Personalize-then-Store: Benchmarking and Learning Personalized Memory for Long-horizon Agents

Yeonjun In, Wonjoong Kim, Sangwu Park, Kanghoon Yoon, Chanyoung Park*
KAIST
{yeonjun.in, wjkim, sangwu.park, ykhoon08, cy.park}@kaist.ac.kr

Abstract

Existing large language model (LLM)-based memory systems apply universal, static policies that overlook a fundamental reality: the contexts that are worth storing in memory are different across users. This misalignment wastes limited memory budget on transient interactions while failing to preserve critical context for long-horizon tasks. To address this gap, we investigate an underexplored question: can LLM-based memory systems learn personalized memory policies? We introduce PerMem-Bench, the first benchmark that jointly models multi-domain interaction histories and cross-user heterogeneity in agent-use patterns to evaluate personalized memory systems. We further present the first empirical study of memory personalization, proposing *session-level storage gating* — a lightweight framework that selectively bypasses memory operations for transient sessions. Our study confirms that personalization yields substantial retention gains under perfect gating, yet reveals that accurate gating remains an open and critical challenge. Our benchmark and source code are available at https://github.com/yeonjun-in/PerMemBench.

1 Introduction

The proliferation of LLM agents has attracted diverse users tasking agents with both transient and long-horizon interactions across various domains. Unlike transient tasks, successful long-horizon interactions require agents to preserve and manage crucial context from past interactions. Since LLMs inherently lack the capacity to memorize prior context, memory systems have emerged as a cornerstone for sustaining effective and coherent long-horizon agent-user dialogues. [1, 2, 3, 4, 5, 6, 7].

Early memory systems relied on storing exhaustive raw dialogue histories within a memory bank or context window. However, this naive approach is impractical for real-world deployment, as it necessitates an infinite memory budget and introduces substantial irrelevant noise. Subsequent research has shifted focus toward deliberately extracting critical information to operate within a fixed budget [8]. Specifically, LLM agents are trained to identify "worth-storing" contexts—i.e., information whose preservation is expected to benefit future interactions, such as user preferences or specific events—and to update or delete existing memories via in-context learning or post-training [1, 9, 2, 3, 4]. These trained policies apply a universal, one-size-fits-all memory system to all users, regardless of individual differences.

However, this paradigm overlooks a fundamental question: **Are the contexts that are worth storing in memory the same for all users?** As illustrated in Figure 1(a), users exhibit heterogeneous agent use patterns across various domains. For Alice, 'Recipe Advice' is a long-horizon project requiring consistent context preservation, whereas 'Travel Plan' involves only spontaneous, transient inquiries. Conversely, Bob regards 'Travel Plan' as a memory-intensive long-horizon task for honeymoon planning, while his 'Recipe Advice' usage is strictly transient. Consequently, the information within

*Corresponding Author

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

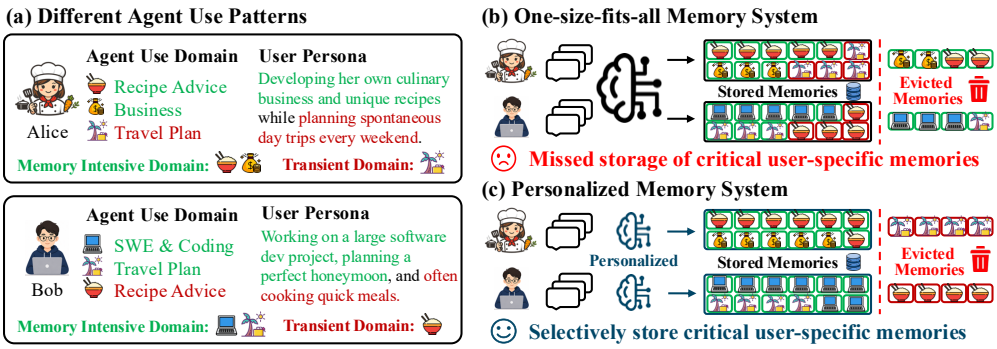

Figure 1: Motivating examples of personalized memory system. (a) Users exhibit distinct agent use patterns. (b) One-size-fits-all memory systems fail to personalize these user-specific needs, leading to the eviction of essential contexts. (c) An ideal personalized memory policy selectively preserves essential contexts tailored to each user's use pattern.

a 'Travel Plan' interaction constitutes a "worth-storing" context for Bob but not for Alice—and the inverse holds true for 'Recipe Advice'.

We observe that existing memory systems fail to account for these heterogeneous **user-specific patterns**, instead managing memory based on universal criteria. This leads to a critical **misallocation of resources**: the system wastes limited memory budget on transient interactions while failing to preserve essential context for vital long-horizon tasks (see Figure 1(b)). To address this, we argue that an ideal memory system should be **personalized**, where the system should infer the user-specific "worth-storing" contexts then selectively store them—bypassing unnecessary storage for transient interactions while prioritizing those requiring long-horizon context accumulation (see Figure 1(c)).

Regarding this observation, we raise an important yet underexplored research question: **Can LLM-based memory systems infer the user-specific "worth-storing" contexts and learn personalized policy?** However, there is no benchmark dataset featuring long-horizon dialogues that capture the heterogeneous and personalized usage patterns observed across diverse users and domains. This absence precludes a rigorous evaluation of a memory system's capacity for fine-grained personalization.

To this end, we introduce PerMem-Bench, a novel benchmark for evaluating personalized memory systems, along with a fully automated data generation pipeline. The pipeline proceeds in three stages: (1) profiling user-specific agent use patterns for diverse personas, (2) constructing a life skeleton per user — a structured blueprint defining their long-horizon interaction trajectory — and (3) synthesizing realistic dialogue sessions via an LLM-based user simulator. By assigning a unique agent use profile to each persona, we instantiate user-specific "worth-storing" contexts, enabling rigorous evaluation of whether a memory system can accurately infer and preserve information tailored to each individual. The resulting dataset comprises multi-year interaction sessions for 20 users spanning diverse domains. Since the pipeline is fully automated and requires no manual intervention, it can be readily scaled to larger and more diverse user cohorts beyond the current set.

Building on PerMem-Bench, we investigate our research question through a systematic empirical study. We propose *session-level storage gating*, a simple yet general personalization framework that identifies whether each session is long-horizon or transient and skips memory operations for the latter, and introduce multiple gating methods as baselines. Our experiments show that perfect gating yields substantial retention and downstream performance gains under a fixed budget, yet current baselines remain suboptimal in gating accuracy, achieving only incremental gains in practice. These results illuminate the difficulty of personalizing memory systems in the wild and provide concrete directions for future research.

Our contributions are as follows:

- We identify and formalize the critical need for personalized memory systems, moving beyond the current "one-size-fits-all" paradigm.
- We present PerMem-Bench, the first benchmark that jointly models multi-domain interaction histories and cross-user heterogeneity in agent-use patterns to evaluate personalized memory systems.

- We introduce the first empirical study on memory personalization, proposing *session-level storage gating* as a novel personalization paradigm and establishing simple baselines as a reference point for future work.

2 Related Work

Agent Memory Systems. AI agents increasingly rely on memory systems to support long-horizon tasks across diverse users. Recent research in this area can be broadly categorized into two directions. The first direction focuses on learning LLM-based memory policies, enabling them to selectively extract and store salient information from interactions [1, 2, 3, 9, 10]. A central challenge in this line of work is determining which information is worth storing for a user. The second direction focuses on structured memory representations, leveraging clustering, graph, and tree-based methods to model relationships among memory units and improve retrieval accuracy [4, 5, 8, 11, 1, 12].

Our work aligns with the first direction but distinguishes itself by moving beyond the uniform criteria of prior approaches. Rather than applying a universal standard for identifying information worth storing, we propose *session-level storage gating* as a novel personalization paradigm that learns to identify each user's worth-storing sessions from their interaction history, and selectively bypasses memory operations for transient ones.

Evaluation of Agent Memory Systems. Evaluation frameworks for agent memory are typically divided into experiential and factual memory: the former distills past interactions into skills and strategies for improved reasoning, while the latter focuses on preserving critical user-centric context over long-horizon interactions. This study focuses on the latter, specifically evaluating whether a memory system effectively stores "worth-storing" information tailored to a user. Existing benchmarks in this space adopt LLM-based user simulations to model realistic interactions and assess memory capabilities [13, 14, 15, 16, 17, 18].

However, we argue that these evaluations largely rely on unrealistic assumptions. First, most benchmarks impose a single-domain constraint. LoCoMo [13] and HalluMem [17] focus exclusively on casual interactions in which users share personal events with agents, whereas real-world users engage with agents across multiple heterogeneous domains and goal-oriented scenarios. Second, they overlook behavioral heterogeneity across users. Benchmarks such as PersonaMem [16], LongMemEval [15], and AmemGym [18] incorporate only **personal attributes**—such as demographics, traits, and preferences—as user profiles, while ignoring **behavioral attributes**, i.e., agent use patterns. As a result, these benchmarks implicitly assume all users exhibit homogeneous agent use pattern, failing to capture the meaningful differences that arise in real-world user–agent interactions.

To bridge these gaps, we introduce a new benchmark, PerMem-Bench, that captures these complex, real-world usage scenarios. Unlike prior work, PerMem-Bench features multi-domain interaction histories and explicitly models *behavioral heterogeneity*. This provides a rigorous environment for evaluating whether memory systems can be effectively personalized across diverse users with heterogeneous interaction patterns.

3 Benchmark Construction: PerMem-Bench^s

This section details the construction of PerMem-Bench^s, a fully automated pipeline comprising three primary stages: (1) user-specific agent use profiling (Section 3.1), (2) life skeleton and timeline construction (Section 3.2), and (3) dialogue generation (Section 3.3). PerMem-Bench^s encompasses diverse agent use scenarios for 20 unique users. This sample size was strategically determined to balance the computational overhead of generation with the subsequent costs of memory system evaluation. While the current scale is optimized for efficiency, the inherent reliability of our automated process facilitates seamless scaling to larger cohorts, as discussed in Section 5. Importantly, PerMem-Bench is constructed entirely from synthetic data, including synthetically generated personas and interaction histories. It contains no real-user interaction data or personally identifiable information.

3.1 User-Specific Agent Use Profiling

We define an agent use profile as the joint configuration of *domain participation* and *memory necessity* across domains. We posit these two dimensions offer a simple yet effective framework for capturing

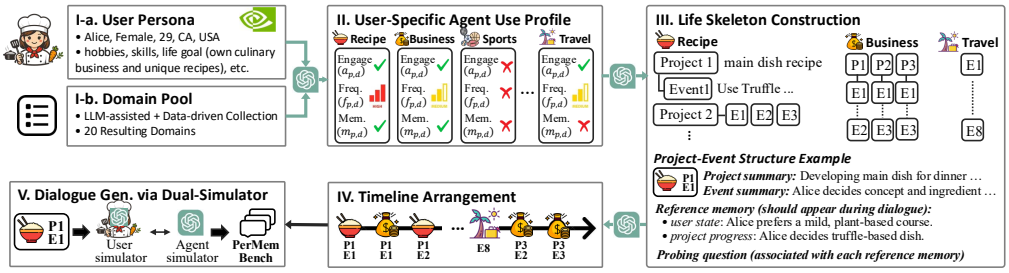

Figure 2: Overview of the construction pipeline for PerMem-Bench^s.

the diverse use patterns. For instance, Alice and Bob in Figure 1(a) demonstrate divergent profiles under this framework. While we recognize more granular patterns exist, we adopt this simple setup as a foundational step toward establishing a baseline for personalized memory management.

User Persona Collection (I-a of Figure 2). To ensure real-world plausibility, we leverage the *Nemotron-Persona-USA* dataset [19]. This collection provides high-fidelity personas with detailed attributes, including personal/professional backgrounds, personal preferences, allowing us to simulate a broad spectrum of user behaviors.

Domain Pool Construction (I-b of Figure 2). To ensure representative coverage of real-world usage, we employ a data-driven approach to construct a domain pool. First, we sample 1,000 personas and prompt `Claude Haiku 4.5` to generate potential usage scenarios without predefined constraints (see Appendix C.1). These candidates are then semantically clustered and assigned representative labels via human review. To align the pool with actual LLM trends, we cross-reference these clusters with industry reports [20, 21], pruning niche cases and supplementing broad-interest domains. This process results in a final taxonomy of 20 domains (see Table 3 in Appendix).

User-Specific Profile Assignment (II of Figure 2). From the collected persona set, we randomly sample 20 personas. For each persona $p \in \mathcal{P}$ and domain $d \in \mathcal{D}$, we employ `Claude-Haiku-4.5` to infer profiles based on the user's lifestyle and objectives (see Appendix C.2 for prompt details). This results in a triplet $\mathcal{M}_{p,d} = (a_{p,d}, f_{p,d}, m_{p,d})$ for every domain:

- **Domain Participation** ($a_{p,d} \in \{0, 1\}$): Whether the user with p uses an agent in domain d.
- **Frequency** ($f_{p,d} \in \{\text{high}, \text{mid}, \text{low}\}$): How often the user interacts within this domain.
- **Memory Necessity** ($m_{p,d} \in \{0, 1\}$): Requirement for context preservation. Crucially, $m_{p,d}$ is determined by user-specific intent rather than inherent domain properties.

We cross-verify the plausibility of the generated profiles using an ensemble of `GPT-5.1`, and `o3-mini`. Any domain is excluded from the user's profile if any model flag its metadata ($a_{p,d}$, $f_{p,d}$, or $m_{p,d}$) as implausible We sample a set $\mathcal{S}p$ of s domains from the persona's active pool $\mathcal{D}^p_{act} = d \mid a_{p,d} = 1$, ensuring a balanced distribution between domains with $m_{p,d} = 1$ and $m_{p,d} = 0$, thereby forming the final user-specific profile metadata.

3.2 Life Skeleton and Timeline Construction (III and IV of Figure 2)

Based on user-specific profiles, we utilize `gpt-5.4` to construct a life skeleton, a structured blueprint for simulating long-horizon user-agent interactions.

For domains requiring memory ($m_{p,d} = 1$), interactions are organized as a sequence of interconnected 'projects'. Each project consists of multiple events, each corresponding to a single dialogue session. An event includes an interaction summary, reference memories, and probing questions. Reference memories represent "worth-storing" information, such as user states and project progress, and serve as the gold standard that the memory system is expected to capture. Each reference memory is paired with a probing question designed to be answerable only if the corresponding information has been stored, enabling the evaluation of downstream memory retrieval and utilization. For transient domains ($m_{p,d} = 0$), interactions consist of independent events covering unrelated topics, without project-level dependencies and reference memories. The number of projects and events is determined by the frequency metadata ($f_{p,d}$).

Once the per-domain skeletons are established, an `gpt-5.4` arranges all events into a coherent, unified timeline. This integrated timeline provides the temporal and contextual structure needed to synthesize multi-turn dialogues that reflect a coherent and personalized long-horizon user experience. Please refer to Appendix C.3.1 and C.3.2 for detailed descriptions of the process.

3.3 Dialogue Generation via Dual-Simulator (V of Figure 2)

Using the life skeleton and integrated timeline, we synthesize realistic interactions through a dual-simulator framework. The user simulator generates context-driven utterances by manifesting the attributes—such as user state and project progress—defined in each event. In contrast, the agent simulator operates without prior access to the skeleton, responding solely based on the user's input and its internal memory. This process yields a long-horizon dialogue corpus that reflects the diverse and personalized requirements of agent use. Detailed procedure is presented in Appendix C.5.

4 Reflecting Shifts in Agent Use Profiles: PerMem-Bench^d

In real-world scenarios, user interests are often dynamic rather than static, evolving in response to significant life events such as career changes, new hobbies, or the conclusion of long-horizon projects. Such transitions inevitably lead to shifts in the user's agent use profiles. In this section, we describe the construction of PerMem-Bench^d, which simulates these profile shifts building upon the foundation of PerMem-Bench^s.

To model these transitions, we modify the user's predefined agent use profile by introducing additional domains from the previously unselected pool ($\mathcal{D}^p_{act} \setminus \mathcal{S}_p$), covering both memory-intensive ($m_{p,d} = 1$) and transient ($m_{p,d} = 0$) domains. Furthermore, we transition an existing domain in $\mathcal{S}_p$ from $m_{p,d} = 1$ to $m_{p,d} = 0$, reflecting the completion of a long-horizon project and its shift toward transactional interaction.

Based on this shifted profile, we leverage `gpt-5.4` to infer plausible life events that justify these transitions and construct a continued life skeleton following the methodology in Section 3.2. The resulting post-shift skeleton is arranged into a timeline and seamlessly appended to the pre-shift sequence. Finally, we perform dialogue generation using the same dual-simulator framework as described in Section 3.3, yielding a continuous, long-horizon trajectory that reflects the user's evolving interests and agent use profiles. Please refer to Appendix C.4 for detailed descriptions of the process.

5 Data Analysis and Meta Evaluation on PerMem-Bench

5.1 Data Analysis

In this section, we provide an exploratory analysis of PerMem-Bench. Table 1 summarizes the core statistics for both PerMem-Bench^s (Static) and PerMem-Bench^d (Dynamic).

Our simulation spans extensive timelines, covering up to 20 months in PerMem-Bench^s and 32 months in PerMem-Bench^d, with up to 1M dialogue-history tokens per user. Individual sessions contain up to 8K tokens, largely driven by detailed agent utterances commonly observed in real-world applications. These dense long-context environments challenge memory systems to distinguish worth-storing information from noise. In total, PerMem-Bench includes up to 146 reference memories and probing questions per user and provides over 1,000 evaluation examples in PerMem-Bench^s and nearly 2,000 in PerMem-Bench^d.

Table 1: Statistics of PerMem-Bench^s and PerMem-Bench^d. Min, Max, and Avg are computed across 20 users.

	PerMem-Bench^s			PerMem-Bench^d		
Metric	**Min**	**Max**	**Avg**	**Min**	**Max**	**Avg**
# Sessions	26	78	54	62	148	104
Timeline (mo)	15	20	17	25	32	28
Total Tokens	126K	522K	314K	340K	1M	634K
Avg. Tokens / Sess.	3.9K	7.7K	5.8K	3.6K	8.1K	6.1K
# Ref. Memories	38	72	53	78	146	97

To ensure the diversity of the generated agent use profiles, which are defined by the combination of active domains and their respective memory necessity, we calculated the Jaccard Similarity between users, treating the domain-memory necessity pairs as features. A similar-

ity value of 1 would indicate identical agent use patterns. As shown in Figure 3(a), the majority of pairs exhibit very low similarity, with no identical profiles existing in the set. To validate the scalability of this diversity, we sample 100 additional personas from the Nemotron-Persona-USA dataset and generate profiles using our pipeline. As illustrated in Figure 3(b), the results consistently demonstrate highly diverse use profiles. These findings confirm that PerMem-Bench effectively covers a broad spectrum of user behaviors in real-world agent application.

5.2 Meta Evaluation

To ensure the integrity of our data generation pipeline, we conduct a three-stage meta-evaluation. For each stage, we employ a panel of two evaluators—one human expert and one strong LLM judge (`Claude Opus 4.6`)—and report the averaged quality score alongside inter-evaluator agreement measured by Gwet's AC1 [22]. Full details are provided in Appendix D.

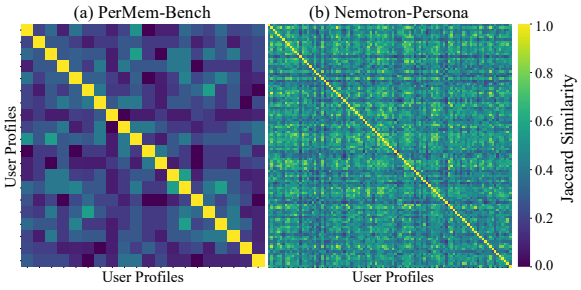

Figure 3: Similarity analysis on cross-user agent use profile. (a) Results on 20 users on PerMem-Bench. (b) Results on random 100 users from Nemotron-Persona.

Stage 1: Profile Plausibility. We assess whether the generated agent use profiles are logically consistent with the assigned user personas, evaluating both relevance and realism. The panel achieves an average quality score of 99.5% with an inter-evaluator agreement of 99.0%, indicating strong alignment between the generated profiles and the intended personas.

Stage 2: Life Skeleton and Timeline Realism. We evaluate the coherence of project sequences and event timelines, verifying that reference memories are appropriate for the user persona and that temporal progressions are realistic. Both evaluators reach perfect agreement, with a quality score and AC1 of 100%.

Stage 3: Dialogue Quality. We randomly sample 100 dialogue sessions and evaluate them along two dimensions: consistency with the life skeleton and seamless integration of reference memories. The panel achieves a quality score of 98.4% with an inter-evaluator agreement of 96.9%, confirming that the synthesized dialogues are faithful to the predefined life trajectories.

Collectively, these results validate the reliability of our fully automated generation pipeline. Since the pipeline requires no manual intervention, PerMem-Bench can be readily scaled to larger and more diverse user cohorts beyond the current 20-user set. The human evaluation for domain review and meta-evaluation was conducted internally by an author who is a Ph.D. student specializing in AI. No external evaluators were recruited, and no financial compensation was provided.

6 Evaluation Protocol of PerMem-Bench

An effective memory system must not only accurately extract, store, and persistently retain user-specific "worth-storing" contexts, but also retrieve and use them when they are needed for downstream tasks. Accordingly, PerMem-Bench evaluates memory systems along two complementary dimensions: whether they preserve tailored reference memories throughout their intended lifespans, and whether these memories remain accessible and useful for various downstream tasks.

6.1 Memory Storage Evaluation: Memory Retention Rate

We leverage the Memory Retention Rate (RR), a metric that measures how consistently a reference memory unit remains in the memory bank throughout its required lifespan. We categorize lifespans based on the nature of the information: user-centric states (e.g., stable preferences or permanent attributes) must be retained until a relevant update occurs or the timeline concludes, whereas project-specific progress (e.g., decisions or milestones) must be retained at least until the corresponding project concludes.

Formally, let $\mathcal{R}$ be the set of reference memory units. For each $r \in \mathcal{R}$, we define $t_{\text{start}}(r)$ as the session at which the information first appears in the dialogue, making it eligible for storage, and $T_{\text{target}}(r)$ as its target retention horizon determined by the information type above. The Memory Retention Rate is:

$$RR = \frac{\sum_{r \in \mathcal{R}} \sum_{t=t_{\text{start}}(r)}^{T_{\text{target}}(r)} \mathbb{I}(r \in \mathcal{M}_t)}{\sum_{r \in \mathcal{R}} \left(T_{\text{target}}(r) - t_{\text{start}}(r) + 1\right)}. \tag{1}$$

where $\mathcal{M}_t$ denotes the memory bank state at session t, and $\mathbb{I}(\cdot)$ is an indicator that equals 1 if r is present in $\mathcal{M}_t$ and 0 otherwise.

Practical Implementation. To determine $\mathbb{I}(r \in \mathcal{M}_t)$, we adopt an LLM-as-a-judge framework (`gpt-5-nano` is used) that verifies whether r is preserved in $\mathcal{M}_t$. Rather than exhaustively checking all entries, the judge considers only the top-10 semantically similar entries retrieved from $\mathcal{M}_t$ using r as the query, and performs a binary verdict. To validate the reliability of our LLM judge (gpt-5-nano), we randomly sampled 100 (reference memory, memory bank state) instances and had a human annotator independently judge whether each reference memory was present, following the same criteria as the judge prompt. Comparing human judgments against the LLM judge's verdicts, we obtain an F1 score of 0.921 (Precision of 0.953 and Recall of 0.891), indicating strong agreement between the LLM judge and human annotation on this binary presence-verification task.

Computing this indicator at every session for every reference memory is nonetheless prohibitively expensive. We therefore approximate the inner summation by sampling $K{=}20$ evenly spaced checkpoints from $[t_{\text{start}}(r),\ T_{\text{target}}(r)]$—always including the first and last sessions—and rescale the sampled scores to approximate the full sum, with the rescaling factor $\frac{|S(r)|}{K}$ treating each sampled checkpoint as representative of $\frac{|S(r)|}{K}$ consecutive sessions. Full implementation details are provided in Appendix E.

6.2 Downstream Task Evaluation: Retrieval and QA

While memory retention measures whether a memory system preserves reference information over time, it does not directly assess whether the retained information can be retrieved and used by an agent. We therefore complement retention evaluation with *probing-question-based* downstream task evaluation. During life-skeleton construction, we associate each reference memory with a probing question whose answer depends on that memory. These questions cover both user-specific facts, such as preferences and traits, and task-state facts, such as decisions, milestones, and plans.

This formulation provides a task-agnostic proxy for downstream utility, as these are precisely the types of information that a downstream agent must recover before it can correctly perform a task grounded in that information. For example, a personalized agent system cannot suggest an appropriate dish without knowing the user's dietary preferences, while it cannot correctly continue a project without knowing its latest state or prior decisions. Correctly answering the associated probing question therefore tests the shared prerequisite of accessing the information needed for such downstream behavior, without tying the evaluation to a particular application or action space.

Crucially, we conduct this downstream evaluation in the same evaluation loop as memory retention. For each reference memory r and each sampled checkpoint $t \in S(r)$ used to estimate RR, we first apply the retention judge described above to the memory bank state $\mathcal{M}_t$. In the same pass, we use the probing question associated with r to query $\mathcal{M}_t$, retrieve the top-5 entries, and prompt the LLM to answer the question using only those entries. Thus, retention, retrieval, and QA are all evaluated on the identical $(r, \mathcal{M}_t)$ instances and sampled checkpoints.

Retrieval Accuracy records whether r is included among the top-5 entries retrieved by its probing question. QA Accuracy records whether the generated answer is judged correct with respect to r. Together with the retention verdict, these measures distinguish whether a reference memory is preserved in the bank, accessible through retrieval, and usable for answering a downstream query.

7 Can Memory Systems Be Personalized?

An ideal personalized memory system should infer user-specific worth-storing information from interaction history and manage its memory bank accordingly. In this section, we conduct an empirical study to evaluate whether current LLM-based memory systems can be effectively personalized, and identify directions for future development.

7.1 Experimental Setup

7.1.1 Memory Personalization via Session-level Storage Gating

We propose a simple yet general framework for personalizing memory systems via *session-level storage gating*. After each session, a gating module inspects the session dialogue and, optionally, prior context to predict whether the session is part of a long-horizon task or a transient interaction. If the session is classified as transient, memory operations for that session are skipped entirely. This lightweight wrapper requires no modification to the underlying memory system, and allows the memory budget to be concentrated on sessions that genuinely benefit from long-term context accumulation.

We evaluate gating methods along a spectrum of increasing contextual richness, from purely session-local signals to explicit structural modeling of cross-session dependencies. To bracket the range of achievable performance, we also define two reference points:

Universal. The memory system operates without any gating, applying its default storage policy uniformly to all sessions. This represents the current state of deployed memory systems.

Oracle. The ground-truth agent-use profile is provided directly, giving perfect knowledge of which sessions are long-horizon and which are transient. This serves as the upper bound for any gating method.

The three gating methods we evaluate are as follows.

Greedy. The simplest instantiation of storage gating. At each session, an LLM predicts whether the session is long-horizon or transient based solely on the current session's dialogue, with no access to prior context. This captures the intuition that long-horizon and transient interactions often exhibit distinguishable surface-level patterns (e.g., references to ongoing goals vs. self-contained queries), without requiring any cross-session reasoning.

Context-aware. To address the absence of historical signal in the Greedy method, each session is summarized in one to two sentences after processing, and a sliding window of the most recent K summaries is passed as context when predicting subsequent sessions. This allows the gating module to exploit sequential patterns in the user's interaction history, particularly for sessions that are ambiguous in isolation.

Structure-aware Method. Rather than treating history as a flat sequence of summaries, this method explicitly models the *relational structure* among sessions—identifying which sessions form coherent long-horizon projects and which are isolated one-off interactions. To this end, we maintain a *structural note*, a structured representation of the user's emerging usage patterns updated every K sessions:

$$\{\texttt{projects} : [\{\texttt{project_id}, \texttt{topic}, \texttt{session_ids}, \texttt{status}\}],\ \texttt{isolated_sessions} : [\texttt{session_ids}]\}$$

Crucially, the note is carried forward across windows rather than reset, allowing the system to retroactively reassign sessions (e.g., recognizing that a previously isolated session belongs to a project identified later). Sessions in `isolated_sessions` are classified as transient; sessions in any project are classified as worth storing. Among the three methods, Structure-aware is the only one that approximates domain-level usage pattern inference—the other two operate purely at the session level.

Implementation details for all methods are provided in Appendix F.2.

7.1.2 Memory Systems

We adopt three recent memory systems as evaluation targets: Mem0 [1], Memory-R1 [3], and RMM [9]. These systems employ LLM-based memory operations—including selective extraction, storage, update, and deletion—to maintain a persistent memory bank. Memory operations are applied at two granularities: *turn-level* and *session-level*.[2] For session-level settings, we use `gpt-5-mini` as the backbone LLM. For turn-level settings, where the higher frequency of LLM calls makes large models cost-prohibitive, we use the open-source `Qwen3-14B`.[3]

Memory entries are stored as text with associated embeddings in a vector database, and the memory budget is defined as the maximum number of entries allowed. To manage budget constraints, we adopt

[2] RMM supports session-level operation only.

[3] As Memory-R1 does not release its trained model weights, we use the base LLM for this system.

Table 2: Session gating classification performance on PerMem-Bench^s and PerMem-Bench^d.

Model	Method	PerMem-Bench^s F1 (↑)	PerMem-Bench^s FNR (↓)	PerMem-Bench^s FPR (↓)	PerMem-Bench^d F1 (↑)	PerMem-Bench^d FNR (↓)	PerMem-Bench^d FPR (↓)
Qwen3 14B	Greedy	0.660	0.457	0.117	0.657	0.444	0.178
	Context	0.434	0.715	**0.008**	0.477	0.665	**0.071**
	Structure	**0.844**	**0.115**	0.280	**0.805**	**0.110**	0.461
gpt-5 mini	Greedy	0.733	0.301	0.259	0.715	0.287	0.378
	Context	0.751	0.287	**0.211**	0.733	0.261	**0.375**
	Structure	**0.795**	**0.018**	0.652	**0.784**	**0.010**	0.782

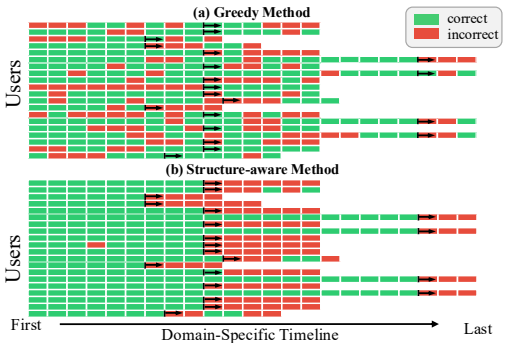

Figure 4: Session gating accuracy under a use profile shift. "→" marks the shift point.

a hybrid deletion strategy combining each system's built-in mechanism with a rule-based time-decay approach following [6, 23, 4, 8], where the oldest entries are evicted first to emulate human memory fading. Unless otherwise stated, all experiments use a default budget of 200 entries.

7.1.3 Evaluation Metrics

For session-wise gating quality, we report **F1**, **False Negative Rate (FNR)**, and **False Positive Rate (FPR)**. A false negative (FN) occurs when a worth-storing session is misclassified as transient. A false positive (FP) occurs when a transient session is stored unnecessarily. Both error types are undesirable, making F1 the primary overall measure.

For memory system performance, we use the Memory Retention Rate (RR), Retrieval Accuracy, and QA Accuracy, defined in Section 6.

7.2 Experimental Results

Finding 1: Structure-aware gating shows promise in recovering domain-level usage patterns, but struggles to detect profile shifts.

As shown in Table 2, the Structure-aware method achieves up to 0.844 F1 on PerMem-Bench^s, demonstrating that explicitly modeling cross-session relational structure enables LLMs to approximate domain-level usage patterns from interaction history alone. In contrast, the lower performance of Greedy and Context-aware methods highlights that session-local signals — whether from the current session alone or a flat window of past summaries — are insufficient for this purpose, as these methods operate purely at the session level without inferring any domain-level structure.

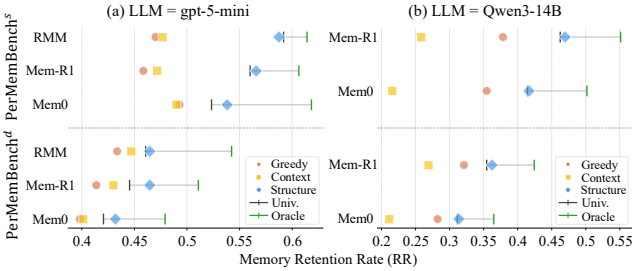

Figure 5: Memory storage performance comparison.

However, as shown in Figure 4, when we analyze prediction accuracy on domains undergoing a "long-horizon → transient shift", Structure-aware predictions are largely correct prior to the shift but collapse immediately afterward. We attribute this to over-reliance on established project structure: once a project cluster is formed in the structural note, the model continues assigning subsequent sessions to it even after the usage pattern has fundamentally changed. Interestingly,

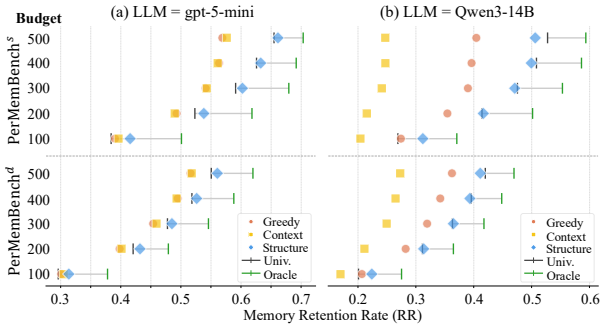

Figure 6: Sensitivity Analyses of memory budget on Mem0. The memory retention rate is reported.

Greedy — which evaluates each session in isolation — shows no such collapse, exhibiting consistent performance before and after the shift. This suggests that combining the complementary strengths of structure-aware and session-level reasoning is a promising direction for robust session-level gating under dynamic conditions.

Finding 2: Memory personalization improves both retention and downstream utility over universal policy, with larger gains under tighter memory budgets.

As shown in Figure 5, comparing Oracle and Universal across all memory systems reveals substantial retention improvements when the agent use pattern is known exactly. By avoiding wasteful storage on transient sessions and allocating the full budget to genuinely worth-storing contexts, personalization dramatically improves the utilization of limited memory capacity. Furthermore, Figure 6 shows that the benefit of personalization is most pronounced at smaller budgets: the Oracle–Universal gap is largest at budgets of 100 and 200, and narrows as the budget increases. Nevertheless, even at a budget of 500, personalization continues to yield meaningful gains, confirming that the benefit is not limited to severely constrained settings.

This benefit extends beyond retention to downstream utility. As shown in Figure 7, Oracle consistently outperforms Universal on both Retrieval and QA across all memory systems and benchmark settings. Thus, when the agent-use profile is known, selectively allocating memory capacity to long-horizon sessions not only preserves more reference memories, but also makes the relevant information more likely to be retrieved and correctly used to answer task-relevant questions. Together with the retention results in Figure 5, these findings establish that the potential benefit of personalization is functional rather than merely representational.

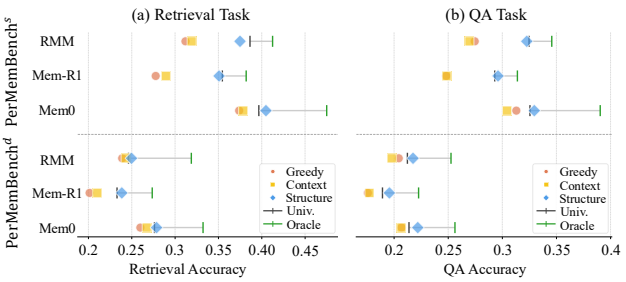

Figure 7: Personalized memory system performance (Retrieval and QA). For the backbone LLM, gpt-5-mini is used.

Finding 3: The potential of personalization is large, but current gating performance is not yet accurate enough to realize it.

However, when session-level gating is imperfect, the picture changes dramatically: Greedy and Context-aware consistently underperform Universal, and Structure-aware yields only marginal improvements despite its higher gating accuracy. The gap between Oracle and the best gating method thus represents unrealized potential — not an argument against personalization, but a precise measure of how much headroom remains. Closing this gap through more accurate session-level gating is therefore the most impactful direction for future work.

8 Conclusion

We formalize the need for personalized memory systems and present PerMem-Bench, the first benchmark that jointly models multi-domain interaction histories and cross-user heterogeneity in agent-use patterns, along with a fully automated construction pipeline whose reliability is validated through rigorous meta-evaluation. We further propose a novel memory personalization paradigm, *session-level storage gating*, along with simple baselines. Our results confirm that personalization yields substantial retention gains when the user's agent use pattern is exactly inferred, with benefits most pronounced under tighter memory budgets — underscoring the critical importance of memory personalization in resource-constrained deployment. Nevertheless, accurate session-level gating remains an open and pressing challenge for future work.

Acknowledgements

This work was supported by National IT Industry Promotion Agency(NIPA) grant funded by the Korea government(MSIT) (RS-2026-25621604), IITP grant funded by the Korea government(MSIT) (RS-2026-25520654), and National Research Foundation of Korea(NRF) funded by Ministry of Science and ICT (RS-2022-NR068758).

References

[1] 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.

[2] Zijian Zhou, Ao Qu, Zhaoxuan Wu, Sunghwan Kim, Alok Prakash, Daniela Rus, Jinhua Zhao, Bryan Kian Hsiang Low, and Paul Pu Liang. Mem1: Learning to synergize memory and reasoning for efficient long-horizon agents. *arXiv preprint arXiv:2506.15841*, 2025.

[3] Sikuan Yan, Xiufeng Yang, Zuchao Huang, Ercong Nie, Zifeng Ding, Zonggen Li, Xiaowen Ma, Jinhe Bi, Kristian Kersting, Jeff Z Pan, et al. Memory-r1: Enhancing large language model agents to manage and utilize memories via reinforcement learning. *arXiv preprint arXiv:2508.19828*, 2025.

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

[5] Ke Yang, Zixi Chen, Xuan He, Jize Jiang, Michel Galley, Chenglong Wang, Jianfeng Gao, Jiawei Han, and ChengXiang Zhai. Plugmem: A task-agnostic plugin memory module for llm agents. *arXiv preprint arXiv:2603.03296*, 2026.

[6] Charles Packer, Vivian Fang, Shishir_G Patil, Kevin Lin, Sarah Wooders, and Joseph_E Gonzalez. Memgpt: towards llms as operating systems. 2023.

[7] Yu Wang, Ryuichi Takanobu, Zhiqi Liang, Yuzhen Mao, Yuanzhe Hu, Julian McAuley, and Xiaojian Wu. Mem-{\alpha}: Learning memory construction via reinforcement learning. *arXiv preprint arXiv:2509.25911*, 2025.

[8] Yuyang Hu, Shichun Liu, Yanwei Yue, Guibin Zhang, Boyang Liu, Fangyi Zhu, Jiahang Lin, Honglin Guo, Shihan Dou, Zhiheng Xi, et al. Memory in the age of ai agents. *arXiv preprint arXiv:2512.13564*, 2025.

[9] Zhen Tan, Jun Yan, I-Hung Hsu, Rujun Han, Zifeng Wang, Long Le, Yiwen Song, Yanfei Chen, Hamid Palangi, George Lee, et al. In prospect and retrospect: Reflective memory management for long-term personalized dialogue agents. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8416–8439, 2025.

[10] Memobase, 2026.

[11] Alireza Rezazadeh, Zichao Li, Wei Wei, and Yujia Bao. From isolated conversations to hierarchical schemas: Dynamic tree memory representation for llms. *arXiv preprint arXiv:2410.14052*, 2024.

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

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

[14] Jiho Kim, Woosog Chay, Hyeonji Hwang, Daeun Kyung, Hyunseung Chung, Eunbyeol Cho, Yeonsu Kwon, Yohan Jo, and Edward Choi. Dialsim: A dialogue simulator for evaluating long-term multi-party dialogue understanding of conversational agents. *arXiv preprint arXiv:2406.13144*, 2024.

[15] Di Wu, Hongwei Wang, Wenhao Yu, Yuwei Zhang, Kai-Wei Chang, and Dong Yu. Longmemeval: Benchmarking chat assistants on long-term interactive memory. *arXiv preprint arXiv:2410.10813*, 2024.

[16] 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.

[17] Ding Chen, Simin Niu, Kehang Li, Peng Liu, Xiangping Zheng, Bo Tang, Xinchi Li, Feiyu Xiong, and Zhiyu Li. Halumem: Evaluating hallucinations in memory systems of agents. *arXiv preprint arXiv:2511.03506*, 2025.

[18] Cheng Jiayang, Dongyu Ru, Lin Qiu, Yiyang Li, Xuezhi Cao, Yangqiu Song, and Xunliang Cai. Amemgym: Interactive memory benchmarking for assistants in long-horizon conversations. *arXiv preprint arXiv:2603.01966*, 2026.

[19] Yev Meyer and Dane Corneil. Nemotron-Personas-USA: Synthetic personas aligned to real-world distributions, June 2025.

[20] Aaron Chatterji, Thomas Cunningham, David J Deming, Zoe Hitzig, Christopher Ong, Carl Yan Shan, and Kevin Wadman. How people use chatgpt. Technical report, National Bureau of Economic Research, 2025.

[21] OpenAI. Chatgpt usage and adoption patterns at work, January 2026.

[22] Kilem Gwet. Handbook of inter-rater reliability. *Gaithersburg, MD: STATAXIS Publishing Company*, pages 223–246, 2001.

[23] Yu Wang, Dmitry Krotov, Yuanzhe Hu, Yifan Gao, Wangchunshu Zhou, Julian McAuley, Dan Gutfreund, Rogerio Feris, and Zexue He. M+: Extending memoryllm with scalable long-term memory. *arXiv preprint arXiv:2502.00592*, 2025.

A Limitations and Future Works

Benchmark scale. PerMem-Bench currently encompasses 20 users, which may limit the diversity of agent use patterns represented. We note, however, that our fully automated construction pipeline can be readily scaled to larger and more diverse user cohorts without manual intervention.

Simplicity of agent use profile modeling. We model agent use profiles as a joint configuration of domain participation and memory necessity — a deliberately simple formulation. While finer-grained behavioral patterns undoubtedly exist, our primary goal is to shed the first light on the necessity of memory personalization and lay a foundational stepping stone for this research direction. Extending PerMem-Bench to richer profile representations remains an important avenue for future work.

Personalization limited to storage operations. Our proposed *session-level storage gating* paradigm exclusively applies to storage, with no mechanism to retroactively correct mistakenly stored entries. A personalized deletion policy that evicts user-specifically unnecessary memories could make memory management substantially more effective, and we leave this as future work.

Session-level gating accuracy. As shown in Finding 3, current gating methods remain insufficient to fully realize the gains of personalization. We believe agentic post-training — optimizing the gating module through agent-environment interaction — is a promising direction for closing this gap.

Human-in-the-Loop Memory Personalization. An important direction for future work is to incorporate explicit user control into personalized memory management. Rather than allowing agents to autonomously determine which information should be retained, practical systems could adopt a user-driven memory personalization paradigm. Specifically, users could explicitly opt in to personalization during onboarding and review candidate memory entries extracted from previous interactions before approving their storage. Furthermore, a dedicated memory management interface could allow users to inspect, edit, correct, or delete stored memories at any time. Such mechanisms would improve transparency and user agency while mitigating the risks of unwanted profiling and incorrect memory retention.

B Societal Impacts and Risks

While long-horizon personalized memory frameworks significantly enhance the continuity and helpfulness of AI agents, deploying persistent memory systems introduces non-trivial ethical, privacy, and safety risks that warrant careful consideration.

Surveillance Dangers and Behavioral Profiling. Silently tracking and aggregating user interactions across multiple life domains over long periods creates a detailed behavioral digital footprint. If mishandled, such long-term profiles could be exploited for intrusive surveillance, unwanted psychological profiling, or targeted behavioral manipulation. To mitigate these risks, memory personalization should be designed as a user-governed process rather than a fully automated and opaque one. In particular, systems should obtain explicit user consent before storing personal information, especially when the information is sensitive or intended to be retained for long periods.

User Control, Auditability, and Memory Correction. A static or opaque memory store risks retaining outdated, incorrect, or sensitive personal inferences. Such errors can lead to inappropriate personalization and may repeatedly reinforce incorrect assumptions about the user. To preserve user agency over personal data, memory systems should provide transparent and granular controls that allow users to inspect, edit, correct, and delete stored memories. They should also maintain auditable records of when, why, and from which interaction a memory was created or updated.

Cross-User Data Isolation and Leakage Safeguards. In multi-user or multi-tenant deployment environments, cross-user memory contamination represents a serious privacy risk. Although our benchmark strictly isolates the memory database of each synthetic persona, real-world systems may still be vulnerable to leakage through shared indexes, caches, retrieval pipelines, or backend infrastructure. Developers should therefore enforce strict tenant-level data boundaries, robust authentication and authorization, encryption, and systematic access verification. Production systems should also be regularly audited and tested to ensure that one user's stored history cannot be retrieved, inferred, or exposed in another user's interactions.

C Benchmark Construction Details

C.1 Domain Pool Construction

To ensure representative coverage of real-world usage, we employ a data-driven approach to construct a domain pool. First, we sample 1,000 personas and prompt `Claude-Haiku-4.5` to generate potential usage scenarios without predefined constraints using following prompt:

Domain Extraction Prompt

You are analyzing user personas to understand how people use AI agents in their daily lives.
User Persona
`{persona}`
Task
Based on this user's background, lifestyle, occupation, hobbies, and goals, infer what kinds of tasks or purposes this user would use an AI agent for.
Instructions
- Generate between 5 and 10 distinct domains of AI agent use for this user.
- Each domain must be grounded in this user's specific context. Do not generate generic domains that would apply to any user.
- domain_name must be short (2-5 words) and general enough to apply to other users with similar backgrounds. All persona-specific context belongs in the reason field only.
- Too specific (bad): "ML/NLP research ideation, experiment design, and literature synthesis"
- Good: "Academic Research & Literature Review"
- Too vague (bad): "Work-related tasks"
- Cover a diverse range of usage purposes across different aspects of the persona (work, hobbies, health, personal goals, etc.). Do not cluster domains around a single aspect of the persona.
Output Format
Return a JSON object in the following format:
[{
"domain_name": "...", "reason": "One sentence grounding this domain in this user's specific context."
}]

These candidates are then semantically clustered and assigned representative labels via human review. To align the pool with actual LLM trends, we cross-reference these clusters with industry reports [20, 21], pruning niche cases and supplementing broad-interest domains. This process results in a final taxonomy of 20 domains (see Table 3).

Table 3: The list of defined domain pools.

#	Domain
1	Academic Study & Learning
2	Business & Entrepreneurship
3	Career Development & Job Search
4	Data Analysis & Visualization
5	Event Planning
6	Health & Wellness
7	Home & Real Estate
8	Language Learning
9	Legal & Administrative Affairs
10	Math & Quantitative Problem Solving
11	Mental Health & Emotional Support
12	News & Current Events
13	Personal Finance & Investment
14	Recipe Advice & Meal Planning
15	Relationship & Social Advice
16	Shopping & Product Research
17	Software Development & Coding
18	Sport & Physical Activity
19	Travel Planning
20	Writing Assistant

C.2 User-Specific Profile Assignment

For each persona $p \in \mathcal{P}$ and domain $d \in \mathcal{D}$, we employ `Claude-Haiku-4.5` to infer profiles based on the user's lifestyle and objectives using the following prompt:

User-Specific Profile Assignment Prompt

You are building a user profile for a personalized AI agent system.
`## User Persona`
{persona}
`## Domains`
{domain_list}
`## Task`
For every domain listed above, determine whether this user would use an AI agent for it, and if so, whether memory is required and how frequently they would use it.
`## Instructions`
- Evaluate all n_domains domains without exception.
- For each domain, first determine whether this user would plausibly use an AI agent for it (use: true/false).
- If use is true, determine whether memory is required (memory_required: true/false) and how frequently this user would use this domain (frequency: "high"/"medium"/"low").
- If use is false, set memory_required to null and frequency to null.
- Base your judgment entirely on this user's specific context, not on general assumptions about the domain.
`## Definition of Memory Required`
Memory is required (true) when this user's usage of the domain is:
- Ongoing and accumulative (e.g., tracking progress over time)
- Connected across multiple sessions (e.g., building on past conversations)
- Tied to long-term goals or evolving personal circumstances
Memory is NOT required (false) when this user's usage of the domain is:
- One-time or ad-hoc (e.g., a single lookup with no follow-up)
- Self-contained within a single session
- Not dependent on past interactions
`## Definition of Frequency`
- "high": This user would use an AI agent for this domain very regularly
- "medium": This user would use an AI agent for this domain occasionally
- "low": This user would use an AI agent for this domain rarely
`## CRITICAL`
Both memory_required and frequency must reflect THIS USER's specific context, not the general nature of the domain.
The same domain can have different memory_required and frequency values for different users.
For example:
- "Recipe Advice & Meal Planning" → memory_required=false, frequency="low" for a user who occasionally looks up recipes, but memory_required=true, frequency="high" for a user who is developing their own menu or working toward opening a restaurant.
- "Fitness & Exercise Planning" → memory_required=false, frequency="low" for a user who casually checks workout tips, but memory_required=true, frequency="high" for a user who follows a structured training routine and tracks progress over time.
- "News & Current Events" → memory_required=false, frequency="low" for a user who reads news casually, but memory_required=true, frequency="high" for a user who actively monitors specific topics for research or professional purposes.
`## Output Format`
Return a JSON object in the following format:

```
[
    {
      "domain_name": "...",
      "use": true/false,
      "memory_required": true/false/null,
      "frequency": "high"/"medium"/"low"/null,
      "reason": "One sentence explaining why, grounded in this user's specific context."
    }
  ]
```

C.3 Life Skeleton and Timeline Construction Details

This subsection provides a detailed description of the Life Skeleton and Timeline Construction pipeline (§3.2), covering both PerMem-Bench^s (static) and PerMem-Bench^d (dynamic).

C.3.1 Life Skeleton Construction for PerMem-Bench^s

Long-Horizon Domains. For each memory-required domain ($m_{p,d} = 1$), an LLM generates a structured life skeleton comprising a sequence of projects and events. The number of projects and events per project is determined by the frequency metadata $f_{p,d}$ as follows:

Frequency	# Projects	Events/Project
High	5	3–5
Medium	3	2–4
Low	2	2–3

Each event is annotated with reference memory items of two types: `user_profile` (stable facts persisting across all projects) and `ongoing_state` (project-scoped decisions and progress). To prevent redundant reference memories across domains, skeletons are generated sequentially — each domain receives a list of facts already recorded by previously generated domains, and is instructed not to duplicate them.

Life Skeleton Generation (Memory-Required Domain) Prompt (condensed version)

[System]
You are designing a Life Skeleton for a personalized AI agent memory benchmark. A Life Skeleton captures how a specific user engages with an AI agent in a particular domain over 1–2 years. . . [see full prompt in code]
[User]
`## User Persona`
{persona}
`## Domain`
- Name: {domain_name} Frequency: {frequency} Why used: {reason}
`## Task`
Generate a Life Skeleton over a 1–2 year period. Scale: {n_projects} projects, {n_events_min}–{n_events_max} events per project.
`## GT Memory Definitions`
user_profile — persists forever after learned. Skills, tools mastered, revealed preferences. Only record if NOT already in persona.
ongoing_state — lives only while the project is active. Decisions made, tools chosen, progress reached. CRITICAL: Must be things decided DURING conversation — not things user already knows.
`## Already Covered (DO NOT duplicate)`
{covered_facts_from_prior_domains}
Output: JSON with `projects[].events[].gt_memory[{type, fact, probing_question, answer}]`

Transient Domains. For transient domains ($m_{p,d} = 0$), independent one-off events are generated without project structure or reference memory. The number of events is derived from the timeline duration estimated from the memory-required skeletons and the domain's interaction frequency:

$$n_{\text{events}} = \left\lfloor \frac{T_{\text{total}} \times 4}{w_{f_{p,d}}} \right\rfloor \tag{2}$$

where T_{total} is the estimated timeline duration in months and w_f is the inter-session interval in weeks ($w_{\text{high}} = 4$, $w_{\text{medium}} = 8$, $w_{\text{low}} = 12$).

One-Off Event Generation (Transient Domain) Prompt (condensed version)

[System]
You are designing one-off AI agent interaction events. . . These events are self-contained, single-session interactions with no longitudinal engagement, no project structure, and no memory required across sessions.
[User]
`## User Persona`
{persona}
`## Domain`
- Name: {domain_name} Frequency: {frequency}
`## Task`
Generate exactly {n_events} one-off events spread across ∼{total_months} months. Each event should reflect a genuinely different moment and need.
Output: JSON with `events[{event_id, event_title, event_description}]`

C.3.2 Timeline Integration for PerMem-Bench^s

Once per-domain skeletons are constructed, an LLM arranges all memory-required events into a unified chronological timeline. The LLM is responsible solely for placing memory-required events; transient events are placed programmatically after the LLM call using the frequency-based spacing formula above. Events beyond the LLM-determined `total_months` are silently truncated.

Timeline Arrangement Prompt (condensed version)

[System]
You are designing an integrated session timeline... Your job is to arrange all events into a single, realistic session timeline — a chronologically ordered list of agent sessions that this person would actually have, given their real life circumstances.
`## Key Principles`
1. Ground in the persona's life.
2. Respect project sequentiality within each domain.
3. Interleave domains realistically.
4. Assign concrete month numbers.
5. Flag cross-domain links.
[User]
`## User Persona`
{persona}
`## Domain Life Skeletons`
{skeleton_summary}
`## Events to Place (all must appear exactly once)`
{domain | project_id | event_id | event_title}
`## Requirements`
- Every event must appear exactly once.
- Assign realistic month (1–{max_month}) to each session.
- Identify anchor life events triggering multiple domains simultaneously.
- Respect project ordering within each domain.
- session_id must be sequential in chronological order.
Output: JSON with `total_months, anchor_life_events, session_sequence`

C.4 Profile Shift and Life Skeleton Construction for PerMem-Bench^d

PerMem-Bench^d extends PerMem-Bench^s by introducing a profile shift at the end of the timeline in PerMem-Bench^s. The shift is determined by three fixed rules applied via rule-based sampling, with no LLM involvement:

1. **Demotion**: one existing memory-required domain is demoted to transient.
2. **New longitudinal domain**: one new memory-required domain is sampled from the unused pool.
3. **New transient domain**: one new transient domain is optionally added from the unused pool.

Domains are sampled with frequency-weighted probability ($w_{\text{high}} = 3$, $w_{\text{medium}} = 2$, $w_{\text{low}} = 1$) and each persona receives a deterministic seed derived from the global seed and its UUID, ensuring reproducibility.

Transition Narrative. A coherent life transition event is generated to justify all three changes simultaneously.

Life Transition Narrative Generation Prompt (condensed version)

[System]
You are writing a life transition narrative... Domain changes have already been decided. Write a short, coherent life transition event that naturally explains all the changes.
[User]
`## User Persona`: {persona}
`## Current Usage (Phase 1, {total_months} months)`
Memory-Required: {mem_domains} One-Off: {oneoff_domains}
`## Phase 2 Changes (already decided)`
1. DEMOTED to occasional use: {mem_to_oneoff_name}
2. NEW longitudinal domain: {added_mem_name}
3. NEW one-off domain: {added_oneoff_name}
Output: JSON with `{name, description}`

Life Skeleton Generation after the Shift. Using the transition narrative, Phase 2 skeletons are generated for each domain with two variants: *added* (new domains starting from scratch) and *retained*

(existing domains continuing into Phase 2 with new projects). The demoted domain is treated as a transient domain and generates one-off events instead. Prompts follow the same structure as in PerMem-Bench generation pipeline, with the transition event appended as additional context and a list of reference memories in PerMem-Bench^s to avoid duplication.

Timeline Integration after the Shift. Events occurring after the shift are arranged into a timeline using the same LLM prompt structure as previous one, with months expressed relative to the point where the shift starts. The final output is a continuous `all_sessions` list spanning both phases.

C.5 Dialogue Generation via Dual-Simulator

Each entry in the unified timeline corresponds to one dialogue session, generated by two separate LLM instances: a *user simulator* and an *agent simulator*. The two simulators are strictly isolated — the agent simulator receives no access to the life skeleton or reference memories, responding solely based on the user's utterances and its parametric knowledge, exactly as a real deployed agent would.

Sessions are generated differently based on `memory_required`: memory-required sessions ($m_{p,d} = 1$) provide the user simulator with the event description and reference memory items from the life skeleton, while transient sessions ($m_{p,d} = 0$) provide only the event description with no reference memory, terminating once the one-off need is met.

A key design challenge is preventing the user simulator from explicitly declaring reference memory facts upfront, which would make the resulting dialogues artificially unnatural. To address this, `user_profile` facts are framed as background traits that should surface implicitly through the user's reactions, while `ongoing_state` facts are converted into open uncertainties before being passed to the simulator — so that decisions emerge organically during the conversation rather than being pre-announced. Cross-session continuity is maintained by providing the user simulator with a summary of facts already established in prior sessions of the same project.

To verify that reference memory facts are covered within each session, an LLM judge tracks which facts have been expressed by the user after each turn, and surfaces any unrevealed facts as gentle nudges in the continuation prompt. The session concludes once all facts have been naturally revealed and a minimum turn count is reached.

The prompt structure provided to the user simulator is as follows.

User Simulator Prompt (condensed version)

[System]
You are roleplaying as a real person interacting with an AI agent. {persona}. You are always the USER seeking help, never the assistant.
[Opening turn]
You are about to start a new conversation about: {domain_name}
`## What is happening in your life right now`
{event_description}
`## What you already know from previous conversations`
{prior_context}
`## Who you are in this conversation`
- *[user_profile]* Traits and preferences — let these shape how you react. DO NOT announce them upfront.
- *[ongoing_state]* Things you have NOT yet decided — let these emerge through the conversation.
Open with 1–3 sentences only. The agent has NO memory of you.
[Continuation turn]
`## Not yet surfaced:` {unrevealed_facts}
`## Already came up - do NOT repeat:` {revealed_facts}
Reply `[END]` if the conversation has reached a natural conclusion.

D Meta Evaluation Details

This section describes the detailed procedures used to validate the reliability of our data generation pipeline across three stages.

D.1 Stage 1: Profile Plausibility

We randomly sample 100 personas from the Nemotron-Persona-USA dataset and run our agent use profile assignment algorithm on each. The resulting profiles are then evaluated for plausibility by the two-evaluator panel. The prompt provided to `Claude Opus 4.6` is as follows.

Profile Plausibility Judge Prompt

You are a strict evaluator for persona-conditioned metadata quality.
You must decide whether the metadata for this user is overall valid and plausible, based on the user's persona.
`## User Persona`
{persona_text}
`## Generated Metadata`
{domains_json}
`## Decision Criteria`
Judge "YES" only if ALL are true:
1) Domain-level choices are generally persona-grounded.
2) There are no major logical contradictions.
3) use/memory_required/frequency are mostly coherent with each other.
4) Reasons are mostly plausible and consistent with the persona.
Judge "NO" if there are clear invalid/unreasonable patterns such as:
- repeated persona-irrelevant domains marked as use=true without evidence,
- frequent inconsistency (e.g., use=false but memory/frequency not null),
- clearly implausible frequency/memory decisions for many domains,
- major contradiction between reasons and persona.
`## Output format (STRICT JSON only, no markdown)`

```
{
  "verdict": "YES" or "NO",
  "confidence": 0.0 to 1.0,
  "summary": "2-4 concise sentences",
  "major_issues": [
    "issue 1",
    "issue 2"
  ]
}
```

The annotation interface provided to the human expert is shown in Figure 8.

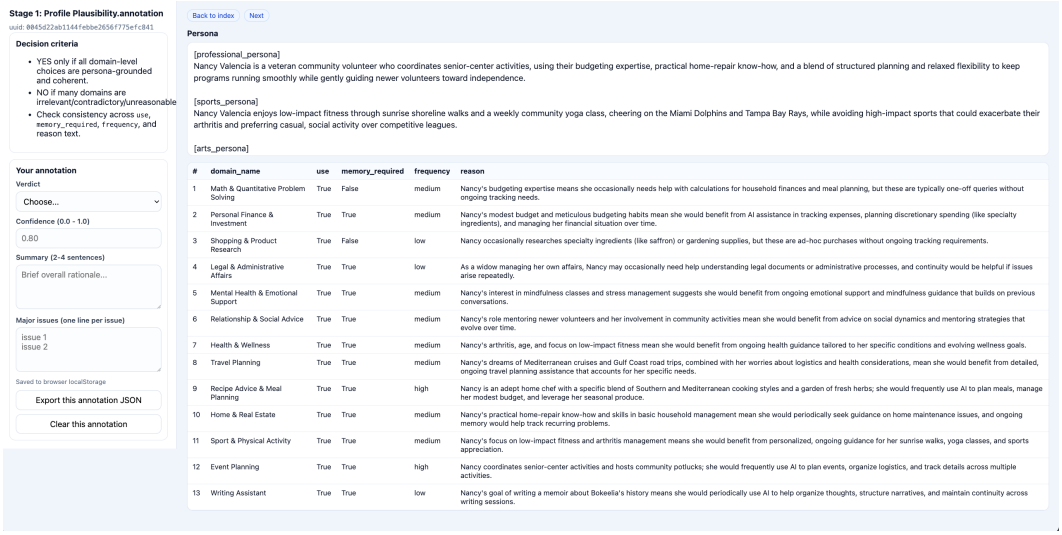

Figure 8: The user interface provided to the annotators for profile plausibility annotation.

D.2 Stage 2: Life Skeleton and Timeline Realism

Using the same 100 sampled personas, we run our life skeleton and timeline construction algorithm and evaluate the quality of the resulting outputs. The prompt provided to `Claude Opus 4.6` is as follows.

Life Skeleton and Timeline Realism Judge Prompt

You are a strict evaluator of synthetic life-skeleton quality.
Your task is to judge whether this user's generated life skeleton (especially project/event structure) is plausible and realistic.
User Persona
{persona_text}
Life Skeleton to Evaluate
{life_skeleton_json}
What to evaluate
Focus on these criteria:
1) persona_alignment:
- Are projects/events grounded in this persona's background, constraints, habits, and goals?
2) project_event_structure:
- Are project scopes and event granularity reasonable for real life?
- Do events represent natural moments when user would open an AI agent?
3) timeline_coherence:
- Is there believable progression across projects/events?
- Do durations and ordering feel coherent over 1-2 years?
4) realism_plausibility:
- Do scenarios feel realistic rather than template-like or contrived?
- Are there major logical contradictions?
Verdict rule
- Return overall_verdict = "YES" only if all 4 criteria are mostly satisfied and no major contradiction exists.
- Return overall_verdict = "NO" if one or more criteria fail clearly.
Output format (STRICT JSON only; no markdown)

```
{
  "overall_verdict": "YES" or "NO",
  "confidence": 0.0 to 1.0,
  "criteria": {
    "persona_alignment": "YES" or "NO",
    "project_event_structure": "YES" or "NO",
    "timeline_coherence": "YES" or "NO",
    "realism_plausibility": "YES" or "NO"
  },
  "summary": "2-4 concise sentences",
  "major_issues": [
    "issue 1",
    "issue 2"
  ]
}
```

The annotation interface provided to the human expert is shown in Figure 9.

D.3 Stage 3: Dialogue Quality

Dialogues in our pipeline are stored at the session level, with each session corresponding to a single event in the life skeleton. Accordingly, we evaluate two aspects: whether the dialogue is consistent with the project and event context defined in the skeleton, and whether the reference memories defined for that event surface naturally during the user–agent interaction. We randomly sample 100 sessions across users for evaluation. The prompt provided to `Claude Opus 4.6` is as follows.

Dialogue Quality Judge Prompt

You are a strict evaluator of synthetic user-agent dialogue quality.
You will evaluate one conversation generated from a life skeleton session.
Session Context (ground truth)

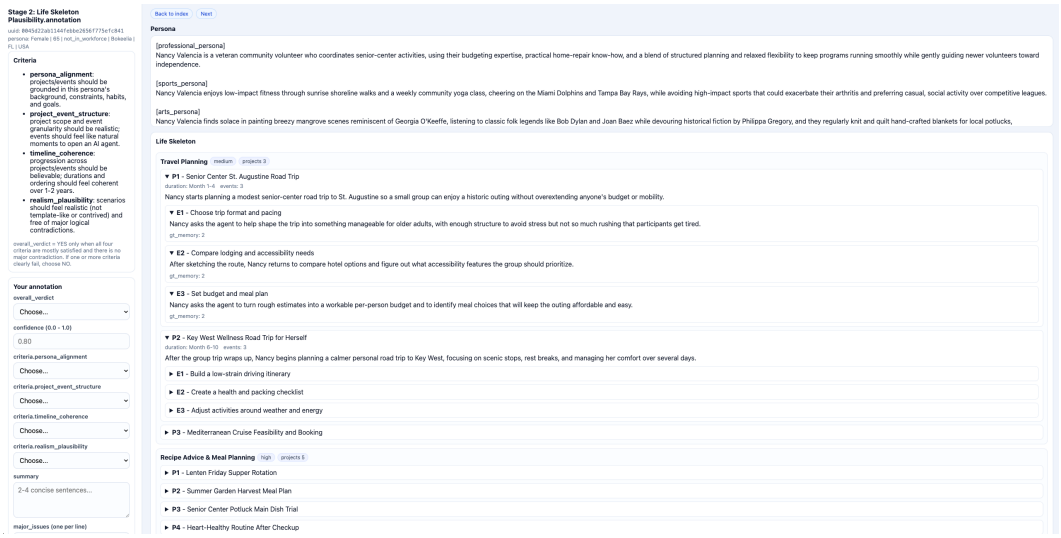

Figure 9: The user interface provided to the annotators for life skeleton and timeline realism annotation.

```
{session_context}
## Conversation
{dialogue_text}
## Evaluation criteria
1) skeleton_alignment (1-5):
- Does the dialogue align with the session's project/event context?
- Is the user request trajectory consistent with event_description?
2) gt_memory_revelation (1-5):
- Do relevant gt_memory facts surface naturally in conversation?
- Avoid penalizing if only truly irrelevant facts are not surfaced.
- Penalize forced, list-like, or unnatural fact dumping.
## Additional rule
- overall_verdict must be "YES" only if all three scores are at least 4 and there is no critical issue.
- Otherwise overall_verdict must be "NO".
## Output format (STRICT JSON only; no markdown)
{

  "overall_verdict": "YES" or "NO",
  "confidence": 0.0 to 1.0,
  "scores": {
    "skeleton_alignment": 1 to 5,
    "gt_memory_revelation": 1 to 5
  },
  "fact_checks": [
    {
      "fact": "original fact text",
      "revealed": "YES" or "PARTIAL" or "NO",
      "evidence": "short quote or reason"
    }
  ],
  "summary": "2-4 concise sentences",
  "major_issues": [
    "issue 1",
    "issue 2"
  ]
}
```

The annotation interface provided to the human expert is shown in Figure 10.

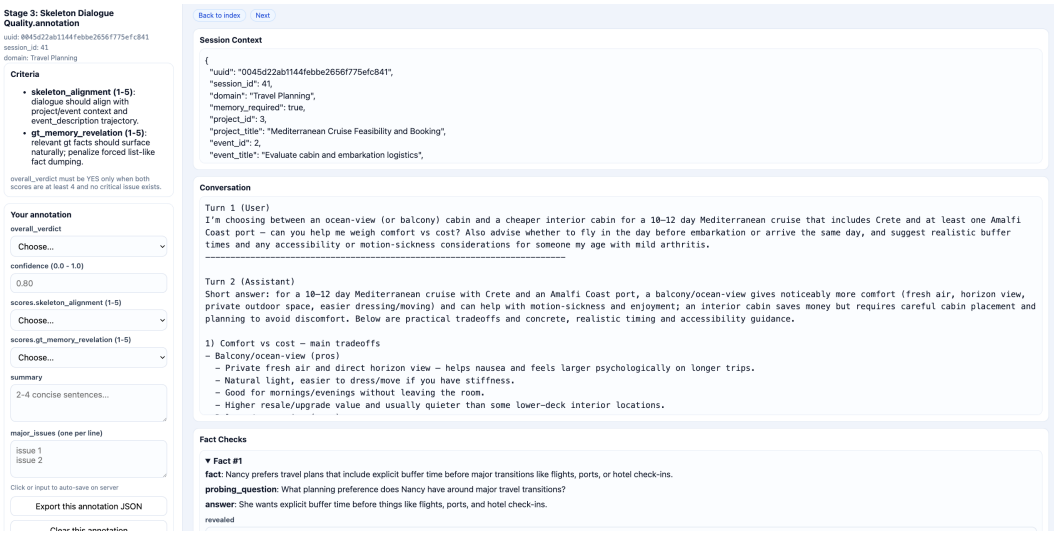

Figure 10: The user interface provided to the annotators for dialogue quality.

E Evaluation Protocol Details

E.1 LLM-as-a-Judge for $\mathbb{I}(r \in \mathcal{M}_t)$

To determine whether reference memory r is present in memory bank $\mathcal{M}_t$, we use a two-step procedure. First, we retrieve the top-10 entries from $\mathcal{M}_t$ by cosine similarity using r as the query (embedding model: `all-MiniLM-L6-v2`). Second, we pass these candidates to an LLM judge (`gpt-5-nano`) with the following prompt, which performs a binary classification on whether the core meaning of r is preserved.

$\mathbb{I}(r \in \mathcal{M}_t)$ Judge Prompt

You are checking whether a specific piece of information is contained in a given text.
Fact to find: "{fact}"
Text to search: {retrieved_entries}
Does the text above contain the core meaning of this fact? Answer YES if the fact is clearly expressed (even if worded differently). Answer NO if the fact is absent or cannot be inferred from the text.
Reply with only YES or NO.

E.2 Checkpoint Sampling Approximation

Computing $\mathbb{I}(r \in \mathcal{M}_t)$ at every session across all reference memories is prohibitively expensive: with up to 100 sessions per user and 50 reference memories per user, a full evaluation would require thousands of LLM calls per user per memory system. We therefore approximate the inner summation in Equation (1) using uniform checkpoint sampling.

Specifically, for each reference memory r, let $S(r) = [t_{\text{start}}(r),\, t_{\text{start}}(r)+1,\, \ldots,\, T_{\text{target}}(r)]$ be the full set of evaluation sessions. We sample $K{=}20$ checkpoints $\hat{S}(r) \subset S(r)$ by selecting evenly spaced indices:

$$\hat{s}_i = S(r)\left[\,\text{round}\left(\frac{i \cdot (|S(r)|-1)}{K-1}\right)\right], \quad i = 0, 1, \ldots, K{-}1, \tag{3}$$

which guarantees that both the first session $t_{\text{start}}(r)$ and the last session $T_{\text{target}}(r)$ are always included. Including the endpoints is important because the first session verifies that the fact was stored at all, and the last session verifies that it survived until the end of its required lifespan.

The full inner sum is then approximated by reweighting the sampled scores:

$$\sum_{t=t_{\text{start}}(r)}^{T_{\text{target}}(r)} \mathbb{I}(r \in \mathcal{M}_t) \;\approx\; \frac{|S(r)|}{K} \sum_{t \in \hat{S}(r)} \mathbb{I}(r \in \mathcal{M}_t), \tag{4}$$

where $\frac{|S(r)|}{K}$ is the rescaling factor under the assumption that each sampled checkpoint is representative of equally-sized intervals of the full evaluation window. Substituting into Equation (1) yields the approximated RR used in all experiments.

F Implementation Details

F.1 LLM Decoding

To ensure reproducible and deterministic evaluation, we set temperature to 0 for all components: the memory systems under evaluation, the personalization methods, and the LLM-based data generation pipeline. For `Qwen3-14B`, inference is served via vLLM on A6000 48GB GPU. For proprietary models, we use their api services.

F.2 Personalization Method

All three inference-based methods share a common interface: at each session, an LLM predicts `memory_required` $\in \{\text{true}, \text{false}\}$, and sessions predicted as transient have their memory operations skipped. The methods differ only in what historical context is provided to the LLM.

Greedy. At each session, the LLM receives only the current session's dialogue (truncated to `max_chars` characters) and outputs a binary prediction with no access to prior sessions.

Greedy Prediction Prompt

You are a memory policy agent. Decide whether this session should be stored in long-term memory.
`## Current Session Dialogue`
{current_dialogue}
memory_required = true: long-horizon session — part of an ongoing project or recurring goal.
memory_required = false: transient session — standalone and self-contained.
Respond ONLY with: `{"memory_required": <true|false>}`

Context-aware. Before prediction, the current session is summarized in 1–2 sentences and appended to a running summary buffer. At each session, the LLM receives the current dialogue alongside the most recent K session summaries as historical context.

Context-aware Prediction Prompt

You are a memory policy agent. Decide whether this session should be stored in long-term memory.
`## Recent Session Summaries (Past` K `Sessions)`
{history_summaries}
`## Current Session Dialogue`
{current_dialogue}
Respond ONLY with: `{"memory_required": <true|false>}`

Structure-aware. Each session is first parsed into a lightweight record $\{\texttt{purpose}, \texttt{summary}, \texttt{topic}\}$ via an LLM extraction call. Every K sessions, these records are passed to a second LLM call that updates the structural note — clustering sessions into inferred projects or marking them as isolated. The note persists across windows (never reset), enabling retroactive reassignment of previously isolated sessions when new evidence connects them. Sessions not yet assigned by any completed window default to `memory_required = true` conservatively.

Session Record Extraction Prompt

Analyze this conversation session and extract a structured summary.
`## Session Dialogue`
{dialogue}
Respond ONLY with: `{"purpose": "...", "summary": "...", "topic": "..."}`

Structural Note Update Prompt

You are managing a structural note that tracks a user's ongoing projects across AI assistant sessions.
`## Current Structural Note`
{current_note}
`## New Session Records (session_id {start}∼{end})`
{session_records}
Rules:
1. Group sessions belonging to the same ongoing project.
2. Self-contained one-off sessions go in `isolated_sessions`.
3. Previously isolated sessions MAY be reassigned to a project if new evidence connects them.
4. Every session_id must appear in exactly one place.
Respond ONLY with the updated note:

```
{
  "projects": [
    {"project_id": "P1", "label": "...",
     "core_topic": "...", "session_ids": [...],
     "status": "ongoing"|"completed"}
  ],
  "isolated_sessions": [...]
}
```

G Further Analyses

G.1 Sensitivity Analysis of K in Estimating Memory Retention Rate

We conduct a sensitivity analysis by varying the number of checkpoints $K \in \{10, 20, 40\}$ and comparing against the full (unsampled) computation on PerMem-Bench. As shown in Table 4, the absolute difference in RR between K=20 and Full is negligible, indicating that our approximation incurs almost no loss in accuracy while substantially reducing computational cost. As expected, reducing the checkpoint count further (K=10) introduces somewhat more instability, while increasing it (K=40) yields results comparable to our default (K=20). This suggests that K=20 offers a reasonable balance between computational cost and estimation accuracy, noting that practitioners with different resource constraints may adjust K accordingly based on this trade-off.

Table 4: Sensitivity analysis of retention rate (RR) across different numbers of evaluation checkpoints (K). The default setting ($K = 20$) is highlighted in bold.

Backbone	**Setting**	$K = 10$	$K = 20$	$K = 40$	**Full**
Mem0	Base	0.5343	**0.5233**	0.5272	0.5258
	Oracle	0.6257	**0.6183**	0.6265	0.6220
Memory-R1	Base	0.5733	**0.5600**	0.5651	0.5594
	Oracle	0.6027	**0.6063**	0.6056	0.6033

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction clearly state the paper's core contributions, including its motivation, dataset construction, and the empirical study.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: See the Limitations and Future works section.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

Justification:

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We provide our source code in the anonymous github repository and detailed implementation details in Appendix.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: We provide our source code including data and running code in the anonymous github repository.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: We provide our source code in the anonymous github repository and detailed implementation details in Appendix.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [N/A]

   Justification: The cost of LLM API calls is prohibitively high for multiple runs.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).

- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Please refer to the implementation detail section.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: To the best of our knowledge, we do not violate the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: Please refer to the Societal Impacts and Risks section.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification:

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We properly cite and state the original papers and resources.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: We provide the proper documentation in Section 3 and 4.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [Yes]

    Justification: See the Meta Evaluation section.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification:

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification:

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.