HELM: Steering Long-Horizon Agents with Learned Hierarchical Memory and Epistemic Governance

Anonymous ACL submission

Abstract

Long-horizon LLM agents must carry state across many tool interactions, yet naïve context extension via periodic summarization or top-k retrieval can discard decisive evidence and makes failures hard to audit. We introduce **HELM** (**H**ierarchical **E**pistemic **L**earned **M**emory), a framework that exposes memory as an explicit, event-driven interface and couples memory access with *epistemic governance*. HELM instantiates a three-tier nested store, **SHNM**, that links episodic traces to consolidated recalls and thematic indices via provenance edges and epistemic metadata (timestamps, source types, tool status). Governance makes memory operations reproducible: retrieval is re-ranked with recency/status-aware scoring and conflict resolution prefers verified, newer evidence, while provenance expansion can trace any recall back to concrete tool spans. On top of SHNM, a learned controller decides when to read, write, consolidate, and prune under task and efficiency budgets, and a tool-aware embedding model indexes tool-augmented trajectories to improve retrieval of procedural and trace-based memories. We evaluate on five long-horizon benchmarks and report diagnostics that jointly measure end-task performance, memory efficiency, and epistemic reliability, including auditable recall metrics that quantify provenance faithfulness.

1 Introduction

Large Language Models (LLMs) are evolving from passive chatbots into autonomous agents capable of pursuing open-ended goals over extended horizons (Yao et al., 2023; Hu et al., 2025). In this regime, an agent must carry forward a complex state—comprising user constraints, intermediate reasoning, and tool-grounded evidence—across hundreds of interaction steps, a volume that frequently exceeds the effective context window.

While recent works have attempted to mitigate this via naive context extension, periodic summarization, or top-k retrieval (Packer et al., 2023; Zhong et al., 2024), these approaches fundamentally misdiagnose the problem. They treat long-horizon memory as a *capacity* challenge, whereas we argue it is primarily an *epistemic* one.

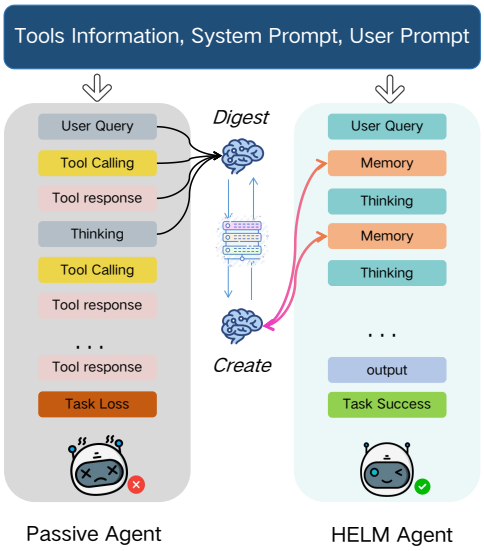


Figure 1: Nested memory intuition: HELM interleaves explicit memory write/recall, enabling coarse-to-fine access from high-level indices to grounded tool traces.

The dominant failure mode in long-horizon agents is not the absence of information, but the high-confidence utilization of *invalid evidence*. Standard retrieval-augmented generation (RAG) retrieves memories based on semantic similarity, blindly surfacing stale observations (e.g., a flight price from 100 steps ago) or failed tool execution logs (e.g., an error trace that matches the query semantics) (Quattrociocchi et al., 2025). Conversely, abstractive summarization sacrifices fidelity for brevity, often discarding decisive constraints (e.g., "pending payment") that are critical for verification. Consequently, agents frequently "hallucinate" successful outcomes based on retrieved but epistemically void traces, creating failures that are difficult

to audit post-hoc.

To bridge this gap, we propose **HELM** (**H**ierarchical **E**pistemic **L**earned **M**emory), a framework that actively steers long-horizon agent reasoning through a nested memory architecture with learned dynamics. The core of HELM is the **Semantic-HNSW Nested Memory (SHNM)**, organized into three layers—episodic traces ($M^{(1)}$), consolidated recalls ($M^{(2)}$), and thematic indices ($M^{(3)}$)—linked by explicit provenance and epistemic metadata (timestamps, source types, tool status) to preserve epistemic integrity. Rather than treating memory as passive storage, HELM uses **epistemic governance** to filter and re-rank memory access with provenance/recency/status signals and to propagate validity through consolidation and pruning (Section 3). On top of SHNM, a learned controller decides when to READ, WRITE, CONSOLIDATE, and PRUNE under explicit budgets, and we fine-tune a tool-aware embedding model to index tool-augmented traces and construct epistemic hard negatives (Section 3.4).

We validate HELM on five long-horizon benchmarks and report end-task performance alongside memory efficiency (CTU), epistemic errors (EpiErr.), and provenance-faithfulness diagnostics. Under matched budgets and scaffolds, HELM improves success on GAIA (L2: 8.9→12.8; L3: 0.8→1.7) and WebArena (19.5→22.4) compared to strong memory baselines, while enabling auditable recall traces for debugging and faithfulness evaluation. Our contributions are threefold: (i) we formulate long-horizon memory as an *epistemic* problem and introduce HELM, coupling governed retrieval with provenance-backed grounding for auditability; (ii) we present SHNM with explicit memory operations and a learned controller, and ablations disentangle which components matter most (with governance and grounding as the primary reliability drivers); and (iii) we evaluate across five benchmarks with process-level diagnostics (CTU/EpiErr.) and faithfulness controls to separate utility from auditability (Sections 4 and 5). Beyond reranking with time/status features, HELM propagates validity through consolidation/pruning and ties recalls to provenance pointers.

2 Related Work

2.1 Agent-Memory Architectures

Agent memory spans token-level episodic stores, latent memories, and parametric edits (Hu et al., 2025). Many deployed systems use **flat retrieval** over interaction logs, e.g., MemGPT and MemoryBank (Packer et al., 2023; Zhong et al., 2024). To improve access under long horizons, recent work organizes memory with graphs or hierarchies (Mem0, GraphRAG, HippoRAG, H-Mem) (Chhikara et al., 2025; Edge et al., 2025; Gutiérrez et al., 2025; Sun and Zeng, 2025). HELM extends this line with a three-tier *nested* store that explicitly links high-level indices to supporting recalls and verifiable traces, enabling coarse-to-fine retrieval with provenance.

2.2 Learned Memory Dynamics

Beyond structure, agents must decide when to write, retrieve, consolidate, and forget. Many systems use hand-crafted heuristics (recency/importance scores or periodic summaries) (Park et al., 2023; Li et al., 2025), which struggle with the stability–plasticity trade-off at long horizons. Recent work explores learning-based control, including RL for retrieval ranking or compression (Tan et al., 2025; Zhang et al., 2025e) and latent-memory systems such as MemGen (Zhang et al., 2025b). In contrast, HELM learns an explicit controller over interpretable memory operations on token-level memory, while keeping recalled content auditable via cross-layer provenance.

2.3 Epistemic Reliability and Trustworthiness

In high-stakes settings, memory must be epistemically reliable because it determines what evidence an agent can recover and justify. Quattrociocchi et al. (2025) argue that apparent "judgment" in LLMs can be a linguistic illusion when generation is not grounded in verifiable traces. Standard retrieval-augmented generation (RAG) grounds responses in retrieved documents (Gao et al., 2024), yet typically treats retrieved content as uniformly valid, ignoring provenance, staleness, and tool-execution status. We discuss privacy and robustness implications of agent memory in the Limitations section, but epistemic confidence is often implicit in current systems. HELM makes epistemic state first-class and uses *epistemic governance* to filter, promote, and ground memories, reducing the risk of reusing stale or invalid traces over long horizons.

3 Methodology

We introduce **HELM**, a hierarchical memory framework that makes long-horizon agent mem-

ory *structured, auditable, and controllable* (Figure 2). HELM combines (i) a three-tier nested store, **SHNM**, that links abstractions to verifiable traces via provenance and epistemic metadata; (ii) *epistemic governance* that filters and re-ranks memory access using recency/status/provenance signals; and (iii) a learned controller that triggers explicit memory operations under fixed budgets. We additionally fine-tune a tool-aware embedding model to index tool-augmented traces for robust retrieval.

3.1 Problem Setup and Explicit Recall

Agent and environment. We consider an LLM-based agent interacting with an environment over steps $t = 1, \ldots, T$. At step t, the agent observes o_t (user input, tool outputs, or state), optionally consults an external memory store $\mathcal{M}$, and produces an action a_t (response or tool call). HELM separates *reasoning* from *memory management*: a reasoner generates task actions, while a controller decides if/when to interact with $\mathcal{M}$. The working context is $c_t = \mathrm{Compose}(o_t, \mathrm{Read}(\mathcal{M}, q_t))$, where q_t is a controller-generated recall query; the controller can also issue WRITE, CONSOLIDATE, and PRUNE operations to update $\mathcal{M}$.

Event-driven recall interface. Memory use in HELM is explicit. During deliberation, the reasoner emits a request span `<recall>...</recall>` describing missing information (e.g., a constraint or a pointer to a past tool result). A separate **memory generator** executes governed retrieval in SHNM, optionally expands provenance to supporting evidence, and replaces the request with a `<memory>...</memory>` block that can be audited against stored traces. For long tool transcripts, we additionally store `<memory>refined log</memory>` entries that extract salient fields while preserving pointers to the original tool trace for later verification. Appendix A.7 specifies the pointer schema used for auditable recall, and Appendix A.8 details the cited-recall validation used in our faithfulness experiments.

Explicit memory operations. HELM exposes memory as a small set of explicit, logged operations: READ (governed retrieval under token budgets), WRITE (store episodic traces with tool status and timestamps), CONSOLIDATE (distill and link abstractions to evidence via provenance), and PRUNE (enforce budgets by removing stale/failed/redundant items). Appendix A.1 provides the full operation summary.

3.2 Semantic-HNSW Nested Memory (SHNM)

We structure the memory store as a three-tier hierarchy $\mathcal{M} = \{M^{(1)}, M^{(2)}, M^{(3)}\}$, each layer representing a different abstraction level. We build an HNSW-style ANN index (Malkov and Yashunin, 2016) per layer and connect layers with provenance links, enabling coarse-to-fine traversal.

Layers. SHNM consists of $M^{(1)}$ **episodic traces** (high-fidelity interaction snippets and tool logs as verifiable evidence), $M^{(2)}$ **consolidated recalls** (procedural/semantic summaries distilled from linked traces), and $M^{(3)}$ **thematic indices** (sparse, high-level entries that act as global retrieval entry points).

Memory items, epistemic state, and provenance. Each memory item is $\Updownarrow = (x, \mathbf{v}, \mathcal{E}, \mathcal{P})$, where x is text, $\mathbf{v}$ is an embedding, $\mathcal{P}$ stores provenance links (including cross-layer parent/child relations), and $\mathcal{E}$ stores epistemic metadata such as timestamps, tool-execution status, and episode/version identifiers. We treat an item as *stale* when it exceeds a step/time window or mismatches the current task version (Appendix A.6).

Coarse-to-fine retrieval with epistemic governance. Given a recall query q with embedding $\mathbf{v}_q$, SHNM first retrieves candidate indices from $M^{(3)}$, expands to linked recalls in $M^{(2)}$, and finally selects supporting traces in $M^{(1)}$ for grounding. We then apply *epistemic governance* to exclude invalid evidence and to prioritize candidates that are both semantically relevant and epistemically reliable. Candidates are re-ranked by a governed score

$$S(\Updownarrow; q) = \lambda_{\text{sim}} \mathrm{sim}(\mathbf{v}_q, \mathbf{v}) + \lambda_{\text{time}}\, g(\Delta t) + \lambda_{\text{status}}\, w(\texttt{status}) + \lambda_{\text{src}}\, w(\texttt{src}), \tag{1}$$

where sim is cosine similarity, $g(\cdot)$ is a monotone recency kernel, and `src` denotes the source type. READ applies hard filters to exclude failed tool traces (and optionally stale items); when evidence is scarce, the filter relaxes into soft downweighting.

Governance signals and fixed weights. We instantiate status and recency as governance signals that are logged and auditable: $\texttt{status} \in \{\text{SUCCESS}, \text{UNKNOWN}, \text{FAILED}\}$ comes from tool wrappers, and Δt is computed from timestamps/step IDs. In all experiments we fix λ_{sim}=1.0

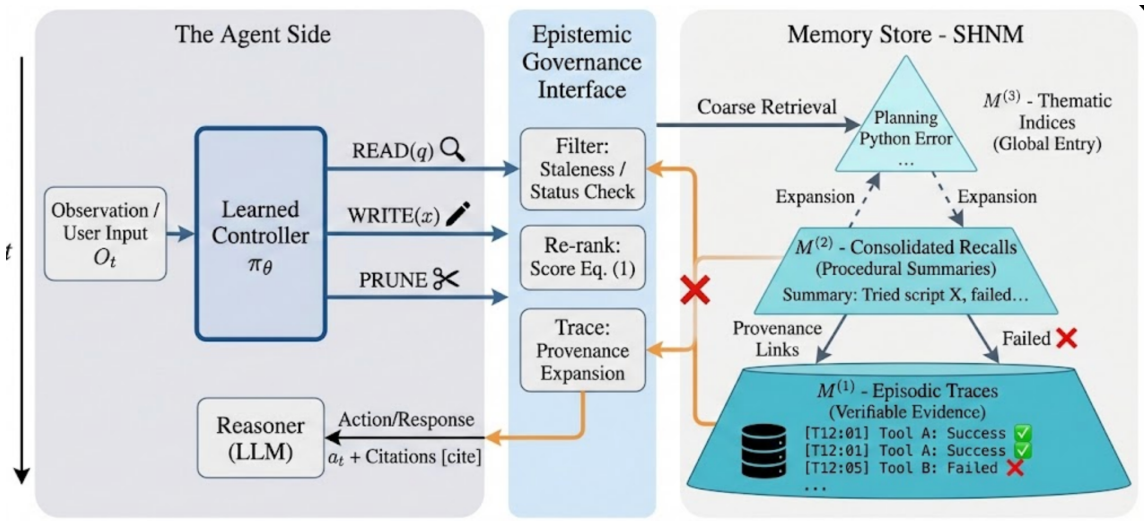


Figure 2: Overview of HELM. Instead of retrieving isolated chunks from a flat store, HELM maintains a three-level hierarchy (episodic traces → consolidated recalls → thematic indices). For each query, the system performs coarse-to-fine retrieval by first selecting high-level indices and then expanding to supporting recalls and verifiable traces, preserving both global task context and grounded evidence.

and use a strong penalty λ_{status}=10.0 so failed traces are excluded (or decisively downweighted) unless no alternative evidence exists. We use λ_{time}=1.0 and λ_{src}=1.0 as default scales; all hyperparameters and budgets are summarized in Appendix A.10. Appendix A.12 provides pseudocode for retrieval, provenance expansion, consolidation, and pruning.

Consolidation and pruning. CONSOLIDATE promotes multi-step evidence from $M^{(1)}$ to $M^{(2)}$ while propagating provenance and resolving conflicts via an epistemic preference order (prefer SUCCESS over UNKNOWN over FAILED, newer over older, trace-supported over ungrounded). We mark two items as **contradictory** when they refer to the same extracted key (e.g., an entity attribute or tool-return field) but disagree on value; consolidation keeps the newer verified value and records the older value as a conflict for downstream governance. PRUNE enforces budgets by removing stale/failed items and demoting redundant entries.

3.3 Learning the Memory Controller

We train a controller that outputs explicit memory operations and recall queries q_t. Training is decoupled from the underlying reasoning model, so we can preserve the reasoner's capabilities while learning specialized memory policies (Lu and Lab, 2025).

Stage 1: On-policy distillation. We sample memory-operation rollouts from the current controller and use a stronger teacher to provide dense per-token supervision on student-visited states via a reverse-KL-style objective (Lu and Lab, 2025; Agarwal et al., 2024) (Appendix A.17).

Stage 2: Strategy exploration. We apply **Group Sequence Policy Optimization (GSPO)** (Zheng et al., 2025a) to maximize a **hybrid reward** $\mathcal{R}$:

$$\mathcal{R} = \alpha\, r_{\text{task}} + \beta\, r_{\text{mem}} \tag{2}$$

where r_{task} is the benchmark task metric and r_{mem} rewards low effective context tokens while penalizing redundant writes and epistemically risky memory access (e.g., promoting stale/failed traces).

We also fine-tune a tool-aware embedding model for SHNM indexing (Section 3.4). Training data uses synthesized hierarchical triplets and epistemic hard negatives (Appendix A.16).

3.4 Tool-Aware Embedding Optimization

Retrieval quality hinges on embeddings that represent both content and tool structure. We fine-tune `Qwen3-Embedding-4B` (Zhang et al., 2025d) to index tool-augmented traces for SHNM.

Architecture Adaptation. We remove the causal mask during fine-tuning to allow bidirectional attention over tool invocations and results, then pool hidden states into a representation $\mathbf{v} \in \mathbf{R}^d$.

Instruction-Aware Contrastive Tuning. We optimize an instruction-conditioned contrastive objective (InfoNCE), where the instruction specifies the

desired retrieval granularity (e.g., evidence span vs. high-level plan) and negatives include *epistemic* counterfactuals that match topic but violate validity conditions (e.g., tool status or timestamps). We provide the full objective and recipe in Appendix A.18.

4 Experiments

4.1 Experimental Setup

Benchmarks. We evaluate HELM on five benchmarks that require long-horizon execution, state tracking, and/or learning from experience: **GAIA** (Mialon et al., 2023) (real-world assistant questions; Level 2–3 stress multi-step tool use), **WebArena** (Zhou et al., 2024) (realistic web interaction with functional validators), **Mind2Web** (Deng et al., 2023) (web generalization with step-level supervision and diagnostics), **StreamBench** (Wu et al., 2024) (online input–feedback streams for continuous improvement), and **LifelongAgentBench** (Zheng et al., 2025b) (lifelong learning under strict sequential execution with automatic label verification). To isolate *memory access quality* from end-task success, we also evaluate retrieval on BRIGHT and run an auxiliary QA sanity check with retrieved context on MMLU and GPQA.

Baselines. We compare against a broad set of representative agent-memory paradigms: (i) **long-context agents** without external memory, (ii) **flat episodic memory (RAG)** with top-k similarity retrieval over interaction logs, (iii) **agent memory frameworks** such as MemGPT (Packer et al., 2023), Mem0 (Chhikara et al., 2025), HippoRAG (Gutiérrez et al., 2025), and GraphRAG (Edge et al., 2025), and (iv) **structured/latent memory** baselines such as MemGen (Zhang et al., 2025c) and G-Memory (Zhang et al., 2025a).

Metrics. We report benchmark-standard **task success** (or accuracy) as the primary outcome metric. To capture process-level effects of memory, we additionally report **effective context tokens** (**CTU**; total input-context tokens consumed across an episode, including retrieved memories and tool outputs) and **epistemic errors** (**EpiErr.**). We measure EpiErr. following the failure taxonomy in §5.2: cases where the agent uses semantically relevant but epistemically invalid memories (e.g., stale timestamps, failed tool status, or contradictory traces).

Table 1: **Faithfulness controls on GAIA and WebArena.** A1 largely preserves SR while enabling auditable recall; A2 maximizes literal faithfulness but degrades success on reasoning-heavy tasks. **Variants**: A0 = baseline recall, A1 = hard cited recall, A2 = extractive-first recall. Cov: citation coverage; C-Valid: citation validity; C-P: citation precision under a sampled judge protocol.

Var.	GAIA (SR %)		WebArena (SR %)	CTU ↓	EpiErr. ↓	Cov ↑	C-Valid ↑	C-P ↑
	L2 ↑	L3 ↑	SR ↑					
A0	**12.8**	**1.7**	**22.4**	**2050**	6.3	N/A	N/A	N/A
A1	12.4	1.6	21.9	2280	**4.2**	**96.5**	**94.2**	**89.5**
A2	10.5	0.8	19.5	2450	2.1	100.0	99.1	96.0

Table 2: Controller ablation.

Controller	Score ↑	Tokens ↓	Write Rate ↓
HeuristicRW	48.0	2350	1.00
DistillOnly	49.1	2200	0.82
Distill+GSPO	**50.8**	**2050**	**0.67**

EpiErr. implementation and verifiability. We compute EpiErr. directly from logged memory access traces: each recalled item carries timestamps, tool-execution status, task/version identifiers, and provenance pointers to episodic spans. We extract lightweight (key, value) claims using tool-type-specific parsers for structured outputs and a constrained regex extractor for free-form text; contradictory is flagged when claims share a key but disagree on value, keeping the newest SUCCESS-supported claim (Appendix A.6). We also report **provenance faithfulness**: whether recalled evidence can be grounded via provenance links to an episodic trace in $M^{(1)}$. We summarize these diagnostics and ablations in Section 5. For Mind2Web, we follow Deng et al. (2023) and report **Step SR** and end-task **SR** across three generalization splits.

Evaluation protocol and fairness. Unless otherwise stated, we follow each benchmark's official evaluation interface (e.g., WebArena functional validators; LifelongAgentBench automatic label verification). All methods share the same agent scaffold and tool interfaces; we vary only the memory system and its controller. Memory-enabled agents invoke recall via `<recall>...</recall>` (Section 3) and we log retrieved items with provenance and epistemic metadata, enabling auditable diagnostics. To make comparisons attributable to memory rather than confounds, we lock the reasoner backbone/decoding, tool wrappers, and memory budgets (entries per layer, retrieved items per

query, and maximum injected recall tokens). Crucially, all baselines are evaluated under **matched token budgets** (Appendix Table 11) to ensure fair comparison. Baselines are evaluated in the same harness under the same budgets whenever feasible; numbers quoted from benchmark papers are included only for context when scaffolds/tools differ.

Data isolation and contamination audit. Tool-based benchmarks (notably WebArena/Mind2Web) are vulnerable to leakage if training trajectories overlap with evaluation domains, URLs, or instruction templates. We therefore enforce (and report) an auditable isolation checklist: (i) benchmark-domain filtering for web-derived trajectories (canonicalized URL/domain patterns), (ii) temporal isolation via snapshot cutoffs where available, and (iii) near-duplicate detection over prompts/instructions. In our training runs, we filtered $\sim$15% of trajectories due to domain overlap; after filtering, URL overlap with evaluation environments is 0.0%. An n-gram audit finds no overlaps longer than 50 tokens between training prompts and evaluation instructions.

Statistical reporting. Unless otherwise noted, we report results over 3 seeds (0, 42, 123). For key comparisons we compute nonparametric bootstrap confidence intervals from episode-level logs; on WebArena we observe HELM 22.4$\pm$1.2% vs. the strongest baseline 19.5$\pm$1.4% (95% CI; $p < 0.05$), and CTU reductions are significant ($p < 0.01$).

Scaling protocol. We stress-test robustness by scaling horizon length (injecting distractor steps/tool logs), memory-store size, and recall-query verbosity, and report scaling curves over success, CTU, and latency.

5 Main Results

We evaluate HELM across five long-horizon benchmarks and report end-task success alongside memory efficiency (CTU) and reliability (EpiErr.). Table 3 summarizes GAIA/WebArena, while Figures 3–4 summarize Mind2Web/StreamBench/LifelongAgentBench and the cost–performance trade-off.

5.1 Component Analysis: Governance Drives Reliability

Under matched scaffolds and token budgets, HELM improves end-task success on GAIA/WebArena (Table 3) while reducing cost, yielding a better trade-off curve (Figure 4). Crucially, the gains are not purchased by longer prompts: compared to flat episodic memory (RAG), HELM nearly halves CTU (4,200→2,050) while also reducing epistemic errors (12.8→6.3), supporting that improvements come from using *more valid evidence* rather than more context.

To isolate which components drive this Pareto shift, Table 4 reports controlled ablations under identical scaffolds and budgets. Removing governance produces the sharpest reliability collapse: EpiErr. more than doubles (6.3→14.2) even though the hierarchy, embeddings, and budgets are unchanged, and success drops accordingly. This pinpoints a core failure mode of similarity-only memory: semantically relevant but *epistemically invalid* traces (stale timestamps, failed tool runs, contradictory states) are promoted and then treated as evidence.

Appendix Figure 5 illustrates two recurring traps: (i) a stale-memory trap where an old but semantically matching value leads to an incorrect tool action, and (ii) a failed-trace trap where an execution error is reused as if it were a successful outcome. Governance explicitly uses timestamps and tool-execution status to filter these traces and, when necessary, trigger fresh evidence acquisition.

Table 5 deconstructs the nested store. Using only $M^{(1)}$ (flat episodic) is both expensive and error-prone; removing $M^{(2)}$ weakens reusable procedure compression; and removing grounding to $M^{(1)}$ reduces cost but substantially increases errors. Full SHNM provides the strongest trade-off, supporting the principle that abstraction improves efficiency only when paired with verifiable grounding.

5.2 Trustworthiness: Auditing Hallucinations and Citation Quality

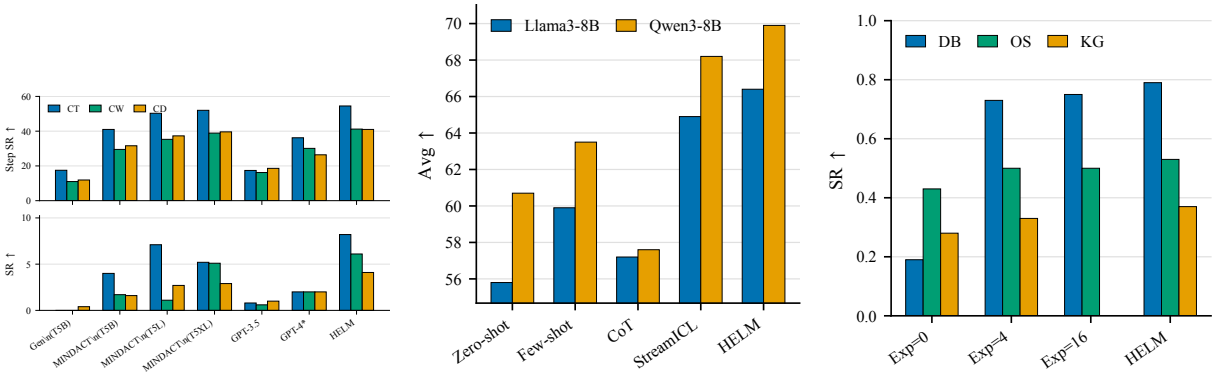
Across episodes, higher EpiErr rates strongly predict failure: we observe Pearson $r \approx -0.65$ between EpiErr and end-task success, indicating that when the system reuses stale/failed/contradictory evidence, tasks fail disproportionately often. Varying the staleness window by $\pm 50\%$ changes final success by less than $\pm 0.8\%$, suggesting the governance signal is not an artifact of a tuned threshold.

We count epistemic errors when the agent uses semantically relevant but epistemically invalid memories: stale timestamps, failed tool executions, or contradictory traces (Appendix Figure 5). These

Table 3: **Main results on GAIA and WebArena.** We report success rate (SR), CTU, and EpiErr. WebArena is mean±95% CI over 3 seeds when available; otherwise we report point estimates. Values marked with * are proxy estimates from partial diagnostics (context only).

Method	Plugins	CoT	UA hint	GAIA (SR %) (Mialon et al., 2023)		WebArena (SR %) (Zhou et al., 2024)	CTU (tokens) ↓	EpiErr. (%) ↓
				L2 ↑	L3 ↑	SR ↑		
Long-context / tool-augmented LLM agents (reported for context)								
GPT-4	✗	✗	✗	2.6	0.0	–	–	–
GPT-4 Turbo	✗	✗	✗	5.5	0.0	–	–	–
AutoGPT (GPT-4)	✗	✔	✗	0.4	0.0	–	–	–
GPT-4 + Plugins	✔	✔	✗	9.7	0.0	–	–	–
Memory-augmented agents (matched scaffolds/budgets)								
Flat episodic (RAG)	✗	✔	✗	6.8	0.2	15.8	4,200	12.8
MemGPT (Packer et al., 2023)	✗	✔	✗	7.1	0.3	16.9	3,500*	10.5*
Mem0 (Chhikara et al., 2025)	✔	✗	✗	7.4	0.4	17.6	–	–
HippoRAG (Gutiérrez et al., 2025)	✔	✗	✗	7.2	0.4	17.2	2,800*	9.2*
GraphRAG (Edge et al., 2025)	✔	✗	✗	7.8	0.5	18.0	3,100*	8.8*
MemGen (Zhang et al., 2025c)	✔	✔	✗	8.5	0.7	19.1	3,121	8.2*
G-Memory (Zhang et al., 2025a)	✔	✗	✗	8.9	0.8	19.5±1.4	2,796	7.5*
HELM (full)	✔	✔	✔	**12.8**	**1.7**	**22.4±1.2**	**2,050**	**6.3**
Upper bound								
Human	–	–	–	91.8	87.3	78.24	–	–

Figure 3: **Main results on Mind2Web, StreamBench, and LifelongAgentBench.** Panels report Mind2Web Step SR/SR across splits, StreamBench average final metric, and LifelongAgentBench success rates under replay in our unified harness.



(a) Mind2Web (b) StreamBench (c) LifelongAgentBench

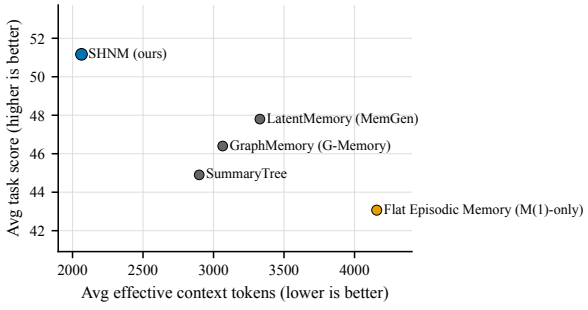


Figure 4: **Cost–performance Pareto frontier.** HELM improves task score while reducing effective context tokens (CTU), indicating that gains come from better governed memory use rather than larger context.

errors are distinct from pure reasoning/planning errors (valid evidence retrieved but mis-executed), motivating EpiErr as a complementary trustworthiness diagnostic beyond success rate.

External memory is only useful if recalled content is faithful to stored evidence. We therefore evaluate HELM under **faithfulness controls** that constrain only the memory generator while keeping the scaffold/backbone/tools fixed: (i) **A0: baseline recall** (current HELM), (ii) **A1: cited recall (hard)** where each sentence must include at least one machine-parsable provenance citation, and (iii) **A2: extractive-first recall** that selects evidence spans before any compression (or outputs spans only).

Table 4: Component ablations under matched budgets. **Variants**: Full = HELM, -Gov = no governance (semantic-only ranking), -Prov = no provenance expansion (no grounding to $M^{(1)}$), -Emb = no tool-aware embedding (base embedding), -Ctrl = heuristic controller (fixed schedules).

Var.	GAIA (SR %)		WebArena (SR %)	CTU ↓	EpiErr. ↓
	L2 ↑	L3 ↑	SR ↑		
Full	**12.8**	**1.7**	**22.4**	2050	6.3
-Gov	11.5	1.4	20.8	2100	14.2
-Prov	11.2	1.3	19.8	1850	11.5
-Emb	9.2	0.5	16.5	2250	7.8
-Ctrl	12.0	1.5	21.0	2350	6.8

Table 5: Layer ablations (EpiErr.: epistemic error).

Variant	Score ↑	Tokens ↓	EpiErr. ↓
M(1) only (Flat)	44.9	4213	12.8
M(3)→M(1) (no M(2))	47.2	2532	10.9
M(3)+M(2) (no grounding to M(1))	48.1	1781	9.7
M(3)+M(2)+M(1) (SHNM)	**50.8**	**2052**	**6.3**

We define the citation schema in Appendix A.7 and implement automatic validation/fallback in Appendix A.8; results are summarized in Table 1.

Hard citation constraints (A1) modestly increase cost (CTU 2050→2280) while largely preserving success and reducing epistemic errors (6.3→4.2), demonstrating that HELM's gains remain grounded under strict auditability. Extractive-first recall (A2) improves literal faithfulness but degrades reasoning-heavy success, highlighting a key trade-off: faithfulness alone is insufficient without controlled abstraction.

5.3 Learned Memory Dynamics

Beyond indexing, HELM treats memory management as a policy learning problem. Table 2 shows that **Distill+GSPO achieves the best task score while reducing token cost and write rate**, compared to heuristic schedules and distillation alone. This indicates the controller learns when *not* to write and when *not* to retrieve, reducing redundancy without dropping decisive evidence. Appendix A.13 reports action-frequency diagnostics and access-path distributions.

5.4 Robustness and Scalability

We stress-test SHNM as a memory access mechanism under reasoning-rich queries and long-horizon distractors.

Table 6: Memory access diagnostics on BRIGHT.

Method	nDCG@10 ↑	R@10 ↑
BM25	17.8	41.2
Dense baseline	19.9	44.7
ReasonIR (Shao et al., 2025)	29.7	58.0
SHNM	**31.6**	**60.3**

On BRIGHT, SHNM outperforms sparse retrieval (BM25) and a reasoning-specialized retriever (ReasonIR) on both nDCG@10 and Recall@10 (Table 6). The gains reflect two complementary mechanisms: coarse-to-fine cueing suppresses distractors, and tool/trace-aware representations improve access to procedural evidence beyond surface semantic matching.

We stress-test horizon length, memory-store size, and recall-query verbosity (§4.1); our harness logs per-step latency by component and supports scaling sweeps. Appendix A.14 summarizes scaling axes and a cost breakdown (Appendix Table 14), showing that HELM's overhead beyond generation is a small fraction of end-to-end runtime at both p50 and p95.

Overall, the results support a design principle for long-horizon agent memory: **abstraction improves efficiency only when paired with controllable navigation and provenance-backed grounding**.

6 Conclusion

We introduced **HELM**, a framework for long-horizon agents that combines a nested memory store (**SHNM**), explicit recall interfaces, and epistemic governance to make memory use auditable and robust to stale or invalid traces. HELM treats memory management as a policy problem—deciding when to read, write, consolidate, and prune under explicit budgets—and pairs this with tool-aware retrieval for tool-augmented trajectories. Across five benchmarks, HELM improves end-task success under matched budgets while reducing effective context usage (CTU) and epistemic errors (EpiErr.), supporting that reliability gains come from better governed evidence rather than larger context. We hope this work helps bridge the gap between scalable long-term memory and verifiable reasoning in deployed agent systems, and enables more reproducible evaluation of memory efficiency and epistemic reliability.

Limitations

HELM stores episodic traces and tool logs, which may contain sensitive content and can be resurfaced by later retrieval or summarization. This enlarges the attack surface: malicious instructions may enter memory via user text or tool outputs and later influence decisions, and consolidation can stabilize incorrect inferences into higher-level recalls that propagate across tasks. Moreover, abstractive recall can omit or subtly rewrite evidence even when provenance links exist, making faithfulness a first-class concern for evaluation and deployment.

These risks motivate operational safeguards and reporting discipline. Deployments require data minimization, automated redaction/PII filtering, strict access control and retention policies, and additional defenses such as quarantining untrusted content, filtering instruction-like spans, and sandboxing high-risk tool execution. HELM also adds runtime components (embedding, indexing, control, provenance expansion) that increase compute/latency under long horizons, and web/tool benchmarks drift over time (page updates, tool/version changes, harness differences), affecting stored traces and the meaning of "stale" evidence; rigorous evaluation should fix versions, record snapshots when possible, and report configurations, budgets, and seeds.

References

Rishabh Agarwal, Nino Vieillard, Yongchao Zhou, Piotr Stanczyk, Sabela Ramos Garea, Matthieu Geist, and Olivier Bachem. 2024. On-policy distillation of language models: Learning from self-generated mistakes. In *The Twelfth International Conference on Learning Representations*.

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

Xiang Deng, Yu Gu, Boyuan Zheng, Shijie Chen, Samuel Stevens, Boshi Wang, Huan Sun, and Yu Su. 2023. Mind2web: Towards a generalist agent for the web. *Preprint*, arXiv:2306.06070.

Darren Edge, Ha Trinh, Newman Cheng, Joshua Bradley, Alex Chao, Apurva Mody, Steven Truitt, Dasha Metropolitansky, Robert Osazuwa Ness, and Jonathan Larson. 2025. From local to global: A graph rag approach to query-focused summarization. *arXiv preprint arXiv:2404.16130*.

Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, Meng Wang, and Haofen Wang. 2024. Retrieval-augmented generation for large language models: A survey. *Preprint*, arXiv:2312.10997.

Bernal Jiménez Gutiérrez, Yiheng Shu, Yu Gu, Michihiro Yasunaga, and Yu Su. 2025. Hipporag: Neurobiologically inspired long-term memory for large language models. *Preprint*, arXiv:2405.14831.

Yuyang Hu, Shichun Liu, Yanwei Yue, Guibin Zhang, Boyang Liu, Fangyi Zhu, Jiahang Lin, and 1 others. 2025. Memory in the age of ai agents: A survey forms, functions and dynamics. *arXiv preprint arXiv:2512.13564*.

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, and 3 others. 2025. Memos: An operating system for memory-augmented generation (mag) in large language models. *Preprint*, arXiv:2505.22101.

Kevin Lu and Thinking Machines Lab. 2025. On-policy distillation. *Thinking Machines Lab: Connectionism*. Https://thinkingmachines.ai/blog/on-policy-distillation.

Yu. A. Malkov and D. A. Yashunin. 2016. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. *arXiv preprint arXiv:1603.09320*.

Grégoire Mialon, Clémentine Fourrier, Craig Swift, Thomas Wolf, Yann LeCun, and Thomas Scialom. 2023. Gaia: A benchmark for general ai assistants. *Preprint*, arXiv:2311.12983.

Charles Packer, Vivian Fang, Shishir G Patil, Kevin Lin, Sarah Wooders, and Joseph E Gonzalez. 2023. Memgpt: Towards llms as operating systems. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Joon Sung Park, Joseph C. O'Brien, Carrie J. Cai, Meredith Ringel Morris, Percy Liang, and Michael S. Bernstein. 2023. Generative agents: Interactive simulacra of human behavior. *Preprint*, arXiv:2304.03442.

Walter Quattrociocchi, Valerio Capraro, and Matjaz Perc. 2025. Epistemological fault lines between human and artificial intelligence. *SSRN Preprint*.

Rulin Shao, Rui Qiao, Varsha Kishore, Niklas Muennighoff, Xi Victoria Lin, Daniela Rus, Bryan Kian Hsiang Low, Sewon Min, Wen-tau Yih, Pang Wei Koh, and Luke Zettlemoyer. 2025. Reasonir: Training retrievers for reasoning tasks. *arXiv preprint arXiv:2504.20595*.

Haoran Sun and Shaoning Zeng. 2025. Hierarchical memory for high-efficiency long-term reasoning in llm agents. *arXiv preprint arXiv:2507.22925*.

Zhen Tan, Jun Yan, I-Hung Hsu, Rujun Han, Zifeng Wang, Long T. Le, Yiwen Song, Yanfei Chen, Hamid Palangi, George Lee, Anand Iyer, Tianlong Chen, Huan Liu, Chen-Yu Lee, and Tomas Pfister. 2025. In prospect and retrospect: Reflective memory management for long-term personalized dialogue agents. *Preprint*, arXiv:2503.08026.

Cheng-Kuang Wu, Zhi Rui Tam, Chieh-Yen Lin, Yun-Nung Chen, and Hung-yi Lee. 2024. Streambench: Towards benchmarking continuous improvement of language agents. *Preprint*, arXiv:2406.08747.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. 2023. React: Synergizing reasoning and acting in language models. In *The Eleventh International Conference on Learning Representations*.

Guibin Zhang, Muxin Fu, Guancheng Wan, Miao Yu, Kun Wang, and Shuicheng Yan. 2025a. G-memory: Tracing hierarchical memory for multi-agent systems. *arXiv preprint arXiv:2506.07398*.

Guibin Zhang, Muxin Fu, and Shuicheng Yan. 2025b. Memgen: Weaving generative latent memory for self-evolving agents. *Preprint*, arXiv:2509.24704.

Guibin Zhang, Muxin Fu, and Shuicheng Yan. 2025c. Memgen: Weaving generative latent memory for self-evolving agents. *arXiv preprint arXiv:2509.24704*.

Yanzhao Zhang, Mingxin Li, Dingkun Long, Xin Zhang, Huan Lin, Baosong Yang, Pengjun Xie, An Yang, Dayiheng Liu, Junyang Lin, Fei Huang, and Jingren Zhou. 2025d. Qwen3 embedding: Advancing text embedding and reranking through foundation models. *arXiv preprint arXiv:2506.05176*.

Yuxiang Zhang, Jiangming Shu, and 1 others. 2025e. Memory as action: Autonomous context curation for long-horizon agentic tasks. *arXiv preprint arXiv:2510.12635*.

Chujie Zheng, Shixuan Liu, Mingze Li, Xiong-Hui Chen, Bowen Yu, Chang Gao, Kai Dang, Yuqiong Liu, Rui Men, An Yang, Jingren Zhou, and Junyang Lin. 2025a. Group sequence policy optimization. *arXiv preprint arXiv:2507.18071*.

Junhao Zheng, Xidi Cai, Qiuke Li, Duzhen Zhang, Zhong-Zhi Li, Yingying Zhang, Le Song, and Qianli Ma. 2025b. Lifelongagentbench: Evaluating llm agents as lifelong learners. *Preprint*, arXiv:2505.11942.

Wanjun Zhong, Lianghong Guo, Qiqi Gao, and Yanlin Wang. 2024. Memorybank: Enhancing large language models with long-term memory. *Proceedings of the AAAI Conference on Artificial Intelligence*.

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

A Additional Details and Diagnostics

A.1 Explicit memory operations (moved from the main text)

Table 7: **Explicit memory operations in HELM.** All operations log provenance and epistemic metadata to enable auditability.

Operation	Role
READ $(\mathcal{M}, q)$	Retrieves a small set of memories under a token budget; applies epistemic governance to filter failed/stale evidence and to re-rank by recency/status/provenance.
WRITE $(\mathcal{M}, x)$	Stores new episodic traces (tool outputs, constraints, intermediate results) with timestamps; records source types and tool status for later grounding.
CONSOLIDATE $(\mathcal{M})$	Distills multi-step evidence into higher-level recalls/indices while preserving provenance links back to supporting traces; resolves conflicts via epistemic preference.
PRUNE $(\mathcal{M})$	Enforces memory budgets by removing stale/failed items and demoting redundancies; retains minimal evidence needed for future verification.

A.2 QA sanity check with retrieved context

We report these numbers as an auxiliary diagnostic for memory access quality; the main paper focuses on BRIGHT memory access diagnostics and long-horizon agent performance.

A.3 Score–cost diagnostic on QA/Math/Code

To complement long-horizon agent benchmarks, we report a controlled multi-domain diagnostic that quantifies the trade-off between task score and memory cost (effective context tokens).

A.4 Benchmark details (moved from main results)

GAIA (Level 2–3). GAIA is particularly unforgiving at higher difficulty levels: even strong tool-augmented assistants score near zero on Level 3 (Mialon et al., 2023).

WebArena. WebArena evaluates end-to-end task completion in a realistic web environment with programmatic validators (Zhou et al., 2024).

Mind2Web. Mind2Web supports step-level evaluation and three generalization regimes (Deng et al., 2023).

Table 8: QA accuracy with retrieved context on MMLU and GPQA (sanity check).

Method	MMLU ↑	GPQA ↑
BM25	63.1	28.4
Dense baseline	64.5	30.0
ReasonIR (Shao et al., 2025)	69.2	36.8
SHNM (ours)	**70.1**	**38.5**

StreamBench. StreamBench evaluates continuous improvement from an input–feedback stream (Wu et al., 2024). In the main paper we report the average final metric across the seven StreamBench tasks (Spider/CoSQL/BIRD/DS-1000/ToolBench/DDXPlus/HotpotQA).

LifelongAgentBench. LifelongAgentBench evaluates agents as lifelong learners across DB/OS/KG environments under strict sequential execution (Zheng et al., 2025b).

A.5 Backbone robustness on LifelongAgentBench (DB)

To contextualize the sensitivity of lifelong settings to the underlying reasoner, Table 10 reproduces DB-environment results reported by Zheng et al. (2025b). These numbers are included for context and are not directly comparable to our unified harness when scaffolds/tools differ.

A.6 Metric sheet (definitions and adjudication)

We summarize the key metrics used throughout the paper and make their computation explicit.

Task success / score. We report each benchmark's official end-task metric (e.g., GAIA success rate, WebArena functional success, Mind2Web Step SR/SR). When multiple runs are used, we report mean and standard error across seeds.

Effective context tokens (CTU). CTU measures *total* input-context tokens consumed over an episode:

$$\mathrm{CTU} = \sum_{t=1}^{T} \mathrm{Tok}\big(\texttt{system} \| \texttt{history}_t \| \texttt{retrieved}_t \| \texttt{tool\_obs}_t\big),$$

where $\texttt{retrieved}_t$ includes recalled memories (and any provenance expansions) injected into the prompt, and $\texttt{tool\_obs}_t$ includes tool outputs exposed to the model. We compute tokens using the backbone tokenizer and report CTU per task (sum) and per step (mean) when relevant.

Epistemic errors (EpiErr.). EpiErr counts steps in which the agent uses *semantically relevant but epistemically invalid* memory. We classify an accessed memory item as invalid if it satisfies any of: (i) **stale** (version mismatch or age beyond a configured window), (ii) **failed** (derived from a tool call with non-success status), or (iii) **contradictory** (marked by consolidation as conflicting with newer verified evidence). EpiErr is reported as a percentage of steps (or recalls) and can be computed directly from logged metadata and access traces. To avoid circularity, "contradictory" is determined from extracted key–value claims in the underlying tool traces and timestamps/status metadata (not from task outcomes): if two memories refer to the same key but disagree on value, we keep the newer SUCCESS-supported value and mark the older value as a conflict for downstream governance.

Claim extraction and robustness checks. EXTRACTCLAIMS uses tool-type-specific parsers for structured outputs (e.g., JSON fields / key–value tables) and a constrained regex extractor for key=value or numeric assertions in free-form text, followed by simple normalization. We additionally run a small manual audit and sensitivity checks over extractor/threshold variants; the qualitative EpiErr trends in Section 5 remain stable.

Provenance faithfulness. We measure whether recalled content is auditable via provenance links back to $M^{(1)}$. Given a recalled memory consisting of sentences $\{s_i\}$, provenance faithfulness is the fraction of sentences that can be matched to at least one valid provenance pointer whose span contains supporting evidence for s_i. In the cited-recall variant (Appendix A.7), this reduces to citation coverage and citation validity.

Citation precision (C-P) protocol. We compute C-P by randomly sampling 100 cited recalls from GAIA Level 2 and judging whether each cited episodic span *explicitly supports* the corresponding sentence-level claim (not merely topical overlap). We fix the judge model and rubric, and report precision as the fraction of claims labeled supported.

Staleness window. We support two interchangeable staleness criteria: **step-based** staleness ($t - \texttt{step\_id} > \Delta_{\text{step}}$) and **time-based** staleness

Table 9: Multi-domain score–cost diagnostic. Higher is better for scores; lower is better for tokens.

Method	Task Score ↑			Eff. Context Tokens ↓		
	QA	Math	Code	QA	Math	Code
Flat Episodic Memory (M(1)-only)	58.4	42.1	28.7	3896	4091	4485
SummaryTree	60.2	44.0	30.5	2614	2889	3192
LatentMemory (MemGen) (Zhang et al., 2025c)	62.8	46.7	33.9	3121	3278	3587
GraphMemory (G-Memory) (Zhang et al., 2025a)	61.9	45.2	32.1	2796	3011	3388
SHNM (ours)	**66.3**	**49.8**	**37.4**	**1847**	**2053**	**2291**

Table 10: **DB environment success rate under replay (Exp).** Numbers are reported in Table 3 of Zheng et al. (2025b).

Backbone	Exp=0	Exp=4	Exp=16	Exp=64
DeepSeek-R1-Distill-Llama-8B	0.07	0.35	OOM	OOM
DeepSeek-R1-Distill-Qwen-7B	0.10	0.18	OOM	OOM
QwQ-32B	0.29	0.21	0.25	OOM
Qwen2.5-7B-Instruct	0.74	0.76	0.74	OOM
Qwen2.5-32B-Instruct	0.82	0.71	0.72	OOM
Llama-3.1-8B-Instruct	0.19	0.73	0.75	0.78
Llama-3.1-70B-Instruct	0.81	0.86	0.88	0.90

(current wall-clock minus `timestamp` $> \Delta_{\text{time}}$). For web benchmarks, we additionally treat environment snapshots/tool versions as part of `task_version`; any mismatch marks the item as stale.

A.7 Provenance and citation schema

Provenance pointers. Each memory item stores a (possibly empty) set of provenance pointers to episodic traces in $M^{(1)}$:

```
(tool, log_id, step_id, field, start,
end, hash)
```

where `field` denotes an observation field (e.g., `title/url/text/json_path`), and (`start`, `end`) indexes a character span (or sentence indices) within that field. `hash` is computed on the referenced span to make pointers stable under log serialization.

Cited recall (machine-parsable). For faithfulness experiments, we constrain the memory generator to emit citations inline using:

```
[[CITE tool=<t> log=<log_id> step=<s>
field=<f> start=<i> end=<j>]]
```

Optionally, a recall can include a `<citations>...</citations>` block listing all cited spans. A citation is **valid** if it parses and resolves to an existing episodic span with matching hash; otherwise it is invalid and triggers regeneration/fallback.

A.8 Cited recall constraints, validation, and fallback

Output format constraint. We constrain the memory generator to output: `<memory>...</memory>` containing the recalled content, and optionally a `<citations>...</citations>` block listing all cited spans. Under the hard constraint, each sentence in `<memory>` must contain at least one inline `[[CITE ...]]`.

Validation and regeneration. After generation, we validate citations against the currently visible episodic-span table in $M^{(1)}$. If validation fails, we provide diagnostics (missing citation, malformed citation, unresolved pointer, hash mismatch) and regenerate up to R times; persistent failures fall back to baseline recall or an extractive-only recall.

Prompt template (for the memory generator). We use a fixed instruction that (i) forbids unsupported claims, (ii) forbids citing non-existent spans, and (iii) requires at least one `[[CITE ...]]` per sentence in `<memory>`. When validation fails, the diagnostics from Algorithm 1 are appended verbatim to the next generation request to steer correction.

Extractive-first recall (control). As a complementary control, we implement an extractive-first pipeline that first selects provenance spans (with citations) and then optionally compresses them into a short summary; this separates retrieval/grounding failures from abstractive compression failures.

Case 1: Stale-memory trap (WebArena/GAIA).
Scenario. The agent should book a flight only if the current price is under $300, but an older tool trace stores a stale price.
Failure (semantic-only).
`Retrieved trace: "UA123 price=$280" (timestamp: T-100)`
`Action: BookFlight(UA123)` **FAIL** (price updated to $450).
Success (HELM + governance).
`Retrieved trace is flagged stale` → `trigger fresh search.`
`New tool output: price=$450` → `refuse booking / ask user.`

Case 2: Failed-trace trap (tool/terminal).
Scenario. The agent must report the output of a script; an earlier run failed, a later run succeeded.
Failure (no status filter).
`Retrieved trace (FAILED): Traceback... FileNotFoundError`
`Generation: "processed successfully"` **hallucination**.
Success (HELM + status-aware read).
`Skip FAILED trace` → `select SUCCESS trace: "Processed 1000 records".`
`Generation: "Processed 1000 records" + citation.`

Figure 5: **Case studies: epistemic traps and governance.** Governance uses timestamps and tool-execution status to prevent reuse of stale or failed traces, while provenance/citations support auditability.

A.9 Case studies: epistemic traps and governance

A.10 Reproducibility: hyperparameters and budgets

We recommend reporting the following parameters for all methods to enable fair comparisons and reproducibility. When baselines are re-implemented in our harness, we lock these values across methods unless the baseline definition requires otherwise.

Algorithm 1 VALIDATECITEDMEMORY for hard cited recall

Require: Model output y, span table $\mathcal{S}$ from visible $M^{(1)}$, max retries R
Ensure: `valid` flag and diagnostics

Parse `<memory>` block; split into sentences $\{s_i\}$
if `<memory>` missing **then**
  return `invalid`, MISSINGMEMORYBLOCK
end if
for all sentence s_i **do**
  if s_i has no `[[CITE ...]]` **then**
    return `invalid`, MISSINGCITE
  end if
  for all citation c in s_i **do**
    if c does not parse **then**
      return `invalid`, MALFORMEDCITE
    end if
    Resolve (`tool`, `log_id`, `step_id`, `field`, `start`, `end`) in $\mathcal{S}$
    if unresolved **then**
      return `invalid`, UNRESOLVEDPOINTER
    end if
    if hash mismatch **then**
      return `invalid`, HASHMISMATCH
    end if
  end for
end for
return `valid`, OK

Algorithm 2 Hard cited recall with regeneration and fallback

Require: Recall request r, max retries R

for $j = 1$ to R **do**
  $y \leftarrow \textsc{GenerateCitedMemory}(r)$
  $(\texttt{valid}, d) \leftarrow \textsc{ValidateCitedMemory}(y, \mathcal{S})$
  if `valid` **then**
    return y
  end if
  $r \leftarrow r \| d$ ▷ append diagnostics to the next prompt
end for
return FALLBACKRECALL(r) ▷ baseline recall or extractive-only

Table 11: **Reproducibility: hyperparameters and budgets used in our unified harness.** We lock these values across methods whenever the baseline definition permits, so improvements are attributable to memory rather than larger context or looser budgets.

Setting	Value	Rationale / justification
Reasoner backbone / context window	`Qwen-3-8B` 128k	Strong backbone; demonstrates gains from memory under a standard long-context budget.
Decoding (temperature, top-p, max tokens)	Temp=0.1 top-p=0.9 max_tokens =1024	Low temperature improves reproducibility for agent actions and reduces stochastic control flow.
Governance weights (Eq. 1)	$\lambda_{\text{sim}} = 1.0$ $\lambda_{\text{time}} = 1.0$ $\lambda_{\text{status}} = 10.0$ $\lambda_{\text{src}} = 1.0$	Strong status penalty enforces tool-grounded validity; other terms are default-scaled.
Staleness window	$\Delta_{\text{step}} = 200$ $\Delta_{\text{time}} = 7$ days + snapshot/version check	Detects version drift and "old but plausible" traces; sensitivity is reported in §5.2.
Memory budgets (max entries in $M^{(1)}, M^{(2)}, M^{(3)}$)	$\lvert M^{(1)}\rvert = 200$ $\lvert M^{(2)}\rvert = 50$ $\lvert M^{(3)}\rvert = 20$	Explicit "pyramid" constraints: forces selective retention and validates efficiency claims under tight budgets.

Retrieval (k per layer; HNSW parameters)	$k_3 = 3,\ k_2 = 5$
	$k_1 = 5$
	`ef_search=64`
	`M=16`
		Small k values test high-precision recall; HNSW settings are standard robust defaults.

Max recall tokens injected per step	1,500 tokens/step	Constrains recall to be substantially smaller than full-log RAG, supporting low-CTU comparisons.
Write / consolidate / prune limits	Write $\leq$ 1/step Consolidate $\leq$ 5/episode Prune on budget overflow	Encourages concise memory management and prevents degenerate "log spamming" behaviors.
Embedding model	Qwen3-Embedding-4B (fine-tuned)	Matches the tool-aware embedding training pipeline in §3.4.
Controller model and reward weights	`Qwen-3-4B` (full FT) $\alpha = 1.0$ $\beta = 0.05$	Lightweight controller keeps overhead low; memory-cost penalty is a soft regularizer while task reward dominates.
Seeds and evaluation splits	3 seeds (0, 42, 123) Official benchmark splits	Standard practice for stability; avoids cherry-picking and supports statistical reporting.

A.11 Training data pipeline and leakage audit

Benchmarks with web interaction are particularly vulnerable to contamination. We therefore make our isolation protocol explicit and auditable. For all training trajectories used to learn the controller or embeddings, we record source metadata (collection timestamp, tool schema, and environment identifiers) and enforce a conservative isolation policy w.r.t. evaluation benchmarks.

Isolation checklist.

- **Domain/URL filtering (web):** canonicalize URLs (scheme, path normalization; strip queries/fragments) and drop trajectories that touch benchmark domains/URLs (WebArena/Mind2Web allowlists/blacklists).
- **Temporal isolation:** enforce a time cutoff preceding the evaluation snapshot whenever snapshots are available; otherwise treat unknown snapshot provenance as unsafe and exclude.
- **Near-duplicate prompts/instructions:** hash normalized instruction templates and run n-gram overlap checks against evaluation instructions.
- **Tool/version drift:** record tool schema and environment version identifiers; mismatches are treated as stale (Appendix A.6).

Quantitative audit (training runs). We filtered $\sim$15% of web-derived training trajectories due to benchmark-domain overlap. After filtering, URL overlap with evaluation environments is 0.0%, and the n-gram audit finds no overlaps longer than 50 tokens between training prompts and evaluation instructions.

A.12 Epistemic governance pseudocode

A.13 Memory operations and controller diagnostics

To make the memory contribution explicit beyond aggregate scores, we further decompose SHNM into *memory operations*: formation (writes), consolidation (promotion across layers), and access (reads). In addition to reporting *Write Rate* as a proxy for memory formation, we summarize: (i) read/write/consolidation frequencies per task, (ii) cross-layer access-path distributions (M(3)$\rightarrow$M(2)$\rightarrow$M(1) vs. early exits), and (iii) an epistemic error breakdown using the failure taxonomy in §5.2. These diagnostics help separate gains from improved *where/what/how* decisions (cueing, reuse, grounding) from gains due to retrieving more text.

Table 12: **Training data pipeline disclosure (counts used in our training runs).** Web-derived trajectories exclude evaluation domains/snapshots to mitigate leakage.

Component	Count	Avg. length	Rationale
Tool trajectories (total episodes)	15,043	128 steps	Scale for RL/finetuning; excludes validation domains.
Distinct tools / tool schemas	3,400	–	Diversity for generalization to unseen tools and schemas.
Synthetic hierarchical triplets	20,320	–	Contrastive supervision for cross-layer alignment (Appendix A.16).
Episodic trace entries ($M^{(1)}$)	10,120	150 tokens	Long raw traces (code/JSON/tool logs), motivating compression.
Consolidated recalls ($M^{(2)}$)	20,024	45 tokens	$\sim$3$\times$ shorter than $M^{(1)}$ (compression evidence).
Thematic indices ($M^{(3)}$)	10,000	8 tokens	Short intent keys for coarse-to-fine routing with minimal prompt footprint.

Table 13: **Memory operations and controller diagnostics.** Mean action frequencies computed from logged memory events (R/W per step; C per episode).

Setting	R/step	W/step	C/ep
GAIA (L2)	**0.32**	**0.45**	**1.2**
GAIA (L3)	**0.58**	**0.62**	**3.5**
WebArena	**0.41**	**0.38**	**0.8**

Suggested controller report. We recommend reporting the following controller diagnostics (computed from logged memory events):

- **Action frequencies:** mean counts per episode for READ/WRITE/CONSOLIDATE/PRUNE.
- **Phase behavior:** action proportions in early/mid/late thirds of an episode.
- **Access paths:** fraction of recalls that traverse M(3)$\rightarrow$M(2)$\rightarrow$M(1) vs. early-exit at M(3) or M(2).
- **Utility vs. cost:** correlation between recall depth and (success, CTU, EpiErr.).

Interpretation. The learned controller exhibits **smart laziness**: it retrieves in only $\approx$32% of GAIA-L2 steps on average, but increases reads/writes on GAIA-L3 as task difficulty rises. On WebArena, the WRITE rate stays low despite frequent web

Algorithm 3 RETRIEVE with epistemic governance (coarse-to-fine)

Require: Query q, embedding $\mathbf{v}_q$, layer budgets (k_3, k_2, k_1), staleness window Δ, weights λ
Ensure: Ranked candidate set $\mathcal{C}$

$\mathcal{C} \leftarrow \emptyset$
for $l \in \{3, 2, 1\}$ **do**
  $\mathcal{H} \leftarrow \text{HNSWSEARCH}(M^{(l)}, \mathbf{v}_q, \text{pool} = c_l)$
  for all $\Updownarrow \in \mathcal{H}$ **do**
    VALID $\leftarrow$ ($\Updownarrow$.status $\neq$ FAILED) $\wedge$ ($\neg$ ISSTALE($\Updownarrow, \Delta$))
    $S(\Updownarrow; q) \leftarrow \lambda_{\text{sim}}\text{sim}(\mathbf{v}_q, \Updownarrow.\mathbf{v}) + \lambda_{\text{time}} g(\Delta t) + \lambda_{\text{status}} w(\texttt{status}) + \lambda_{\text{src}} w(\texttt{source\_type})$
    if $\neg$ VALID **then**
      $S(\Updownarrow; q) \leftarrow S(\Updownarrow; q) - \infty$
    end if
  end for
  $\mathcal{C} \leftarrow \mathcal{C} \cup \text{TOPK}(\mathcal{H}, k_l, \text{by } S)$
end for
return SORT($\mathcal{C}$, by S)

Algorithm 4 PROVENANCEEXPANSION for auditable recalls

Require: Seed set $\mathcal{C}$, token budget B, max depth D
Ensure: Expanded evidence set $\mathcal{Z}$

$\mathcal{Z} \leftarrow \emptyset$; queue $\leftarrow \mathcal{C}$
while queue not empty **and** $\text{Tok}(\mathcal{Z}) < B$ **do**
  pop $\Updownarrow$ from queue
  $\mathcal{Z} \leftarrow \mathcal{Z} \cup \{\Updownarrow\}$
  if $\text{depth}(\Updownarrow) < D$ **then**
    for all $p \in \Updownarrow.\mathcal{P}$ **do**
      $\Updownarrow_1 \leftarrow \text{RESOLVEPOINTER}(p)$ ▷ episodic span in $M^{(1)}$
      push $\Updownarrow_1$ to queue
    end for
  end if
end while
return $\mathcal{Z}$

interaction, indicating effective noise filtering (e.g., skipping scroll/click chatter). Across successful recalls, 85% traverse the full coarse-to-fine path M(3)→M(2)→M(1), suggesting the hierarchy is actively utilized rather than merely decorative.

A.14 Scaling and robustness diagnostics

We evaluate robustness along the three scaling axes described in §4.1: (i) horizon length (with increasing distractors), (ii) memory-store size, and (iii) query rewriting length. For each axis, we report a scaling curve over (a) retrieval success/quality, (b) end-task performance, and (c) effective context tokens / latency.

System cost breakdown. To characterize overhead, we recommend decomposing per-step latency into: (i) embedding + ANN search, (ii) controller inference, (iii) memory generation, and (iv) provenance expansion, and reporting p50/p95 over episodes. We also report index build time and peak memory/CPU/GPU usage as the store grows.

Takeaway. At p50, EMB+ANN+CTRL+PROV contributes ≈4–5% of total latency; even at p95 it remains small (up to ≈6%), with generation accounting for the vast majority of runtime.

Table 14: **System cost breakdown (ms per step).** We report p50/p95 over episodes; GEN (reasoner generation) dominates total latency.

Setting	Emb+ANN	Ctrl	Gen	Prov	Total
GAIA (L2) p50	45	25	1,850	12	**1,932**
GAIA (L2) p95	120	40	3,200	45	**3,405**
WebArena p50	55	28	2,100	15	**2,198**
WebArena p95	145	48	4,500	60	**4,753**

A.15 Additional qualitative cases

We provide extended case studies with explicit provenance trails M(3)→M(2)→M(1), illustrating: (i) correct coarse-to-fine traversal that recovers a decisive tool trace, (ii) an epistemic mismatch case where stale memory is rejected under governance, and (iii) an over-compression case where M(2)/M(3) loses a constraint and the agent must fall back to M(1) for verification.

A.16 Nested-memory synthesis details

We build training instances that teach the retriever to (i) align queries with the appropriate abstraction level, (ii) remain robust under long-horizon distractors, and (iii) distinguish semantically relevant but epistemically invalid memories. Concretely, we synthesize hierarchical supervision by mapping each seed document or interaction into a three-level memory triplet $\mathcal{M} = \{M^{(1)}, M^{(2)}, M^{(3)}\}$.

Seed selection and filtering. We draw seeds from FineWeb-Edu and high-quality tool-augmented interaction logs. To bias toward

Algorithm 5 CONSOLIDATE with conflict resolution

Require: Supporting traces $\mathcal{T} \subseteq M^{(1)}$, policy thresholds τ
Ensure: Consolidated memory item $\Updownarrow^{(2)}$

```
Extract structured claims K ← EXTRACTCLAIMS(T)
Group by key: {K_j} ← GROUPBYKEY(K)
for all K_j do
    Rank candidates by epistemic preference: SUCCESS ≻ UNKNOWN ≻ FAILED, newer ≻ older, trace-supported ≻ ungrounded
    Select winner k*; mark alternatives as CONFLICT if score gap < τ
end for
Synthesize summary x^(2) ← SUMMARIZE({k*}) with pointers to supporting spans
return ⇕^(2) = (x^(2), v^(2), E^(2), P^(2))
```

Algorithm 6 PRUNE under memory budgets

Require: Memory layers $M^{(1:3)}$, budget limits, utility scores $U(\cdot)$
Ensure: Updated memory layers

```
for l ∈ {1, 2, 3} do
    Mark items stale/failed for removal; compute redundancy clusters
    Remove lowest-utility items until budget satisfied
end for
```

procedural knowledge, we filter by educational value scores ($\geq$ 3.0) and prefer examples that exhibit multi-step reasoning or tool-use structure rather than factual recitation.

Vertical abstraction synthesis. For each seed item, we prompt a teacher model to synthesize:

1. **Episodic traces ($M^{(1)}$):** raw interaction snippets (e.g., tool calls, tool outputs, user dialogue) that could have produced the seed information;
2. **Consolidated recalls ($M^{(2)}$):** procedural summaries that preserve causal structure while removing irrelevant noise;
3. **Thematic indices ($M^{(3)}$):** high-level keys that capture intent and serve as entry points for coarse-to-fine traversal.

We then sample queries q whose positive target m^+ can come from any layer $l \in \{1, 2, 3\}$, forcing the embedding space to represent cross-layer relationships.

Long-horizon contextual expansion. To model "needle-in-a-haystack" retrieval, we generate varied-length interaction trajectories (512–8,192 tokens) by interleaving the target signal with semantically plausible distractor episodes. This yields supervision where the model must rely on high-level cues ($M^{(3)}$) to recover precise evidence ($M^{(1)}$) under heavy distraction.

Generative epistemic contrast. Beyond semantic hard negatives, we generate *epistemic* negatives: counterfactual traces that match topic and lexical content but violate validity conditions (e.g., mismatched timestamps, failed tool status, or inconsistent entity state). These negatives encourage the retriever to prefer justified evidence over superficially relevant but invalid memory.

A.17 Stage 1 on-policy distillation details

Stage 1 initializes the memory controller with *on-policy distillation*, the goal is to combine the *on-policy* relevance of RL (training on states visited by the current policy) with a *dense* learning signal (token-level supervision), reducing exposure bias and avoiding brittle, heuristic write/retrieval schedules.

Setup. Let π_θ denote the current controller policy that produces a sequence of memory-operation tokens/actions $x_{1:T}$ (e.g., READ/WRITE/CONSOLIDATE decisions and associated arguments) conditioned on a trajectory prefix $x_{1:t}$ and the current interaction state. We also maintain a fixed teacher policy π_T (a stronger model) used only for scoring.

Dense supervision via per-token reverse KL. We sample rollouts from the student/controller π_θ and query the teacher for token-level log-probabilities on the *same sampled tokens*. Following Lu and Lab (2025), we use the per-token reverse KL:

$$\mathrm{KL}(\pi_\theta \,\|\, \pi_T) = \mathbf{E}\Big[\log \pi_\theta(a_{t+1} \mid s_t) - \log \pi_T(a_{t+1} \mid s_t)\Big].$$

where the expectation is taken over token/action samples (s_t, a_{t+1}) induced by student rollouts from π_θ. Operationally, this yields a token-level "advantage" signal $A_{t+1} = -\left(\log \pi_\theta(a_{t+1} \mid s_t) - \log \pi_T(a_{t+1} \mid s_t)\right)$, which penalizes tokens that deviate from the teacher in states the student actually visits. Consistent with

the blog recipe, we use a zero discount (each update targets the immediate next token), and we do not backpropagate through the teacher.

Training loop (recipe). Each iteration consists of:

1. **Sample trajectories:** roll out the current controller π_θ to obtain $x_{1:T}$ and record student log-probabilities $\log \pi_\theta(x)$.
2. **Score with teacher:** compute teacher log-probabilities $\log \pi_T(x)$ on the sampled tokens (e.g., via a `compute_logprobs`-style call).
3. **Compute advantages:** set per-token advantages to the negative reverse KL, $A = -\left(\log \pi_\theta(x) - \log \pi_T(x)\right)$.
4. **Update student:** apply a standard RL-style policy-gradient/importance-sampling objective using these advantages, which yields a supervised-like update while remaining on-policy.

Practical implications. The per-token signal enables (i) **partial rollouts** (no need to wait for sequence termination to obtain a reward), and (ii) **prompt reuse**: unlike multi-epoch RL on a fixed prompt that can overfit to a single final answer, on-policy distillation targets matching the teacher distribution and empirically tolerates reusing prompts with many samples per prompt (Lu and Lab, 2025). In our setting, this is particularly valuable for stabilizing the controller early in training: the controller learns to avoid pathological writes/retrievals in the states it induces, before Stage 2 optimizes task-specific objectives.

A.18 Tool-aware embedding fine-tuning details

We fine-tune `Qwen3-Embedding-4B` to embed tool-augmented traces for SHNM indexing. Since the base model is decoder-only, we remove the causal mask during fine-tuning and use bidirectional attention to allow each token to attend to both tool invocations and tool results. We pool the final hidden states into a vector $\mathbf{v} \in \mathbf{R}^d$ using weighted mean pooling.

Instruction-conditioned contrastive objective. For each input, we prepend an instruction I that specifies the retrieval intent (e.g., retrieving a tool log vs. a high-level plan). We optimize an InfoNCE loss:

$$
\begin{aligned}
s(m) &= \mathrm{sim}(\mathbf{v}_q, \mathbf{v}_m), \\
Z &= \exp\big(s(m^+)/\tau\big) + \sum_{m^- \in \mathcal{N}} \exp\big(s(m^-)/\tau\big), \\
\mathcal{L}_{\mathrm{cl}} &= -\log \frac{\exp\big(s(m^+)/\tau\big)}{Z},
\end{aligned}
\tag{3}
$$

where $\mathcal{N}$ includes in-batch negatives and our synthesized epistemic counterfactuals. This makes the embedding space sensitive to both semantics and tool-structure metadata (e.g., API schemas, return codes, and execution status).