Meta-Cognitive Memory Policy Optimization for Long-Horizon LLM Agents

Ziyan Liu[1] * **Zhezheng Hao**[2] * **Yeqiu Chen**[1] * **Hong Wang**[1] **Jingren Hou**[1]
Ruiyi Ding[4] **Yongkang Yang**[1] **Wence Ji**[3] **Wei Xia**[3] † **Feng Liu**[3]

[1]University of Science and Technology of China [2]Zhejiang University
[3]WorkBuddy Team, Tencent [4]Fudan University
*Equal contribution. †Corresponding author: xwellxia@tencent.com

Abstract

Memory-augmented LLM agents tackle complex long-horizon tasks by recursively summarizing interaction trajectories into compact memory. However, existing approaches typically train these memory policies using outcome-based reinforcement learning, failing to localize where intermediate memory quality degrades. As interactions unfold, ambiguous recursive summaries progressively discard task-relevant information and introduce semantic noise. This exacerbates *belief deviation*, obscuring the agent's estimate of the latent task state and ultimately derailing long-horizon reasoning. We therefore argue that memory optimization should target not only trajectory-level success but also *belief preservation* during recursive summarization. To this end, we introduce Belief Entropy, a self-supervised proxy that probes how uncertain the model remains about the latent task state given its current memory. Based on this proxy, we propose Metacognitive Memory Policy Optimization (MMPO). Instead of relying only on sparse outcome-based signals, MMPO provides fine-grained, memory-specific supervision via explicitly penalizing summaries that induce high epistemic uncertainty. Experiments show that MMPO consistently outperforms existing methods on diverse long-horizon tasks, maintaining 97.1% performance even when scaled to 1.75M-token contexts.

1 Introduction

Solving complex problems in long-horizon environments with reliable internal memory forms a cornerstone of human intelligence, which is also an important component in building Artificial General Intelligence (AGI). Recently, Large Language Models (LLMs) [46, 29, 12, 7] have demonstrated remarkable reasoning capabilities. However, during extended long-horizon interactions, they are often bottlenecked by limited context windows and the "lost-in-the-middle" phenomenon [26, 25]. To address these limitations, memory-augmented agents have emerged as a prominent paradigm [9, 30, 22, 4], which recursively summarize past interaction trajectories into compact memory. This memory supports continued reasoning and task execution within a bounded context window.

Despite this advantage, recursive summarization can accumulate LLM-generated semantic noise, inducing cascading hallucinations and long-horizon reasoning collapse [45, 16, 35]. To improve memory management, recent work adopts Reinforcement Learning with Verifiable Rewards (RLVR) to train summary policies using final task outcomes [44, 8, 38, 42]. However, these sparse rewards cannot localize intermediate memory degradation or directly supervise individual summaries, still allowing noisy or irrelevant information to accumulate into *memory explosion* and performance decay [35]. This limitation motivates a principled criterion for intermediate summary optimization.

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

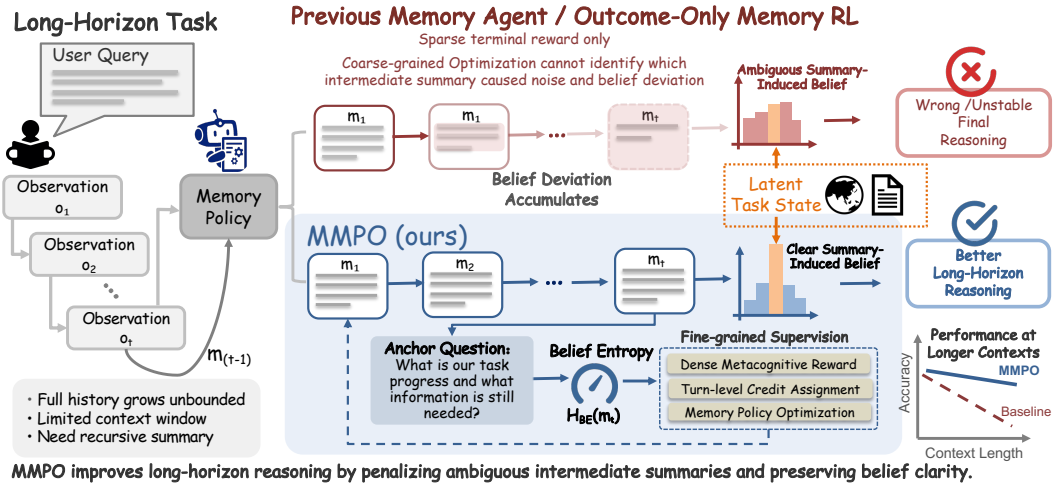


Figure 1: **Overview of MMPO. (Top)** Existing outcome-based memory policies suffer from sparse credit assignment, failing to prevent ambiguous summaries from accumulating *belief deviation*. **(Bottom)** MMPO uses a progress-and-gap anchor to compute *Belief Entropy*, providing dense, memory-specific supervision based on response uncertainty.

To address this limitation, we first analyze what accumulated summary noise disrupts in the agent's decision process. We formulate multi-turn agentic tasks as Partially Observable Markov Decision Processes (POMDPs), where hidden task states require the agent to act according to a **belief state** [33, 19]—an internal probabilistic estimate derived from the interaction history. Under this formulation, recent work [49] has attributed long-horizon reasoning collapse to *belief deviation*, i.e., the progressive drift between the agent's internal belief and the underlying latent task state as interactions extend. In summary-based workflows, textual memory replaces the full interaction history as the agent's decision context, thereby inducing the belief that guides subsequent reasoning and actions. Therefore, semantic noise accumulated through recursive summarization manifests as deviation in this *summary-induced* belief. This makes *belief preservation* the principled criterion for intermediate memory optimization: a reliable intermediate summary should preserve the task-relevant posterior supported by the full history, including uncertainty that the available evidence leaves unresolved.

This perspective suggests that final task outcomes are insufficient for memory optimization: reliable long-horizon memory requires fine-grained supervision on the task information retained in intermediate summaries. Since the latent task state is unobserved, we assess this information through the model's response to its current memory. We adopt a metacognitive probe inspired by cognitive science [11, 28] to elicit an account of task progress and remaining information needs. Instantiated with a dedicated *progress-and-gap anchor*, this probe yields **Belief Entropy**, a self-supervised predictive-uncertainty score used as a proxy for intermediate memory quality. Local analysis characterizes its semantic-ranking conditions, and controlled experiments evaluate its reliability. Based on this signal, we propose *Metacognitive Memory Policy Optimization* (MMPO), which augments outcome-based RL with dense, memory-specific supervision at intermediate memory states.

Our main contributions are threefold. (1) We formulate recursive memory optimization as belief preservation within a POMDP framework. (2) We introduce Belief Entropy, an anchor-based, self-supervised signal for intermediate memory quality, and MMPO, which combines this signal with verifiable outcomes for memory-level credit assignment. (3) We validate BE through controlled studies and demonstrate gains across frameworks and backbones: on RULER-HotpotQA, MMPO retains 97.1% of 7K accuracy at 1.75M tokens, with gains of up to 5.47 points over RL-MemAgent.

2 Belief Entropy

2.1 POMDP Formulation

We model long-horizon reasoning as a Partially Observable Markov Decision Process (POMDP) [33, 36, 19]: $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, \Omega, \mathcal{T}, \mathcal{O}, \mathcal{R}, \gamma \rangle$. At each step t, the agent observes $o_t \in \Omega$ (e.g., a retrieved

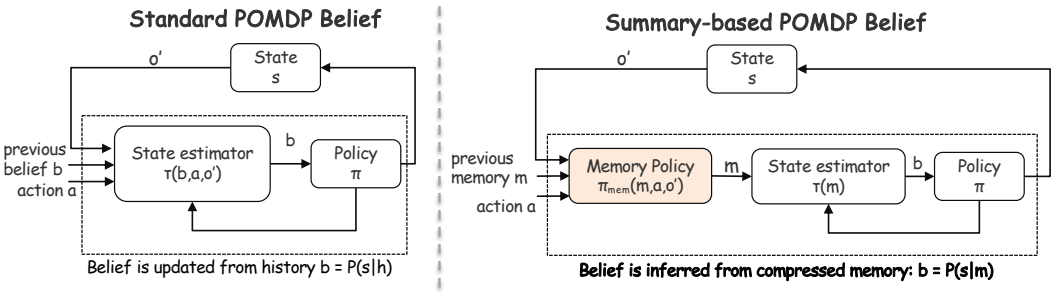


Figure 2: **Belief-state under standard and summary-based POMDPs.** (a) In standard POMDPs, the belief $b = P(s \mid h)$ is updated from the full interaction history. (b) In summary-based POMDPs, the memory policy compresses the history into a summary m, inducing a belief $b = P(s \mid m)$ from the compressed representation.

document snippet) while the true task state $s_t \in \mathcal{S}$ remains hidden. It then executes action $a_t \in \mathcal{A}$; the environment transitions via $\mathcal{T}(s_{t+1}|s_t, a_t)$ and emits the next observation via $\mathcal{O}(o_{t+1}|s_{t+1}, a_t)$.

The Belief State. Since s_t is unobservable, the optimal policy $\pi^*(a_t|b_t)$ conditions on the *belief state* $b_t \in \Delta(\mathcal{S})$, a distribution over hidden states. The belief summarizes the full interaction history $h_t = \{o_{\leq t}, a_{<t}\}$ as the posterior $b_t(s) \equiv P(s_t = s|h_t)$, updated recursively by the Bayesian filter: $b_t(s') = \eta \cdot \mathcal{O}(o_t|s', a_{t-1}) \sum_{s \in \mathcal{S}} \mathcal{T}(s'|s, a_{t-1}) b_{t-1}(s)$, where $\eta = 1/P(o_t|b_{t-1}, a_{t-1})$ normalizes. As a sufficient statistic of h_t, the belief provides all information needed for optimal decision-making.

2.2 Belief Preservation for Memory Optimization

For long-horizon LLM agents, since the interaction history grows monotonically and suffers from "lost-in-the-middle" degradation [26, 25], conditioning on the full history h_t is impractical. Such long-horizon partial observability induces *belief deviation*, where the agent's internal state estimate can drift over extended interactions [49]. To maintain finite context, memory-augmented agents use recursive summarization: at turn t, the memory policy updates a bounded textual summary $m_t = \pi_{\text{mem}}(m_{t-1}, a_{t-1}, o_t)$, and the action policy selects actions conditioned on this memory, $a_t \sim \pi_{\text{act}}(\cdot \mid m_t)$. This limits long-horizon task execution within a fixed context budget.

Summary-Induced Belief. We compare memory representations at a fixed task query and turn, suppressing this shared conditioning. Let $P_0(s_t, h_t)$ be a fixed reference state-history distribution, with s_t discrete and of finite entropy. Within the memory budget, the policy π_{mem}, parameterized by θ, induces a compression channel $K_\theta(m_t \mid h_t)$ and the joint law $P_\theta(s_t, h_t, m_t) = P_0(s_t, h_t)K_\theta(m_t \mid h_t)$. The full-history belief is $b_t(s) = P_0(s_t = s \mid h_t)$, and the summary-induced reference belief is

$$b_t^M(s) \triangleq P_\theta(s_t = s \mid m_t). \tag{1}$$

Here b_t^M describes the task-state information retained by memory; its dependence on the memory policy is implicit. Appendix B gives the reference-distribution construction.

Belief-Preservation Objective. An ideal memory policy preserves the full-history posterior. Under the reference compression law, memory contains no additional state information once the full history is given. The expected posterior information loss therefore satisfies

$$\mathbb{E}_{P_\theta(h_t, m_t)} D_{\text{KL}}(b_t \| b_t^M) = H_{P_\theta}(s_t \mid m_t) - H_{P_0}(s_t \mid h_t). \tag{2}$$

Zero KL loss preserves the full-history posterior and its uncertainty (Proposition 1, Appendix B). With P_0 fixed, the second term is constant, giving the equivalent objective

$$\min_\theta H_{P_\theta}(s_t \mid m_t). \tag{3}$$

Minimizing this conditional entropy is equivalent to maximizing the mutual information between the summary and the task state. This objective can also be viewed as making m_t an informative representation of the underlying task state, thereby preserving a reliable belief.

2.3 Belief Entropy as a Practical Proxy

Directly optimizing Equation 3 is intractable since the latent state s_t is not directly observable. To bridge this gap, we use a metacognitive probe inspired by cognitive science [11, 28, 14] to probe the model's uncertainty when assessing the task state from the current memory. We instantiate this probe with a task-state anchor question q and define **Belief Entropy (BE)** as the uncertainty of the model's response to this anchor question q:

$$\mathcal{H}_{\mathrm{BE}}(m_t) \triangleq H(y \mid m_t, q), \qquad y \sim \pi_{\mathrm{LLM}}(\cdot \mid m_t, q), \tag{4}$$

where y denotes the model's response to the anchor question. Intuitively, a clear memory can induce a concentrated response distribution, while an ambiguous or incomplete memory can lead to higher response uncertainty.

Following token-entropy scoring in entropy-based RLVR [13, 34], we compute a practical BE score by averaging token-level predictive entropy along one greedy anchor response y^*, capped at 64 tokens. At each scored position ℓ, $\mathcal{V}_\ell$ contains the top 50 predictive tokens and $\tilde{\pi}_{\mathrm{LLM}}$ renormalizes their probabilities. The score is

$$\widehat{\mathcal{H}}_{\mathrm{BE}}(m_t) = \frac{1}{|y^*|} \sum_{\ell=1}^{|y^*|} \left[- \sum_{v \in \mathcal{V}_\ell} \tilde{\pi}_{\mathrm{LLM}}(v \mid m_t, q, y^*_{<\ell}) \log \tilde{\pi}_{\mathrm{LLM}}(v \mid m_t, q, y^*_{<\ell}) \right], \tag{5}$$

Below, $\mathcal{H}_{\mathrm{BE}}$ denotes this implemented score. It averages predictive entropy along a single response; Appendix C.4 analyzes its relation to full-response entropy through prefix selection, vocabulary truncation, and length normalization. Appendix E.10 evaluates token-set sensitivity.

Progress-and-Gap Anchor Question. The anchor question is designed to probe task-state uncertainty through observable response uncertainty. It should satisfy two criteria: it should condition explicitly on the current memory m_t, and it should probe task-state uncertainty rather than generic model confidence. We therefore ask for both established task progress and remaining information needs. Progress records what has been established; information gaps identify what the agent still needs for continued execution. This *progress-and-gap design* jointly captures established evidence and unresolved information needs before the final answer is known:

> **TASK-STATE ANCHOR**
>
> "Based on current memory, what is our task progress and what information is still needed?"

Response entropy combines uncertainty over a report's meaning with variation in its wording. Under bounded expression and score-approximation variation across candidate memories, BE preserves sufficiently separated semantic-entropy rankings (Proposition 3). Appendix C.3 gives report-coverage and calibration conditions connecting these local rankings to the task-state conditional entropy in Equation 3, linking BE to belief preservation. Table 4 evaluates the anchor design.

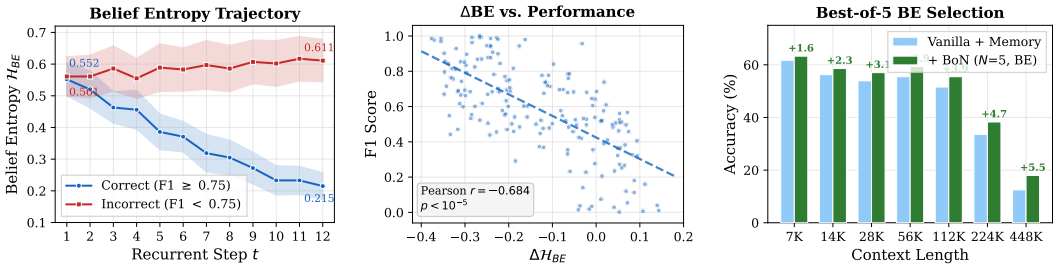
Empirical Validation. We empirically validate $\mathcal{H}_{\mathrm{BE}}$ in the MemAgent [42] RULER-HotpotQA setting using Qwen2.5-7B (Figure 3). The analysis examines whether BE provides a useful intermediate memory-quality signal. Controlled comparisons additionally evaluate semantic ranking, content sensitivity, and downstream memory selection (Appendix E.5).

Finding 1 (Trajectory Dynamics). Successful trajectories exhibit a consistent decrease in $\mathcal{H}_{\mathrm{BE}}$ as relevant evidence is accumulated, whereas failed trajectories show increasing entropy (Figure 3a). This suggests that lower Belief Entropy corresponds to a clearer summary-induced task state.

Finding 2 (Outcome Correlation). The total entropy change $\Delta\mathcal{H}_{\mathrm{BE}}$ is strongly correlated with final task accuracy (Pearson $r = -0.684$; Figure 3b), indicating that stronger entropy reduction is empirically associated with better task performance.

Finding 3 (Inference-Time Selection). Without any training, selecting the lowest-entropy trajectory among $N = 5$ candidates improves accuracy over Vanilla+Memory (Figure 3c). This demonstrates that $\mathcal{H}_{\mathrm{BE}}$ provides an actionable memory-quality signal independent of the training objective.

Reliability of BE. Semantic entropy has been validated for uncertainty estimation and confabulation detection [21, 10]; learned probes further estimate it from single-generation hidden states [20]. Our



(a) BE trajectory trend. (b) $\Delta\mathcal{H}_{\text{BE}}$ vs. accuracy. (c) Best-of-5 BE selection.

Figure 3: Empirical validation of Belief Entropy. (a) Successful trajectories show decreasing $\mathcal{H}_{\text{BE}}$, while failed ones generally stagnate or increase. (b) Entropy reduction correlates with task accuracy. (c) Test-time Best-of-N selection by $\mathcal{H}_{\text{BE}}$ improves performance. All panels use the same frozen pre-MMPO Qwen2.5-7B prober (Appendix E.5).

controlled tests assess three properties: semantic ranking—BE agrees with semantic-entropy rankings in 71.6% of memory pairs, with mean within-group Spearman correlation 0.59; content sensitivity—corruption increases BE while paraphrases leave it stable; and task utility—lower BE selects memories with better subsequent performance in 66.8% of pairs. Together, these tests support reliable memory-quality ranking within the same task and turn. Appendix E.5 provides the protocols and results; Table 4 validates the progress-and-gap design.

3 Metacognitive Memory Policy Optimization (MMPO)

Equipped with Belief Entropy as a tractable proxy for memory quality, we propose **Metacognitive Memory Policy Optimization (MMPO)**. MMPO addresses the credit assignment challenge in long-horizon reasoning by injecting dense process supervision at the level of intermediate summaries. We formalize this using a group-relative paradigm that evaluates sub-trajectories based on both their intermediate belief quality and their contribution to the final outcome. The complete training pipeline proceeds through three stages: trajectory sampling, Belief Entropy computation, and policy optimization.

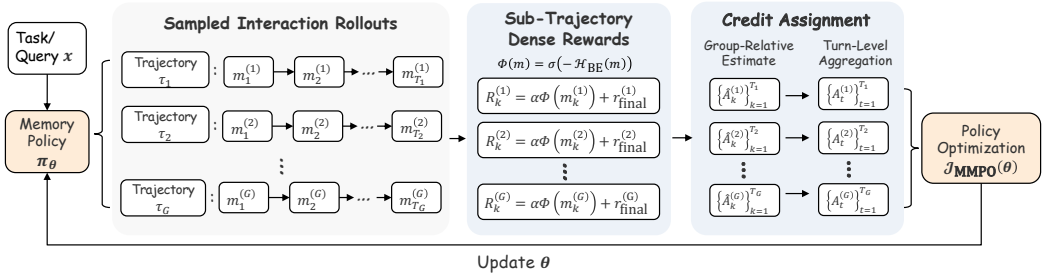


Figure 4: Overview of the MMPO training pipeline. **Stage 1:** The memory policy π_θ samples G trajectories per task. **Stage 2:** Each trajectory is decomposed into sub-trajectories $\tau_{\leq 1}, \ldots, \tau_{\leq T}$, and Belief Entropy $\mathcal{H}_{\text{BE}}(m_k)$ is computed at every turn to produce dense per-step rewards R_k. **Stage 3:** Sub-trajectory rewards are normalized via GRPO and aggregated into future-aware turn-level advantages for policy optimization.

3.1 Sub-Trajectory Dense Rewards

In MMPO, we evaluate the agent's performance through the lens of *sub-trajectories*. A sub-trajectory $\tau_{\leq k}^{(i)}$ represents the reasoning path of the i-th sample from the initial step up to turn k ($1 \leq k \leq T$).

Unlike standard sparse-reward RL, we assign a reward $R_k^{(i)}$ to each sub-trajectory that incorporates both local memory quality and the global task outcome. Let $m_k^{(i)}$ be the memory summary at turn k,

and $r_{\text{final}}^{(i)} \in [0,1]$ be the terminal outcome reward of the i-th complete trajectory (e.g., token-level F1 score for QA tasks, following MemAgent [42] and MEM1 [48]). To control the contribution of predictive uncertainty relative to the outcome reward, we map the entropy score to a bounded clarity reward using a sigmoid transformation:

$$R_k^{(i)} = \alpha \cdot \underbrace{\sigma\Big(-\mathcal{H}_{\text{BE}}(m_k^{(i)})\Big)}_{\text{Normalized BE Reward}\in(0,0.5]} + \underbrace{r_{\text{final}}^{(i)}}_{\text{Outcome Reward}\in[0,1]} \tag{6}$$

where $\sigma(\cdot)$ is the sigmoid function and $\alpha > 0$ controls the relative weight of the intrinsic belief reward against the outcome reward. Since $\mathcal{H}_{\text{BE}}$ is non-negative and may vary in scale across tasks, the sigmoid transformation converts it into a bounded reward signal: lower Belief Entropy yields a larger reward, while higher Belief Entropy yields a smaller reward. With $\alpha = 0.5$, the BE bonus is at most 0.25. Appendix E.4 compares this construction with task-aware progress rewards and local potential differences. By attaching r_{final} to every sub-trajectory within the same reasoning path, the reward still remains anchored to final task success, while the Belief Entropy term provides dense intermediate supervision for clearer memory states.

3.2 Turn-Level Advantage Estimation

To optimize the policy without the instability of a learned value network, MMPO utilizes Group Relative Advantage Estimation. For each task, we sample a group of G complete trajectories. At each reasoning depth k, we aggregate the sub-trajectory rewards across the group $\{R_k^{(1)}, R_k^{(2)}, \ldots, R_k^{(G)}\}$. The *sub-trajectory advantage* $\hat{A}_k^{(i)}$ is computed by standardizing the rewards within this group at turn k:

$$\hat{A}_k^{(i)} = \frac{R_k^{(i)} - \text{mean}(R_k)}{\text{std}(R_k) + \varepsilon_A} \tag{7}$$

For unequal trajectory lengths T_i, statistics use the valid set $\mathcal{I}_k = \{i : T_i \geq k\}$; Appendix F specifies numerical stability and masking. By comparing sub-trajectories at the same step, $\hat{A}_k^{(i)}$ isolates the relative contribution of the i-th path's memory policy to the task's progress. A positive advantage indicates that the k-step prefix of the i-th sample is superior to its peers in balancing memory quality and goal attainment.

To update the memory policy for generating m_t, we must aggregate the advantages of all future sub-trajectories influenced by m_t. Since the summary at turn t serves as the context for all subsequent steps $k \in \{t, t+1, \ldots, T\}$, its overall quality is reflected in the performance of all these overlapping sub-trajectories.

We compute the *turn-level advantage* $A_t^{(i)}$ by averaging the advantages of all sub-trajectories containing turn t:

$$A_t^{(i)} = \frac{1}{T - t + 1} \sum_{k=t}^{T} \hat{A}_k^{(i)} \tag{8}$$

Here T denotes the evaluated trajectory length; unequal-length trajectories use their respective T_i. This aggregation mechanism provides a robust credit assignment: a memory summary m_t is reinforced if it leads to a sequence of subsequent states that are consistently clearer and more successful than the alternatives in the group.

3.3 Optimization Objective

The turn-level advantage $A_t^{(i)}$ is distributed to all tokens comprising the summary m_t. Let the summary consist of L tokens $\{w_1, \ldots, w_L\}$. We optimize the memory policy π_θ using the clipped PPO surrogate objective:

$$\mathcal{J}_{\text{MMPO}}(\theta) = \mathbb{E}\left[\sum_{t=1}^{T}\sum_{j=1}^{L} \min\left(\rho_{t,j}(\theta)A_t^{(i)},\ \text{clip}(\rho_{t,j}(\theta), 1-\epsilon, 1+\epsilon)A_t^{(i)}\right)\right] - \beta\mathbb{D}_{\text{KL}}(\pi_\theta \| \pi_{\text{ref}}) \tag{9}$$

where $\rho_{t,j}(\theta)$ is the token-level importance ratio and $\mathbb{D}_{\text{KL}}$ is the penalty against the reference model π_{ref}. Only generated memory tokens contribute to the policy loss and KL penalty; rewards are detached

Table 1: Main results on RULER-HotpotQA across context lengths. MMPO follows the same recursive memory workflow as RL-MemAgent and adds Belief Entropy supervision during training.

Model	7K	14K	28K	56K	112K	224K	448K	896K	1.75M	3.5M
QwenLong-L1-32B	72.66	75.00	72.66	60.94	31.25	17.19	13.28	11.72	N/A	N/A
Qwen2.5-14B-1M	60.16	60.94	50.00	57.03	50.00	37.50	8.59	0.00	N/A	N/A
Qwen2.5-7B-1M	61.72	56.25	53.91	55.47	51.56	33.59	12.50	0.00	N/A	N/A
DS-Qwen-32B	70.31	66.41	65.62	46.88	23.44	13.28	7.81	7.03	N/A	N/A
DS-Qwen-14B	64.06	64.84	57.03	40.62	14.84	8.59	3.12	6.25	N/A	N/A
DS-Qwen-7B	30.47	12.50	3.12	0.00	0.00	0.78	0.00	0.00	N/A	N/A
RL-MemAgent-14B	80.47	82.03	82.03	83.59	81.25	77.34	79.69	75.78	78.91	71.09
MMPO-14B	82.16	**83.38**	**84.32**	**83.75**	82.47	**80.59**	**80.91**	**80.69**	**79.77**	**76.47**
RL-MemAgent-7B	81.25	81.25	82.03	80.47	79.69	75.78	76.56	74.22	77.34	71.88
MMPO-7B	**82.38**	82.81	81.03	82.98	**82.81**	79.56	78.12	79.69	78.91	75.22

scalar feedback. This objective reinforces memory updates that support informative task-state reports and successful long-horizon execution (Appendix F).

3.4 Algorithm Overview

The complete MMPO training procedure is given in Algorithm 1 (Appendix A). At each iteration, the policy samples a group of trajectories, computes per-turn Belief Entropy, constructs sub-trajectory rewards, and updates parameters via the clipped objective in Eq. (9).

Implementation Details. At each turn t, the memory policy receives the previous summary m_{t-1} and the current observation o_t, and generates a new summary m_t. Belief Entropy is computed using Eq. 5. The anchor question is adapted per task type: for QA tasks, "Based on current memory, what is our task progress and what information is still needed?"; for agentic tasks with tool use, "Based on current memory, what is our task progress and what steps remain?"

4 Experiments

We evaluate MMPO against two representative memory-agent frameworks: MemAgent [42] and MEM1 [48]. The MemAgent comparison tests whether Belief Entropy supervision improves recursive memory summarization under extreme context-length scaling on RULER-HotpotQA. The MEM1 comparison tests whether the same supervision is complementary to MEM1's outcome-based training on multi-objective QA and WebShop. Since MMPO keeps the corresponding memory workflow unchanged and only augments the training signal, we focus on task performance under the original evaluation protocols.

4.1 Experimental Setup

For the **MemAgent**-based comparison, we evaluate MMPO on RULER-HotpotQA following the MemAgent setting [42]. RULER-HotpotQA combines HotpotQA-style multi-hop questions with controllable long-context scaling, requiring the agent to update memory over progressively longer distractor contexts. This benchmark tests whether a memory agent can retain task-relevant evidence under extreme context-length scaling. We train MMPO with the same recursive memory workflow to isolate the effect of Belief Entropy supervision on intermediate memory states.

For the **MEM1**-based comparison, we evaluate MMPO on multi-objective QA and WebShop [41]. In multi-objective QA, the agent is trained on 2-objective tasks and evaluated on longer objective horizons, requiring memory to track multiple unresolved information needs. In WebShop, the agent interacts with the environment through search and click actions, testing whether the same memory supervision benefits interactive decision-making beyond retrieval-based QA.

Table 2: Comparison with MEM1 on Multi-objective QA. We follow the MEM1 evaluation protocol and report task scores. Following MEM1, EM/F1 are aggregated over objectives.

Model	2-Objective		8-Objective		16-Objective	
	EM ↑	F1 ↑	EM ↑	F1 ↑	EM ↑	F1 ↑
Qwen2.5-14B-Inst	0.732	0.902	1.55	1.87	0.567	0.703
Qwen2.5-7B-Inst	0.268	0.366	0.87	1.10	0.165	0.213
Qwen2.5-7B-Inst (A-MEM)	0.286	0.371	1.13	1.43	0.730	0.961
Qwen2.5-7B-Inst (truncate)	0.262	0.336	0.97	1.23	0.396	0.497
Search-R1	0.452	0.531	0.064	0.080	0.009	0.011
DeepResearcher	0.536	0.650	0.73	0.90	0.071	0.106
MEM1-QA	0.709	0.838	1.87	2.31	1.97	2.39
MMPO	**0.725**	**0.852**	**2.15**	**2.63**	**2.43**	**2.84**

Table 3: WebShop results under the MEM1 evaluation protocol.

Method	Avg. Reward ↑
Qwen2.5-7B	18.42
Qwen2.5-14B	12.34
MEM1-WebShop	70.87
MMPO	**77.25**

Table 4: Anchor question ablation on Ruler HQA with Qwen2.5-7B at 56K context length.

Anchor Question	Acc. ↑	$\Delta\mathcal{H}_{\text{BE}}$ Corr.
Outcome Only	80.47	–
Direct-answer probe	78.17	-0.54
Gap-only probe	82.02	-0.62
Progress + gap	**82.98**	$\mathbf{-0.68}$

We further compare with additional agent baselines, including A-MEM [40], Search-R1 [17], and DeepResearcher [47]. We also evaluate Qwen3.5-9B [32] under the same recurrent memory workflow (Appendix E.8). Appendix D details the compared memory-agent frameworks.

4.2 Main Results

Comparison with MemAgent Table 1 compares MMPO with MemAgent and other long-context/reasoning baselines on RULER-HotpotQA. We evaluate MMPO with both Qwen2.5-7B and Qwen2.5-14B backbones, using the same memory workflow as MemAgent. MMPO improves the average accuracy over RL-MemAgent for both model sizes. The improvement is especially clear in the long-context regime: from 224K to 3.5M context length, MMPO improves accuracy over RL-MemAgent by an average of 3.14 percentage points on Qwen2.5-7B and 3.12 points on Qwen2.5-14B. The largest gains are 5.47 points at 896K for Qwen2.5-7B and 5.38 points at 3.5M for Qwen2.5-14B. These results suggest that Belief Entropy provides useful intermediate supervision for recursive memory summarization, helping the memory policy preserve clearer summary-induced beliefs and reduce noise accumulation over long contexts.

Experiments with Qwen3.5-9B. On the fixed 500-question 56K evaluation, MMPO improves Qwen3.5-9B accuracy from 84.2% to 86.0%. A separate 7K–3.5M length sweep shows improvements in nine of ten settings, with gains of 2.06–3.16 percentage points at 224K–3.5M (Appendix E.8).

Comparison with MEM1 Table 2 evaluates whether MMPO is complementary to MEM1's outcome-based memory training. On multi-objective QA, MMPO improves over MEM1-QA across the evaluated objective horizons, with larger gains at harder settings such as 8-objective and 16-objective QA, where the agent must maintain multiple unresolved information needs over longer trajectories. On WebShop, MMPO also improves over MEM1-WebShop (see Table 3), showing that the proposed supervision is not limited to retrieval-based QA but also benefits interactive tool-use tasks. Since the underlying memory architecture is unchanged, these improvements suggest that the gain comes from the denser memory-quality signal introduced by Belief Entropy.

4.3 Analysis

Anchor Question Ablation. Table 4 compares anchor-question designs on RULER-HQA with Qwen2.5-7B at 56K context length. *Outcome Only* is the standard RLVR baseline using only

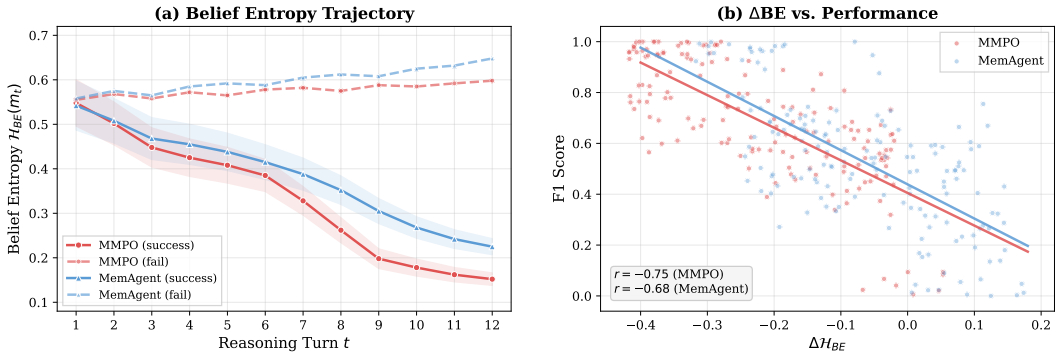


Figure 5: **Belief Entropy analysis. (a)** Belief Entropy trajectories over reasoning turns at 56K context length. Successful trajectories show consistent entropy decrease, while failed trajectories stagnate or increase. **(b)** Correlation between total entropy change $\Delta\mathcal{H}_{\text{BE}}$ and task accuracy across 500 test episodes. MMPO strengthens this correlation compared with MemAgent, supporting Belief Entropy as a proxy for intermediate memory quality. Both methods use the same frozen pre-MMPO Qwen2.5-7B prober, anchor, and decoding settings (Appendix E.5).

final task reward. The *Direct-answer probe* asks for the final answer from memory, the *Gap-only probe* asks for missing evidence, and our default *Progress + gap* probe asks for both task progress and missing information. The direct-answer probe remains correlated with task accuracy ($r = -0.54$), but underperforms Outcome Only, suggesting that rewarding low answer entropy can encourage premature confidence before sufficient evidence is collected. In contrast, the gap-only and progress+gap probes better target intermediate memory quality by exposing unresolved information needs. The progress+gap probe performs best, showing that jointly tracking progress and missing information provides a more informative memory-quality signal.

Belief Entropy Dynamics. Figure 5 analyzes Belief Entropy dynamics during long-horizon reasoning. Successful trajectories show decreasing entropy as evidence accumulates, while failed trajectories stagnate or increase. MMPO yields a steeper entropy decline than MemAgent and strengthens the correlation between entropy reduction and task accuracy, indicating that BE supervision better aligns intermediate memory clarity with final task success.

Additional validation and generalization. On Qwen2.5-7B, three 56K training seeds yield a mean gain of 2.00 percentage points (paired-bootstrap 95% CI: $[0.68, 3.31]$). Further experiments demonstrate transfer to RULER synthetic tasks, SQuAD, and LongBench summarization, and a 2.20-point gain with separate memory-policy, prober, and solver models. Training profiles show approximately 12% overhead, with no prober cost during standard inference. Appendix E reports these results, reward controls, and implementation details.

5 Related Work

Memory-Augmented LLM Agents. Managing long interaction histories is a central challenge for long-horizon LLM agents. Existing methods rely on truncation, retrieval-augmented context construction [25], or KV-cache management [3], while recent memory agents compress trajectories into compact memory states. MemGPT [30] introduces an operating-system-inspired memory hierarchy, and Mem0 [4] and MemOS [22] develop learnable memory management layers. More recent works further formulate memory as a trainable policy within the agent loop [9, 42, 48]. However, they are typically optimized by final task outcomes, leaving intermediate summaries weakly supervised. MMPO addresses this gap by supervising the quality of summary-induced beliefs during memory optimization.

Reinforcement Learning for LLMs. RLHF and related outcome-based objectives are widely used for LLM alignment [8, 44]. To improve credit assignment in multi-step reasoning [27], process-supervision methods such as PRM [23], PRIME [6], Miracle [43], and AgentPRM [39] reward intermediate reasoning states, while GRPO [8] stabilizes RL through group-relative normalization

without a value model. In memory-agent settings, MemAlpha [38], MemAgent [42], and MEM1 [48] apply RL to memory policies, but still largely rely on outcome-level rewards. MMPO assigns task-conditioned uncertainty rewards to recursive memory updates through same-depth comparison and future-aware aggregation. Appendix E.11 details its relationship to entropy-based reasoning objectives and process reward models.

Belief States and Uncertainty Estimation. Belief states are central to decision-making under partial observability [33, 36, 19], with related work studying history compression through predictive state representations and information-theoretic sufficient statistics [24, 37]. Recent studies identify belief deviation as a key failure mode of long-horizon LLM agents [49], motivating memory mechanisms that preserve reliable state estimates. In parallel, predictive entropy, semantic entropy, verbalized confidence, and self-consistency have been used to quantify LLM uncertainty [18, 21, 10, 20]. MMPO connects these directions by using a self-supervised entropy signal as dense supervision for memory-policy optimization, rather than for selective prediction.

6 Conclusion

We introduced Metacognitive Memory Policy Optimization (MMPO), a memory optimization framework for long-horizon LLM agents. Motivated by posterior information preservation, MMPO uses predictive uncertainty in progress-and-gap reports as dense supervision alongside verifiable task outcomes. Local analysis characterizes the score's semantic-ranking conditions, and controlled validation establishes its practical value for memory diagnosis and selection. Experiments demonstrate improved long-horizon task performance.

Acknowledgement

This work was conducted during Ziyan Liu and Zhezheng Hao's internships with the WorkBuddy team at Tencent, which provided computational support. Wei Xia and Feng Liu are employees of Tencent. This research did not receive any external research funding. The authors declare no additional competing interests beyond the affiliations and employment relationships disclosed above.

References

[1] Shivam Agarwal, Zimin Zhang, Lifan Yuan, Jiawei Han, and Hao Peng. The unreasonable effectiveness of entropy minimization in llm reasoning. *Advances in Neural Information Processing Systems*, 38:107150–107180, 2026.

[2] Yushi Bai, Xin Lv, Jiajie Zhang, Hongchang Lyu, Jiankai Tang, Zhidian Huang, Zhengxiao Du, Xiao Liu, Aohan Zeng, Lei Hou, Yuxiao Dong, Jie Tang, and Juanzi Li. Longbench: A bilingual, multitask benchmark for long context understanding. In *ACL*. Association for Computational Linguistics, 2024.

[3] Yeqiu Chen, Ziyan Liu, Zhenxin Huang, Runquan Gui, Hong Wang, and Lei Liu. ArborKV: Structure-aware KV cache management for scaling tree-based LLM reasoning. *arXiv preprint arXiv:2605.22106*, 2026.

[4] 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.

[5] Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory*. John Wiley & Sons, 2 edition, 2006. ISBN 9780471241959. doi: 10.1002/047174882X.

[6] Ganqu Cui, Lifan Yuan, Zefan Wang, Hanbin Wang, Wendi Li, Bingxiang He, Yuchen Fan, Tianyu Yu, Qixin Xu, Weize Chen, Jiarui Yuan, Huayu Chen, Kaiyan Zhang, Xingtai Lv, Shuo Wang, Yuan Yao, Xu Han, Hao Peng, Yu Cheng, Zhiyuan Liu, Maosong Sun, Bowen Zhou, and Ning Ding. Process reinforcement through implicit rewards. *arXiv preprint arXiv:2502.01456*, 2025.

[7] DeepSeek-AI, Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Daya Guo, Dejian Yang, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Haowei Zhang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Li, Hui Qu, J. L. Cai, Jian Liang, Jianzhong Guo, Jiaqi Ni, Jiashi Li, Jiawei Wang, Jin Chen, Jingchang Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, Junxiao Song, Kai Dong, Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Lei Xu, Leyi Xia, Liang Zhao, Litong Wang, Liyue Zhang, Meng Li, Miaojun Wang, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Mingming Li, Ning Tian, Panpan Huang, Peiyi Wang, Peng Zhang, Qiancheng Wang, Qihao Zhu, Qinyu Chen, Qiushi Du, R. J. Chen, R. L. Jin, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, Runxin Xu, Ruoyu Zhang, Ruyi Chen, S. S. Li, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shaoqing Wu, Shengfeng Ye, Shengfeng Ye, Shirong Ma, Shiyu Wang, Shuang Zhou, Shuiping Yu, Shunfeng Zhou, Shuting Pan, T. Wang, Tao Yun, Tian Pei, Tianyu Sun, W. L. Xiao, Wangding Zeng, et al. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*, 2024.

[8] DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, Bing Xue, Bingxuan Wang, Bochao Wu, Bei Feng, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Qu, Hui Li, Jianzhong Guo, Jiashi Li, Jiawei Wang, Jingchang Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, J. L. Cai, Jiaqi Ni, Jian Liang, Jin Chen, Kai Dong, Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Liang Zhao, Litong Wang, Liyue Zhang, Lei Xu, Leyi Xia, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Meng Li, Miaojun Wang, Mingming Li, Ning Tian, Panpan Huang, Peng Zhang, Qiancheng Wang, Qinyu Chen, Qiushi Du, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, R. J. Chen, R. L. Jin, Ruyi Chen, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shengfeng Ye, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, S. S. Li, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

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

[10] Sebastian Farquhar, Jannik Kossen, Lorenz Kuhn, and Yarin Gal. Detecting hallucinations in large language models using semantic entropy. *Nature*, 630(8017), 2024. doi: 10.1038/s41586-024-07421-0.

[11] John H Flavell. Metacognition and cognitive monitoring: A new area of cognitive–developmental inquiry. *American psychologist*, 34(10), 1979.

[12] Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

[13] Zhezheng Hao, Hong Wang, Haoyang Liu, Jian Luo, Jiarui Yu, Hande Dong, Qiang Lin, Can Wang, and Jiawei Chen. Rethinking entropy interventions in rlvr: An entropy change perspective. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 31105–31133, 2026.

[14] Julian T Hart. Memory and the feeling-of-knowing experience. *Journal of educational psychology*, 56(4), 1965.

[15] Cheng-Ping Hsieh, Simeng Sun, Samuel Kriman, Shantanu Acharya, Dima Rekesh, Fei Jia, Yang Zhang, and Boris Ginsburg. Ruler: What's the real context size of your long-context language models? *arXiv preprint arXiv:2404.06654*, 2024.

[16] Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Yejin Bang, Andrea Madotto, and Pascale Fung. Survey of hallucination in natural language generation. *ACM Computing Surveys*, 55(12), 2023.

[17] Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan Arik, Dong Wang, Hamed Zamani, and Jiawei Han. Search-R1: Training LLMs to reason and leverage search engines with reinforcement learning. In *COLM*, 2025.

[18] Saurav Kadavath, Tom Conerly, Amanda Askell, Tom Henighan, Dawn Drain, Ethan Perez, Nicholas Schiefer, Zac Hatfield-Dodds, Nova DasSarma, Eli Tran-Johnson, et al. Language models (mostly) know what they know. *arXiv preprint arXiv:2207.05221*, 2022.

[19] Leslie Pack Kaelbling, Michael L Littman, and Anthony R Cassandra. Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 101(1-2), 1998.

[20] Jannik Kossen, Jiatong Han, Muhammed Razzak, Lisa Schut, Shreshth Malik, and Yarin Gal. Semantic entropy probes: Robust and cheap hallucination detection in LLMs. *arXiv preprint arXiv:2406.15927*, 2024.

[21] Lorenz Kuhn, Yarin Gal, and Sebastian Farquhar. Semantic uncertainty: Linguistic invariances for uncertainty estimation in natural language generation. In *ICLR*, 2023.

[22] Zhiyu Li, Shichao Song, Hanyu Wang, Simin Niu, Ding Chen, Jiawei Yang, Chenyang Xi, Huayi Lai, Jihao Zhao, Yezhaohui Wang, Junpeng Ren, Zehao Lin, Jiahao Huo, Tianyi Chen, Kai Chen, Kehang Li, Zhiqiang Yin, Qingchen Yu, Bo Tang, Hongkang Yang, Zhi-Qin John Xu, and Feiyu Xiong. Memos: An operating system for memory-augmented generation (MAG) in large language models. *arXiv preprint arXiv:2505.22101*, 2025.

[23] Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *ICLR*, 2024.

[24] Michael L. Littman, Richard S. Sutton, and Satinder Singh. Predictive representations of state. In *NeurIPS*, volume 14, 2001.

[25] Jiaheng Liu, Dawei Zhu, Zhiqi Bai, Yancheng He, Huanxuan Liao, Haoran Que, Zekun Wang, Chenchen Zhang, Ge Zhang, Jiebin Zhang, Yuanxing Zhang, Zhuo Chen, Hangyu Guo, Shilong Li, Ziqiang Liu, Yong Shan, Yifan Song, Jiayi Tian, Wenhao Wu, Zhejian Zhou, Ruijie Zhu, Junlan Feng, Yang Gao, Shizhu He, Zhoujun Li, Tianyu Liu, Fanyu Meng, Wenbo Su, Yingshui Tan, Zili Wang, Jian Yang, Wei Ye, Bo Zheng, Wangchunshu Zhou, Wenhao Huang, Sujian Li, and Zhaoxiang Zhang. A comprehensive survey on long context language modeling. *arXiv preprint arXiv:2503.17407*, 2025.

[26] Nelson F Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *Transactions of the association for computational linguistics*, 12:157–173, 2024.

[27] Ziyan Liu, Xueda Shen, Yuzhe Gu, Songyang Gao, Kuikun Liu, Guangran Cheng, Chengqi Lyu, Dahua Lin, Wenwei Zhang, and Kai Chen. Thoughtfold: Folding reasoning chains via introspective preference learning. *arXiv preprint arXiv:2606.03503*, 2026.

[28] Thomas O. Nelson and Louis Narens. *Metamemory: A Theoretical Framework and New Findings*, volume 26. Elsevier, 1990.

[29] OpenAI. GPT-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

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

[31] Mihir Prabhudesai, Lili Chen, Alex Ippoliti, Katerina Fragkiadaki, Hao Liu, and Deepak Pathak. Maximizing confidence alone improves reasoning. *arXiv preprint arXiv:2505.22660*, 2025.

[32] Qwen Team. Qwen3.5: Towards native multimodal agents, February 2026. URL https://qwen.ai/blog?id=qwen3.5.

[33] Karl Johan Åström. Optimal control of Markov processes with incomplete state information. *Journal of Mathematical Analysis and Applications*, 10(1), 1965. doi: 10.1016/0022-247X(65)90154-X.

[34] Han Shen. On entropy control in llm-rl algorithms. In *International Conference on Learning Representations*, volume 2026, pages 136535–136556, 2026.

[35] Leheng Sheng, Yongtao Zhang, Wenchang Ma, Yaorui Shi, Ting Huang, Xiang Wang, An Zhang, Ke Shen, and Tat-Seng Chua. When to memorize and when to stop: Gated recurrent memory for long-context reasoning. *arXiv preprint arXiv:2602.10560*, 2026.

[36] Richard D. Smallwood and Edward J. Sondik. The optimal control of partially observable Markov processes over a finite horizon. *Operations Research*, 21(5), 1973. doi: 10.1287/opre.21.5.1071.

[37] Susanne Still and Doina Precup. An information-theoretic approach to curiosity-driven reinforcement learning. *Theory in Biosciences*, 131(3), 2012.

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

[39] Zhiheng Xi, Chenyang Liao, Guanyu Li, Zhihao Zhang, Wenxiang Chen, Binghai Wang, Senjie Jin, Yuhao Zhou, Jian Guan, Wei Wu, et al. Agentprm: Process reward models for llm agents via step-wise promise and progress. In *Proceedings of the ACM Web Conference 2026*, pages 4184–4195, 2026.

[40] Wujiang Xu, Zujie Liang, Kai Mei, Hang Gao, Juntao Tan, and Yongfeng Zhang. A-mem: Agentic memory for llm agents. *Advances in Neural Information Processing Systems*, 38: 17577–17604, 2026.

[41] Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. Webshop: Towards scalable real-world web interaction with grounded language agents. In *NeurIPS*, volume 35, 2022.

[42] Hongli Yu, Tinghong Chen, Jiangtao Feng, Jiangjie Chen, Weinan Dai, Qiying Yu, Ya-Qin Zhang, Wei-Ying Ma, Jingjing Liu, Mingxuan Wang, et al. Memagent: Reshaping long-context llm with multi-conv rl-based memory agent. In *International Conference on Learning Representations*, volume 2026, pages 39458–39486, 2026.

[43] Youliang Yuan, Qiuyang Mang, Jingbang Chen, Hong Wan, Xiaoyuan Liu, Junjielong Xu, Jen-tse Huang, Wenxuan Wang, Wenxiang Jiao, and Pinjia He. Curing miracle steps in LLM mathematical reasoning with rubric rewards. *arXiv preprint arXiv:2510.07774*, 2025.

[44] Kaiyan Zhang, Yuxin Zuo, Bingxiang He, Youbang Sun, Runze Liu, Che Jiang, Yuchen Fan, Kai Tian, Guoli Jia, Pengfei Li, Yu Fu, Xingtai Lv, Yuchen Zhang, Sihang Zeng, Shang Qu, Haozhan Li, Shijie Wang, Yuru Wang, Xinwei Long, Fangfu Liu, Xiang Xu, Jiaze Ma, Xuekai Zhu, Ermo Hua, Yihao Liu, Zonglin Li, Huayu Chen, Xiaoye Qu, Yafu Li, Weize Chen, Zhenzhao Yuan, Junqi Gao, Dong Li, Zhiyuan Ma, Ganqu Cui, Zhiyuan Liu, Biqing Qi, Ning Ding, and Bowen Zhou. A survey of reinforcement learning for large reasoning models. *arXiv preprint arXiv:2509.08827*, 2025.

[45] Yue Zhang, Yafu Li, Leyang Cui, Deng Cai, Lemao Liu, Tingchen Fu, Xinting Huang, Enbo Zhao, Yu Zhang, Yulong Chen, et al. Siren's song in the ai ocean: A survey on hallucination in large language models. *Computational Linguistics*, 51(4), 2025.

[46] Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, Yifan Du, Chen Yang, Yushuo Chen, Zhipeng Chen, Jinhao Jiang, Ruiyang Ren, Yifan Li, Xinyu Tang, Zikang Liu, Peiyu Liu, Jian-Yun Nie, and Ji-Rong Wen. A survey of large language models. *arXiv preprint arXiv:2303.18223*, 2023.

[47] Yuxiang Zheng, Dayuan Fu, Xiangkun Hu, Xiaojie Cai, Lyumanshan Ye, Pengrui Lu, and Pengfei Liu. Deepresearcher: Scaling deep research via reinforcement learning in real-world environments. In *EMNLP*, 2025.

[48] Zijian Zhou, Ao Qu, Zhaoxuan Wu, Sunghwan Kim, Alok Prakash, Daniela Rus, Bryan Kian Hsiang Low, and Paul Liang. Mem1: Learning to synergize memory and reasoning for efficient long-horizon agents. In *International Conference on Learning Representations*, volume 2026, pages 58413–58438, 2026.

[49] Deyu Zou, Yongqiang Chen, Jianxiang Wang, Garry Yang, Mufei Li, Qing Da, James Cheng, Pan Li, and Yu Gong. Reducing belief deviation in reinforcement learning for active reasoning of LLM agents. In *ICLR*, 2026.

A MMPO Algorithm

Algorithm 1 Metacognitive Memory Policy Optimization (MMPO)

Require: Policy π_θ, reference policy π_{ref}, anchor q, group size G, max turns T, coefficients α, β, ϵ

for each training iteration **do**
  Take a rollout-start actor snapshot as the BE prober
  Sample a batch of tasks from the training set
  for each task **do**
    Sample G complete trajectories $\{\tau^{(i)}\}_{i=1}^{G}$ using π_θ
    Retain valid completed trajectories according to Appendix F
    for each retained trajectory $\tau^{(i)}$ **do**
      Read memory summaries $m_1^{(i)}, \ldots, m_{T_i}^{(i)}$ and terminal reward $r_{\text{final}}^{(i)}$
      Score each memory with the fixed snapshot using Eq. 5
      Form $R_k^{(i)} = \alpha\sigma(-\mathcal{H}_{\text{BE}}(m_k^{(i)})) + r_{\text{final}}^{(i)}$
    end for
    for each valid depth k **do**
      Normalize $R_k^{(i)}$ over $\mathcal{I}_k = \{i : T_i \geq k\}$ to obtain $\hat{A}_k^{(i)}$ (Eq. 7)
    end for
    for each trajectory i and memory turn t **do**
      Aggregate $A_t^{(i)} = \frac{1}{T_i - t + 1} \sum_{k=t}^{T_i} \hat{A}_k^{(i)}$
    end for
  end for
  Update θ through memory tokens using Eq. 9; keep rewards detached
end for

B Summary-Induced Belief: Architectural Justification

This appendix defines the information available through a summary and the reference distribution used to compare compression policies. Known task context is included in all conditioning sets and suppressed below where convenient. For proofs, S, H, M denote the state, history, and memory at the fixed turn t; the beliefs retain the main-text notation b_t and b_t^M.

POMDP and full-history belief. A POMDP is specified by $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, \Omega, \mathcal{T}, \mathcal{O}, \mathcal{R}, \gamma \rangle$ [33, 36, 19]. At turn t, the agent observes $o_t \in \Omega$ while the task state $s_t \in \mathcal{S}$ remains hidden. After action $a_t \in \mathcal{A}$, the environment transitions according to $\mathcal{T}(s_{t+1} \mid s_t, a_t)$ and emits o_{t+1} according to $\mathcal{O}(o_{t+1} \mid s_{t+1}, a_t)$. The reward function $\mathcal{R}$ and discount γ define the return. For history $h_t = \{o_{\leq t}, a_{<t}\}$, the full-history posterior is

$$b_t(s) \triangleq P(s_t = s \mid h_t), \tag{10}$$

and its Bayesian update is

$$b_t(s') = \eta\, \mathcal{O}(o_t \mid s', a_{t-1}) \sum_{s \in \mathcal{S}} \mathcal{T}(s' \mid s, a_{t-1}) b_{t-1}(s), \tag{11}$$

where η normalizes the distribution. Under the POMDP model, this posterior is a sufficient statistic of the history for optimal decision-making.

Reference distribution for memory compression. A recursive memory policy updates $m_t \sim \pi_\theta(\cdot \mid m_{t-1}, a_{t-1}, o_t, c)$, and subsequent actions use $\pi_{\text{act}}(\cdot \mid m_t, c)$. To analyze information preservation at a fixed task and turn, let S, H, M denote the state, history, and memory at that turn.

Fix a reference law $P_0(s, h \mid c)$ and vary the compression channel $K_\theta(m \mid h, c)$ within the memory budget:

$$P_\theta(s, h, m \mid c) = P_0(s, h \mid c) K_\theta(m \mid h, c). \tag{12}$$

This law isolates how a memory representation preserves information in the reference histories. It induces the summary-conditioned reference posterior

$$b_t^M(s \mid m, c) \triangleq P_\theta(S = s \mid M = m, C = c). \tag{13}$$

The posterior describes the task-state information available from memory. The architecture fixes this information source; the actual prober distribution Q_m describes the model's responses to it. Appendix C.3 connects these two distributions through a shared semantic report space.

Information constraint and optimization objective. Under Eq. 12, $S \to H \to M$ is a Markov chain conditional on $C = c$, equivalently $S \perp M \mid H, C = c$. The compression channel depends on the history and known context: once these are given, observing the generated memory leaves the state posterior unchanged, $P_\theta(S \mid H, M, c) = P_0(S \mid H, c)$. Memory can nevertheless retain state information across histories, as quantified by $I_{P_\theta}(S; M \mid C = c)$. The data processing inequality [5] gives

$$I_{P_\theta}(S; M \mid C = c) \le I_{P_0}(S; H \mid C = c). \tag{14}$$

For a discrete state with finite entropy,

$$\begin{aligned} I_{P_\theta}(S; M \mid C = c) &= H_{P_0}(S \mid C = c) - H_{P_\theta}(S \mid M, C = c), \\ H_{P_\theta}(S \mid M, C = c) &= -\mathbb{E}_{P_\theta(\cdot \mid c)}[\log b_t^M(S \mid M, c)]. \end{aligned} \tag{15}$$

Because P_0 is fixed, minimizing $H_{P_\theta}(S \mid M, c)$ is equivalent to maximizing either the mutual information or the expected log probability of the reference state. The following identity connects this objective directly to the posterior information loss in Eq. 2.

Proposition 1 (Posterior information loss under compression)**.** *Under Eq. 12, suppose S is discrete and $H_{P_0}(S \mid C = c) < \infty$. Write $b_t = P_0(S \mid H, c)$ and $b_t^M = P_\theta(S \mid M, c)$. Then*

$$\begin{aligned} \mathcal{L}_{\text{pres}}(\theta; c) &:= \mathbb{E}_{P_\theta(H, M \mid c)} D_{\text{KL}}(b_t \| b_t^M) \\ &= H_{P_\theta}(S \mid M, c) - H_{P_0}(S \mid H, c) \\ &= I_{P_\theta}(S; H \mid M, c) \\ &= I_{P_0}(S; H \mid c) - I_{P_\theta}(S; M \mid c). \end{aligned} \tag{16}$$

The loss is zero if and only if $b_t = b_t^M$ for $P_\theta(H, M \mid c)$-almost every pair, equivalently $S \perp H \mid M, C = c$.

Proof. The conditional Markov property gives $P_\theta(S \mid H, M, c) = P_0(S \mid H, c)$. Expanding the expected KL divergence therefore yields

$$\begin{aligned} \mathcal{L}_{\text{pres}}(\theta; c) &= \mathbb{E}_{P_\theta(S, H, M \mid c)} \left[\log \frac{P_0(S \mid H, c)}{P_\theta(S \mid M, c)} \right] \\ &= H_{P_\theta}(S \mid M, c) - H_{P_0}(S \mid H, c). \end{aligned}$$

The two conditional entropies are finite under the stated assumption. Substituting $H_{P_\theta}(S \mid H, M, c) = H_{P_0}(S \mid H, c)$ gives the conditional mutual information. Expanding $I_{P_\theta}(S; H, M \mid c)$ in both orders gives the final equality. Nonnegativity of KL and its equality condition show that zero expected loss is equivalent to equality of the two posteriors almost surely, which is precisely the stated conditional independence. □

Thus memory optimization minimizes the information about S that remains in the history after conditioning on memory. Zero compression loss retains the entire full-history posterior even when $H_{P_0}(S \mid H, c) > 0$: uncertainty left unresolved by the original evidence is preserved. Equation 16 averages over (H, M) under P_θ. Conditioning on a particular memory instead averages full-history entropy under $P_\theta(H \mid M = m, c)$, which can vary with m; averaging across memories recovers the fixed term $H_{P_0}(S \mid H, c)$. The objective therefore compares compression policies under the reference law. Appendix C.3 analyzes the prober's report distribution for candidate memories. In interactive training, memory updates also change the visited histories and states; matched training experiments evaluate the resulting execution performance.

Mixture view of the reference belief. The Markov property $S \perp M \mid H, C = c$ yields

$$b_t^M(s \mid m, c) = \sum_h P_0(S = s \mid H = h, c)\, P_\theta(H = h \mid M = m, c). \tag{17}$$

Thus the reference belief mixes the full-history posteriors of histories that can produce the same memory. Compressing histories with different task-state implications into one summary merges their posterior information. Preserving supporting evidence and entity distinctions helps keep those implications distinguishable to subsequent reasoning.

C Information-Theoretic Justification for Belief Entropy

We distinguish the reference task state S, its task-relevant partition U, and the meaning Z of a model-generated anchor report. An exact decomposition separates report semantic uncertainty from expression variation; local stability conditions yield report-ranking and selection guarantees. A further comparison with ideal state-related reports specifies the conditions for transferring these guarantees to the state-information objective.

C.1 Response Distributions and Semantic Decomposition

Fix a comparison context $C = c$ containing the task query, turn, anchor question q, prober checkpoint, and response-distribution settings. This context specifies known task inputs and evaluation settings; the interaction history is represented by memory. Let $\mathcal{M}_c$ be a finite, nonempty set of candidate memories. For each $m \in \mathcal{M}_c$, let Q_m be the autoregressive prober distribution over complete responses Y, under a common finite vocabulary, output horizon, and termination rule. Sampling from Q_m defines the theoretical response distribution; greedy decoding selects the path used by the practical score in Eq. 5.

Fix a deterministic semantic map $g_c : \mathcal{Y} \to \mathcal{Z}$ throughout the comparison, forming an exhaustive finite partition of responses. Its classes identify responses with the same task-state account, including established progress and remaining information needs, while merging differences in wording. Thus $Z = g_c(Y)$ records response meaning on a shared semantic alphabet. Define, using natural logarithms,

$$F(m) = H_{Q_m}(Y), \qquad T(m) = H_{Q_m}(Z), \qquad W(m) = H_{Q_m}(Y \mid Z).$$

Here T is semantic uncertainty and W is the uncertainty of linguistic realization after meaning is fixed.

Proposition 2 (Semantic decomposition)**.** *For every candidate memory,*

$$F(m) = T(m) + W(m). \tag{18}$$

Proof. Determinism of g_c gives $H_{Q_m}(Z \mid Y) = 0$. Expanding $H_{Q_m}(Y, Z)$ in the two orders yields $H_{Q_m}(Y) = H_{Q_m}(Z) + H_{Q_m}(Y \mid Z)$. □

The decomposition identifies the stability relevant to BE: when linguistic-realization uncertainty varies little across candidate memories, changes in full-response entropy track changes in semantic uncertainty. The semantic partition specifies which distinctions matter for continued task execution. Semantic-entropy clustering in Appendix E.5 operationalizes this distinction between meanings and paraphrases.

C.2 Local Ranking and Memory Selection

Write $B(m) = \widehat{\mathcal{H}}_{\mathrm{BE}}(m)$ for the implemented score. Let $a_c > 0$ and b_c be shared constants, and define

$$\overline{B}(m) = \frac{B(m) - b_c}{a_c}, \qquad e_E(m) = \overline{B}(m) - F(m). \tag{19}$$

The common affine transformation expresses a comparison scale and leaves candidate rankings unchanged. It is an analytical device; training uses B directly.

Proposition 3 (Local semantic ranking and selection)**.** *Suppose that, for all* $m \in \mathcal{M}_c$*,*

$$|W(m) - w_c| \le \epsilon_W, \qquad |e_E(m)| \le \epsilon_E, \qquad \epsilon = \epsilon_W + \epsilon_E. \tag{20}$$

For $\widehat{m} \in \arg\min_{m \in \mathcal{M}_c} B(m)$*,*

$$T(m_2) - T(m_1) > 2\epsilon \implies B(m_1) < B(m_2), \tag{21}$$

$$T(\widehat{m}) - \min_{m \in \mathcal{M}_c} T(m) \le 2\epsilon. \tag{22}$$

Proof. Proposition 2 implies $\overline{B}(m) = T(m) + w_c + \eta(m)$, with $\eta(m) = W(m) - w_c + e_E(m)$ and $|\eta(m)| \le \epsilon$. Consequently, $\overline{B}(m_2) - \overline{B}(m_1) \ge T(m_2) - T(m_1) - 2\epsilon$, proving the first claim. For $m^\star \in \arg\min_m T(m)$, optimality of $\widehat{m}$ gives

$$T(\widehat{m}) + w_c - \epsilon \le \overline{B}(\widehat{m}) \le \overline{B}(m^\star) \le T(m^\star) + w_c + \epsilon.$$

Subtracting the common terms proves the second claim. □

The result gives both a ranking guarantee and a selection guarantee on the specified candidate set: BE preserves report semantic-entropy gaps exceeding $2(\epsilon_W + \epsilon_E)$, and its minimum-score candidate has report semantic entropy within the same bound of the candidate minimum. The bounds in Eq. 20 control the largest expression and estimation deviations across this set. Appendix C.3 extends both guarantees to task-state uncertainty under explicit report-coverage and calibration conditions; Appendix C.4 separates the score-approximation terms. The empirical validation in Appendix E.5 measures pairwise agreement with semantic entropy, responses to matched memory interventions, and the downstream utility of the selected memory.

C.3 Connection to Task-State Uncertainty

Let P be a reference task-generating law over latent state S, memory M, and known context C, with $P(M = m \mid C = c) > 0$ for every candidate. Define a finite task-relevant state partition $U = f_C(S)$, fixed across candidate memories. The state variable U represents distinctions in the task, whereas $Z = g_c(Y)$ represents meanings expressed by the prober. At fixed c, the state posterior uncertainty is $J(m) = H_P(U \mid M = m, C = c)$. For fixed m, c, define an ideal semantic report $Z^\star_m = \psi_{m,c}(U)$ on the same alphabet $\mathcal{Z}$ as the prober report. Dependence of $\psi_{m,c}$ on memory allows the report to describe information needs relative to the current memory. Let $P^\star_m$ be the distribution of $Z^\star_m$ under $P(U \mid M = m, c)$, and let Q^Z_m be the prober's induced semantic distribution. Define

$$R(m) = H_P(U \mid Z^\star_m, M = m, c),$$
$$D(m) = H(Q^Z_m) - H(P^\star_m).$$

R measures state information left unresolved by the ideal report; D measures the entropy discrepancy between actual and ideal semantic reports. The map $\psi_{m,c}$ specifies which state distinctions an ideal report conveys. Its coverage and the agreement of Q^Z_m with $P^\star_m$ determine the connection between report uncertainty and state uncertainty.

Proposition 4 (State-report decomposition and calibration)**.** *For every candidate,*

$$F(m) = J(m) + W(m) - R(m) + D(m). \tag{23}$$

If $|\mathcal{Z}| = K \ge 2$ *and* $\mathrm{TV}(Q^Z_m, P^\star_m) \le \delta \le 1 - 1/K$ *for every* m*, then*

$$|D(m)| \le f_K(\delta) := h_2(\delta) + \delta \log(K - 1), \tag{24}$$

where $h_2(x) = -x \log x - (1 - x) \log(1 - x)$*. If, additionally, Eq. 20 holds and* $|R(m) - r_c| \le \epsilon_R$*, then*

$$\left|\overline{B}(m) - J(m) - (w_c - r_c)\right| \le \epsilon_{\text{state}}, \qquad \epsilon_{\text{state}} := \epsilon_W + \epsilon_E + \epsilon_R + f_K(\delta). \tag{25}$$

Equations 21–22 consequently hold with T, ϵ *replaced by* $J, \epsilon_{\text{state}}$*.*

Proof. Because $Z_m^\star$ is deterministic given U, m, c, the entropy chain rule gives $H(P_m^\star) = J(m) - R(m)$. Substituting $T(m) = H(P_m^\star) + D(m)$ into Eq. 18 proves Eq. 23.

For the continuity bound, write $d = \mathrm{TV}(Q_m^Z, P_m^\star)$. Couple random variables A, B with these marginal distributions so that $\Pr(A \neq B) = d$: place mass $\min\{Q_m^Z(z), P_m^\star(z)\}$ on each diagonal pair (z, z), then couple the disjoint residual masses. For $E = \mathbf{1}\{A \neq B\}$,

$$H(A \mid B) \leq H(E) + \Pr(E = 1)\log(K - 1) = f_K(d),$$

since A is determined by B when $E = 0$ and has at most $K - 1$ possible values when $E = 1$. The same bound holds for $H(B \mid A)$. Using $H(A) - H(B) = H(A \mid B) - H(B \mid A)$ gives $|H(A) - H(B)| \leq f_K(d)$. The function f_K is increasing on $[0, 1 - 1/K]$, hence $f_K(d) \leq f_K(\delta)$. Finally, substitute Eq. 23 into Eq. 19 and apply the triangle inequality. The ranking and selection conclusions follow by the proof of Proposition 3. □

The conditions distinguish variation in linguistic realization (ϵ_W), score approximation (ϵ_E), state information left unresolved by the ideal report (ϵ_R), and prober calibration on semantic report classes ($f_K(\delta)$). Averaging $J(m)$ over $P(M = m \mid C = c)$ gives $H_P(U \mid M, C = c)$. For a discrete latent state with finite entropy, its relation to the full state objective is exact:

$$\begin{aligned} H_P(S \mid M, C) &= H_P(U \mid M, C) + H_P(S \mid U, M, C), \\ H_P(S \mid M) &= H_P(S \mid M, C) + I_P(S; C \mid M). \end{aligned} \tag{26}$$

These follow from $U = f_C(S)$ and the definition of conditional mutual information. For finite S, taking $U = S$ removes the projection residual. At fixed context and with $P = P_\theta$ from Eq. 12, minimizing the average state target $H_{P_\theta}(S \mid M, C = c)$ is equivalent to minimizing Eq. 2, since $H_{P_0}(S \mid H, c)$ is fixed. For a coarser finite partition U, Eq. 26 identifies the remaining state uncertainty. The alphabet size K is the number of semantic report classes in $\mathcal{Z}$.

A definite report of unresolved uncertainty. Let $S = U$ be uniform on $\{0, 1\}$ and let $H = U$. A compression channel retains U with probability $1/2$ and otherwise outputs an erasure $m_?$, independently of U. The candidate memories are $m_0, m_1, m_?$. A truthful deterministic prober reports "the state is u" for m_u and "the state is unresolved" for $m_?$. These are three classes on a shared semantic alphabet. Choose $\psi_{m_u,c}$ to return the known-state report on its posterior support, and $\psi_{m_?,c}$ to return the unresolved report for both states. Then $Q_m^Z = P_m^\star$ and $T(m) = D(m) = 0$ for every candidate, while

$$J(m_0) = J(m_1) = R(m_0) = R(m_1) = 0, \qquad J(m_?) = R(m_?) = \log 2.$$

The prober accurately describes the available information in every case. Here R accounts for the state uncertainty expressed as unresolved, so a definite report can coexist with an uncertain state. Its candidate-to-candidate variation is therefore a substantive part of the state-ranking condition in Proposition 4.

Why combine progress and information gaps? Write the ideal report as $(Z_P^\star, Z_G^\star)$, with progress and gap components. Their residual state uncertainties obey

$$\begin{aligned} &H_P(U \mid Z_P^\star, M = m, c) - H_P(U \mid Z_P^\star, Z_G^\star, M = m, c) \\ &\quad = I_P(U; Z_G^\star \mid Z_P^\star, M = m, c) \geq 0. \end{aligned} \tag{27}$$

Thus an informative gap component resolves distinctions that progress alone leaves open, reducing $R(m)$. Exchanging the roles of the two components gives the information added by progress beyond a gap-only report. This identity characterizes the coverage of an ideal joint report at each memory. State ranking additionally uses the across-memory variation $|R(m) - r_c| \leq \epsilon_R$ in Proposition 4. On a finite candidate set, the smallest such uniform bound over shared centers r_c is half the range of $R(m)$. Coverage improvements can differ across memories, so this residual-range condition remains a separate part of the ranking analysis.

Each implemented anchor induces its own response distribution, semantic partition, and entropy scale. We evaluate the resulting design empirically in Table 4 and Appendix E.1: adding progress to the gap-only probe improves accuracy by 0.96 percentage point at 56K and improves accuracy at every evaluated context length. The ideal-report identity explains complementary coverage, while these comparisons establish the utility of the implemented progress-and-gap probe.

C.4 From Complete Responses to the Implemented Score

For the theoretical distribution with horizon $L_{\max}$, pad terminated responses with a deterministic absorbing symbol. Let $h_\ell(u) = H(Q_m(\cdot \mid Y_{<\ell} = u))$ under this representation. The token chain rule yields

$$F(m) = \sum_{\ell=1}^{L_{\max}} \mathbb{E}_{Y_{<\ell}\sim Q_m}[h_\ell(Y_{<\ell})]. \tag{28}$$

For the practical score, let $\mathcal{I}(m)$ index the positions evaluated along the greedy response, let $L(m) = |\mathcal{I}(m)| \geq 1$, and use $h_\ell^{(k)}$ for their renormalized top-k entropies, with $k = 50$. The score and its full-vocabulary counterpart are

$$B(m) = \frac{1}{L(m)} \sum_{\ell\in\mathcal{I}(m)} h_\ell^{(k)}(y^*_{<\ell}),$$
$$G(m) = \sum_{\ell\in\mathcal{I}(m)} h_\ell(y^*_{<\ell}).$$

Thus $G(m)$ sums full-vocabulary predictive entropies at precisely the positions used by $B(m)$. This representation isolates path selection and termination, vocabulary truncation, and length normalization.

At any scored prefix, let τ be the probability outside the top-k set, and, for $\tau > 0$, let h_{tail} be its renormalized entropy. Set the tail contribution τh_{tail} to zero when $\tau = 0$. The entropy of the indicator for set membership gives

$$\begin{aligned} h_\ell - h_\ell^{(k)} &= h_2(\tau) + \tau\big(h_{\text{tail}} - h_\ell^{(k)}\big), \\ |h_\ell - h_\ell^{(k)}| &\leq h_2(\tau) + \tau \log|\mathcal{V}|. \end{aligned} \tag{29}$$

The right side is zero when $\tau = 0$. Summing these bounds along the greedy path bounds $|L(m)B(m) - G(m)|$.

For a common reference length $L_c > 0$, choose $a_c = 1/L_c$ and $b_c = 0$ in Eq. 19. If, uniformly over candidates,

$$\begin{aligned} |G(m) - F(m)| &\leq \epsilon_{\text{path}}, \\ |L(m)B(m) - G(m)| &\leq \epsilon_{\text{vocab}}, \\ |L(m) - L_c| &\leq \epsilon_L, \end{aligned}$$

then $0 \leq B(m) \leq \log k$ gives the explicit sufficient bound

$$|L_c B(m) - F(m)| \leq \epsilon_{\text{path}} + \epsilon_{\text{vocab}} + \epsilon_L \log k. \tag{30}$$

Indeed, insert $L(m)B(m)$ and $G(m)$ between $L_cB(m)$ and $F(m)$ and apply the triangle inequality. The path term includes replacing the prefix expectation by greedy-prefix entropies and any termination contribution omitted by the scored positions; the vocabulary term is controlled by Eq. 29; the final term measures variation in scored length. This deterministic analysis specifies sufficient uniform control of ϵ_E for the actual scoring procedure. Appendix E.10 holds responses fixed to isolate vocabulary sensitivity; the path term concerns the additional comparison with the prefix expectation in Eq. 28. Appendix E.5 evaluates the implemented score through semantic rankings and downstream task performance.

C.5 Connection to the Local Reward

MMPO maps the score to $\Phi(m) = \sigma(-B(m))$. Strict monotonicity reverses the ordering of B, so the ranking and selection results transfer directly to maximizing clarity. At a fixed terminal reward and $\alpha > 0$, they also transfer to the prefix reward in Eq. 6. Shared same-depth normalization has a positive denominator and therefore preserves these prefix-reward comparisons.

Moreover, $|\sigma'(x)| \leq 1/4$. Under Proposition 3, define the semantic reference score $B_0(m) = a_c[T(m) + w_c] + b_c$. Then

$$|\Phi(m) - \sigma(-B_0(m))| \leq \frac{a_c\epsilon}{4}. \tag{31}$$

This bounds how expression and estimation variation affect the local clarity reward. The monotone transform preserves candidate order; its expectation depends on the distribution of scores across visited memories. Equation 2 averages posterior information loss under a fixed reference law, while Eq. 9 defines the clipped policy update using sampled rewards and turn-level advantages. Terminal outcomes supply the task-success component, and future-prefix aggregation assigns the resulting comparisons to recursive memory updates. Matched training and reward-construction controls evaluate the combined effect of this supervision and the MMPO update on final performance (Appendix E.4).

D Details of Compared Memory-Agent Frameworks

This appendix summarizes the two memory-agent frameworks used in our experiments and clarifies how MMPO is applied to them.

MemAgent Framework. MemAgent [42] formulates long-context reasoning as a recurrent memory-update process. At each turn, the model receives the task query, the previous memory, and the current context segment, and produces an updated memory summary. After processing all segments, the final answer is generated from the accumulated memory. This design enables a fixed context window to process inputs much longer than the model's native context length, but the memory policy is mainly optimized through final task outcomes. In our MemAgent-based experiments, MMPO keeps this recurrent memory workflow unchanged and adds Belief Entropy supervision to the intermediate memory summaries. Thus, the comparison evaluates whether dense belief-quality feedback improves recursive memory summarization beyond outcome-only memory RL.

MEM1 Framework. MEM1 [48] studies long-horizon agents that jointly maintain internal memory and perform task-directed reasoning. Instead of relying only on raw interaction history, the agent maintains a compact internal memory state across steps and uses it to support subsequent reasoning, querying, or environment interaction. This framework is evaluated in both multi-objective QA and WebShop-style interactive tasks, where the agent must preserve multiple information needs or action-relevant constraints over a long trajectory. In our MEM1-based experiments, MMPO keeps the memory workflow and task interaction format unchanged, while adding Belief Entropy as an intermediate reward for memory states. This tests whether the proposed supervision can improve memory optimization not only in recursive summarization, but also in broader memory-augmented agent workflows.

E More Results

E.1 Full Anchor Question Robustness Study

Table 5 reports the anchor-question robustness study across all evaluated RULER-HotpotQA context lengths with Qwen2.5-7B. We compare three anchor designs: a direct-answer probe, a gap-only probe, and the default progress+gap probe. The direct-answer probe underperforms Outcome Only on average, suggesting that directly rewarding answer confidence can encourage premature certainty when the memory is still incomplete. In contrast, the gap-only and progress+gap probes better target intermediate memory quality by exposing unresolved information needs. The progress+gap probe achieves the best average performance, indicating that jointly tracking task progress and missing information provides a more informative signal for memory optimization.

Table 5: Full anchor question robustness study on RULER-HotpotQA with Qwen2.5-7B. All values are accuracy (%). Outcome Only denotes the standard RLVR setting without Belief Entropy supervision.

Anchor Question	7K	14K	28K	56K	112K	224K	448K	896K	Avg.
Outcome Only	81.25	81.25	82.03	80.47	79.69	75.78	76.56	74.22	78.91
Direct-answer probe	80.10	79.80	79.20	78.17	78.60	75.10	75.90	74.80	77.71
Gap-only probe	81.90	82.10	80.80	82.02	82.20	78.70	77.30	78.40	80.43
Progress + gap	**82.38**	**82.81**	81.03	**82.98**	**82.81**	**79.56**	**78.12**	**79.69**	**81.17**

We use the following anchor prompts. The direct-answer probe asks: "Based on current memory, what is the answer to the question?" The gap-only probe asks: "Based on current memory, what key information is still needed to answer the question?" The progress+gap probe asks: "Based on current memory, what is our task progress and what information is still needed?" The stronger performance of the progress+gap probe suggests that Belief Entropy benefits from explicitly tracking both the current task state and the remaining uncertainty.

E.2 Comparison with Alternative Proxy Signals

To examine whether Belief Entropy captures task-relevant memory quality rather than generic model confidence, we compare it with several low-cost proxy signals. As shown in Table 6, Belief Entropy achieves the strongest correlation with task accuracy and yields the best downstream performance when used as the dense reward signal.

Table 6: Comparison of proxy signals as dense rewards on RULER-HotpotQA with Qwen2.5-7B at 56K context length. $|r|$ denotes the absolute Pearson correlation with task accuracy.

Proxy Signal	$\|r\|$ **with Accuracy**	**Accuracy (%)**
Selected-token NLL (no anchor q)	0.41	79.12
Direct-answer entropy	0.54	78.17
Memory length	0.22	79.84
Random-question entropy	0.31	80.05
Belief Entropy ($\widehat{\mathcal{H}}_{\mathrm{BE}}$)	**0.68**	**82.98**

Selected-token NLL without the anchor question serves as a generic confidence baseline, but it does not directly probe task-state uncertainty. Direct-answer entropy remains correlated with task accuracy, but it can encourage premature answer confidence before sufficient evidence is collected. Memory length and random-question entropy show weaker correlations. These results indicate that the anchor-question-based Belief Entropy is better aligned with intermediate memory quality than generic confidence or length-based heuristics.

E.3 Computational Overhead

MMPO generates one short autoregressive anchor response per memory state, capped at 64 tokens. Entropy is accumulated directly from the predictive logits produced during decoding. At 56K, the anchor contains 34.7 generated tokens on average, with a 95th percentile of 51 tokens. Relative to Outcome Only, anchor decoding and entropy computation add 10.6% wall-clock time and reward bookkeeping adds 1.3%, for 11.9% total training overhead. Peak GPU memory increases by 0.3%.

We profile Qwen2.5-7B at 28K, 56K, and 112K using identical hardware, model, batch configuration, precision, optimizer, and rollout configuration within each matched comparison. Each setting is measured in three repetitions, each with ten warm-up and thirty measured steps. Costs are normalized to Outcome Only at the same document length (Table 7).

Table 7: Matched training profiles. Anchor and bookkeeping columns are incremental costs; total time and peak memory are ratios to Outcome Only.

Length	**Anchor tokens mean/P95**	**Anchor + entropy**	**Bookkeeping**	**Total time**	**Peak memory**
28K	35.4/52	0.109×	0.013×	1.122×	1.004×
56K	34.7/51	0.106×	0.013×	1.119×	1.003×
112K	34.9/52	0.108×	0.012×	1.120×	1.003×

Across the three profiled lengths, total training overhead is 11.9–12.2%, with at most 0.4% additional peak memory. Standard inference performs the original memory-update and answer-generation workflow without invoking the entropy prober, so BE supervision adds no inference cost.

E.4 Dense-Reward and Reward-Construction Controls

Task-aware dense-reward baseline. We compare full MMPO with a task-aware progress-reward baseline using the same Qwen2.5-7B backbone, 56K setting, terminal verifier, rollout budget, group optimization, and training steps. The baseline uses a frozen rubric-based scorer that reads only the task query and current memory. Let $z_{k,j}$ be its first-token logit for score label $j \in \{0,1,2,3,4\}$ at memory update k. We renormalize over these five labels and compute their probability-weighted score:

$$p_{k,j} = \frac{\exp z_{k,j}}{\sum_{a=0}^{4} \exp z_{k,a}}, \qquad s_k = \sum_{j=0}^{4} j\, p_{k,j} \in [0,4].$$

With $g(s) = s/8$ and $\lambda = 0.5$, the prefix target is

$$R_{k,\mathrm{prog}} = r_{\mathrm{final}} + \lambda g(s_k) = r_{\mathrm{final}} + \frac{s_k}{16}.$$

The auxiliary reward lies in $[0, 0.25]$, matching the maximum amplitude of Original MMPO's BE bonus. This range matching concerns the reward bound; the two signals retain their own empirical distributions. Scorer outputs and rewards are detached. The baseline retains MMPO's same-depth normalization, future-prefix averaging, and memory-token-only policy and KL loss masks. It uses no gold answers, supporting-fact labels, future observations, additional supervision, or separately trained reward model.

The progress scorer remains fixed throughout training, while Original MMPO refreshes its actor-snapshot BE prober at the start of each rollout batch (Appendix F). Table 8 therefore compares the task-aware dense-reward baseline with the full MMPO configuration, including both the scoring signal and its update rule. The progress baseline improves accuracy from 80.47% to 81.60%; full MMPO reaches 82.98%, a further gain of 1.38 percentage points.

Table 8: Task-aware dense-reward baseline and full MMPO on RULER-HotpotQA with Qwen2.5-7B at 56K. The progress scorer is frozen; Original MMPO uses a BE prober refreshed each rollout batch. Costs are relative to Outcome Only.

Method	Acc. (%)	Extra labels / RM	Cost
Outcome Only	80.47	0 / No	1.000×
Frozen progress score	81.60	0 / No	1.084×
Original MMPO	82.98	0 / No	1.119×

Joint reward and correctness ordering. Equation 6 jointly rewards final task success and intermediate memory quality. Since $\mathcal{H}_{\mathrm{BE}} \geq 0$ and $\alpha = 0.5$, the local contribution satisfies $0 < \alpha\sigma(-\mathcal{H}_{\mathrm{BE}}) \leq 0.25$. With a binary verifier, incorrect trajectories therefore have raw prefix rewards in $(0, 0.25]$, while correct trajectories have rewards in $(1, 1.25]$. The same positive affine transformation is applied to all trajectories at a given depth, preserving this correctness ordering while using BE to differentiate their memory quality.

Absolute versus local potential-difference BE rewards. We compare supervision based on current memory clarity with supervision based on its change between consecutive updates. Both controls use the same frozen Qwen2.5-7B prober and solver configuration, with matched training settings. Let $m_0^{(i)}$ denote the initial memory, the literal string `No previous memory`, before any document chunk is processed, and let T_i be the number of valid memory updates in trajectory i. With $\Phi(m) = \sigma(-\widehat{\mathcal{H}}_{\mathrm{BE}}(m))$ and $\alpha = 0.5$, the two prefix targets are

$$\begin{aligned} R_{k,\mathrm{abs}}^{(i)} &= r_{\mathrm{final}}^{(i)} + \alpha\Phi(m_k^{(i)}), \\ R_{k,\Delta}^{(i)} &= r_{\mathrm{final}}^{(i)} + \alpha\big[\Phi(m_k^{(i)}) - \Phi(m_{k-1}^{(i)})\big], \qquad 1 \leq k \leq T_i. \end{aligned} \tag{32}$$

The difference is undiscounted ($\gamma = 1$). Each valid prefix receives the verifier reward from its completed trajectory. For each question, we first standardize targets across trajectories reaching the same depth, then average the resulting prefix advantages over future prefixes:

$$\widehat{A}_{k,\Delta}^{(i)} = \frac{R_{k,\Delta}^{(i)} - \mu_{k,\Delta}}{s_{k,\Delta} + \varepsilon_A}, \qquad A_{t,\Delta}^{(i)} = \frac{1}{T_i - t + 1} \sum_{k=t}^{T_i} \widehat{A}_{k,\Delta}^{(i)}.$$

Here $\mu_{k,\Delta}$ and $s_{k,\Delta}$ are computed over $\mathcal{I}_k = \{i : T_i \geq k\}$, exactly as in Eq. 7; the absolute-clarity control follows the same calculation. Each $A_t^{(i)}$ is assigned to the tokens generating $m_t^{(i)}$ in Eq. 9. Thus, local targets are normalized before future-prefix aggregation, without forming a reward-to-go from raw differences or introducing a critic.

For the local potential-difference control, we evaluate $\Phi(m_0)$ once per question using the same frozen Qwen2.5-7B prober and progress-and-gap anchor as for later memories, then cache and reuse it across all trajectories in that question group. The shared offset $-\alpha\Phi(m_0)$ cancels under same-depth centering at $k = 1$ and leaves the group standard deviation unchanged.

Processing the prescribed document length and generating the final answer completes the task. The final valid target retains the scored potential of m_{T_i} and uses the same local difference as every earlier update. No potential-reset correction or additional terminal prefix is appended to the targets, normalization, aggregation, or token loss. Both auxiliary terms have absolute magnitude at most 0.25; the difference term can be negative, so this common bound does not imply identical reward distributions. This experiment compares two local reward constructions within the same MMPO prefix estimator. All variants are evaluated across three training seeds on the fixed 500-question 56K held-out set.

Table 9: Absolute and local potential-difference BE rewards on the fixed 500-question 56K set. The two reward-construction controls use the same frozen prober and MMPO prefix estimator. Original MMPO refreshes its actor-snapshot prober at each rollout batch (Appendix F). Values are mean accuracy (%) $\pm$ sample SD across three training seeds.

Method	**Accuracy (%)**
Outcome Only	80.47 ± 0.31
Local potential-difference BE	81.67 ± 0.23
Absolute BE, same frozen prober	82.13 ± 0.31
Original MMPO	82.47 ± 0.42

The local potential-difference control improves over Outcome Only by 1.20 percentage points. With the prober and prefix estimator held fixed, absolute BE achieves approximately 0.47 percentage points higher accuracy, calculated from unrounded means. This comparison favors directly supervising current memory clarity under the evaluated training budget.

E.5 Reliability Validation of Belief Entropy

We validate BE through three controlled tests: *semantic ranking*, *content sensitivity*, and *downstream memory selection*. Semantic entropy supplies a meaning-level comparison; matched memory interventions separate changes in task content from changes in wording; independent continuations measure the task performance supported by the preferred memory. These tests evaluate the practical behavior of the score analyzed in Appendix C. Throughout this section, BE denotes the implemented score $\widehat{\mathcal{H}}_{\mathrm{BE}}(m)$.

Evaluation sets and matched comparisons. We evaluate memories in the Qwen2.5-7B RULER-HotpotQA setting. BE and semantic entropy use exactly the same frozen Qwen2.5-7B initialization checkpoint from before MMPO optimization. This checkpoint also serves as the common prober for all panels of Figure 3 and for the cross-method comparison of MMPO and MemAgent in Figure 5. BE scoring uses the same task-state anchor, greedy decoding, 64-token response cap, and renormalized top-50 predictive probabilities across candidates and methods. The semantic-entropy protocol uses sampled anchor responses from this same checkpoint, with its sampling settings fixed across candidate memories.

The natural-memory set contains 100 held-out questions, 200 question-turn groups, and 800 memories, with four sampled memories in each group. Their six pre-specified pairs yield 1,200 within-group comparisons. A separate intervention study uses 400 original memories and matched modifications. Semantic-entropy interventions are evaluated on a fixed 200-memory subset. Within-group comparison holds the question and turn fixed, aligning the empirical ranking problem with the candidate comparisons in Proposition 3.

Test 1: Semantic ranking. For each of the 800 natural memories, we sample ten anchor responses and cluster them by bidirectional entailment to compute semantic entropy (SE), following semantic-uncertainty evaluation [21, 10]. The clustering groups responses by their expressed meaning and provides an offline semantic diagnostic for the single-response BE score. We compute Spearman correlation between BE and SE within each question-turn group and average across the 200 groups. BE achieves mean within-group correlation 0.59 and agrees with the SE ranking in 71.6% of the 1,200 memory pairs (Table 10). These measurements establish that the inexpensive score tracks differences in meaning-level uncertainty across memories at the same stage of task execution.

On the fixed 200-memory intervention subset, supporting-fact deletion, entity/number substitution, and contradiction insertion increase SE by 0.184, 0.143, and 0.267 nats, respectively. Meaning-preserving paraphrases produce a mean absolute change of 0.036 nats. BE–SE corruption-direction agreement is 79.3%. The natural-memory rankings and matched interventions provide complementary evidence: BE tracks semantic uncertainty across naturally occurring candidates, and both scores respond to changes in the information preserved in memory.

Table 10: Semantic-entropy validation on 800 natural memories and a fixed 200-memory intervention subset. SE changes are measured in nats.

Diagnostic	**Result**
Mean within-group Spearman $\rho(\mathrm{BE}, \mathrm{SE})$	0.59
Pairwise BE–SE ranking agreement	71.6%
Mean ΔSE: deletion	$+0.184$
Mean ΔSE: entity/number substitution	$+0.143$
Mean ΔSE: contradiction	$+0.267$
Mean $\lvert\Delta\mathrm{SE}\rvert$: paraphrase	0.036
BE–SE corruption-direction agreement	79.3%

Test 2: Content sensitivity. For each of the 400 original memories, we construct variants by deleting a supporting fact, substituting an entity or number, inserting a contradiction, or paraphrasing while preserving meaning. We define $\Delta\mathrm{BE}$ and $\Delta\mathrm{F1}$ as the modified-memory score minus the original-memory score. Deletion, substitution, and contradiction increase BE by 0.076, 0.058, and 0.118, respectively, with all three confidence intervals above zero (Table 11). Pooled corruption detection reaches AUROC 0.811, and lower BE correlates with supporting-fact recall at $\rho = 0.56$ (Table 12).

Paraphrases produce a mean absolute BE change of 0.011, compared with the larger signed increases under each corruption type. Their mean absolute continuation-F1 change is 0.006, whereas corruption reduces continuation F1 by 0.119. This matched contrast links BE to task-relevant content: changing the wording while preserving the memory's meaning leaves both the score and its downstream utility stable, while altering supporting information changes both. The paraphrase intervention measures the stability of the complete scoring procedure under changes to memory wording; the content interventions test its response to information loss, factual substitutions, and internal inconsistency. Confidence intervals use bootstrap resampling over original memories, retaining all associated interventions.

Table 11: BE response to controlled memory interventions. Corruption entries report signed mean changes; the paraphrase entry reports a mean absolute change.

Intervention	**BE change**	**95% CI**
Supporting-fact deletion	$+0.076$	[0.064, 0.088]
Entity/number substitution	$+0.058$	[0.047, 0.069]
Contradiction insertion	$+0.118$	[0.104, 0.132]
Paraphrase, mean $\lvert\Delta\mathrm{BE}\rvert$	0.011	—

Test 3: Downstream memory selection. We return to the 800 natural memories and estimate each memory's downstream value using five independent continuations. Starting from each candidate memory, any subsequent memory updates are performed by the same frozen pre-MMPO Qwen2.5-7B

memory policy. A common frozen Qwen2.5-14B solver then generates the final answer. These two models, their prompts, and their decoding parameters remain fixed across all candidates; the five continuations vary only the sampling seed.

Let $V(m) = \mathbb{E}[\mathcal{R}_{\text{task}} \mid m, c]$, where $\mathcal{R}_{\text{task}}$ is the task evaluation score, c fixes the question and turn, and the expectation is over sampling by the fixed memory policy and solver. We estimate $V(m)$ by the mean score of the five continuations. We compute Spearman correlation between $-\text{BE}$ and estimated continuation value within each group, then average across groups. For every pre-specified pair, we select the lower-BE memory and compare its continuation value with that of the alternative; continuation-value ties receive half credit.

The resulting correlation is 0.47 (95% CI: $[0.37, 0.56]$), and lower BE selects the higher-value memory in 66.8% of the 1,200 pairs (95% CI: $[62.9\%, 70.5\%]$). Both confidence intervals use question-clustered bootstrap resampling, retaining the dependence among turns, memories, and candidate pairs from the same question. By evaluating the task value of the preferred memory, this experiment connects BE-based candidate selection to successful continuation.

Table 12: BE alignment with downstream memory utility. Continuation statistics use 800 memories in 200 question-turn groups; controlled-study statistics use 400 original memories and their interventions. CI denotes a 95% bootstrap confidence interval.

Diagnostic	**Estimate**	**95% CI**
$\rho(-\text{BE}, V)$, question-turn macro	0.47	[0.37, 0.56]
Higher-value memory selection	66.8%	[62.9%, 70.5%]
Corruption detection AUROC	0.811	[0.791, 0.830]
$\rho(-\text{BE}$, supporting-fact recall)	0.56	[0.50, 0.61]
Corruption Δcontinuation F1	-0.119	$[-0.142, -0.097]$

From memory diagnosis to optimization. Together, these experiments evaluate meaning-level rankings, sensitivity to task information, and the task performance supported by selected memories. Pairwise agreement measures ranking behavior within question-turn groups; intervention averages quantify responses to controlled content changes; independent continuations measure downstream utility. The token-set sensitivity study in Appendix E.10 separately tests vocabulary choices along fixed responses. These empirical quantities characterize the implemented score. The sufficient bounds in Propositions 3–4 concern uniform expression, estimation, residual-state, and report-distribution deviations over a specified candidate set.

The anchor and training comparisons evaluate how this diagnostic becomes useful supervision. At 56K, the progress+gap anchor reaches 82.98% accuracy, compared with 82.02% for gap-only and 78.17% for the direct-answer probe (Table 4). This supports probing the current task-state account and its unresolved information needs. The frozen progress-score baseline reaches 81.60%, while full MMPO with its rollout-snapshot BE prober reaches 82.98% (Table 8). This comparison evaluates the complete MMPO configuration against task-aware dense feedback; both the scoring signal and scorer update rule differ. With the prober and solver separated from the trainable memory policy, MMPO retains a mean gain of 2.20 percentage points across three seeds (Appendix E.9). These results complete the empirical connection from intermediate memory diagnosis to improvements in the learned memory policy.

E.6 Reliability across Training Seeds

We evaluate three independent Qwen2.5-7B training seeds at 56K on a fixed, expanded 500-question held-out set constructed from the HotpotQA validation split. Outcome Only and MMPO use the same evaluation questions for every seed. Table 13 reports each run, the mean, and the sample standard deviation (SD) across training seeds.

MMPO improves accuracy in all three runs, with a mean paired gain of 2.00 points. A question-clustered paired bootstrap over the fixed evaluation set, averaging the matched seed pairs within each replicate, gives a 95% CI of $[0.68, 3.31]$ for this gain. The across-seed SD measures training-run variation, while the bootstrap interval measures evaluation-set uncertainty. These results complement

Table 13: Three-seed reliability at 56K. Accuracy is in percent; paired gains are in percentage points.

Training seed	Outcome Only	MMPO	Paired gain
1	80.2	82.8	+2.6
2	80.8	82.0	+1.2
3	80.4	82.6	+2.2
Mean ± sample SD	80.47 ± 0.31	82.47 ± 0.42	$+2.00 \pm 0.72$

the main Qwen2.5 model-by-length grid, where MMPO improves 19 of 20 comparisons and achieves average gains of 3.14 and 3.12 points over 224K–3.5M on 7B and 14B, respectively.

E.7 Transfer across Tasks and Data Sources

Training and evaluation separation. Following MemAgent [42], Stage I trains on synthetic QA instances derived from the HotpotQA training split, and Stage II uses a mixture of DocQA-RL and Stage-I data. Held-out RULER-HotpotQA evaluation instances are synthesized from the HotpotQA validation split. Outcome Only and MMPO use the same training records and evaluation manifest, and their checkpoints remain fixed throughout evaluation. This setup measures held-out-instance and raw-document-length transfer. In the MEM1 setting, training uses two-objective compositions, while evaluation on held-out eight- and sixteen-objective compositions measures transfer to longer composition horizons. The additional evaluations below change task format and dataset source.

RULER synthetic tasks and SQuAD. We evaluate frozen Qwen2.5-7B checkpoints on the MemAgent transfer suite: ten synthetic RULER tasks [15], comprising eight needle-in-a-haystack variants, Variable Tracking, and Frequent Words Extraction, at 8K–512K document lengths, together with SQuAD-based QA at lengths up to 256K. Evaluation uses no task-specific fine-tuning. We first average over lengths within each task and then weight the eleven tasks equally. MMPO improves the overall macro score from 90.58 to 91.86, a 1.28-point gain with paired-bootstrap 95% CI $[0.62, 1.94]$. Both the synthetic-task macro and SQuAD component improve (Table 14).

Table 14: Transfer to RULER synthetic tasks and SQuAD with frozen Qwen2.5-7B checkpoints. The overall macro gives equal weight to each of the eleven tasks.

Evaluation component	Outcome Only	MMPO	Gain
Ten synthetic tasks, macro	91.80	93.00	+1.20
SQuAD QA	78.40	80.50	+2.10
Eleven-task macro	90.58	91.86	+1.28

LongBench summarization. To evaluate transfer from QA training to summarization, we test Qwen2.5-7B-Instruct-based MMPO on GovReport and QMSum under the LongBench-SUM protocol [42, 2]. Both datasets are outside the QA training mixture. We retain the recurrent memory workflow and report ROUGE-1, ROUGE-2, and ROUGE-L recall, together with their arithmetic mean. Table 15 includes the Qwen2.5-Instruct and RL-MemAgent reference results from MemAgent [42]. MMPO improves the GovReport average from 19.34 to 19.99 and the QMSum average from 31.27 to 32.33, demonstrating transfer of the learned memory policy to a different task format.

Table 15: LongBench-SUM transfer. R-1/2/L denote ROUGE recall scores; Avg is their arithmetic mean. Qwen2.5-Instruct and RL-MemAgent reference results follow MemAgent [42].

	GovReport				QMSum			
Method	R-1	R-2	R-L	Avg	R-1	R-2	R-L	Avg
Qwen2.5-Instruct-7B	30.91	11.68	15.20	19.26	46.64	12.01	28.33	28.99
RL-MemAgent-7B	30.28	12.37	15.37	19.34	48.49	14.41	30.91	31.27
MMPO-7B	31.48	12.29	16.20	19.99	49.72	15.16	32.10	32.33
MMPO gain	+1.20	−0.08	+0.83	+0.65	+1.23	+0.75	+1.19	+1.06

E.8 Generalization to Qwen3.5-9B

We extend the comparison to Qwen3.5-9B [32] using the same recurrent memory workflow. On a fixed, expanded 500-question 56K set constructed from the HotpotQA validation split, matched training data, rollout budget, memory limit, optimizer, and decoding give 84.2% accuracy for Outcome Only and 86.0% for MMPO, a 1.8-point gain.

Table 16 separately reports the full raw-document-length sweep from 7K to 3.5M. Each recurrent call processes a bounded document chunk and memory, while longer documents require more recursive updates. MMPO improves nine of the ten length settings, with gains of 2.06–3.16 points at 224K–3.5M. The results show that the memory supervision transfers to the newer backbone and remains effective over extended recursive horizons.

Table 16: Qwen3.5-9B accuracy (%) across raw document lengths. This length sweep is reported separately from the expanded 500-question 56K evaluation.

Document length	RL-MemAgent	MMPO	Gain
7K	87.50	88.28	+0.78
14K	87.50	86.92	−0.58
28K	85.94	86.71	+0.77
56K	84.38	85.94	+1.56
112K	82.81	85.13	+2.32
224K	81.25	84.38	+3.13
448K	80.47	82.81	+2.34
896K	78.91	80.97	+2.06
1.75M	77.34	80.19	+2.85
3.5M	75.78	78.94	+3.16

E.9 Decoupling the Memory Policy, Entropy Prober, and Solver

The memory policy, entropy prober, and answer solver are functional roles that can be instantiated by separate models. We evaluate a trainable Qwen2.5-7B memory policy with a separate frozen Qwen2.5-7B entropy prober and a frozen Qwen2.5-14B solver. Outcome Only and MMPO use the same solver, and both reward signals are detached so that only the memory policy is updated. The comparison matches training settings across three seeds and uses the same fixed 500-question 56K evaluation set.

Table 17: Separate memory-policy, prober, and solver models. Accuracy is in percent; gains are in percentage points. The final row reports across-seed mean ± sample SD.

Training seed	Outcome Only	MMPO	Paired gain
1	82.8	84.8	+2.0
2	83.4	85.6	+2.2
3	82.8	85.2	+2.4
Mean ± sample SD	83.00 ± 0.35	85.20 ± 0.40	$+2.20 \pm 0.20$

MMPO improves accuracy in every seed, reaching $85.20 \pm 0.40\%$ compared with $83.00 \pm 0.35\%$ for Outcome Only. The paired gain of 2.20 ± 0.20 points demonstrates that BE supervision improves the learned memory policy when evaluation and final solving are performed by separate frozen models.

E.10 Entropy Estimator and Token-Set Sensitivity

At each anchor decoding step, we select the top-50 predictive tokens and renormalize their probabilities before computing entropy. Writing $c_\ell = (m_t, q, y^*_{<\ell})$ for the conditioning context, the normalized distribution is

$$\tilde{\pi}_{\mathrm{LLM}}(v \mid c_\ell) = \frac{\pi_{\mathrm{LLM}}(v \mid c_\ell)}{\sum_{u \in \mathcal{V}_\ell} \pi_{\mathrm{LLM}}(u \mid c_\ell)}, \qquad v \in \mathcal{V}_\ell. \tag{33}$$

We average token entropies along one greedy anchor response as in Eq. 5. To isolate vocabulary truncation, the sensitivity analysis holds the response and scored positions fixed and recomputes entropy with top-50, top-100, top-$p = 0.95$, and full-vocabulary probabilities. All truncated sets use the same renormalization rule. This comparison varies the vocabulary term in Appendix C.4 while keeping the decoding path and length fixed.

Table 18: BE rank correlation with full-vocabulary entropy under fixed responses.

Candidate set	**Rank correlation**
Top-50 (default)	0.968
Top-100	0.984
Top-$p = 0.95$	0.972

All three vocabulary choices closely preserve the full-vocabulary ranking along the fixed responses (Table 18). Across these choices, the BE–task-value correlation changes by at most 0.021 and Best-of-5 accuracy by at most 0.4 percentage point. These results support top-50 scoring as a compact vocabulary approximation for memory supervision and selection.

E.11 Relationship to Entropy-Based Learning and Process Rewards

Uncertainty supervision for recursive memory. Entropy-based reasoning methods demonstrate that model confidence can provide supervision without labeled answers. RENT reinforces reasoning outputs with high model confidence [31], while entropy minimization has also been applied through fine-tuning, reinforcement learning, and inference-time distribution adjustment [1]. MMPO applies uncertainty supervision to the evolving memory that mediates later reasoning: the progress+gap anchor evaluates the task state represented by each recursive summary, and the resulting signal is combined with verifiable success and assigned to the responsible memory updates. The evaluation target and credit assignment are therefore specific to intermediate memory. At 56K, direct-answer entropy obtains 78.17% accuracy, while the progress+gap anchor obtains 82.98%, demonstrating the value of probing the information required for continued task execution.

Connection to process reward models. AgentPRM evaluates step-wise promise and progress and constructs training targets using temporal-difference estimation and generalized advantage estimation [39]. MMPO obtains its intermediate signal directly from one greedy anchor response, with no external labels or separately trained reward model. Same-depth comparison and future-aware aggregation connect this memory-specific signal to recursive policy updates. Appendix E.4 compares a frozen task-aware progress-reward baseline with full MMPO, whose BE prober is refreshed from the actor each rollout batch.

Connection to semantic uncertainty. Semantic entropy groups paraphrase-equivalent responses and measures predictive uncertainty over their meanings [21, 10]. Semantic Entropy Probes estimate this multi-sample signal from the hidden states of a single generation [20]. These methods establish semantic uncertainty as an effective tool for uncertainty estimation and hallucination detection. MMPO uses a short task-state response to evaluate recursive memory during training. The semantic-entropy comparison in Appendix E.5 connects this inexpensive estimator to meaning-level uncertainty, while continuation and intervention experiments connect it to memory utility.

F Implementation Details

Memory Generation. We follow the MemAgent recurrent memory workflow and prompt format. At each turn t, the model receives the task query, the previous memory m_{t-1}, and the current document chunk o_t, and generates an updated memory summary m_t. Before any document chunk is processed, m_0 is initialized to the literal string `No previous memory`. The first update uses the same template as later turns, with the question and first observation o_1 filling its problem and section fields, and `<memory>No previous memory</memory>` as its memory field. Following MemAgent, the context budget is allocated to a 1,024-token query, a 5,000-token document chunk, a 1,024-token memory, and a 1,024-token memory output, with the remaining tokens reserved for the chat template. The main experiments retain the shared memory-generation and answer-generation weights of the

baseline workflow. Appendix E.9 evaluates separate memory-policy, entropy-prober, and solver models.

Training Setup. We train the memory policy with the clipped actor objective in Eq. 9 using same-depth prefix normalization and future-aware advantage aggregation. We use AdamW with learning rate 1×10^{-6} for the policy and a constant learning-rate schedule with linear warm-up. The KL coefficient is set to 1×10^{-3}, and the entropy loss is disabled. The group size is set to $G = 16$. Each rollout batch contains 128 trajectories, corresponding to 8 prompts with 16 sampled trajectories per prompt. The Belief Entropy reward weight is set to $\alpha = 0.5$.

Normalization and numerical stability. For each question and depth k, Eq. 7 uses the population standard deviation of rewards over $\mathcal{I}_k = \{i : T_i \geq k\}$, with variance computed using divisor $|\mathcal{I}_k|$. We add $\varepsilon_A = 10^{-8}$ to the standard deviation in the denominator. For singleton or zero-variance groups, we explicitly set all normalized prefix advantages $\widehat{A}_k^{(i)}$ to zero. These depths contribute zero to future-prefix averaging and supply no reward-driven policy-gradient signal; each turn-level advantage still aggregates contributions from its other future prefixes. These numerical rules apply to the same-depth estimator in Original MMPO and the reward controls.

Infrastructure interruptions. Rollouts interrupted by infrastructure failures are discarded in full and excluded from reward normalization and policy optimization; their partial prefixes are not used for training. We resample whenever possible to restore the intended trajectory-group size and discard a question group if fewer than two valid trajectories remain. For retained groups, same-depth statistics are computed from valid trajectories reaching each depth, using the numerical rules above. Infrastructure interruptions are handled separately from normal task completion, including completion at the prescribed document-length limit described in Appendix E.4.

Prober snapshots and model roles. Original MMPO uses a snapshot of the actor at the start of each rollout batch as its BE prober. All trajectories, candidate memories, and depths in that batch are scored by the same snapshot, which remains fixed throughout the batch's policy updates. After those updates, the next rollout batch takes a new snapshot of the updated actor. The KL reference model is used exclusively for the KL penalty. Table 19 summarizes the scoring roles across controls. The frozen-prober absolute and local potential-difference controls share one fixed Qwen2.5-7B prober throughout training; their solver configuration matches Original MMPO. The decoupled experiment additionally uses an independent frozen Qwen2.5-14B solver (Appendix E.9).

Table 19: Scoring models and refresh rules. Each setting trains the memory actor through generated memory tokens; scoring produces detached scalar feedback.

Setting	Scoring model	Refresh rule
Original MMPO	Rollout-start actor snapshot	Once per rollout batch
Absolute BE control	Frozen Qwen2.5-7B prober	Fixed throughout training
Local potential-difference BE	Same frozen prober as absolute BE	Fixed throughout training
Progress-score baseline	Frozen rubric-based scorer	Fixed throughout training
Decoupled MMPO (Appendix E.9)	Independent frozen Qwen2.5-7B prober	Fixed throughout training

Gradient paths and token masks. Only newly generated memory tokens contribute to the clipped policy loss and KL penalty in Eq. 9. Query, observation, and previous-memory input tokens are fully masked. Final-answer and environment-action tokens are used to complete the task and obtain the verifier reward; anchor-response tokens are used to compute BE. These generated tokens are also excluded from both losses. BE and terminal verifier rewards are detached scalar feedback, so gradients enter the actor through the memory-token objective rather than through reward computation. In the shared-weight workflow, memory-policy updates also change the actor parameters used for answer generation, despite the answer tokens being masked. Likewise, the next batch's prober changes when its actor snapshot is refreshed; detaching the reward fixes the gradient boundary, while the snapshot schedule determines when the scoring model changes.

Fixed models for diagnostic evaluation. Figures 3 and 5 and the BE/semantic-entropy comparison in Appendix E.5 use the frozen Qwen2.5-7B checkpoint that initializes MMPO. In the continuation evaluation, the frozen pre-MMPO 7B memory policy performs subsequent memory updates and a common frozen 14B solver produces the final answer. The frozen 14B solver is used in this evaluation and the role-decoupling experiment in Appendix E.9; default training retains the shared actor weights for memory and answer generation described above.

Prompt Templates. The memory update prompt follows MemAgent's original format. At each turn, the model receives the task problem, the previous memory, and the current article section, and then updates the memory by retaining previous relevant details while incorporating new useful information:

```
You are presented with a problem, a section of an article that may contain the
answer to the problem, and a previous memory.  Please read the provided section
carefully and update the memory with the new information that helps to answer
the problem.  Be sure to retain all relevant details from the previous memory
while adding any new, useful information.
<problem>
{prompt}
</problem>
<memory>
{memory}
</memory>
<section>
{chunk}
</section>
Updated memory:
```

For QA tasks, Belief Entropy is computed using the following anchor-question template:

```
Based on the problem and current memory, what is the current task progress and
what information is still needed?
<problem>
{prompt}
</problem>
<memory>
{memory}
</memory>
Your assessment:
```

G Limitations

MMPO's controlled reliability studies focus on recursive text memory in long-document question answering, using a fixed progress-and-gap anchor. Tasks with changing objectives or multimodal observations may require different report formats; extending and evaluating the anchor design in these settings is a direction for future work.

H Impact Statement

This work aims to improve the reliability of long-horizon LLM agents by providing denser supervision for intermediate memory states. Better memory optimization may benefit applications that require sustained context tracking, such as long-document reasoning, multi-step question answering, interactive search, and task-oriented assistants.

However, MMPO does not guarantee factual correctness or safe autonomous behavior. Inaccurate memories, overconfident summaries, or poorly designed anchor questions may still lead to incorrect decisions, especially in long interactions or high-stakes settings. In addition, memory-based agents may process sensitive user information, so practical deployments should include appropriate privacy controls, retention policies, and user-facing transparency.

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 state the main contributions (Belief Entropy as a proxy, MMPO framework, and empirical improvements), all of which are supported by the theoretical analysis and experimental results in the main body.

   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: A dedicated Limitations section is included in the appendix, discussing assumptions, computational costs, and scope of evaluation.

   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: [Yes]

Justification: All theoretical results (propositions and bounds) are stated with explicit assumptions in the main body, with complete proofs provided in the appendix.

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: The paper provides model architectures, training hyperparameters, dataset descriptions, and evaluation protocols in the main body and appendix. Implementation details are described sufficiently for reproduction.

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: The code repository is publicly available.

   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: Training hyperparameters and evaluation protocols are provided in the experimental setup and implementation details 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: [Yes]

   Justification: Selected experiments report results across three training seeds and bootstrap confidence intervals.

   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: [No]

   Justification: Hardware configurations and absolute training times are withheld due to company confidentiality restrictions.

   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: This research conforms with the NeurIPS Code of Ethics. No human subjects or sensitive data are involved.

   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: An Impact Statement is included in the appendix discussing both positive applications and potential risks of improved autonomous agent capabilities.

    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: The paper proposes a training methodology rather than releasing a pre-trained model or dataset with high misuse risk.

    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: All datasets and models used are properly cited. The benchmarks used are publicly available for research purposes.

    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: [N/A]

    Justification: The paper does not introduce new datasets or pre-trained models as standalone released assets.

    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: [N/A]

    Justification: This work does not involve crowdsourcing or research with human subjects.

    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: This work does not involve human subjects research.

    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: LLMs were used only for writing and editing assistance, which does not impact the core methodology or originality of this research.

    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.