BELIEF MEMORY: AGENT MEMORY UNDER PARTIAL OBSERVABILITY

Anonymous authors
Paper under double-blind review

ABSTRACT

LLM agents that operate over long context depend on external memory to accumulate knowledge over time. However, existing methods typically store each observation as a single deterministic conclusion (e.g., inferring "API X failed" from temporary errors), even though such observations are inherently partial and potentially ambiguous. By committing to one conclusion and discarding uncertainty, these methods results in *self-reinforcing error*: the agent acts on the stored conclusion, never revisits alternatives, and reinforces the conclusion over time. To address this issue, we propose *BeliefMem*, which shifts the memory paradigm from committing to a single conclusion per observation to retaining multiple candidate conclusions with their probabilities. Concretely, BeliefMem stores the candidate conclusions as separate memory entries, each carrying a probability that is updated via Noisy-OR rules as new observations arrive. At retrieval, all candidates surface together with their probabilities, keeping alternatives visible to the agent. Since each conclusion in memory retains its probability, BeliefMem preserves the uncertainty that the deterministic paradigm discards, enabling the agent to act with high confidence on well-evidenced knowledge while retaining the capacity to update its confidence when new evidence arrives. Empirical evaluations on LoCoMo and ALFWorld benchmarks show that, even with limited data, BeliefMem achieves the best average performance, remarkably outperforming well-known baselines. More broadly, such probabilistic memory produces substantial gains and explores a new direction for agent memory in partially observable environments.

1 INTRODUCTION

Large language model (LLM) agents deployed in long-horizon, multi-session tasks increasingly rely on persistent external memory to accumulate knowledge across interactions (Hu et al., 2025b; Du, 2026). *Factual memory* methods store observations about users and environments as structured entries, from natural-language memory streams (Park et al., 2023) to vector-based extracted facts (Chhikara et al., 2025). While these methods record what was observed, *self-improving memory* methods distill actionable lessons from past experience, from reflections (Shinn et al., 2023; Zhao et al., 2024) to reusable skill libraries (Zhang et al., 2026a). Despite this diversity, these methods share a common paradigm: every memory entry is stored as a single deterministic conclusion inferred from observations, and every operation over it produces an all-or-nothing outcome.

This deterministic paradigm results in errors that persist over time. Consider an agent that observes repeated API X timeouts (Figure 1): since each memory entry holds only a single categorical conclusion, the agent stores "API X failed" while the possibility of transient failure (e.g., temporary rate limiting) is permanently discarded. Self-improving methods amplify this problem by distilling experience such as "avoid API X," and even methods that update entries cannot escape, as correcting to "API X is operational" merely replaces one deterministic conclusion with another and the next transient error flips it right back. Furthermore, when such flawed conclusions conflict with user instructions (e.g., "Use API X to ..."), the agent struggles to act reliably (Hu et al., 2025a; Lam et al., 2026). We refer to this issue as *self-reinforcing error*: the agent acts on stored conclusions, generating observations that reinforce them rather than exploring alternative possibilities.

Fundamentally, these agents operate in a *partially observable Markov decision process* (POMDP): they never directly access the true state of the world but only receive partial, noisy observations

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

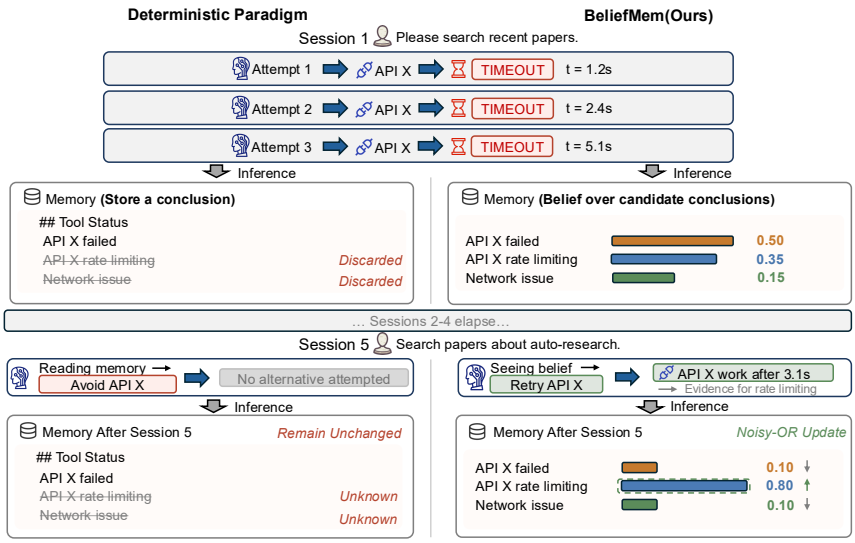

Figure 1: Deterministic memory vs. BeliefMem with an API timeout example. After repeated API X timeouts, the deterministic paradigm stores "API X failed" and avoids it in later sessions, reinforcing the error. In contrast, BeliefMem keeps multiple hypotheses (e.g., failure vs. rate limiting) with probabilities, retries the API, and updates beliefs with new evidence, enabling correction over time.

such as user messages and tool outputs (Kaelbling et al., 1998). For instance, whether API X is permanently or temporarily down is a hidden state that must be inferred from observations. Yet existing deterministic memory methods equate each observation with ground truth, leaving alternative hypotheses unrepresented and allowing self-reinforcing errors to persist across sessions (Figure 1).

To bridge this gap, we propose *BeliefMem*, which fundamentally shifts the memory paradigm from storing deterministic conclusions to maintaining an attribute-level belief representation over the environment. Specifically, BeliefMem maintains active candidate conclusions for each piece of stored knowledge, assigning each conclusion a probability updated via Noisy-OR update (Pearl, 2014) as new observations arrive. At retrieval, the candidate conclusions of each attribute surface with their probabilities, keeping competing hypotheses visible to the agent instead of reducing them to a single deterministic conclusion. This combination of belief-aware memory storage and probability-aware retrieval directly mitigates self-reinforcing error at its root: the alternative conclusions that the deterministic paradigm discards during the storage phase are now preserved and accessible to the agent. For example, in Figure 1, repeated timeouts on API X keep candidate conclusions viable alongside permanent failure. Therefore, the agent can revisit previously unfavorable actions in the future, and each new observation incrementally refines the probability assignment of each conclusion, strengthening well-supported conclusions and downweighting those with weak evidence.

To evaluate BeliefMem, we conduct experiments on both LoCoMo (Maharana et al., 2024) and ALFWorld (Shridhar et al., 2020) benchmarks, from long-term conversation to embodied agent interaction settings. Empirical evaluations show that our method achieves the best average performance on both benchmarks, outperforming existing memory methods, even with limited memory corpus size. Furthermore, ablation studies and adversarial experiments confirm the effectiveness of BeliefMem in preserving uncertainty and refining memories. More broadly, these results demonstrate that replacing deterministic memory entries with probabilistic belief representation yields promising gains, exploring a new direction for agent memory paradigm in partially observable environments.

2 Related Work

2.1 Factual and RL-Based Memory

Factual and RL-based memory methods follow the deterministic paradigm, reducing each observation's candidate conclusions to a single categorical one and discarding the alternatives. Within

this shared paradigm, early factual memory methods differ mainly in how they organize and access stored entries. Generative Agents (Park et al., 2023) maintains a natural language memory stream and retrieves memories with various signals, whereas MemGPT (Packer et al., 2023) manages memories across context, recall, and storage through virtual context management. Subsequent work further improves extraction, organization, and retrieval without changing the underlying representation: Mem0 (Chhikara et al., 2025) dynamically extracts and consolidates salient facts for vector-based retrieval, and A-MEM (Xu et al., 2025) organizes memories as structured notes with indexing and linking. Other work enriches the storage structure itself, with MemoryBank (Zhong et al., 2024) updating retrieval strength with a forgetting curve, Zep (Rasmussen et al., 2025) preserving evolving information in a temporal knowledge graph, and MemOS (Li et al., 2025) unifying heterogeneous memory blocks within a single system. Meanwhile, RL-based memory methods replace this hand-crafted memory management with learnable policies to add/update/delete entries, including Memory-R1 (Yan et al., 2025), MEM1 (Zhou et al., 2025), Agentic Memory (Yu et al., 2026), and MemRL (Zhang et al., 2026b). Across these studies, the main differences lie in storage management and retrieval strategy, not in memory representation, where each memory entry generally still records only one categorical conclusion inferred from noisy and ambiguous observations.

2.2 Self-improving Memory

Beyond recording factual observations, self-improving memory methods store actionable lessons distilled from past experience to instruct agent future actions. There are several studies that summarize raw experience into verbal lessons, such as Generative Agents (Park et al., 2023) summarizing interaction history as reflective memory, Reflexion (Shinn et al., 2023) generating self-corrective guidance from failed experiences, and ExpeL (Zhao et al., 2024) aggregating recurring patterns across trajectories into reusable insights. Beyond verbal lessons, concurrent work records feasible actions in skill libraries. Voyager (Wang et al., 2023) expands the library through an automatic curriculum as the agent explores new environments, and MemSkill (Zhang et al., 2026a) constructs a set of skills that transfer reusable knowledge across related problems. Despite shifting from factual observations to distilled experience, these methods retain the same deterministic paradigm, storing each lesson as a single categorical entry while ignoring uncertainty in observations.

2.3 Belief State under Partial Observability

In the standard POMDP, uncertainty under partial observability is represented by a belief state, a probability distribution over hidden states conditioned on the observation history (Kaelbling et al., 1998). Recent work views LLM agents as operating under partial observability and uses belief based representations for action selection and coordination (Lidayan et al., 2025; Jiang et al., 2026; Wang et al., 2025b). Additionally, Belief Engine (Yang et al.) externalizes and updates beliefs in a specific multi-agent debate setting, and empirical work shows that the mismatch between agent's beliefs and the true states of the environment can result in unreliable opinions and actions (Geng et al., 2025). However, existing memory systems still ignore the key implication of such partial observability, namely that an agent's observations provide only partial evidence about hidden states (e.g., user preference) rather than direct access to the true states. As a result, memory is represented as deterministic conclusions inferred from noisy observations, collapsing their uncertainty into a single ground truth. This motivates a memory representation that preserves such uncertainty dynamically updates as new observations arrive, instead of storing each memory entry as ground truth.

3 Methodology

3.1 Problem Formulation

POMDP (Partially Observable Markov Decision Process) setting. We consider an agent interacting with partially observable environments. At decision time t, the agent has access to an observation $o_t \in \mathcal{O}$ and selects an action $a_t \in \mathcal{A}$. Let $s_t \in \mathcal{S}$ denote the latent environment state at time t, the environment transitions according to $s_{t+1} \sim T(\cdot \mid s_t, a_t)$ (Kaelbling et al., 1998). Bayes-optimal action selection depends on the belief state, i.e., the posterior distribution over latent states induced by the interaction history. Defining $\eta_t := (o_{1:t}, a_{1:t-1})$, we write:

$$b_t(s) := \Pr(s_t = s \mid \eta_t), \qquad b_t \in \Delta(\mathcal{S}), \qquad a_t \sim \pi(\cdot \mid b_t). \tag{1}$$

Therefore, b_t is a sufficient statistic of the action-observation history for action selection.

External Memory as Belief Approximation. Existing memory methods can be viewed as approximating b_t through an external memory module M_t, which compresses task-relevant information from past interactions into a retrievable structure. At t, the agent queries M_t with the current observation o_t to obtain the memory context:

$$m_t = \mathrm{Read}(M_t, o_t), \tag{2}$$

and selects an action conditioned on both the observation and the retrieved context: $a_t \sim \pi(\cdot \mid o_t, z_t)$. After executing a_t and observing o_{t+1}, the memory is updated as:

$$M_{t+1} = \mathrm{Update}(M_t, o_t, o_{t+1}), \tag{3}$$

where Update encompasses memory writing and management operations (Wang et al., 2025a; Wang & Chen, 2025; Yan et al., 2025). In this way, M_t serves as a tractable approximation of the belief state b_t, supporting future decisions without maintaining the inaccessible full posterior.

3.2 Motivation

The Deterministic Bottleneck. However, in practice, many existing memory methods store point estimates of latent attributes relevant to the task, i.e., a deterministic conclusion of each attribute inferred from observations, thus discarding uncertainty that would be retained in a representation of complete belief b_t. Let c denote a task-relevant attribute of the latent state (e.g., user preference, tool status, or object-location relation), and let $\mathcal{H}(c) = \{h_1^{(c)}, \ldots, h_{M_c}^{(c)}\}$ denote a set of mutually exclusive and collectively exhaustive hypotheses representing the possible conclusions of c. A reliable memory would maintain, for each c, a local posterior:

$$b_t^{(c)}(h) := \Pr(s_t \in h \mid o_{1:t}, a_{1:t-1}) = \textstyle\sum_{s \in h} b_t(s), \qquad h \in \mathcal{H}(c). \tag{4}$$

However, in the deterministic memory paradigm, the write operation stores only a single conclusion $\hat{h}_t^{(c)}$ rather than the full local posterior:

$$M_t = \{(c, \hat{h}_t^{(c)}) : c \in \mathcal{C}_t\}, \qquad \hat{h}_t^{(c)} \in \mathcal{H}(c), \tag{5}$$

where $\mathcal{C}_t$ denotes all attributes preserved in M_t. Fundamentally, this corresponds to writing the most probable attribute-level hypothesis, $\hat{h}_t^{(c)} \in \mathrm{argmax}_{h \in \mathcal{H}(c)} b_t^{(c)}(h)$, while discarding the remaining alternatives and their associated probabilities. Therefore, M_t in current methods is a collection of attribute-level point estimates rather than a probabilistic approximation to the full belief state b_t, and the discarded uncertainty is no longer available for subsequent retrieval or update.

Self-Reinforcing Error. This point estimate can induce self-reinforcing error. Suppose the retrieved memory m_t in Eq. 2 exposes a stored conclusion $(c, \hat{h}_t^{(c)}) \in M_t$. The agent selects a_t conditioned on o_t and m_t, and the resulting transition (o_t, a_t, o_{t+1}) is written back to memory via Eq. 3. Since memory retains no posterior support for alternative hypotheses in $\mathcal{H}(c) \setminus \{\hat{h}_t^{(c)}\}$, the agent is unlikely to select actions that would test these alternatives. If $\hat{h}_t^{(c)}$ is incorrect or prematurely consolidated, the agent instead collects further evidence consistent with the flawed conclusion, reinforcing it over time. For example, if memory stores "API X failed," the agent becomes less likely to retry the API, thereby missing observations that could contradict the stored memory entry. Once uncertainty is collapsed to a point estimate, posterior support for discarded alternatives cannot be reconstructed from memory alone and must be re-established through entirely new evidence. This motivates a memory paradigm that retains a belief over the uncertainty rather than collapsing it to a point estimate.

3.3 Belief Memory

Belief-based Memory Formulation. To bridge this gap, we propose BeliefMem, which replaces the deterministic paradigm with an attribute-level belief representation that approximates the belief state $b_t^{(c)}$. We first introduce the ideal representation for each memory entry:

$$\big(c,\ b_t^{(c)}\big), \qquad b_t^{(c)} : \mathcal{H}(c) \to [0,1], \quad \text{s.t.} \textstyle\sum_{h \in \mathcal{H}(c)} b_t^{(c)}(h) = 1, \tag{6}$$

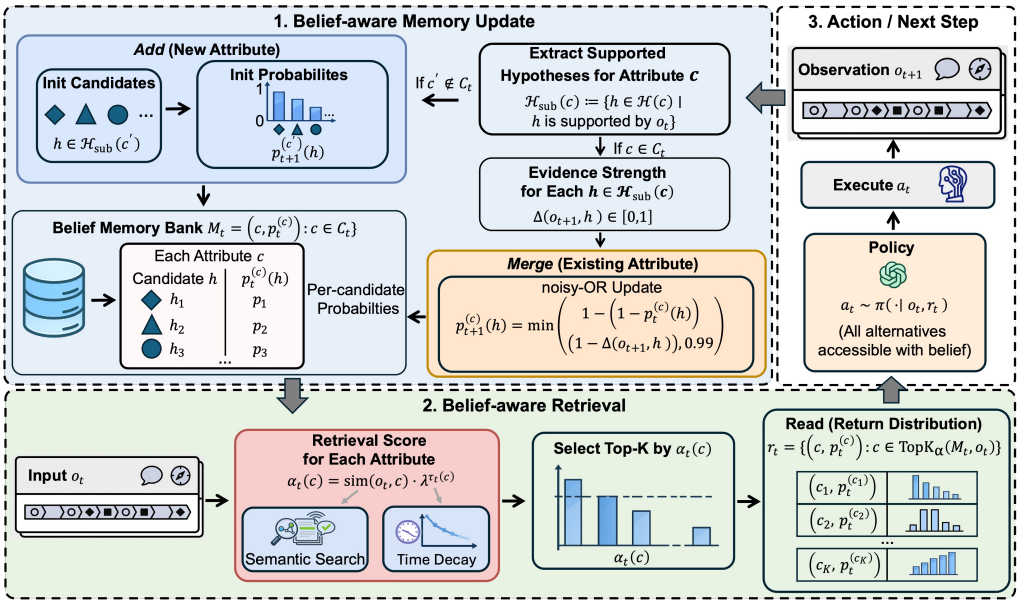

Figure 2: Overview of BeliefMem. i) Upon receiving an observation, BeliefMem updates memories via Add (initializing candidates for new attributes) or Merge (incorporating new evidence via Noisy-OR update). ii) Retrieval scores entries by semantic similarity and temporal decay, returning a full belief rather than a single conclusion. iii) The agent acts conditioned on both the current observation and the retrieved belief, keeping all alternative hypotheses visible at decision time.

where $b_t^{(c)}$ denotes the distribution of all possible conclusions for attribute c at time t. Therefore, the ideal representation of memory in BeliefMem is:

$$M_t = \{(c, b_t^{(c)}) : c \in \mathcal{C}_t\}, \tag{7}$$

which replaces the deterministic collection of point estimates in Eq. 5 with a belief state which can represent the uncertainty of each attribute in the environment.

In practice, this idealized representation is not directly feasible, because the conclusion space associated with an attribute is open-ended or dynamically expanding, making exact posterior maintenance over the full set impractical. This leads to two practical challenges: i) In open-ended settings, $\mathcal{H}(c)$ is not fixed and may expand online as new candidate conclusions are generated. A fully normalized distribution over all candidate conclusions is therefore difficult to define. ii) Even under a fixed set of possible conclusions, updating the distribution for all candidates after each new observation is computationally expensive. Therefore, classical POMDP methods rely on approximate belief representations, such as representative belief points, rather than exact update across all possible states.

Belief Update in Memory. To overcome these challenges, in this work, BeliefMem leverages two coupled ways to practically approximate Eq. 6. First, for each attribute c, BeliefMem only stores candidates that previous observations have actually evidenced, so preserved conclusions grow with evidence rather than with $|\mathcal{H}(c)|$, and unseen candidates incur no storage or update cost. Specifically, for each observation o_t, the agent identifies the supported hypotheses to form the subset $\mathcal{H}_{\text{sub}}(c) := \{h \in \mathcal{H}(c) \mid h \text{ is supported by } o_t\}$. The agent then assigns to each $h \in \mathcal{H}_{\text{sub}}(c)$ a probability $p_t^{(c)}(h) \in [0, 1]$ measuring how strongly "h is true" under o_t, and stores each resulting (attribute c, candidate h, probability $p_t^{(c)}(h)$) in memory bank. Secondly, BeliefMem keeps per-candidate probability $\{p_t^{(c)}(h)\}$ instead of a normalized joint posterior over $\mathcal{H}_{\text{sub}}(c)$ to prevent the magnitude of a supported h from being affected by the number of alternative candidates sharing the same c. Thus, rather than representing mutually exclusive posteriors, these values are stored independently as evidence-based probabilities that are updated jointly whenever new evidence for c is observed.

Building upon these principles, BeliefMem dynamically maintains M_t via the following operations:

Add works when the agent reports a new attribute $c' \notin \mathcal{C}_t$ and stores a new entry in M_t,

$$\big(c',\ h,\ p_{t+1}^{(c')}(h)\big),$$

where $h \in \mathcal{H}_{\text{sub}}(c')$ is supported by the current observation o_{t+1}.

Merge activates when the new observation supports an attribute $c \in \mathcal{C}_t$ already present in M_t. For any candidate conclusion h, if the observation provides supporting evidence, its belief is updated via Noisy-OR (Pearl, 2014) evidence merge:

$$p_{t+1}^{(c)}(h) = \min\Big(1 - \big(1 - p_t^{(c)}(h)\big)\big(1 - \Delta(o_{t+1}, h)\big),\ 0.99\Big), \tag{8}$$

where $\Delta(o_{t+1}, h) \in [0, 1]$ quantifies the strength of evidence provided by o_{t+1} for the stored conclusion h (details are provided in Appendix B.1). The upper bound of 0.99 prevents any candidate from being stored with certainty. After *Merge*, BeliefMem archives the old version $p_t^{(c)}(h)$ for later retrieval. Additionally, if the observation supports a competing candidate for the same attribute c, its probability would be reduced to the uniform value $1/|\mathcal{H}_{\text{sub}}(c)|$.

Belief-aware Retrieval. Belief update alone is insufficient if retrieval discards the uncertainty that storage has carefully preserved. To close this gap, retrieval is redefined as an operation conditioned on the stored belief rather than on a single chosen conclusion. Specifically, given an observation o_t, the retrieval score of each entry is:

$$\alpha_t(c) \ = \ \text{sim}(o_t,\ c) \cdot \lambda^{\tau_t(c)}, \qquad \lambda \in (0, 1], \tag{9}$$

where $\text{sim}(\cdot) \in \mathbb{R}_{\geq 0}$ measures the relevance of c to o_t through semantic similarity. The specific choice of sim is shown in Appendix B.3. $\lambda \in [0, 1]$ is the decay rate to control temporal importance during retrieval. $\tau_t(c) \in \mathbb{N}$ denotes the staleness of entry c (i.e., the time elapsed since its last update). It increases by one at each time step, unless the corresponding entry c is updated by *Add* or *Merge*, in which case it resets to 0. Thus, an entry's retrieval priority decays with staleness, and its probability mass remains unchanged. Read then selects the top-K entries by $\alpha_t(c)$ and returns

$$r_t \ = \ \big\{\big(c,\ p_t^{(c)}\big) : c \in \text{TopK}_\alpha(M_t, o_t)\big\}, \tag{10}$$

so that each retrieved attribute has its candidate probabilities over $\mathcal{H}_{\text{sub}}(c)$. The agent then selects an action as $a_t \sim \pi(\cdot \mid o_t, r_t)$, and every alternative conclusion in $\mathcal{H}_{\text{sub}}(c)$ is now accessible to the agent with its confidence, rather than being erased at storage time in the deterministic paradigm.

Overall, BeliefMem mitigates self-reinforcing error through two coupled principles. Specifically, it preserves memory as an approximated belief representation and returns the candidate beliefs at retrieval so that alternative hypotheses remain visible to the agent at decision time.

4 Experiments

Benchmarks. We conduct experiments on two benchmarks to evaluate long-term memory capabilities of BeliefMem in both long-term conversation and embodied agent interaction settings: i) *LoCoMo* (Maharana et al., 2024), a long-term conversational memory benchmark whose dialogues contain roughly 9,000 tokens on average and up to 35 sessions, stressing multi-session retrieval and temporal reasoning. Following this, we evaluate along four question categories: *single-hop*, which asks the model to extract a specific fact from a single session; *multi-hop*, which requires composing information scattered across multiple sessions; *temporal reasoning*, which evaluates ordering and duration of events along the dialogue timeline; and *open-domain*, which demands combining the contextual history with external commonsense knowledge. We report F1 for token-level precision and recall, and BLEU-1 for lexical overlap against ground-truth answers. ii) *ALFWorld* (Shridhar et al., 2020), a text-based embodied benchmark whose tasks cover six household goal categories. The evaluation is split into an in-distribution *Seen* set and an out-of-distribution *Unseen* set whose room layouts and object instances are held out from training, so the latter directly probes memory transfer rather than pattern memorization. We report success rate (SR), the fraction of tasks whose goal condition is satisfied within a 50-step horizon, and the average step used on solved episodes, following Zhang et al. (2026a). More details on the evaluations are provided in the Appendix C.1.

Baselines. On LoCoMo, we compare BeliefMem with six well-known memory methods: LoCoMo baseline from Maharana et al. (2024), ReadAgent (Lee et al., 2024), MemoryBank (Zhong et al.,

Table 1: LoCoMo results across four categories with GPT-4o-mini and GPT-4o. Each cell reports F1 / BLEU-1. Best and second numbers per column are in **bold** and underline, respectively.

Backbone	Method	Multi-Hop	Temporal	Open Domain	Single-Hop	Avg.
GPT-4o-mini	LoCoMo	25.02 / 19.75	18.41 / 14.77	12.04 / 11.16	40.36 / 29.05	23.96 / 18.68
	ReadAgent	9.15 / 6.48	12.60 / 8.87	5.31 / 5.12	9.67 / 7.66	9.18 / 7.03
	MemoryBank	5.00 / 4.77	9.68 / 6.99	5.56 / 5.94	6.61 / 5.16	6.71 / 5.72
	MemGPT	26.65 / 17.72	25.52 / 19.44	9.15 / 7.44	41.04 / 34.34	25.59 / 19.74
	A-MEM*	27.02 / 20.09	45.85 / 36.67	12.14 / 12.00	44.65 / 37.06	32.42 / 26.46
	A-MEM	27.08 / 20.46	29.14 / 24.08	16.60 / 13.80	40.70 / 32.63	28.38 / 22.74
	Mem0*	38.72 / 27.13	48.93 / 40.51	28.64 / 21.58	47.65 / 38.72	40.99 / 31.99
	Mem0	36.83 / 26.50	34.52 / 26.38	22.57 / 16.54	46.89 / 37.63	35.20 / 26.76
	BeliefMem (ours)	**40.51 / 32.24**	**51.88 / 45.78**	**28.73 / 25.12**	**48.41 / 42.07**	**42.38 / 36.30**
GPT-4o	LoCoMo	28.00 / 18.47	9.09 / 5.78	16.47 / 14.80	**61.56 / 54.19**	28.78 / 23.31
	ReadAgent	14.61 / 9.95	4.16 / 3.19	8.84 / 8.37	12.46 / 10.29	10.02 / 7.95
	MemoryBank	6.49 / 4.69	2.47 / 2.43	6.43 / 5.30	8.26 / 7.10	5.91 / 4.88
	MemGPT	30.36 / 22.83	17.29 / 13.18	12.24 / 11.87	60.18 / 53.35	30.02 / 25.31
	A-MEM*	32.86 / 23.76	39.41 / 31.23	17.10 / 15.84	48.43 / 42.97	34.45 / 28.45
	A-MEM	31.66 / 23.34	41.11 / 34.72	17.45 / 15.58	47.04 / 41.02	34.32 / 28.67
	Mem0	**42.57** / 30.92	44.55 / 32.60	23.04 / 17.62	48.49 / 37.00	39.66 / 29.54
	BeliefMem (ours)	41.24 / **35.68**	**50.81 / 45.05**	**27.60 / 23.78**	51.81 / 43.80	**42.87 / 37.08**

2024), MemGPT (Packer et al., 2023), A-MEM (Xu et al., 2025), and Mem0 (Chhikara et al., 2025). * denotes official reported performance. On ALFWorld, we extend these baselines with LangMem (LangChain, 2025) and MemoryOS (Kang et al., 2025), and include No-Memory that chooses actions directly from the current observation to show the contribution of memory.

Implementation Details. On LoCoMo, we use text-embedding-3-small for embedding, and utilize GPT-4o and GPT-4o-mini (Hurst et al., 2024) as base models and Qwen3-Next-80B-A3B-Instruct (Yang et al., 2025) as the base model for ALFWorld. All baselines are run with their released configurations. The hyperparameter configurations of all methods are listed in Appendix B.3.

4.1 Main Results

Effectiveness in long conversational scenarios. As demonstrated in Table 1, BeliefMem achieves the highest average performance across both base models, producing substantial improvements in multi-hop and temporal reasoning tasks. These specific tasks rigorously test an agent's ability to resolve observation conflicts and aggregate evidence over interactions. The effectiveness of BeliefMem arises from its dynamic belief update mechanism, which continuously refines and retains essential historical context while mitigating memory degradation. Furthermore, by archiving prior memory states with explicit temporal metadata, BeliefMem supports more precise retrieval of former environmental states, directly facilitating its superior temporal reasoning.

Superiority in embodied interactive scenarios. As detailed in Table 2, BeliefMem consistently outperforms all baselines across seen and unseen tasks. Specifically, BeliefMem outperforms the second-best method (ReadAgent) by 11%, and exceeds the average of the remaining baselines by 99% overall. This advantage grows to 12.4% over the second-best baseline in unseen (out-of-distribution) scenarios, demonstrating BeliefMem's robust generalizability in real agent scenario with memory. Crucially, BeliefMem achieves this superior performance using only half of the standard memory corpus. In Section 4.3, we provide a detailed analysis of this remarkable efficiency, showing that only 16.67% of the memory corpus is sufficient to outperform 5 out of 6 baselines.

4.2 Ablation Studies

We conduct comprehensive ablation studies to investigate probabilistic memory, belief-aware retrieval, and memory update operations (*Add* and *Merge*) on LoCoMo and ALFWorld (Table 3). As shown, replacing probabilistic memory with standard deterministic memory (w/o belief-based memory) results in clear performance drops in both benchmarks, highlighting the necessity of retaining uncertainty under partial observability. Furthermore, in Appendix D.3, our experiment using shuffling the probability of each candidate in BeliefMem shows notable performance decrease, further validating our probabilistic paradigm. Removing belief-aware retrieval eliminates access to mem-

Table 2: ALFWorld results with the Qwen3-Next-80B-A3B-Instruct, on the in-distribution (Seen) split and the out-of-distribution (Unseen) split. SR(%): Success rate (↑); #Steps: Average steps on solved episodes (↓). Δ indicates the difference relative to the best result in each column. BeliefMem*: 50% memory corpus used. BeliefMem: full memory corpus used.

Method	ALF-Seen				ALF-Unseen				Avg.			
	SR ↑		#Steps ↓		SR ↑		#Steps ↓		SR ↑		#Steps ↓	
	Val.	Δ	Val.	Δ	Val.	Δ	Val.	Δ	Val.	Δ	Val.	Δ
No-Memory	18.57	-45.00	42.48	+14.99	26.12	-35.07	39.35	+11.94	22.35	-37.53	40.92	+13.27
ReadAgent	53.57	-10.00	27.88	+0.39	54.48	-6.71	27.41	-	54.03	-5.85	27.65	-
MemoryBank	37.86	-25.71	35.15	+7.66	38.06	-23.13	34.99	+7.58	37.96	-21.92	35.07	+7.42
A-MEM	25.00	-38.57	40.28	+12.79	29.10	-32.09	39.04	+11.63	27.10	-32.78	39.66	+12.01
Mem0	38.57	-25.00	33.64	+6.15	41.04	-20.15	33.16	+5.75	39.81	-20.07	33.40	+5.75
LangMem	37.14	-26.43	34.42	+6.93	31.34	-29.85	37.17	+9.76	34.24	-25.64	35.80	+8.15
MemoryOS	19.29	-44.28	42.43	+14.94	18.66	-42.53	42.95	+15.54	18.98	-40.90	42.69	+15.04
BeliefMem* (ours)	58.57	-5.00	29.77	+2.28	**61.19**	-	29.34	+1.93	**59.88**	-	29.56	+1.91
BeliefMem (ours)	**63.57**	-	27.49	-	53.75	-7.44	30.49	+3.08	58.66	-1.22	28.99	+1.34

ory uncertainty, forcing the agent to discard candidate probabilities at retrieval and consequently degrading performance on both benchmarks. Furthermore, ablating the update mechanisms undermines BeliefMem's capabilities: removing the *Add* operation prevents the incorporation of new attributes of latent states into the memory bank, while removing *Merge* disables the probability updates over evidence for existing attributes. Without them, the memory bank of BeliefMem remains static, notably decreasing the performance arising from dynamic memory updates. Overall, these results show that each part of our method is vital for achieving reliable memory under partial observability. The full results and detailed analyses are provided in Appendix D.2.

4.3 Analysis and Discussion

BeliefMem scales robustly under limited memory data. Figure 4 evaluates BeliefMem on ALFWorld with memory corpus sizes ranging from 500 to 3,000. To ensure a fair evaluation and prevent cherry-picking, these subsets are strictly extracted in the original sequential order of the original dataset. When using just 50% of the corpus, BeliefMem already outperforms all baselines, and even with 500 samples, it still surpasses 5 out of 6 baselines. Beyond this advantage, we observe a trade-off within BeliefMem: using the full memory corpus produces the best performance on seen tasks, whereas a 50% subset results in superior generalization on unseen tasks. This trade-off arises because a richer set of in-distribution (Seen tasks) memories can bias the agent toward memorizing seen trajectories, at the cost of out-of-distribution (Unseen tasks) generalizability. These analyses demonstrate that BeliefMem's probabilistic memory enables effective knowledge retention even when data is highly limited. Full results are in Appendix D.1.

Table 3: Results of ablation studies on LoCoMo (GPT-4o-mini) and ALFWorld (Qwen3-Next-80B-A3B-Instruct). w/o memory: without belief-based memory; w/o retrieval: without belief-aware retrieval.

	LoCoMo		ALFWorld	
	F1↑	BLEU↑	SR↑	#Steps ↓
w/o memory	22.58	16.92	28.71	38.87
w/o retrieval	28.50	19.02	51.77	32.57
w/o *Add*	14.48	10.14	22.58	41.53
w/o *Merge*	20.38	13.31	40.81	33.03
BeliefMem	**42.38**	**36.30**	**59.88**	**29.56**

BeliefMem achieves reliable belief convergence. To validate whether BeliefMem enables candidate probabilities to converge to the ground truth, we report the Top-1 rate, defined as the proportion of instances where the true conclusion attains the highest confidence among all candidates, compared with Mem0 and A-Mem. In Figure 4, the Top-1 rate of BeliefMem steadily increases as the gold evidence accumulates, achieving 88.46% of the cases where the true conclusion receives the highest probability. In contrast, although Mem0 and A-Mem exhibit a rapid initial increase, their coverage degrades sharply in later stages as more noisy observations arrive. This degradation fundamentally stems from their deterministic update mechanisms, which erroneously overwrite previously established correct memories with incoming noise. However, BeliefMem overcomes this vulnerability

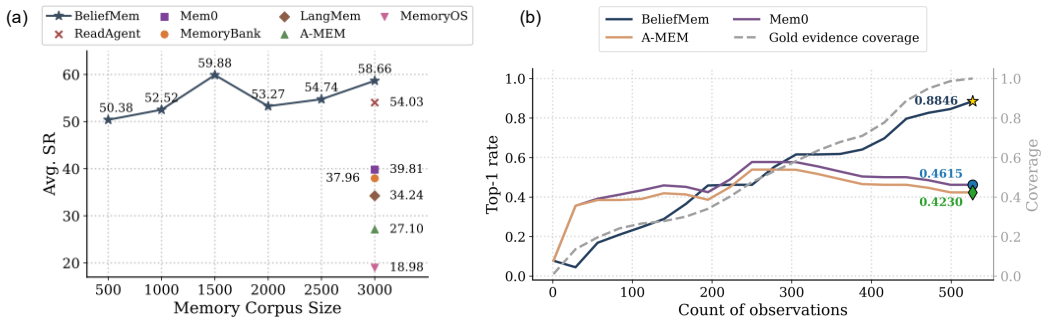

Figure 4: (a) BeliefMem maintains competitive performance across varying memory corpus sizes on ALFWorld, outperforming all baselines with only 50% of memory corpus. (b) BeliefMem's candidate probabilities reliably converge to the true conclusion as evidence accumulates on LoCoMo, whereas deterministic paradigm (Mem0, A-Mem) fails to converge under noisy observations.

through its probabilistic updating mechanism; by carefully weighing incoming noise against accumulated gold evidence, it effectively prevents the catastrophic overwriting of true memories and ensures robust, stable convergence toward the ground truth. Details are provided in Appendix C.4.

BeliefMem shows strong memory correction in adversarial settings. We conduct adversarial experiments on ALFWorld benchmark by injecting strongly flawed memory conclusions into the memory bank and observing the correction process (see Appendix C.5 for detailed pipeline). As shown in Figure 3, after updates with valid and noisy observations, BeliefMem achieves a correction rate nearly twice that of the deterministic memory baseline. Furthermore, it achieves this correction notably faster, requiring an average of only 4.75 steps. These results highlight BeliefMem's robustness and stability when handling flawed memories under noisy observations.

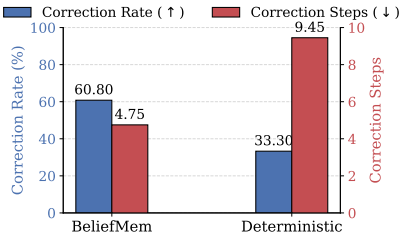

Figure 3: BeliefMem vs. deterministic memory under adversarial setting on ALFWorld dataset.

Hyperparameter analysis. We evaluate the impact of the retrieval size K and decay rate λ on BeliefMem (Table 8 in Appendix D). Performance on ALFWorld scales positively with K up to an optimal $K = 20$. Beyond this ($K = 30$), BeliefMem suffers a trade-off: although its SR on seen tasks reaches the best, the SR on unseen tasks drops by 5.22% as broader retrieval may surface noisy, in-distribution memories that hinder generalization. Additionally, variations in λ affect the performance of BeliefMem, highlighting the critical role of the decay mechanism in controlling the agent's reliance on early memory, thereby striking a crucial balance between efficient in-distribution exploitation and robust out-of-distribution generalization.

5 Conclusion

In this work, we identify a key drawback of prior memory methods in partially observable environments: their deterministic paradigm of storing categorical conclusions inferred from observations results in self-reinforcing error. To address this issue, we propose BeliefMem, which reframes memory as an approximation of the environment's belief state. Specifically, BeliefMem maintains multiple candidate conclusions with probabilities for each attribute of the evolving environment, updated via Noisy-OR evidence merge as new observations arrive. During retrieval, these probabilistic conclusions enable the agent to reason under uncertainty and select reliable actions toward task goals. Experiments on the LoCoMo and ALFWorld benchmarks show that our method outperforms well-known baselines on average across diverse scenarios. Additionally, various analyses in our work illustrate our method's promising capabilities in memory correction and data efficiency. Overall, our work introduces a novel perspective on agent memory in partially observable environments and demonstrates its empirical benefits under various settings.

AI USE STATEMENT

In this paper, we employed the commercial large language model GPT-5.6-Chat for language refinement and manuscript polishing. It was not used for generating research ideas, designing methods, or conducting a literature search and discovery.

REFERENCES

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.

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

Jiayi Geng, Howard Chen, Ryan Liu, Manoel Horta Ribeiro, Robb Willer, Graham Neubig, and Thomas L Griffiths. Accumulating context changes the beliefs of language models. *arXiv preprint arXiv:2511.01805*, 2025.

Yuanzhe Hu, Yu Wang, and Julian McAuley. Evaluating memory in llm agents via incremental multi-turn interactions. *arXiv preprint arXiv:2507.05257*, 2025a.

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*, 2025b.

Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*, 2024.

Haitao Jiang, Lin Ge, Hengrui Cai, and Rui Song. Pabu: Progress-aware belief update for efficient llm agents. *arXiv preprint arXiv:2602.09138*, 2026.

Leslie Pack Kaelbling, Michael L Littman, and Anthony R Cassandra. Planning and acting in partially observable stochastic domains. *Artificial intelligence*, 101(1-2):99–134, 1998.

Jiazheng Kang, Mingming Ji, Zhe Zhao, and Ting Bai. Memory os of ai agent. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 25972–25981, 2025.

Chingkwun Lam, Jiaxin Li, Lingfei Zhang, and Kuo Zhao. Governing evolving memory in llm agents: Risks, mechanisms, and the stability and safety governed memory (ssgm) framework. *arXiv preprint arXiv:2603.11768*, 2026.

LangChain. Langmem. https://github.com/langchain-ai/langmem, 2025. GitHub repository.

Kuang-Huei Lee, Xinyun Chen, Hiroki Furuta, John Canny, and Ian Fischer. A human-inspired reading agent with gist memory of very long contexts. *arXiv preprint arXiv:2402.09727*, 2024.

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

Aly Lidayan, Jakob Bjorner, Satvik Golechha, Kartik Goyal, and Alane Suhr. Abbel: Llm agents acting through belief bottlenecks expressed in language. *arXiv preprint arXiv:2512.20111*, 2025.

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

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

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

Judea Pearl. *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Elsevier, 2014.

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

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

Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. Alfworld: Aligning text and embodied environments for interactive learning. *arXiv preprint arXiv:2010.03768*, 2020.

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

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

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

Zhimin Wang, Shaokang He, Duo Wu, Jinghe Wang, Linjia Kang, Jing Yu, and Zhi Wang. Cobel-world: Harnessing llm reasoning to build a collaborative belief world for optimizing embodied multi-agent collaboration. *arXiv preprint arXiv:2509.21981*, 2025b.

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.

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.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Joshua C Yang, Damian Dailisan, and Maurice Flechtner. Belief engine: Bayesian memory for configurable opinion dynamics in llm agents. In *ICLR 2026 Workshop on Memory for LLM-Based Agentic Systems*.

Yi Yu, Liuyi Yao, Yuexiang Xie, Qingquan Tan, Jiaqi Feng, Yaliang Li, and Libing Wu. Agentic memory: Learning unified long-term and short-term memory management for large language model agents. *arXiv preprint arXiv:2601.01885*, 2026.

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

Shengtao Zhang, Jiaqian Wang, Ruiwen Zhou, Junwei Liao, Yuchen Feng, Zhuo Li, Yujie Zheng, Weinan Zhang, Ying Wen, Zhiyu Li, et al. Memrl: Self-evolving agents via runtime reinforcement learning on episodic memory. *arXiv preprint arXiv:2601.03192*, 2026b.

Andrew Zhao, Daniel Huang, Quentin Xu, Matthieu Lin, Yong-Jin Liu, and Gao Huang. Expel: Llm agents are experiential learners. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 19632–19642, 2024.

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

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.

A Appendix

B More Implementation Details of BeliefMem.

B.1 More Details about Memory Update

Given a new observation, the agent first extracts a set of candidate conclusions using the prompt shown in Figure 6. Each candidate is represented as a structured memory object, including its normalized conclusion, semantic slots, evidence references, temporal information, and belief scores. In practice, an attribute c is formed from stable semantic slots such as subject, predicate, entities, and qualifiers; a candidate conclusion h is the normalized conclusion text/object for that attribute. Notably, the extracted `prob` field is used as the evidence strength $\Delta(o_{t+1}, h)$ in Eq. 8, which measures how strongly the new observation support the conclusion h. We use this value as an LLM-extracted confidence, not as a calibrated posterior probability. For *Add*, we clip this extracted value to $[p_{\min}, p_{\max}]$, and the default setting is $[p_{\min}, p_{\max}] = [0.7, 0.9]$. Throughout the implementation, the stored `prob` values are confidence scores used for ranking and updating, not calibrated probabilities.

After extraction, BeliefMem updates the existing memory bank through several operations. If the candidate describes a new conclusion that is not covered by the current memory bank, BeliefMem applies *Add* and inserts it as a new memory entry. If the candidate provides compatible evidence for an existing conclusion (using keyword matching over attribute conclusions), BeliefMem uses *Merge*: the new evidence is attached to the existing memory, and the truth belief is updated with the Noisy-OR evidence aggregation in Eq. 8.

B.2 Contradictory Memory

For any candidate conclusion h, if the observation o_{t+1} provides evidence to support a contradictory conclusion, the current belief of h is reduced to 0.25, called *Version*. And the previous value is retained as a historical version. Specifically, we use a rule-based criterion to identify contradictory conclusions: Formally, let (c, h) denote an existing memory conclusion of attribute c and (c, h') denote a newly extracted candidate from o_{t+1} via the operation in Appendix B.1. When $h \neq h'$, for same attribute c, the new candidate is treated as a contradictory conclusion for h.

B.3 Hyperparameter Configuration

The decay rate λ in Eq. 9 is set to 0.5 for LoCoMo and 0.1 for ALFWorld. For LoCoMo, we use random seed 20260413. Top-$K = 20$ for single-hop questions and top-$K = 30$ for multi-hop, temporal, and open-domain questions. For ALFWorld, we follow the official evaluation, which sets chunk size 512, query source `objective`, Contriever retrieval, memory top-$K = 20$, and a maximum of 50 environment steps. The action model is run with seed 42, temperature 0.0, top-$p = 1.0$, and a maximum generation length of 32.

$\mathrm{sim}(\cdot)$ in Eq. 9 employs a hybrid design. Specifically, it is computed as a linear combination of embedding cosine similarity and lexical overlap (both attribute and evidence), with weights of 0.7 and 0.3, respectively, across all tasks. In addition, to reduce cost and latency in BeliefMem, we set the maximum number of candidate conclusions per attribute to 4 during retrieval.

B.4 Memory Costs

BeliefMem stores one candidate set for each active attribute and optionally preserves historical versions after *Merge*. If attribute c has M_c active candidates and v_c retained versions, memory storage is $O(\sum_c M_c v_c)$ textual entries plus embeddings. Updating an observed attribute costs $O(M_c)$ after LLM extraction, since only candidates under the matched attribute are updated. Retrieval first scores attributes by semantic similarity and decay, then serializes the top-K attributes and their active candidates, so token cost grows with the number of retrieved candidates rather than only with K.

To reduce this cost, we cap the number of retrieved candidates per attribute. Figure 5 reports the average token consumption per generation on LoCoMo. Our method uses fewer tokens than the competitive baselines, confirming that the strategy effectively limits overhead.

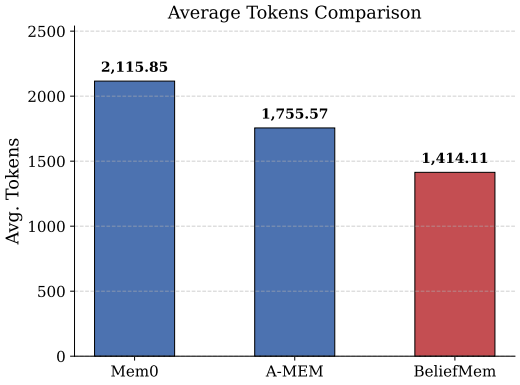

Figure 5: Average token consumption of BeliefMem and competitive baselines on LoCoMo using GPT-4o-mini for each generation.

B.5 Hardware and Software

All base models and benchmarks used in this work are publicly accessible. All experiments were conducted using NVIDIA A800-80GB GPUs with Python 3.11 and PyTorch 2.4.1.

C Further Experiment Setup

C.1 ALFWorld Evaluation Details

Evaluation split. For all methods in Section 4.1, we conduct experiments on the official ALFWorld (Shridhar et al., 2020), evaluating the full 140 episodes of the in-distribution *Seen* set and the full 134 episodes of the out-of-distribution *Unseen* set. Both splits cover the six household goal templates (Pick & Place, Examine in Light, Clean & Place, Heat & Place, Cool & Place, and Pick Two & Place), and every episode runs under the standard 50-step environment horizon.

Memory bank construction and retrieval. We follow the ALFWorld pipeline of Zhang et al. (2026a) for all memory methods in this paper. Expert trajectories are collected from the official training split, where each trajectory records the full sequence of observations, actions, and outcomes produced by the demonstrating agent, and are then grouped by task type. For every task type, a random subset of training trajectories is sampled as the experience corpus used for memory construction. Evaluation uses the official Seen/Unseen episodes described above, so no evaluation trace is used during memory construction. We fix the total bank size at 3,000 expert trajectories, distributed across the six task types unless otherwise stated, such as BeliefMem* and the data-size analysis. Specifically, the memory bank is constructed once before evaluation begins, with each trajectory written through the native write operation of each baseline method. At test time, every method, including BeliefMem and all baselines, retrieves up to 20 memory items per observation (Top-$K = 20$). The sampled corpus and evaluation episodes are kept identical across methods.

C.2 LoCoMo Evaluation Details

For LoCoMo, we follow the official setting in Maharana et al. (2024). All baseline methods are reproduced with their open-source settings described in their papers. Additionally, we also present the performance of Mem0 and A-MEM reported in their papers in Section 4.1 for clarification. They are denoted with *.

C.3 Preserving Historical Beliefs

Instead of overwriting an existing belief p_t during *Merge*, BeliefMem retains $p_t^{(c)}$ as an independent historical entry alongside the newly updated current version p_{t+1}. This update mechanism is essential for handling temporal queries. While recent observations naturally update the agent's current belief of the environment, queries targeting specific past contexts necessitate access to historical states.

This temporal awareness is achieved through timestamp management. The updated entry p_{t+1} receives the latest timestamp, while the old entry p_t retains its original timestamp. Following the decay mechanism in Eq. 9, the current version is naturally prioritized during default retrieval due to its recency. Simultaneously, the historical version remains fully accessible for queries explicitly referencing earlier time steps, ensuring comprehensive temporal grounding without information loss.

C.4 Belief Coverage Analysis Experiment Setup

To ensure reproducibility and clarity, we detail the setup of belief convergence in Section 4.3. We conduct our evaluation on the multi-hop task of LoCoMo benchmark, maintaining all hyperparameters at their defaults. Specifically, we choose the gold-standard answers of 211 selected samples that can be mapped to a single attribute-level conclusion as the target true states. The observations are sampled from the memory corpus associated with these questions, ensuring they contain the necessary evidence to support these ground-truth conclusions. Through this rigorous configuration, we effectively validate the capacity of BeliefMem to make the memory belief converge toward the true conclusion.

Top-1 rate of Mem0 and A-MEM. Since Mem0 and A-MEM rely on a single-conclusion memory storage rather than a probabilistic mechanism, they update their entire memory deterministically. Therefore, to ensure a fair comparison, we adapt the "Top-1 Rate" metric for Mem0 and A-MEM: it measures the fraction of gold-evidence items that are successfully preserved in the memory without semantic conflicts, out of all gold-evidence items. This setup objectively evaluates how much gold evidence is effectively covered by Mem0 and A-MEM.

C.5 Detailed Pipeline for Adversarial Memory Correction

To provide further clarity on the adversarial experiments discussed in Section 4.3, this section details the complete experimental pipeline, including adversarial sample construction, update procedures, and evaluation metrics.

Experimental Setup. The correction process is evaluated through the following steps:

- **Flawed Memory Injection:** We scan the BeliefMem memory bank evaluated on the ALFWorld benchmark to identify strongly flawed conclusions. A memory entry is selected as an adversarial sample if it meets three criteria: (1) it contradicts the optimal action, (2) it is highly ranked (retrieved in the Top-K), and (3) the correct conclusion is entirely excluded from the Top-K. This strict filtering yields 102 adversarial samples.
- **Observation Generation:** For each sample, we construct a sequence of observations to simulate the update process. We generate 5 *valid observations* based on the correct actions, providing sparse, ground-truth hints. Simultaneously, we construct 5 *noisy observations* derived from incorrect candidate conclusions (strictly excluding the correct one) to serve as adversarial perturbations during the memory update phase.
- **Update Protocol:** BeliefMem is updated using the default settings specified in Appendix B.3. As a baseline, the deterministic memory method only stores and updates a single conclusion per sample, as autonomously determined by the agent. Additionally, we run 10 update steps, where each step randomly includes one of the valid or the noisy observation.

Evaluation Metrics. We assess memory correction performance using two primary metrics:

Table 4: The performance of BeliefMem on the ALFWorld dataset with varying corpus sizes.

Corpus Size	ALF-Seen		ALF-Unseen		Avg.
	SR ↑	#Steps ↓	SR ↑	#Steps ↓	SR ↑
500	50.00	32.51	50.75	33.80	50.38
1000	54.29	32.78	50.75	32.62	52.52
1500	58.57	29.77	**61.19**	**29.34**	**59.88**
2000	54.29	31.09	52.24	31.56	53.27
2500	55.00	30.60	54.48	31.81	54.74
3000	**63.57**	**27.49**	53.75	30.49	58.66

- **Correction Rate:** The proportion of samples where the correct conclusion successfully outranks the injected flawed conclusion during retrieval after the update process.
- **Correction Steps:** The average number of update steps required for the correct conclusion to achieve a stably higher retrieval ranking than the flawed conclusion.

D Further Empirical Results

D.1 Full results of BeliefMem on ALFWorld with different memory corpus size.

In this section, we provide the full results of Figure 4, as shown in Table 4. As detailed, we observe a generalization trade-off related to memory corpus size. Specifically, BeliefMem achieves its highest out-of-distribution (ALF-Unseen) success rate of 61.19% and optimal average performance of 59.88% using only 1,500 samples, representing exactly 50% of the sampled memory corpus. Additionally, the agent also exhibits maximum behavioral efficiency in novel environments, requiring a minimum of only 29.34 steps to complete tasks.

Conversely, scaling the memory corpus to the full 3,000 samples maximizes in-distribution (ALF-Seen) performance, reaching a peak success rate of 63.57% with the fewest interaction steps (27.49). However, this data increase results in a sharp 7.44% decline in unseen success rates compared to the 1,500-sample performance. This divergence suggests that, in this setting, excessive environment-specific data may induce corpus size overfitting, biasing the agent toward memorizing seen trajectories at the expense of generalizability. We treat this explanation as plausible rather than conclusive, since no additional controlled test of this hypothesis is performed. Furthermore, BeliefMem demonstrates exceptional low-data robustness; with merely 500 samples (16.67% of the data), it maintains an average success rate of 50.38%. Overall, these results show that BeliefMem efficiently distills actionable, generalizable memories from highly limited interactions, whereas simply increasing the memory corpus does not monotonically improve its robust environmental understanding.

D.2 Full Results of Ablation Studies

The complete results of the ablation on ALFWorld and LoCoMo are provided in Tables 5 and 6, respectively.

w/o belief-based memory. In this setting, we collapse the probabilistic memory to a deterministic one. Specifically, for each attribute, only the single most likely conclusion is kept and retrievable. As shown, success rate on ALFWorld drops from 59.88 to 28.71, and average F1 on LoCoMo falls from 42.38 to 22.58. Without belief representation over conclusions, the agent acts on overconfident, often incorrect, memories and loses the ability to reason under partial observability.

w/o belief-aware retrieval. All candidate conclusions for an attribute are still stored, but their probabilities are discarded during retrieval, making them appear equally likely. As observed, the performance drop is more moderate, where ALFWorld success rate declines to 51.77 (a drop of 8.11 absolute SR points compared to full BeliefMem) and LoCoMo average F1 decreases to 28.50.

Table 5: Full results of ablation studies on ALFWorld benchmark (memory corpus size = 1500).

Method	ALF-Seen (SR / #Steps↓)	ALF-Unseen (SR / #Steps↓)	Average (SR / #Steps↓)
w/o belief-based memory	27.65 / 39.44	29.76 / 38.29	28.71 / 38.87
w/o belief-aware retrieval	54.29 / 31.59	49.25 / 33.54	51.77 / 32.57
w/o *Add*	19.87 / 43.24	25.29 / 39.82	22.58 / 41.53
w/o *Merge*	39.29 / 32.38	42.32 / 33.67	40.81 / 33.03
BeliefMem (Ours)	**58.57 / 29.77**	**61.19 / 29.34**	**59.88 / 29.56**

Table 6: Full results of ablation studies on LoCoMo benchmark using GPT-4o-mini.

Method	Multi-Hop (F1 / BLEU)	Temporal (F1 / BLEU)	Open Domain (F1 / BLEU)	Single Hop (F1 / BLEU)	AVG. (F1 / BLEU)
w/o belief-based memory	22.45 / 16.57	16.82 / 12.34	10.11 / 8.76	40.93 / 30.02	22.58 / 16.92
w/o belief-aware retrieval	27.12 / 18.98	25.67 / 15.43	15.89 / 10.12	45.31 / 31.55	28.50 / 19.02
w/o *Add*	14.56 / 10.21	9.14 / 6.88	8.73 / 6.54	25.48 / 16.91	14.48 / 10.14
w/o *Merge*	16.84 / 12.63	17.39 / 10.17	11.05 / 7.99	36.22 / 22.46	20.38 / 13.31
BeliefMem (Ours)	**40.51 / 32.24**	**51.88 / 45.78**	**28.73 / 25.12**	**48.41 / 42.07**	**42.38 / 36.30**

This indicates that merely retaining multiple hypotheses already preserves a useful degree of uncertainty. However, these results change on the more challenging LoCoMo sub-tasks: on multi-hop and open-domain questions, F1 decreases from 40.51 to 27.12 and from 28.73 to 15.89, respectively. In these settings, the agent must overcome conflicting evidence, and without probabilities it is unable to judge between competing claims, leading to ambiguous retrieval.

w/o *Add*. When *Add* is entirely removed, no new memory is inferred from observations. This destroys the dynamic memory update in our method. As a result, the ALFWorld success rate collapses to 22.58%, and LoCoMo F1 drops to 14.48%, showing that correct attribution of new evidence is a crucial condition for memory to remain organized and usable.

w/o *Merge*. Removing *Merge* means that every new observation creates a separate attribute entry rather than updating an existing one with accumulated evidence. Consequently, probabilities are never refined by subsequent observations and remain frozen at their initial values. ALFWorld success rate falls to 40.81% and LoCoMo F1 drops to 20.38%, as the memory stays static and cannot integrate sequential information.

D.3 Effectiveness of probability presented in memory

To empirically validate that the LLM leverages these numerical signals in retrieved memory, we conducted an additional experiment by shuffling the probabilities assigned to the top three hypotheses of each attribute. As shown in Table 7, shuffling these probability distributions causes substantial performance degradation across all tasks. This shows that the agent actively relies on the accurate probabilities provided in the context to guide its actions.

Our ablation studies further support this conclusion in Table 3. BeliefMem without retrieval essentially treats all hypotheses as having a uniform probability distribution. The notable performance decline observed in Table 3 (from 42.38 to 28.50 on LoCoMo) confirms the effectiveness of incorporating explicit probabilistic distributions into the memory paradigm.

In summary, these results demonstrate that the agent successfully perceives the uncertainty presented in numerical probabilities and adapts its decision making accordingly. This design enables the system to handle POMDP settings effectively and yields a more reliable memory system.

Table 7: Performance of BeliefMem with shuffling probabilities and the original on LoCoMo.

Method	Multi-hop	Temporal	Open-domain	Single-hop
Shuffling Probabilities	29.72 / 24.10	31.22 / 27.61	17.93 / 14.81	32.17 / 25.12
BeliefMem (ours)	**40.51 / 32.24**	**51.88 / 45.78**	**28.73 / 25.12**	**48.41 / 42.07**

D.4 Results of hyperparameter analysis

Top-K analysis. Table 8 presents a sensitivity analysis of the retrieval size K on ALFWorld, evaluating its impact on both task success rate (SR) and interaction efficiency (Steps). The empirical results demonstrate a clear non-linear relationship between memory retrieval scale and the agent's generalization capabilities, identifying $K = 20$ as the optimal threshold for BeliefMem. Specifically, at $K = 20$, the model achieves the best generalization and overall efficiency, achieving an average SR of 59.88% while minimizing the average step to 29.55 steps. When the retrieval size is overly restricted ($K \leq 10$), the agent exhibits degraded performance across all metrics. This indicates that an insufficient K fails to retrieve adequate contextual memory. Conversely, expanding the retrieval size to $K = 30$ exposes a generalization trade-off. While a larger memory context maximizes the SR on in-distribution tasks (Seen SR peaks at 61.43%), it severely compromises out-of-distribution reasoning. The Unseen SR experiences a sharp 8.5% absolute degradation (from 61.19% down to 55.97%), with a corresponding degradation in execution efficiency (Unseen steps increase to 32.43). This structural divergence suggests a trade-off: excessive retrieval surfaces redundant, task-specific noisy memories from seen environments. Rather than augmenting the belief state, these extraneous in-distribution memories may act as noise, impairing the agent's generalization in unseen environments.

Decay rate λ. To investigate the impact of the hyperparameter decay rate λ on both task efficacy and generalization, we conduct a comprehensive sensitivity analysis on ALFWorld. As illustrated in Table 8, removing this term (w/o decay) yields the highest in-distribution SR (63.57%) but results in the poorest out-of-distribution performance (55.97% unseen SR) with the longest unseen trajectory length, clearly indicating strong reliance on earlier memories from seen environments. However, $\lambda = 0.1$ drastically shifts this dynamic, achieving the best unseen success rate (61.19%) while sacrificing seen performance. As λ is incrementally increased towards 0.9, BeliefMem exhibits a steady recovery in seen environments while maintaining robust unseen generalization, achieving one of the highest average success rates. Furthermore, the step metrics reveal that higher values of λ (≥ 0.9) consistently induce more efficient decision-making. Specifically, the agent uses fewer steps, driving the average trajectory length down to its minimum of 29.00 steps at $\lambda = 1.0$, as the agent can retrieve more related early memories. Consequently, the choice of λ explicitly dictates the agent's reliance on its historical memory. Increasing λ encourages the recall of past experiences to achieve optimal in-distribution execution, whereas decreasing λ prevents the model from being constrained by prior patterns, strictly benefiting out-of-distribution performance.

E Limitations and Future Work

While BeliefMem successfully shifts the memory paradigm from storing deterministic conclusions to maintaining a belief representation of the underlying true states, achieving promising performance across diverse scenarios, several limitations remain to be addressed in future work:

- **Lack of theoretical guarantees for belief approximation.** BeliefMem maintains the probability of each candidate conclusion via noisy-OR evidence aggregation rather than a complete normalized posterior distribution, because exact belief maintenance over an open-ended hypothesis space is computationally infeasible. Although this approximation provides no formal convergence guarantees, the experimental results in Figure 4 show that as evidence accumulates the candidate probabilities reliably converge toward the true conclusion, demonstrating that the approximation is effective in practice.
- **LLM-extracted evidence strength.** The evidence strength Δ used in the noisy-OR update is extracted by LLMs instead of being derived from a calibrated observation likelihood model. This can introduce noise when the model's confidence estimates are inaccurate.

Table 8: Sensitivity analysis of hyperparameters (Top-K and λ) on ALFWorld.

Hyperparameter	ALF-Seen		ALF-Unseen		Avg.	
	SR↑	#Steps↓	SR↑	#Steps↓	SR↑	Step↓
Sensitivity to Top-K						
5	50.71	33.37	53.73	32.18	52.22	32.78
10	50.00	32.60	52.24	32.60	51.12	32.60
20	58.57	29.77	61.19	29.34	59.88	29.55
30	61.43	29.45	55.97	32.43	58.70	30.94
Sensitivity to λ						
w/o decay	63.57	28.42	55.97	31.72	59.77	30.07
0.1	58.57	29.77	61.19	29.34	59.88	29.55
0.3	59.29	28.84	58.21	29.95	58.75	29.40
0.5	59.29	28.90	58.96	30.01	59.13	29.46
0.7	60.00	29.06	58.21	30.13	59.10	29.60
0.9	62.86	27.60	58.21	30.72	60.53	29.16
1.0	62.14	27.81	58.96	30.19	60.55	29.00

However, the adversarial correction experiments in Section 4.3 indicate that BeliefMem is robust to noisy observations, achieving a correction rate for flawed memory entries that is nearly twice that of the deterministic baseline.

- **Computational overhead.** Although we leverage an approximated belief representation, the computational cost remains non-trivial compared to standard deterministic memory baselines, especially during memory writing and merging. Given that our method uses fewer tokens than competitive baselines (Table B.4), exploring more cost-effective architectures for maintaining and updating belief-based memory represents a promising direction for future work.

```
You are extracting structured long-term memory from one interaction
    session.

Return JSON only. Do not include markdown fences.
Return a JSON object with a single key "memories" whose value is a
    list of memory objects.

Return between 14 and 20 memory objects whenever the session contains
    enough facts.
Prioritize atomic memories that help answer future QA questions.

Each memory object must contain:
- type: one of observation, event, profile, anchor, episode
- canonical_text
- subject
- predicate
- object
- participants
- entities
- qualifiers
- dialog_ids
- time_text
- relative_time
- prob

Rules:
- Only use facts supported by the provided session.
- Keep each memory atomic: one memory should capture one fact, event,
    or stable profile attribute.
- Prefer profile nodes for stable attributes and preferences.
- Prefer event nodes for time-bounded actions or episodes.
- Prefer observation nodes for one-off factual statements grounded in
    1-2 turns.
- Use episode only for a very short episode label, not a long summary
    paragraph.
- Use exact dialog ids for evidence whenever possible.
- Every returned memory must cite dialog ids from the current session
    only.
- canonical_text should be concise and specific, usually one sentence
    or a short clause.
- object should be short when possible.
- time_text must be a short time phrase only, such as "last Saturday",
     "2022", or "7 May 2023". Never copy an entire sentence into
    time_text.
- If no time phrase is available, leave time_text empty.
- Do not invent unsupported details.
- Include memories for:
  - important plans, goals, identity, relationship status, jobs,
    preferences
  - concrete events, activities, meetings, races, support groups,
    volunteer work
  - key artifacts or entities that future questions may mention
- Capture proper nouns exactly when they appear, including countries,
    cities, book titles, bands, workshops, conferences, parks, museums
    , and artists.
- Capture background/profile facts even when mentioned indirectly,
    such as home country, roots, family structure, children, marriage,
     support network, books read, favorite activities, places visited,
     and recurring hobbies.
- For list-like evidence, prefer one memory per distinct item when
    that will help later QA, such as one book title, one camping
    location, one music artist, one event, or one activity per memory.
- Favor exact, evidence-grounded wording over paraphrastic summaries.

Session date: {session_date}
Conversation:
{conversation}
```