the paper, and an LLM judge is part of the evaluation protocol; the paper reports the human review of these artifacts and of the judge. Second, we used LLM assistance to edit the manuscript for readability. Third, we used LLM assistance to write parts of the supporting code and to search for related literature. We did not use generative AI tools to develop theoretical models or conceptual frameworks, to formulate mathematical claims or assist in proving them, to propose or refine hypotheses, to design or give feedback on our research methodology or experiments, or to interpret results. We have reviewed all AI-assisted work. The authors reviewed and tested all code. All quantitative claims were checked against the generation and evaluation logs. The edited text was checked against our draft, and the related-work survey was verified manually against the cited papers. We take responsibility for the final content of this work, including its text, claims, and artifacts.

ETHICS STATEMENT

All data in LIFEFORGE-BENCH is synthetic. Personas, contacts, addresses, organisations, documents and photos are generated from short free-text briefs; no real person's records were used at any stage, and generated email addresses use fictitious domains. Generated faces are synthetic and not conditioned on images of real people. Outsourced review of generated records was carried out by contracted annotators, and the release audit of the questions by members of our team; both saw only synthetic data. The benchmark evaluates memory systems that may operate on real personal data; we encourage users to treat abstention and memory-conflict scores as first-class, since fabricating personal facts is a direct harm.

REPRODUCIBILITY STATEMENT

The generator, the question compiler, the validators, the judge prompt, the evaluation harness, the eleven lives with their hidden provenance and the 1,034 questions will be released. §3.1 and Appendix A specify the deterministic operators; Appendices A and B list models, temperatures, chunk sizes and thresholds; Appendix E gives the versions and settings of every evaluated system; all random choices in question compilation are seeded by the life identifier, the reference date and the structure identifier, and question identifiers are content hashes, so re-running the compiler on the same life and date reproduces the same set. Every LLM call, validator report and dropped candidate is persisted, and the gate statistics of Table 6 are computed from these logs.

REFERENCES

Anthropic. Use Google Workspace connectors. Claude Help Center, `https://support.claude.com/en/articles/10166901-use-google-workspace-connectors`, 2026. Accessed September 2026.

Apple. Apple Intelligence and Siri. `https://www.apple.com/apple-intelligence/`, 2026. Accessed September 2026.

Huacan Chai, Yukai Wang, Yingxuan Yang, Dan Peng, Yuanyi Song, Zhihui Fu, Weiwen Liu, Jianghao Lin, Jun Wang, and Weinan Zhang. SMMBench: A benchmark for source-distributed multimodal agent memory. *arXiv preprint arXiv:2605.15710*, 2026.

Zihao Cheng, Weixin Wang, Yu Zhao, Ziyang Ren, Jiaxuan Chen, Ruiyang Xu, Shuai Huang, Yang Chen, Guowei Li, Mengshi Wang, Yi Xie, Ren Zhu, Zeren Jiang, Keda Lu, Yihong Li, Xiaoliang Wang, Liwei Liu, and Cam-Tu Nguyen. LifeBench: A benchmark for long-horizon multi-source memory. *arXiv preprint arXiv:2603.03781*, 2026.

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.

Zhuyun Dai, Vincent Y. Zhao, Ji Ma, Yi Luan, Jianmo Ni, Jing Lu, Anton Bakalov, Kelvin Guu, Keith B. Hall, and Ming-Wei Chang. Promptagator: Few-shot dense retrieval from 8 examples. In *International Conference on Learning Representations (ICLR)*, 2023.

Pengfei Du. Memory for autonomous LLM agents: Mechanisms, evaluation, and emerging frontiers. *arXiv preprint arXiv:2603.07670*, 2026.

Tao Ge, Xin Chan, Xiaoyang Wang, Dian Yu, Haitao Mi, and Dong Yu. Scaling synthetic data creation with 1,000,000,000 personas. *arXiv preprint arXiv:2406.20094*, 2024.

Google. Personal intelligence: Connecting Gemini to Google apps. The Keyword, `https://blog.google/innovation-and-ai/products/gemini-app/personal-intelligence/`, 2026. Accessed September 2026.

Zexue He, Yu Wang, Churan Zhi, Yuanzhe Hu, Tzu-Ping Chen, Lang Yin, Ze Chen, Tong Arthur Wu, Siru Ouyang, Zihan Wang, Jiaxin Pei, Julian McAuley, Yejin Choi, and Alex Pentland. MemoryArena: Benchmarking agent memory in interdependent multi-session agentic tasks. In *International Conference on Machine Learning (ICML)*, 2026.

Yuanzhe Hu, Yu Wang, and Julian McAuley. Evaluating memory in LLM agents via incremental multi-turn interactions. In *International Conference on Learning Representations (ICLR)*, 2026.

Wei-Chieh Huang, Weizhi Zhang, Yueqing Liang, Yuanchen Bei, Yankai Chen, Tao Feng, et al. A survey of agent memory in the second half: Towards self-evolving and long-horizon agents. *arXiv preprint arXiv:2602.06052*, 2026.

Bowen Jiang, Zhuoqun Hao, Young-Min Cho, Bryan Li, Yuan Yuan, Sihao Chen, Lyle Ungar, Camillo J. Taylor, and Dan Roth. Know me, respond to me: Benchmarking LLMs for dynamic user profiling and personalized responses at scale. *arXiv preprint arXiv:2504.14225*, 2025.

Jiazheng Kang, Mingming Ji, Zhe Zhao, and Ting Bai. Memory OS of AI agent. *arXiv preprint arXiv:2506.06326*, 2025.

Bryan Klimt and Yiming Yang. The Enron corpus: A new dataset for email classification research. In *European Conference on Machine Learning (ECML)*, 2004.

Patrick Lewis, Yuxiang Wu, Linqing Liu, Pasquale Minervini, Heinrich Küttler, Aleksandra Piktus, Pontus Stenetorp, and Sebastian Riedel. PAQ: 65 million probably-asked questions and what you can do with them. *Transactions of the Association for Computational Linguistics*, 9:1098–1115, 2021.

Boyan Li, Zhuowen Liang, Yupeng Xie, Xiaotian Lin, Tianqi Luo, Xinyu Liu, Yizhang Zhu, Zhangyang Peng, Yuan Li, Zhengxuan Zhang, Jiayi Zhang, Nan Tang, Guoliang Li, and Yuyu Luo. DataSpace: Benchmarking data agents for verifiable analytics over heterogeneous workspaces. *arXiv preprint arXiv:2608.03451*, 2026.

Guanming Liu, Yuqi Ren, Hansu Gu, Peng Zhang, Weihang Wang, Jiahao Liu, Ning Gu, and Tun Lu. StreamMemBench: Streaming evaluation of agent memory for future-oriented assistance. *arXiv preprint arXiv:2606.14571*, 2026.

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 (ACL)*, 2024.

Microsoft. Microsoft 365 Copilot enhanced personalization control. Microsoft Learn, `https://learn.microsoft.com/en-us/graph/control-enhanced-personalization-privacy`, 2025. Accessed September 2026.

OpenAI. Google app for ChatGPT: Data controls FAQ. OpenAI Help Center, `https://help.openai.com/en/articles/10408842-google-connector-for-chatgpt-data-controls-faq`, 2026. Accessed September 2026.

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 C. O'Brien, Carrie J. 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 (UIST)*, 2023.

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.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. GPQA: A graduate-level Google-proof Q&A benchmark. In *First Conference on Language Modeling (COLM)*, 2024.

Haoran Tan, Zeyu Zhang, Chen Ma, Xu Chen, Quanyu Dai, and Zhenhua Dong. MemBench: Towards more comprehensive evaluation on the memory of LLM-based agents. *arXiv preprint arXiv:2506.21605*, 2025.

Zirui Tang, Xuanhe Zhou, Yumou Liu, Linchun Li, Yukai Wu, Weizheng Wang, Hongzhang Huang, Wei Zhou, Jun Zhou, Jiachen Song, Shaoli Yu, Jinqi Wang, Zihang Zhou, Hongyi Zhou, Yuting Lv, Jinyang Li, Jiashuo Liu, Ruoyu Chen, Chunwei Liu, Guoliang Li, Jihua Kang, and Fan Wu. Workspace-Bench 1.0: Benchmarking AI agents on workspace tasks with large-scale file dependencies. *arXiv preprint arXiv:2605.03596*, 2026.

Mohammad Tavakoli, Alireza Salemi, Carrie Ye, Mohamed Abdalla, Hamed Zamani, and J. Ross Mitchell. Beyond a million tokens: Benchmarking and enhancing long-term memory in LLMs. In *International Conference on Learning Representations (ICLR)*, 2026.

Harsh Trivedi, Tushar Khot, Mareike Hartmann, Ruskin Manku, Vinty Dong, Edward Li, Shashank Gupta, Ashish Sabharwal, and Niranjan Balasubramanian. AppWorld: A controllable world of apps and people for benchmarking interactive coding agents. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (ACL)*, 2024.

Kuan Wang. MemDelta: Controlled baselines and hidden confounds in agent memory evaluation. *arXiv preprint arXiv:2606.29914*, 2026.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. Self-instruct: Aligning language models with self-generated instructions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (ACL)*, 2023.

Yu Wang and Xi Chen. MIRIX: Multi-agent memory system for LLM-based agents. *arXiv preprint arXiv:2507.07957*, 2025.

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

Tianxin Wei, Noveen Sachdeva, Benjamin Coleman, Zhankui He, Yuanchen Bei, Xuying Ning, Mengting Ai, Yunzhe Li, Jingrui He, Ed H. Chi, Chi Wang, Shuo Chen, Fernando Pereira, Wang-Cheng Kang, and Derek Zhiyuan Cheng. Evo-Memory: Benchmarking LLM agent test-time learning with self-evolving memory. *arXiv preprint arXiv:2511.20857*, 2025.

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.

Di Wu, Zixiang Ji, Asmi Kawatkar, Bryan Kwan, Jia-Chen Gu, Nanyun Peng, and Kai-Wei Chang. LongMemEval-V2: Evaluating long-term agent memory toward experienced colleagues. *arXiv preprint arXiv:2605.12493*, 2026a.

Tingyu Wu, Zhisheng Chen, Ziyan Weng, Shuhe Wang, Chenglong Li, Shuo Zhang, Sen Hu, Silin Wu, Qizhen Lan, Huacan Wang, and Ronghao Chen. KnowMe-Bench: Benchmarking person understanding for lifelong digital companions. *arXiv preprint arXiv:2601.04745*, 2026b.

Zidi Xiu, David Q. Sun, Kevin Cheng, Maitrik Patel, Josh Date, Yizhe Zhang, Jiarui Lu, Omar Attia, Raviteja Vemulapalli, Oncel Tuzel, Meng Cao, and Samy Bengio. ASTRA-bench: Evaluating tool-use agent reasoning and action planning with personal user context. *arXiv preprint arXiv:2603.01357*, 2026.

Wujiang Xu, Zujie Liang, Kai Mei, Hang Gao, Juntao Tan, and Yongfeng Zhang. A-MEM: Agentic memory for LLM agents. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

Sikuan Yan, Xiufeng Yang, Zuchao Huang, Ercong Nie, Zifeng Ding, Zonggen Li, Xiaowen Ma, Jinhe Bi, Kristian Kersting, Jeff Z. Pan, Hinrich Schütze, Volker Tresp, and Yunpu Ma. Memory-R1: Enhancing large language model agents to manage and utilize memories via reinforcement learning. *arXiv preprint arXiv:2508.19828*, 2025.

Zhe Yang, Shulin Tian, Kairui Hu, Shuai Liu, Hoang-Nhat Nguyen, Yichi Zhang, Zujin Guo, Mengying Yu, Zinan Zhang, Jingkang Yang, Chen Change Loy, and Ziwei Liu. HippoCamp: Benchmarking contextual agents on personal computers. *arXiv preprint arXiv:2604.01221*, 2026.

Shunyu Yao, Noah Shinn, Pedram Razavi, and Karthik Narasimhan. τ-bench: A benchmark for tool-agent-user interaction in real-world domains. *arXiv preprint arXiv:2406.12045*, 2024.

Lingyang Zeng, Guangze Chen, Kaichen Yu, Zhicheng Pan, Siyang Weng, Zirui Hu, Xiangyun Du, Hailin He, Rong Zhang, Chengcheng Yang, Kai Huang, and Xuan Zhou. Setoka: A benchmark for hierarchical user understanding in personalized agents over heterogeneous data. *arXiv preprint arXiv:2607.27056*, 2026.

Haozhen Zhang, Quanyu Long, Jianzhu Bao, Tao Feng, Weizhi Zhang, Haodong Yue, and Wenya Wang. MemSkill: Learning and evolving memory skills for self-evolving agents. *arXiv preprint arXiv:2602.02474*, 2026a.

Weizhi Zhang, Xiaokai Wei, Wei-Chieh Huang, Zheng Hui, Chen Wang, Michelle Gong, and Philip S. Yu. MemoryCD: Benchmarking long-context user memory of LLM agents for lifelong cross-domain personalization. *arXiv preprint arXiv:2603.25973*, 2026b.

Zeyu Zhang, Xiaohe Bo, Chen Ma, Rui Li, Xu Chen, Quanyu Dai, Jieming Zhu, Zhenhua Dong, and Ji-Rong Wen. A survey on the memory mechanism of large language model based agents. *arXiv preprint arXiv:2404.13501*, 2024.

Zeyu Zhang, Quanyu Dai, Luyu Chen, Zeren Jiang, Rui Li, Jieming Zhu, Xu Chen, Yi Xie, Zhenhua Dong, and Ji-Rong Wen. MemSim: A Bayesian simulator for evaluating memory of LLM-based personal assistants. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

Yujie Zhao, Boqin Yuan, Junbo Huang, Haocheng Yuan, Zhongming Yu, Haozhou Xu, Lanxiang Hu, Abhilash Shankarampeta, Zimeng Huang, Wentao Ni, Yuandong Tian, and Jishen Zhao. AMA-Bench: Evaluating long-horizon memory for agentic applications. *arXiv preprint arXiv:2602.22769*, 2026.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging LLM-as-a-judge with MT-bench and chatbot arena. In *Advances in Neural Information Processing Systems (NeurIPS), Datasets and Benchmarks Track*, 2023.

Wanjun Zhong, Lianghong Guo, Qiqi Gao, He Ye, and Yanlin Wang. MemoryBank: Enhancing large language models with long-term memory. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 19724–19731, 2024.

Shuyan Zhou, Frank F. Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, Uri Alon, and Graham Neubig. WebArena: A realistic web environment for building autonomous agents. In *International Conference on Learning Representations (ICLR)*, 2024.

A CONSTRUCTION DETAILS AND VALIDATORS OF STAGE ONE

Given H, the observable life is generated as

$$p_\theta(\mathcal{D} \mid H) = \prod_{j=1}^{J} \pi_\theta\big(R_{c_j} \mid P, c_j, R_{\mathrm{rel}(c_j)}\big) \prod_{d} \pi_\theta\big(F_d \mid \sigma_d, d, F_{\mathrm{prior}(d)}\big) \times \prod_{r \in R_{\mathrm{photo}}} \psi\big(I_r \mid \mathrm{prompt}_\theta(r, A, \mathrm{sib}(r)),\, I^0\big), \tag{4}$$

where $c_1, \ldots, c_J$ are the chunks of at most eight events into which the timeline is cut, $R_{\mathrm{rel}(c_j)}$ are the records already generated for the events that c_j refers to through `triggered_by`, σ_d is the content specification of file d, $F_{\mathrm{prior}(d)}$ is the earlier related file that d continues (empty if none), A is the structured appearance description of the persona, $\mathrm{sib}(r)$ are the photos of the same event as r, and ψ is an image model conditioned on the reference portraits I^0 of the participants. The chain of (1) and (4) is executed in four steps: write the persona, generate the timeline window by window, project the events onto the platforms, and finally materialise files and photos.

Persona. One call maps the brief to a persona with five typed parts: a biography, a career, a social network $\mathcal{C}$ of at least twenty contacts, a per-platform data profile, and life patterns with structured recurring templates (cadence, weekday set, time, participants, expected projections). Every contact receives a stable identifier; people are referred to by identifier and rendered as names only when records are written, so a name that is translated or abbreviated on one platform still resolves to the same person.

Windowed timeline. Each window makes two calls. The first proposes the *series delta* Δ_k: series that start, end or change. The second receives the schedule preview Π_k, computed in code from each cadence, and writes one occurrence per scheduled slot plus the ad hoc events of the window. Occurrences carry a state in $\{$`none`, `rescheduled`, `canceled`, `terminated`, `patched`$\}$ together with scheduled and actual times, so a moved meeting is one event rather than two; events carry an optional `triggered_by` link, which later lets a projection continue an email thread and lets Stage two find causal chains. Cadences are expanded in code because LLMs are unreliable at calendar arithmetic and readily miscompute dates such as "every second Monday".

Projection. Events declare which platforms they touch and are projected in chunks of at most eight; each call receives the persona, the chunk, and the already-generated records of every event the chunk refers to as read-only context. Each recurring series is projected once into a calendar master with an `RRULE`, and only rescheduled occurrences receive an exception row, as a real client stores them. Records carry the fields usual on each platform (threads, read flags, spam labels and attachment specifications for mail; type, creator, sharing and summary for drive; time, location, description and tags for photos).

Files and photos. Drive records and standalone attachments become real `.docx`, `.xlsx`, `.pptx` and `.pdf` files: a content specification σ_d (language, length, sections, tables) comes first, and content is generated in *continuation* mode when an earlier related file exists, so that a v2 of the minutes continues v1; rendering is done in code, and a file is accepted only if the format's SDK can open it. Photos take three steps: a structured appearance description of the persona, one reference portrait for the protagonist and each frequent contact, and scene photos generated with the participants' portraits as reference images, siblings of one event sharing environment and lighting (Figure 5).

Models, sizes and prompts. Stage one used a frontier LLM (GPT-5.5) for persona, timeline, projection and file content with temperatures 0.4 (persona), 0.5 (timeline), 0.35 (projection) and 0.5 (files); attachment-mention checks run at temperature 0. Reference portraits were generated with a text-to-image model (GPT-Image-2) and scene photos with a reference-conditioned model (Gemini-3-Pro-Image) at 1024×1024; photos are down-scaled to a 720-pixel long side before upload, and the visible view of a photo is defined at that resolution. Windows are 14 days, projection chunks hold 8 events, and a chunk receives at most 3 repair calls. Stage one uses 17 prompt pairs (persona; series delta; events; occurrence repair; projection; recurring master; three repair prompts; attachment

semantics; content specification; document, sheet and slide content; appearance; portrait prompt; scene prompt).

Validators and gates. Table 2 lists, for every step of Stage one, the checks computed in code and the repair or fallback applied when a check fails; the two gates re-run the accumulated checks on the persisted data and halt the pipeline instead of repairing.

Table 2: Deterministic validators in Stage one and the repair or fallback applied on failure.

Stage	Checks	On failure
Persona	schema of the five parts; contact and template identifier syntax; cadence grammar (weekly: weekday set; monthly: month-day or set position); closed vocabularies for projections and frequencies	warnings persisted; pipeline continues only if the schema holds
Timeline	one occurrence per scheduled slot; no duplicate `series_id`+`scheduled_time`; legal `change_type`/`state`/`actual_time` combinations; participants $\subseteq$ contact ids; a rescheduled time lies between neighbouring slots; series never edited in place	targeted repair of the single offending occurrence; fallback to a canceled occurrence
Projection	projection contract (every declared platform has a record); drive references resolve; attachment specifications normalised; unique drive and photo file names; exactly one master row per calendar series; anchor and source revisions equal the current event revision	three targeted repair kinds (missing projection, drive name collision, photo name collision), ≤ 3 per chunk; stale records dropped and regenerated
Gate 1 (`verify_core`)	all of the above on the persisted files; names closed under the social network; naive local time stamps; attachment extensions in the allowed set; LLM check that a body semantically mentions its attachment	pipeline halts; report persisted
Files	content specification present; rendered file opens with the format's SDK; extension matches the actual format; manifest, file path and disk agree; no record left in a placeholder or stale state	fallback from continuation to direct generation; `repair` command
Gate 2 (`verify_assets`)	all of the above; re-run before a life is released	pipeline halts
Images	reference portrait exists for every participant that appears; sibling photos share scene text; no legible text requested	regeneration of the single image

B QUESTION TYPES, SELECTORS, GATES AND EXAMPLES IN STAGE TWO

Table 3: Question types, sub-types, selectors and ground-truth form. Selectors run on the hidden structure H; phrasing sees only the visible view.

Type (operation): sub-types	**Selector on H**	**Ground truth**
Single-hop (locate, extract): mail, calendar, drive, photo	one record; platform quota, realised 38/25/25/12%	core / detail / wrong
Multi-hop (associate, aggregate, track): chain, entity bridge, time slice, knowledge update	*chain*: causal line of three or four events with projections *entity bridge*: projections of one event on ≥ 2 platforms, or records of one contact within one month (≤ 10, evenly subsampled) *time slice*: all mail and calendar records of one day (≥ 2; no drive or photo), one of them as anchor and all the others as evidence *knowledge update*: causal line with a state change, span ≥ 30 days and tail ≤ 30 days before τ, with a supersession check	core / detail / wrong *knowledge update*: current / outdated / discriminator
Abstention (boundary, honesty): fictitious person, fictitious activity, memory conflict	*fictitious person / activity*: profile-fitting entities with zero hits in a superset corpus *memory conflict*: real record with one perturbed detail	*fictitious*: "no record" rubric *memory conflict*: true memory + wrong detail
Preference (internalise, apply): communication, naming, schedule, content, venue, other	habit mined from the pool with ≥ 3 supporting records	rubric of the habit

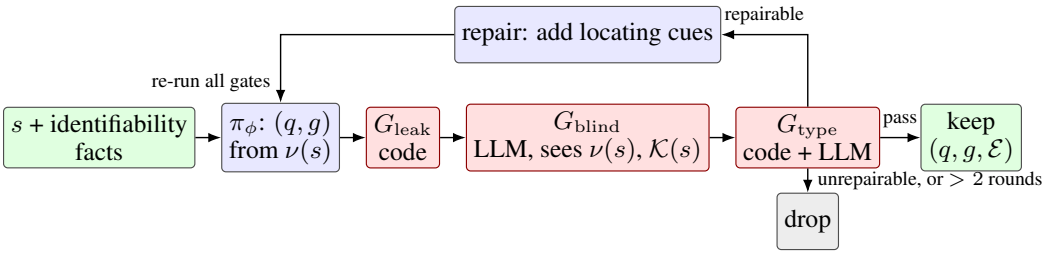

Figure 4: Gates applied to every candidate question, in the notation of (3). Repairable failures go to a repair call that adds locating cues, and the repaired question re-enters all gates.

The three identifiability facts of §3.3 are stored with every structure as an *identifiability* block computed in code: `candidate_total` (the number of pool items satisfying the structure's predicate before subsampling), `anchor_span` (for each record, the number of distinct dates on which its title recurs) and the set of records that can serve as unique anchors (span ≤ 1 day). In one development life of about 3,200 records, 12 of 26 contact bridges had been subsampled from cardinalities of up to 120 to the cap of ten records; without the block, a question phrased as "which three times" would have had a hundred valid answers. A structure that lacks the block makes the compiler stop rather than degrade, because an empty block would silently disable the cardinality and anchor gates while leaving the logs clean.

Examples. The following examples are translated from the Chengdu life (a web novelist and late-night game streamer); the originals are in Chinese. Ground truths are abbreviated.

- **Single-hop, mail.** *In the late-May reply from Zhou Qiming about the comic-convention rehearsal changes, what did he say I should answer if the host presses me on my writing progress?* Core: "still serialising, no spoilers for now". Detail: date 2026-05-23; the chapter-90 question should not be expanded on stage. Wrong: answering about chapter 90, or paraphrasing "don't spoil" without the phrase.
- **Single-hop, calendar.** *The late-night authors' chat in the group voice call with Shen Lu and Han Che on 19 July—what time did it start?* Core: 03:30. (The evidence resolves to an occurrence of the recurring series "authors nudging each other to update".)
- **Single-hop, drive.** *In the file I prepared on 20 April for Duan Lin to check the first-half-of-April income and the fan-meet material receipts, which fan-meet item still had only a receipt and no formal invoice?* Core: the spare blank cards. Wrong: the other pending item (table signs and number stickers).
- **Multi-hop, chain.** *Around the May Day holiday, while chapter 86 moved to the chapter-87 preview, what did I first fix based on reader feedback, and how did I then turn that feedback into what I could and could not say on stream?* Ten core points spanning four drive documents (fixed the unclear evidence hand-over in chapter 86; fixed timeline and wording slips in the preview; . . . ; must not promise a release time). Each record contributes points the others lack.
- **Multi-hop, entity bridge.** *In late March, Cheng Xiaoxi and I went to the alley-corner café to write—which two times exactly?* Core: a Friday afternoon in late March and the following Friday. Detail: 20 and 27 March, 16:00. The repair loop added "late March" after the blind validator found that the same café session recurs across months.
- **Multi-hop, time slice.** *One afternoon in my living room, Mili and I sorted the fan-meet raffle list and passcodes. What else was on my schedule that day?* Five core points (a pre-dawn writing sprint, an evening equipment check with Jingchuan, a video call with family, the night's fan-meet and moderator rota, the late-night stream).
- **Multi-hop, knowledge update.** *In mid-June I had a progress sync with my editor Li Man about submitting chapter 92 and the opening of chapter 93. As of 22 July, what is the latest state?* Current: chapter 95 v4 is ready to publish; chapter 96 must give a clearer cold clue within the first 800 characters and a 2,000-character preview is due by 24 July. Outdated: on 17 June chapter 92 had just been cleared for submission. Discriminator: the chapter numbers.
- **Abstention, fictitious person.** *Did I once show the villain arc of my novel to Lu Chenzhou? Which two revision suggestions did he give, and in what order?* Ground truth: no record of "Lu Chenzhou" anywhere in the life (zero hits over mail, calendar, drive full text and photo descriptions); three similar real records exist and may be mentioned as clarification.
- **Abstention, memory conflict.** *When I revised the chapter-91 ending rhythm on the evening of 3 June, did I save the document as v3? . . .* True memory: the saved file is "chapter 091 ending rhythm v2". The assistant must point out the mismatch; confirming v3 is a failure.
- **Preference, naming.** *I am tidying a batch of drafts, asset sheets and memos and want to fix a naming rule first. What would you suggest, with a few examples?* Rubric: topic_content_date (or time-of-day) structure, long descriptive names, an explicit date field—the pattern of the user's own files; generic advice ("keep names short, add a version number") fails.

Models, thresholds and prompts. Stage two used a mid-size model (GPT-5.4) for phrasing (temperature 0.35) and for the blind validator (temperature 0). Quotas and thresholds not given in Table 3 or §3.2: the multi-hop sub-type quota is 40/30/30% (chain/bridge/slice) and realises 45/34/21%, time slices falling short because a day with at least two mail or calendar records is not always available; knowledge updates are selected separately by their line selector; a bridge draws $N \in [2, 4]$ records from its group of at most ten; core ground-truth points are at most 60 characters; a knowledge update requires a current state of at most 120 characters and cross-container recall with a topic-token overlap of at least 4; the abstention sub-type quota is 20/20/60% (person/activity/conflict) and realises 27/13/60%, fictitious activities being harder to accept than fictitious people, a fictitious activity being accepted only if its hypernym covers at least 0.6% of the corpus and an entity demoted to an anchor if its sub-phrases cover more than 0.1%; drive full text enters the context up to 8,000 characters, and extraction must succeed for at least 95% of extractable files or the run fails. Thresholds that depend on corpus size are ratios so that they transfer across lives. Stage two uses one phrasing

and one blind-validation prompt per type, a repair prompt, a line-selection prompt for knowledge updates, a preference-mining prompt and a supersession judge.

C DATASET DETAILS

Table 4: LIFEFORGE-BENCH at a glance. All record counts are the rows as they reach the systems under test: the account contents up to each life's reference date, with recurring calendar series expanded into one record per occurrence (Table 7). Per-life details are in Table 5.

Lives		**Per life (median / max)**	
digital lives	11	events	530 / 952
countries / cities / time zones	6 / 11 / 8	calendar records	751 / 1,774
languages: records / biographies	4 / 9	emails	724 / 2,682
horizon, days (median / max)	90 / 133	drive files	517 / 1,317
contacts (total)	269	photos	244 / 701
Totals		**Questions (11 lives)**	
events / recurring series	6,615 / 160	single-hop (mail/cal/drive/photo)	310 (119/77/77/37)
records fed to the systems	28,952	multi-hop (chain/bridge/slice/update)	339 (132/99/61/47)
mail threads / body chars	3,732 / 2.3M	abstention (person/act./conflict)	220 (59/29/132)
openable office files	3,584	preference (6 dimensions)	165
photos / reference portraits	3,565 / 218	with an evidence set	781

Lives. We sampled the lives from short free-text briefs (one sentence to one paragraph), which name an occupation, a city and at most a few storylines; the contacts, schedules, files and photos counted in Table 4 were filled in by the model. Occupations range from undergraduate to group CEO, and the cities lie in Asia, North America and South America. Each life writes in the language its persona would use at work, with technical vocabulary left in English where a speaker would leave it: the records are Chinese in three lives, English in six, Japanese in one and Portuguese in one, while the biographies list nine languages across the personas. Emails are threaded, carry labels and attachment lists and include spam; calendars mix single events with `RRULE` masters and exception rows; drive holds 3.6K openable files: 1.6K documents, 1.6K spreadsheets, 179 slide decks, 175 PDFs and 33 images. Mail and calendar records are emitted in schemas that real accounts can import, so a memory system sees the same fields an account sync would deliver.

Questions. One set was compiled per life at that life's reference date (Table 7), 1,034 questions in total, 86 to 104 per life, each written in the language of the life it comes from. They are first-person and colloquial (median length 83 characters), multi-hop questions cite 2–18 evidence records (median 3), and each carries its layered ground truth, evidence set and reference date (examples in Appendix B and Figure 5). The 192 questions of L01 and L11 were additionally audited item by item (Appendix D); the other nine sets are raw compiler output.

Cost. The longest life (L10, 133 days) consumed $155 of text-LLM calls (about 8 cents per record) plus image generation; over the released lives, text-LLM spend was $82–282 per life including repairs. Compiling 100 questions takes 280–410 LLM calls and $4.8–6.0, two orders of magnitude below expert-written agentic benchmarks (He et al., 2026).

D DATASET QUALITY

The 192 questions of L01 and L11 were audited one by one against the raw data by native Chinese speakers on our team (§4.1); the other 842 are raw compiler output, and this appendix is what we know about them. Independently of that audit, an outsourced annotation team reviewed generated records during pipeline development and a separate group of annotators reviewed raw compiler output; Table 6 summarises both.



Figure 5: Left: reference portrait of one protagonist (a New York content creator) and three scene photos generated for different events (client shoot, spring outing, brand video); the face is conditioned on the portrait, the scene on the event. Right: the photo behind a single-hop photo question of the Hangzhou life (translated): *"In late July at the photography club we went over the warm-up shots for the August outing; the projector showed several backlit lake shots side by side as cover candidates. How many were there?"* Core answer: *six*; the exact timestamp, the place, the file name and the names are forbidden anchors, while an approximate time is allowed.

Table 5: The eleven released lives. Occupations are abbreviated from the generated persona; counts are the records fed to the systems under test, on the same basis as Table 7.

Life	City	Languages	Occupation (from brief)	Days	Events	Cal.	Mail	Drive	Photos
L01	Hangzhou, CN	Chinese	CS undergraduate, student-project lead, photography club	88	566	601	400	546	142
L02	San Francisco, US	English, Chinese	VP of business	90	722	1,208	1,916	665	244
L03	Austin, US	English	Senior undergraduate, job seeking	90	479	683	498	291	219
L04	Seattle, US	English, Korean	Software engineer	90	463	751	663	151	149
L05	New York, US	English	Freelance content creator, brand strategist	90	425	633	811	343	701
L06	Singapore, SG	English, Chinese, Malay, German	Group CEO	90	947	1,774	2,682	1,317	271
L07	Tokyo, JP	Japanese, English	Front-end engineer	93	433	627	805	246	357
L08	Rio de Janeiro, BR	Portuguese, English, Spanish	Freelance content strategist and lifestyle photographer	93	606	903	864	484	404
L09	Jakarta, ID	Indonesian, English	Freelance food content creator	93	492	787	724	545	687
L10	Shanghai, CN	Chinese, English	AI product lead, senior algorithm engineer	133	952	810	702	607	113
L11	Chengdu, CN	Chinese	Full-time web novelist and late-night game streamer	120	530	455	556	517	100

Table 6: Dataset quality summary. Record pass rates come from the final outsourced review of generated records during development (calendar and mail over 12 development lives, 3,563 records; drive and photos over 3 lives after the final fixes); question metrics come from a review of 80 question–answer pairs of raw compiler output by twelve annotators; gate metrics come from the generation logs of three question compilations.

Metric	Value	Notes
Human review pass rate of records: calendar / mail / drive / photos	99% / 99% / 85% / 97%	a record fails on any of the five criteria listed in the text
Questions directly usable / usable with flaws / unusable	87.5% / 11.3% / 1.3%	flaws are mostly questions that read as machine-written because of too many anchors
Judge–human agreement	93.8%	after auditing disagreements, two genuine judge errors remain, both lenient
Candidates dropped by gates / kept questions that needed repair	36% / 13%	173 of 480 candidates dropped; multi-hop candidates fail mostly in the blind validator and on uniqueness

Human review of the data. An outsourced annotation team of native Chinese speakers reviewed sampled generated records during pipeline development; for lives in other languages they saw a machine translation next to every original field and judged the two side by side. A record fails on internal contradiction or factual error, violation of common sense or of the persona, implausibility, missing information or format anomaly, or inconsistency with another platform. Drive remains the component with most room to improve; its failures are mostly wrong contact names inside a document and contradictions with the event or an earlier version of the file. The 11 released lives were generated with the pipeline version after this review and were not sent for outsourced review.

Question usability and judge agreement. Twelve annotators reviewed 80 question–answer pairs covering 75 of the 100 questions in the first compiled set, with the layered ground truth and evidence records in view. Flaws were of two kinds: questions that read as machine-written because many anchors were packed into one sentence—the visible price of identifiability in a dense, repetitive life—and one ground truth that inherited an ambiguity from its question. Annotators agreed with the LLM judge on 93.8% of pairs (91.3% when their own correctness labels are cross-tabulated against the judge); auditing the disagreements against the full data left two genuine judge errors (both lenient: core points hit, an accompanying detail wrong), one annotator error from an incomplete annotation view, and one ambiguous ground truth—which motivated the three-level verdict with a factual-error veto used in the evaluation.

E EVALUATION SETUP

Table 7: Text volume of the eleven lives as fed to the systems under test. Records are rendered as text with their visible view, including only records with event time $\leq \tau$ and expanding recurring calendar series into one record per occurrence; tokens are counted with the GPT-5-series tokenizer (o200k).

Life	Reference date τ	Mail	Calendar	Drive	Photos	Records	Chars	Tokens
L01 (Hangzhou, undergraduate)	2026-08-11	400	601	546	142	1,689	2.37M	1.52M
L02 (San Francisco, VP of business)	2026-07-26	1,916	1,208	665	244	4,033	5.32M	1.45M
L03 (Austin, job-seeking senior)	2026-07-26	498	683	291	219	1,691	2.00M	0.55M
L04 (Seattle, software engineer)	2026-07-26	663	751	151	149	1,714	1.44M	0.40M
L05 (New York, content creator)	2026-07-26	811	633	343	701	2,488	2.45M	0.67M
L06 (Singapore, group CEO)	2026-07-26	2,682	1,774	1,317	271	6,044	7.27M	2.00M
L07 (Tokyo, front-end engineer)	2026-07-26	805	627	246	357	2,035	2.02M	0.64M
L08 (Rio de Janeiro, content strategist)	2026-07-26	864	903	484	404	2,655	2.67M	0.76M
L09 (Jakarta, food content creator)	2026-07-26	724	787	545	687	2,743	3.70M	1.02M
L10 (Shanghai, AI product lead)	2026-07-18	702	810	607	113	2,232	3.54M	1.54M
L11 (Chengdu, web novelist and streamer)	2026-07-22	556	455	517	100	1,628	2.03M	1.54M
Total		10,621	9,232	5,712	3,387	28,952	34.8M	12.1M

Setup. Each system received exactly the records of the eleven lives with event time $\leq \tau$ (Table 7), one record per message, grouped into sessions by source and day (5,179 sessions, split into 6,169 chunks of at most eight records), each life in its own memory namespace, rendered with the visible fields: recurring calendar series expanded into per-occurrence instances as an account sync does, drive bodies capped at the compiler's limit, and photos replaced by captions produced by a vision model from the uploaded 720-pixel images (the same captions for every system). Versions and retrieval settings: verbatim RAG is Mem0's store with extraction disabled; Mem0 2.0.20 makes one extractor call per chunk; Letta 0.16.8 writes one archival passage per record and retrieves server-side by vector (in the MemGPT-style memory hierarchy, core memory consists of a few short text blocks that always stay in context and are edited by the agent itself, while archival memory is paged in by the agent through a search tool when needed; we connect only the latter's storage and a single retrieval, without the agent's multi-round autonomous search or core memory, so the Letta row represents the class of "verbatim per-record storage plus vector retrieval"); EverOS 1.3.0rc1 receives records in batches of 25, runs its episode/fact/profile cascade inside the service with hybrid keyword and vector retrieval, and is forced to a boundary and a drain of the cascade queue after every batch; Graphiti 0.30.1 runs on Neo4j with one episode per record and reciprocal-rank fusion

over edges, nodes and episodes. Ingestion costs are in Table 1; Graphiti's are highest because every record becomes an episode whose entities and edges are extracted, deduplicated against the existing graph and summarised by the LLM, and EverOS calls its own LLM inside the service, so its calls and tokens cannot be metered. Ingestion times are wall-clock times on one machine, with several systems ingesting and answering in parallel during measurement, so they indicate orders of magnitude only.

Answering and grading. Retrieval is performed online by the harness, which saves the context of every question (the retrieved memories and their order); the main results and the reader ablation generate answers on the saved contexts: the three GPT-5.6 models with no temperature setting and an output cap of 6,000 tokens, GPT-4.1-mini at temperature 0. The prompt is identical to the online run and asks for a complete answer with dates and details, for the latest state when records conflict, and for an explicit "no record" when the memories do not support one.

Oracle evidence. The Oracle column of Table 1 replaces retrieval by the evidence set of the question: the gold records (for a recurring series, the occurrence named in the evidence) are rendered exactly as retrieved memories are, in chronological order, and passed to the same reader with the same prompt, clock and judge. A time-slice question additionally receives its anchor record, the one mail or calendar record of that day that the question describes and that the evidence set excludes by construction, since the reader cannot identify the day without it. The 253 questions without an evidence set (fictitious entities and preferences) are not run.

Context cap. The $k = 30$ retrieved memories are concatenated in the system's own order under a cap of 150K characters for the verbatim store and 60K for the other systems, beyond which memories would be dropped from the tail of the ranking. Median contexts are 9–22K characters. Across the 5,170 question–system pairs the cap truncates the tail of the ranking 148 times: 122 for EverOS, 23 for Graphiti and 3 for the verbatim stores.

F JUDGE PROTOCOL

Free-text ground truths let judges invent criteria and accumulate per-fact penalties (we measured pass rates of only 8–9% for long ground truths against 74% for short ones on the same system), so ground truths are layered: *core* points that must all be hit, *detail* points that cost nothing if omitted, and a typical *wrong* answer; knowledge-update, abstention and preference questions instead declare the current and outdated states, what counts as fabrication, and the habit that serves as the rubric, respectively. The judge is an LLM (Zheng et al., 2023) that receives the question, the layered ground truth and the answer, and outputs PASS, PARTIAL or FAIL with a rationale. PASS requires every core point to be hit and no factual error; PARTIAL requires at least one core point and no factual error, fabrication or acceptance of a false premise; any statement that contradicts the record is a FAIL regardless of the other points (factual-error veto). Type-specific clauses map "old and new state listed side by side" (knowledge update), "evasive non-answer" (abstention), "does not correct the user but does not elaborate the error" (memory conflict) and "generic template advice" (preference) to PARTIAL. The judge additionally reports, without influence on the verdict, a quality status for the item (*ok*, *stale_gt*, *ambiguous_anchor*, *answer_leaked*); items flagged *stale_gt*—the answer cites a later state with a time and source—are graded FAIL by the judge but sent to human audit, and the ground truth of the item is corrected if the later state is real; this mechanism was used during the release audit to check the currency of knowledge-update questions.

G METRICS BY QUESTION TYPE

Definitions. A memory in the verbatim store is one record; in Mem0 one fact extracted from a chunk of records; in EverOS an episode, fact or profile; in Graphiti a node, edge or episode. Retrieval metrics are therefore defined on the source records of each memory (§4.1); occurrences of a recurring series are attributed to the series and drive chunks to the file. The 781 questions with an evidence set are 310 single-hop, 339 multi-hop and 132 memory-conflict questions. Recall@k is the share of a question's evidence records covered by the top-k memories, Precision@k the share of the top-k memories that hit evidence, full recall the indicator that every evidence record is covered,

and MRR the reciprocal rank of the first memory that hits evidence, all averaged over questions. Reader input tokens count the whole prompt sent to the reader: the retrieved memories, the system prompt and the question. The attribution bias noted in §4.1 arises because Mem0 attributes the facts extracted from a chunk to all records of the chunk and EverOS does not return record identifiers, so a memory can be attributed only to its session (records of one source on one day); the memories of verbatim RAG and Letta trace back exactly to a single record, and Graphiti's episodes to one record and its edges to one or two, while its entity summaries carry no record identifier in the returned payload, so they can never count as a hit. Graphiti's record-level retrieval metrics are therefore a lower bound. Attributing each entity summary offline to every record that mentions the entity raises its Recall@30 by about twenty points on the two audited lives. We report the unattributed number for every system.

Table 8: Retrieval and cost metrics by question type (means, $k = 30$, reader GPT-5.6-sol, eleven lives). Retrieval metrics are computed over the memories that actually reach the reader; fictitious-entity abstention and preference questions have no evidence set, so retrieval metrics do not apply.

Metric	Type	n	RAG	Mem0	Letta	EverOS	Graphiti
Recall@30 (%)	single-hop	310	89.7	90.3	75.2	74.5	82.3
	multi-hop	339	69.7	64.9	47.7	55.7	50.5
	memory conflict	132	87.1	84.1	70.5	78.8	84.8
Precision@30 (%)	single-hop	310	12.2	11.2	2.5	11.1	8.2
	multi-hop	339	17.9	13.9	5.0	14.5	9.3
	memory conflict	132	18.6	15.9	2.3	16.2	9.0
Full recall (%)	single-hop	310	89.7	90.3	75.2	74.5	82.3
	multi-hop	339	42.8	33.6	20.1	28.0	26.8
	memory conflict	132	87.1	84.1	70.5	78.8	84.8
MRR	single-hop	310	0.61	0.56	0.50	0.50	0.38
	multi-hop	339	0.62	0.50	0.45	0.52	0.35
	memory conflict	132	0.67	0.52	0.49	0.62	0.45
Reader input tokens (K), all types		1,034	9.3	2.9	8.6	9.4	12.7

Table 9 breaks the multi-hop questions down by the number of evidence records a question needs and by the number of platforms its evidence spans. The two platform groups differ in sub-type mix, knowledge-update questions falling mostly on one platform and chains and bridges mostly on two or more, which is one reason the platform split does not separate the systems the way the record count does.

Table 9: Weighted score of multi-hop questions grouped by the number of evidence records a question needs, and, for questions with 2–5 records, by the number of platforms the evidence spans (number of questions in parentheses; reader GPT-5.6-sol, eleven lives). Oracle is the same reader given the gold evidence records.

Records needed	RAG	Mem0	Letta	EverOS	Graphiti	Oracle
2 (108 questions)	57.4	49.5	64.8	47.7	58.8	94.4
3 (102 questions)	66.7	61.3	73.5	57.4	65.2	95.6
4–5 (80 questions)	63.7	41.9	63.1	61.9	48.1	93.1
6 or more (49 questions)	8.2	9.2	11.2	41.8	12.2	72.4
2–5 records, one platform (95)	59.5	52.6	64.7	57.9	50.5	94.7
2–5 records, two or more platforms (195)	63.8	51.0	68.7	53.6	61.8	94.4

H Ablations

All ablations in this appendix run on the 192 audited questions of L01 and L11, rather than on all eleven lives, because each of them re-answers or re-ingests the whole set. Table 10 gives the

baseline they vary from. Their purpose is to show which design choices move a score, not to rank the systems.

Table 10: The baseline these ablations vary from (weighted score, %, $k = 30$, reader GPT-5.6-sol).

	RAG	Mem0	Letta	EverOS	Graphiti	Oracle
Two audited lives (192 questions)	75.3	58.9	**78.6**	74.7	74.2	94.9

Number of retrieved memories. The k ablation re-retrieves at $k = 5, 10, 20$ on the kept memory stores of the main run and saves the contexts, then answers and grades offline with the same reader and judge as the main results (Table 11, and by type Table 12). Multi-hop gains most over the range of k, 16.2–33.8 points from $k = 5$ to $k = 30$ for all five systems, although RAG peaks at $k = 20$. Single-hop saturates after $k = 10$ for Letta, gains little for RAG, and rises all the way to $k = 30$ for Mem0 (39.8 to 64.8), whose short fact entries need several memories to cover one record. More memories do not distract the strong reader (RAG's abstention score stays at 100 for $k \geq 20$). The small reader behaves differently. Answering the same contexts with GPT-4.1-mini gives RAG 56.2 / 62.5 / 65.4 / 62.2, Mem0 44.3 / 48.2 / 54.7 / 58.3, Letta 57.3 / 65.1 / 66.7 / 69.0, EverOS 50.3 / 56.5 / 62.8 / 62.8 and Graphiti 52.3 / 59.9 / 60.4 / 63.8 for $k = 5/10/20/30$ (the $k = 30$ column is Table 13). RAG then loses three points between $k = 20$ and $k = 30$, its abstention score falling from 82.5 to 67.5, while Letta gains two.

Table 11: Ablation over the number of retrieved memories k (each cell: weighted score / Recall@k / full recall, %; reader GPT-5.6-sol). Same memory store, only the number of memories passed to the reader varies; the $k = 30$ column is the two-audited-lives row of Table 10. Graphiti's recall columns here resolve entity summaries offline to the records that mention the entity, as Table 15 does, so they are higher than the unattributed Graphiti column of Table 8; the k with the best weighted score in each row is in bold.

System	$k = 5$	$k = 10$	$k = 20$	$k = 30$
RAG	63.8 / 63.0 / 48.6	68.2 / 68.7 / 52.1	**75.8** / 76.4 / 61.6	75.3 / 79.8 / 67.8
Mem0	43.5 / 47.8 / 36.3	48.4 / 58.5 / 46.6	55.5 / 71.1 / 57.5	**58.9** / 77.1 / 61.6
Letta	63.8 / 46.3 / 39.0	70.3 / 51.5 / 41.8	73.7 / 58.2 / 47.3	**78.6** / 61.6 / 50.7
EverOS	59.4 / 55.2 / 43.2	64.3 / 64.9 / 50.7	**74.7** / 71.2 / 56.8	**74.7** / 74.7 / 61.0
Graphiti	57.8 / 45.8 / 37.0	62.0 / 59.0 / 47.3	69.8 / 70.1 / 59.6	**74.2** / 74.4 / 63.0

Reader. The reader ablation reads the $k = 30$ contexts of the main run and swaps only the reader, keeping the judge unchanged (Table 13). The drop from the GPT-5.6 readers to GPT-4.1-mini concentrates on abstention and preference questions (RAG's abstention score falls from 100 to 67.5, and all systems fall from 85–88 to 65–78 on preference, where the small model gives generic advice); on single-hop questions verbatim RAG and EverOS drop 8–11 points while Letta rises slightly, and on multi-hop questions every system except Mem0 drops 3.7–8.1 points.

Repeated runs. We re-ran RAG, Mem0 and EverOS completely (ingestion, retrieval, answering and grading each performed afresh) under identical settings: $k = 30$, the same batching, the GPT-4.1-mini reader of the online runs and the same judge; the run kept for these two lives is the one that enters Table 1 and every ablation here. The verbatim store, whose ingestion involves no LLM call, differs by 2.6 points between runs (66.4 against 63.8), EverOS, whose extraction runs inside its service, by 1.6 (62.5 against 64.1), and Mem0, whose ingestion depends on LLM extraction in the harness, by 7.3 (50.8 against 58.1), since the randomness of extraction changes the contents of the memory store: the two stores of about 3,100 memories each share only 0.2–0.4% of their memories verbatim and about 13% near-verbatim, the records retrieved for the same question overlap by 40%, and 53 of the 192 questions change verdict. Letta was run twice, but the second run was not graded; Graphiti was run once.

Table 12: Ablation over the number of retrieved memories k by question type (weighted score, %, reader GPT-5.6-sol, two audited lives, 192 questions); the last two columns are the mean reader input tokens (K) and answer seconds; the $k = 30$ rows are the GPT-5.6-sol rows of Table 13.

System	k	Single-hop	Multi-hop	Abstention	Preference	Tokens (K)	Seconds
RAG	5	82.4	27.9	92.5	73.3	2.7	6.1
RAG	10	87.0	32.4	97.5	76.7	5.1	6.8
RAG	20	87.0	46.3	100.0	90.0	9.8	7.4
RAG	30	90.7	44.1	100.0	85.0	14.5	10.9
Mem0	5	39.8	15.4	72.5	75.0	0.9	6.0
Mem0	10	50.0	26.5	67.5	70.0	1.3	6.5
Mem0	20	62.0	28.7	77.5	75.0	2.1	7.3
Mem0	30	64.8	33.1	72.5	88.3	3.0	9.8
Letta	5	82.4	22.8	97.5	78.3	2.6	6.0
Letta	10	88.9	36.0	100.0	75.0	4.8	6.6
Letta	20	87.0	47.8	95.0	80.0	9.3	7.5
Letta	30	87.0	56.6	97.5	88.3	14.2	8.7
EverOS	5	64.8	28.7	87.5	81.7	1.8	6.8
EverOS	10	71.3	35.3	92.5	80.0	3.0	7.6
EverOS	20	77.8	53.7	97.5	86.7	5.4	8.0
EverOS	30	77.8	55.9	92.5	88.3	9.2	9.6
Graphiti	5	59.3	33.8	85.0	73.3	4.0	7.8
Graphiti	10	66.7	39.0	85.0	75.0	7.8	9.1
Graphiti	20	65.7	52.9	92.5	85.0	15.7	10.1
Graphiti	30	75.9	56.6	92.5	86.7	23.6	10.5

Table 13: Reader ablation (weighted score, %; the Overall column also gives the PASS rate in parentheses). Contexts retrieved at $k = 30$ are fixed and only the reader changes; the judge is unchanged; the GPT-5.6-sol rows are the two-audited-lives row of Table 10. The GPT-4.1-mini rows re-answer offline on the saved contexts and differ from the online run that produced those contexts by 0.2–3.4 points.

System	Reader	Overall (PASS)	Single-hop	Multi-hop	Abstention	Preference
RAG	GPT-5.6-sol	75.3 (69.3)	90.7	44.1	100.0	85.0
RAG	GPT-5.6-terra	76.0 (71.9)	88.9	46.3	100.0	88.3
RAG	GPT-5.6-luna	77.1 (70.8)	90.7	50.0	100.0	83.3
RAG	GPT-4.1-mini	62.2 (53.1)	82.4	40.4	67.5	68.3
Mem0	GPT-5.6-sol	58.9 (51.6)	64.8	33.1	72.5	88.3
Mem0	GPT-5.6-terra	59.4 (50.0)	64.8	36.8	72.5	83.3
Mem0	GPT-5.6-luna	60.7 (51.0)	61.1	36.8	87.5	78.3
Mem0	GPT-4.1-mini	58.3 (46.4)	65.7	37.5	75.0	70.0
Letta	GPT-5.6-sol	78.6 (72.4)	87.0	56.6	97.5	88.3
Letta	GPT-5.6-terra	76.8 (70.3)	85.2	53.7	97.5	86.7
Letta	GPT-5.6-luna	77.3 (70.8)	88.9	52.9	97.5	85.0
Letta	GPT-4.1-mini	69.0 (58.3)	90.7	48.5	77.5	65.0
EverOS	GPT-5.6-sol	74.7 (66.1)	77.8	55.9	92.5	88.3
EverOS	GPT-5.6-terra	73.2 (65.6)	73.1	55.1	92.5	88.3
EverOS	GPT-5.6-luna	74.2 (66.7)	77.8	55.9	90.0	88.3
EverOS	GPT-4.1-mini	62.8 (49.5)	66.7	51.5	65.0	78.3
Graphiti	GPT-5.6-sol	74.2 (69.3)	75.9	56.6	92.5	86.7
Graphiti	GPT-5.6-terra	73.4 (68.8)	74.1	55.9	95.0	83.3
Graphiti	GPT-5.6-luna	76.0 (71.9)	79.6	55.9	97.5	86.7
Graphiti	GPT-4.1-mini	63.8 (54.2)	75.9	49.3	70.0	66.7

Extraction model of Mem0. Mem0 is the system that loses most at ingestion, so we re-ingested both audited lives twice more, changing only the model that performs its fact extraction and leaving the reader, the judge, the retrieval budget and the embedding model fixed (Table 14). A larger non-reasoning model (GPT-4.1) does not help overall. A reasoning model (GPT-5-mini) gains 2.3 points. Both shift the profile rather than lift it. Memory-conflict questions improve most, from 54.2 to 62.5 and 79.2, because the extracted facts stay closer to the record they came from and can be checked against a false premise. Multi-hop questions stay in the 27.9–33.1 band that Mem0 occupies with every extractor. Extracting more carefully does not recover what extraction discards, and the reasoning model costs 4.8 times the ingestion time.

Table 14: Mem0 with three extraction models on the two audited lives (weighted score, %, $k = 30$, reader GPT-5.6-sol, judge GPT-5-mini; best in each column in bold). Only the extraction model changes; GPT-4.1-mini is the configuration used everywhere else in the paper. Ingestion hours are the summed per-batch durations for the two lives.

Extraction model	Overall	Single-hop	Multi-hop	Memory conflict	Preference	Ingestion (h)
GPT-4.1-mini (Table 10)	58.9	64.8	**33.1**	54.2	88.3	2.6
GPT-4.1	56.8	**68.5**	27.9	62.5	73.3	**1.9**
GPT-5-mini	**61.2**	63.9	29.4	**79.2**	**93.3**	12.4

Episode granularity of Graphiti. Graphiti turns each ingested unit into an *episode* from which entities and edges are extracted. Our first run grouped the eight records of a batch into one episode; the run used in the paper makes one episode per record, matching the other systems. Table 15 compares the two configurations on the same records, reader and judge. Per-record episodes cost about three times the LLM calls and twice the wall-clock time, because every record is extracted, deduplicated against the graph and summarised separately, but they score 9.6 points higher: a batch episode carries eight records of text, so 30 of them exceed the 60K-character cap and only about half reach the reader in nearly every question, single-hop questions suffering most.

Table 15: Graphiti with one episode per batch of eight records (the first run) against one episode per record (the configuration used in the paper); $k = 30$, reader GPT-5.6-sol, judge GPT-5-mini. Retrieval metrics are over the 146 questions of these two lives that carry an evidence set, and over the memories that reach the reader. Both columns resolve entity summaries offline to the records that mention the entity, which is why they are higher than the unattributed Graphiti column of Table 8; the batch run is a lower bound, because its store was not kept and only the summaries still on disk could be resolved.

	Episode per batch of 8	Episode per record
Weighted score / PASS rate (%)	64.6 / 58.3	74.2 / 69.3
Single-hop / multi-hop / abstention / preference	59.3 / 47.1 / 85.0 / 86.7	75.9 / 56.6 / 92.5 / 86.7
Recall@30 / full recall / Precision@30 (%)	44.9 / 31.5 / 6.0	74.4 / 63.0 / 9.9
Memories reaching the reader (median) / questions truncated	15 / 189	30 / 0
Reader input tokens per question	44.7K	23.6K
Ingestion: LLM calls / tokens / hours (3,317 records)	7,783 / 123M / 5.1	22,465 / 135M / 11.1