

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

Jialong Liu², Yuling Shi⁴, Zuchao Li^{1*}, Ping Wang³

¹School of Artificial Intelligence, Wuhan University, Wuhan, China

²School of Advanced Interdisciplinary Information Sciences, Wuhan University, Wuhan, China

³School of Information Management, Wuhan University, Wuhan, China

⁴School of Computer Science, Shanghai Jiao Tong University, Shanghai, China

{galleons, zcli-charlie, wangping}@whu.edu.cn, yuling.shi@sjtu.edu.cn

Abstract

Long-horizon LLM agents often fail even when the relevant information is present in memory: semantically relevant recalls can be epistemically invalid—stale observations, failed tool traces, or claims contradicted by newer verified evidence—yet are retrieved and treated as support. We introduce **HELM** (**H**ierarchical **E**pistemic **L**earned **M**emory), whose three-tier **Semantic-HNSW Nested Memory** (SHNM) stores every item with provenance and epistemic state, linking episodic tool traces to consolidated recalls and thematic indices so that recalls stay auditable back to concrete evidence spans. *Epistemic governance* filters and re-ranks recalls by status, freshness, consistency, recency, and provenance, while a learned controller decides when explicit read/write/consolidate/prune operations are worth their cost. Across five long-horizon benchmarks—GAIA, WebArena, Mind2Web, StreamBench, and LifelongAgentBench—HELM improves success over strong memory baselines while reducing effective context tokens and epistemic errors. Retriever-controlled comparisons and per-signal ablations attribute the reliability gains specifically to epistemic governance, and the improvements persist with a 32B backbone and in a training-free variant. Code is available at <https://github.com/Galleons2029/HELM>.

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., 2026). In this regime, an agent must carry forward user constraints, intermediate plans, environment state, and

tool-grounded evidence across hundreds of interaction steps. The usual diagnosis is a capacity problem—the context window is too short, so more retrieval, longer context, or more aggressive compression should help—which motivates a large literature on reducing what long context costs (Shi et al., 2024). Our central claim is different. For long-horizon agents, *epistemic* failures are a major and directly addressable source of memory errors, complementary to reasoning and planning failures: the agent retrieves information that is semantically on-topic but no longer valid evidence for the current decision. In a manual taxonomy of 100 failed episodes (Appendix A.1), 38% of failures trace to this epistemic misuse of memory—stale, failed, or contradicted evidence—alongside 31% planning and 14% retrieval-miss failures: not dominant, but large enough to matter and, unlike reasoning failures, attackable directly at the memory layer.

Consider two common failures. In a WebArena-style purchase task, an agent observes a flight price of \$280 at step 10 and stores it. By step 85, the page has changed and the current price is \$450; semantic retrieval still returns the old “flight price” trace, leading the agent to book a now-invalid option. In a tool-use task, an earlier script run fails with `FileNotFoundException`, while a later run succeeds; a query for “script output” may retrieve the failed trace because it is lexically similar, after which the model fabricates a successful result. These errors are not ordinary generation-side hallucinations: the retrieved text exists in memory. They are also not standard time-sensitive RAG over static documents: the invalid evidence is the agent’s own dynamic interaction history, whose status, timestamp, version, and contradictions determine whether it should be trusted.

Recent memory systems mitigate long horizons with context extension, periodic summarization, or top-*k* retrieval (Packer et al., 2024; Zhong et al., 2024). These methods improve access, but they

*Corresponding author. This work was supported by the Technology Innovation Program of Hubei Province (No. 2024BAB043), the National Natural Science Foundation of China (No. 62306216), and the National Social Science Foundation of China (No. 24&ZD186).

Table 1: **Each HELM mechanism answers a specific memory failure mode.** Attribution of the resulting gains is in Table 4.

Failure mode	HELM mechanism
Stale evidence	<i>Freshness</i> over timestamps and task/environment versions
Failed tool trace	<i>Status</i> filter, with a diagnostic fallback when the failure is the evidence sought
Contradicted evidence	<i>Consistency</i> ; consolidation records conflicts instead of overwriting
Unauditable abstraction	Provenance pointers and cited recall back to source traces
Ops at the wrong time	Learned controller under explicit budgets

usually treat retrieved memories as uniformly valid once they are semantically relevant. Abstractive summarization introduces a second failure mode: it can compress away decisive constraints such as “pending payment” or “tool call failed,” making later failures hard to audit. We therefore frame long-horizon memory as governed evidence management rather than passive storage.

We propose **HELM** (**H**ierarchical **E**pistemic **L**earned **M**emory), a framework for steering long-horizon agents with auditable, governed memory. Rather than assembling components for their own sake, we introduce each mechanism to answer a specific failure mode observed in the taxonomy above; Table 1 states this correspondence, and Section 5.2 measures what each mechanism actually buys.

Concretely, the substrate is **Semantic-HNSW Nested Memory (SHNM)**, a three-tier store over episodic traces ($M^{(1)}$), consolidated recalls ($M^{(2)}$), and thematic indices ($M^{(3)}$), in which every item carries provenance pointers and epistemic metadata (tool status, timestamp, source type, task/environment version). **Epistemic governance** then uses that metadata at retrieval time to filter and re-rank candidates, and at consolidation and pruning time to propagate validity rather than letting abstractions overwrite evidence.

We evaluate HELM on five long-horizon benchmarks and report not only task success but also effective context tokens (CTU), epistemic errors (EpiErr.), and provenance faithfulness. Under matched scaffolds and budgets, HELM improves over strong memory baselines on GAIA (L2: 8.9→12.8; L3: 0.8→1.7) and WebArena (19.5→22.4), while reducing CTU and EpiErr. Additional analyses address the main practical concerns: gains persist with Qwen3-32B, a training-

free HELM-ZS variant improves GAIA-L2 over the strongest training-free baseline (8.9→10.2) while matching it on WebArena, HELM-lite isolates governance as the largest reliability driver, and latency diagnostics show that memory overhead is small relative to reasoner generation.

Our contributions are: (i) a formalization of agent-memory failure as *epistemic error*, distinct from hallucination and static temporal RAG; (ii) SHNM, a nested memory architecture with provenance-backed recall and epistemic governance over explicit memory operations; and (iii) a broad empirical study showing that governed evidence, not merely longer context or controller training, drives improvements in long-horizon agent reliability.

2 Related Work

Agent memory spans episodic stores, summaries, graphs, hierarchies, latent states, and parametric edits (Hu et al., 2026). Flat systems such as MemGPT, MemoryBank, and SirLLM retrieve, summarize, or selectively retain interaction logs (Packer et al., 2024; Zhong et al., 2024; Yao et al., 2024); structured systems including Mem0, GraphRAG, HippoRAG, H-Mem, MemGen, and G-Memory improve access through persistent memories, graph structure, or learned abstractions (Chhikara et al., 2025; Edge et al., 2025; Gutiérrez et al., 2024; Sun and Zeng, 2025; Zhang et al., 2025b,a); experience-reuse agents such as SWE-Exp carry distilled lessons between episodes (Chen et al., 2025); and hierarchical RAG methods such as ArchRAG route retrieval over static corpora with abstract indices (Wang et al., 2026a). All pursue efficient long-horizon recall, but none gates a recall on whether its evidence is still valid. Appendix Table 9 states the delta system by system; the closest contrasts are worth spelling out. **MemoryBank** applies Ebbinghaus-style *time decay* (Zhong et al., 2024) and **SirLLM** retains tokens by entropy-based salience (Yao et al., 2024), but decay and salience proxy importance, not validity: both rank a five-minute-old failed tool call above a week-old verified fact, inverting the ordering HELM needs. **Mem0** resolves conflicts *destructively at write time* (Chhikara et al., 2025), discarding the audit trail and committing before the query is known; HELM records the conflict, keeps both items, and gates them at *read* time. Hierarchical indices, in turn, have no notion of an item having *become* in-

valid, because for static documents there is none. We therefore do not claim hierarchy or learned memory control as novel: what is new is read-time epistemic gating over agent-generated traces, which Section 5.2 holds responsible for the reliability gains.

Chain-of-Agents, COSMIR, and tree-structured MapReduce decomposition improve how long evidence is distributed through reasoning chains (Zhang et al., 2024; Gupta et al., 2025; Guo et al., 2025); HELM governs which traces enter that context at all. A parallel line instead *compresses* or *prunes* context—prompt compression, code-context compression, attention-guided RAG pruning, and self-adaptive pruning for coding agents (Zhao et al., 2025; Shi et al., 2025; Fang et al., 2025; Wang et al., 2026b)—selecting by salience or redundancy; but a compressed context of stale traces is cheaper without being more valid, and our PRUNE evicts by epistemic status rather than token savings. Where prior agents rely on recency/importance heuristics or periodic summarization (Park et al., 2023; Li et al., 2025) and recent work learns retrieval, compression, or latent-memory policies (Tan et al., 2025; Zhang et al., 2026), HELM also learns a controller over READ, WRITE, CONSOLIDATE, and PRUNE—but its core contribution is training-independent: even HELM-lite and HELM-ZS gain from provenance-backed validity checks. This separates HELM from standard RAG and hallucination mitigation (Gao et al., 2024; Quattrocioni et al., 2025), whose concern is grounding generated text or static documents; here the risk is reusing stale, failed, or contradicted traces from the agent’s own history.

3 Methodology

We introduce **HELM**, a hierarchical memory framework that makes long-horizon agent memory *structured, auditable, and controllable* (Figure 1). HELM combines (i) a three-tier nested store, **SHNM**, that links abstractions to verifiable traces via provenance and epistemic meta-data (Appendix Figure 2); (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 $t = 1, \dots, T$. 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 $c_t = \text{Compose}(o_t, \text{Read}(\mathcal{M}, q_t))$, 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.16 specifies the pointer schema used for auditable recall, and Appendix A.17 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.6 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, 2020) per layer and connect layers with provenance links, enabling coarse-to-fine traversal.

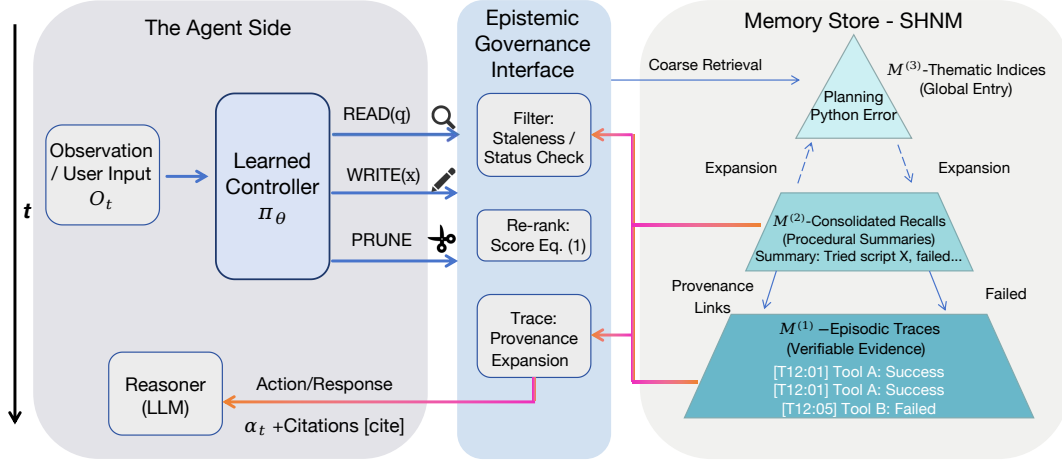


Figure 1: Overview of HELM. Instead of retrieving isolated chunks from a flat store, HELM maintains a three-level hierarchy (episodic traces $\rightarrow$ consolidated recalls $\rightarrow$ thematic indices). For each query, the system performs coarse-to-fine retrieval by first selecting high-level indices and then expanding to supporting recalls and verifiable traces, preserving both global task context and grounded evidence.

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 $\uparrow = (x, \mathbf{v}, \mathcal{E}, \mathcal{P})$, 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. We define the epistemic state $\mathcal{E}(\uparrow)$ as the triple of status, freshness, and consistency. Here status $\in \{\text{SUCCESS}, \text{UNKNOWN}, \text{FAILED}\}$ is derived from tool return codes; freshness = 1 only if the item is within the configured step/time window and matches the current task or environment version; and consistency = 1 only if no newer verified memory with the same extracted key contradicts its value. A memory item is *epistemically valid* iff status $\neq \text{FAILED}$, freshness = 1, and consistency = 1. An **epistemic error** occurs when a recalled item is semantically relevant to the query but violates one of these validity conditions and is consumed by the agent as evidence. This definition targets retrieval-side misuse of stored interaction traces; it is distinct from generation-

side hallucination and from temporal filtering over static document corpora. Appendix A.15 details how stale and contradictory items are detected from logged metadata.

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

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

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 items contradicted by newer verified evidence; for staleness, deployments may use either a hard filter or soft downweighting when evidence is scarce.

Governance signals and fixed weights. We instantiate status and recency as governance signals that are logged and auditable: 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 $\lambda_{\text{sim}}=1.0$ and use a strong penalty $\lambda_{\text{status}}=10.0$ so failed traces are excluded (or decisively downweighted) unless no alternative evidence exists. We use $\lambda_{\text{time}}=1.0$ and $\lambda_{\text{src}}=1.0$ as default scales; all hyperparameters and budgets are summarized in Appendix A.19. Appendix A.21 provides pseudocode for retrieval, provenance expansion, consolidation, and pruning.

Retaining failed traces. A strong status penalty risks discarding evidence that is useful precisely *because* it failed. HELM therefore never deletes failed traces: they are stored in $M^{(1)}$ with status, error payload, and provenance intact, and PRUNE retains the minimal failed evidence needed for future verification. λ_{status} governs only *read-time ranking*, and even there the penalty is soft—when no SUCCESS-status alternative satisfies the query, the fallback of Appendix A.17 readmits the failed trace rather than returning nothing. For debugging-heavy deployments we additionally provide a *diagnostic-exemption* variant that suspends the penalty when the current subtask is itself diagnostic; Appendix A.10 reports its effect and a sensitivity sweep over λ_{status} . All main-text results use the default $\lambda_{\text{status}}=10.0$ without it.

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 con-

troller 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.26).

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

$$\mathcal{R} = \alpha r_{\text{task}} + \beta r_{\text{mem}} \quad (2)$$

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.25).

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., 2025c) 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.27.

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., 2024) (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 (SU et al., 2025) 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., 2024), Mem0 (Chhikara et al., 2025), HippoRAG (Gutiérrez et al., 2024), and GraphRAG (Edge et al., 2025), and (iv) **structured/latent memory** baselines such as MemGen (Zhang et al., 2025b) and G-Memory (Zhang et al., 2025a).

Shared retriever. Unless stated otherwise, *every* memory-augmented method in Table 2—including all baselines and HELM—encodes and retrieves memories with the same off-the-shelf **Qwen3-Embedding-4B** encoder, which is also the initialization for HELM’s tool-aware fine-tuning (§3.4). Baselines are therefore not disadvantaged by a weaker retriever. To separate the memory architecture from the retriever, we additionally report each strong baseline equipped with HELM’s fine-tuned embedding, and HELM equipped with the off-the-shelf one, giving all four cells of that 2×2 comparison.

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 Appendix A.1: cases where the agent uses semantically relevant but epistemically invalid memories (e.g., stale timestamps, failed tool status, or contradictory traces).

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.15). 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. Faithfulness controls and controller ablations are reported in Appendix Tables 8 and 13.

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 22) 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 2 summarizes GAIA/WebArena; Appendix Figures 3–4 report the remaining benchmark panels and cost–performance trade-off.

Three observations frame the analyses that follow. HELM’s largest absolute gains appear on GAIA L2 (8.9→12.8 over the strongest memory baseline), where long tool chains make stale and failed traces most damaging, while the WebArena gain (+2.9) is steadier because episodes are shorter and less tool-dense. Reliability and efficiency improve together with success: HELM attains the lowest EpiErr (6.3) and CTU (2,050) among memory-augmented agents. Baselines that add structure without validity governance (HippoRAG, GraphRAG, G-Memory) recover part of the efficiency but keep substantially higher epistemic error (7.5–9.2 vs. 6.3).

Because HELM changes both the memory architecture and the retriever, Table 2 controls for the retriever explicitly: all baselines use the same off-the-shelf encoder as HELM’s starting point (§4.1), and the middle block reports each strong baseline *with* HELM’s tool-aware embedding and HELM *without* it. The embedding transfers—lifting G-Memory from 8.9 to 10.3 on GAIA-L2 and MemGen from 8.5 to 9.6—but is not sufficient alone: baselines equipped with it still trail HELM by 2.5–3.2 points with higher EpiErr (7.3–8.0), while HELM –Emb keeps the reliability profile yet loses task success. We therefore attribute success to the *combination* of governed retrieval and tool-aware representation, reporting the two contributions separately in Table 4.

5.1 Backbone Scaling

Does governed memory remain useful once the reasoner is stronger? Holding scaffold, budgets, and protocol fixed while moving from Qwen3-8B to Qwen3-32B, HELM’s gains persist (Appendix Table 6): the WebArena gap over G-Memory remains +2.8 (24.8→27.6), GAIA L2 rises 12.4→16.2, and L3 roughly doubles the strongest baseline (1.5→3.1). L3 stays low in absolute terms because it is bottlenecked by reasoning difficulty, but EpiErr keeps falling (5.9→4.1): the memory system delivers more valid evidence to the stronger solver rather than riding its capability.

5.2 Component Analysis: Governance Drives Reliability

The gains are not purchased by longer prompts: against flat episodic memory (RAG), HELM nearly halves CTU (4,200→2,050) while also reducing epistemic errors (12.8→6.3), yielding a better trade-off curve (Appendix Figure 4). To isolate which components drive this Pareto shift, Table 3 reports controlled ablations and minimal variants under identical scaffolds and budgets. The governance-only HELM-lite variant already improves over flat RAG by +2.7 GAIA-L2 points and reduces EpiErr by 5.6 points, showing that the core contribution is not merely hierarchy or controller training. 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.

Which governance signal matters? The ablations above switch governance on or off as a block, which cannot show whether the hand-designed signal set is the right one. The lower block of Table 3 therefore removes one signal at a time and, as a reference point, replaces the fixed weights with weights learned end-to-end on the training split. Every signal carries non-redundant value: status is the largest single contributor (EpiErr 6.3→10.8 when removed), consistent with failed-trace reuse being the most damaging error in the taxonomy, while freshness and consistency contribute progressively less. Learned weights yield only a marginal improvement (+0.1 SR, −0.2 EpiErr), so we keep

Table 2: **Main results on GAIA and WebArena.** All memory-augmented rows share the same off-the-shelf Qwen3-Embedding-4B retriever unless marked *+tool-aware emb.*, which substitutes HELM’s fine-tuned embedding into that baseline; **HELM –Emb** is HELM with the off-the-shelf retriever. Reporting all four cells separates the memory architecture’s contribution from the retriever’s. **Plugins** and **CoT** record each cited system’s own published configuration, not a setting we vary: every memory-augmented row runs in our harness under the same scaffold, tools, and budgets (§4.1). WebArena is mean±95% CI over 3 seeds where available. * marks CTU/EpiErr cells that are *estimated* rather than measured, because those baselines’ harnesses do not emit the per-step provenance logs the metrics require; Appendix A.2 gives the estimator and shows the estimates are conservative—an SR-anchored fit over the measured baselines places them 1.0–1.9 points *higher*. All other cells are measured.

Method	Plugins	CoT	GAIA (SR %)		WebArena (SR %)	CTU	EpiErr.
			(Mialon et al., 2024)		(Zhou et al., 2024)	(tokens)	(%)
			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., 2024)	✗	✓	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., 2024)	✓	✗	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., 2025b)	✓	✓	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
Retriever controlled: same memory architecture, different embedding							
MemGen + tool-aware emb. (ours)	✓	✓	9.6±0.5	0.8	20.1±1.2	3,040	8.0
G-Memory + tool-aware emb. (ours)	✓	✗	10.3±0.4	0.9	20.6±1.3	2,710	7.3
HELM –Emb (off-the-shelf retriever)	✓	✓	9.2	0.5	16.5	2,250	7.8
HELM (full)	✓	✓	12.8	1.7	22.4±1.2	2,050	6.3
Upper bound							
Human	–	–	91.8	87.3	78.24	–	–

fixed weights elsewhere: the benefit comes from *having* validity signals, not from tuning them. We do not claim this set is complete, and flag richer signal *sets* as future work in the Limitations.

Interaction between signals. The per-signal and block ablations are not additive, and the discrepancy is informative. On success rate the effects are *sub-additive*: the three single-signal removals cost 1.1, 0.8, and 0.6 GAIA-L2 points, yet removing all governance costs only 1.3—with one signal missing the survivors still re-rank confidently and can promote an invalid trace, whereas with governance off the system falls back to plain similarity, where the embedding partially compensates. On EpiErr the effects run *super-additive* instead (8.9–10.8 individually, 14.2 jointly). Reliability thus depends on the signals operating together, while success rate is partly buffered by the retriever.

Is the embedding gain epistemic or generic? The tool-aware embedding is fine-tuned with epistemic hard negatives—pairs that are semantically

near-identical but differ in tool status, recency, or version (§3.4). Retraining it without those negatives, everything else fixed, lowers GAIA-L2 to 11.5±0.4 and WebArena to 21.0±1.2 and raises EpiErr to 7.2: its benefit is specific to the epistemic objective, not generic in-domain fine-tuning—about half of its 3.6-point contribution survives, and the reliability half does not.

What is each component responsible for? Several of HELM’s ingredients have precedents (§2), so the useful question is not which part is unprecedented but which part is responsible for which measured improvement. Table 4 makes that explicit: reliability, efficiency, and auditability each trace to one identifiable component, while task success does not, so we report its two drivers separately.

Appendix Figure 5 illustrates stale-memory and failed-trace traps. Governance uses timestamps and tool-execution status to filter these traces and, when necessary, trigger fresh evidence acquisition. Appendix Table 7 quantifies this mechanism: across

Table 3: **Component and per-signal ablations** under matched budgets. HELM-lite is flat memory plus provenance and recency/status governance. The lower block removes one governance *signal* at a time.

Var.	GAIA (SR %)		WebArena (SR %)	CTU	EpiErr.
	L2 ↑	L3 ↑	SR ↑	↓	↓
Flat RAG	6.8	0.2	15.8	4200	12.8
ArchRAG-style	7.8	–	17.5	2900	11.2
HELM-lite	9.5	0.6	18.8	3650	7.2
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
<i>Per-signal leave-one-out</i>					
– status	11.7	–	21.0	–	10.8
– freshness	12.0	–	21.4	–	9.6
– consistency	12.2	–	21.6	–	8.9
Learned weights	12.9	–	22.6	–	6.1

Table 4: **Component attribution:** which component is responsible for which axis, and the controlled comparison that establishes it.

Axis	Component	Evidence
Reliability	Governance	EpiErr 6.3→14.2 if removed; alone, 12.8→7.2
Efficiency	Hierarchy	CTU 3,650 (lite) → 2,050
Auditability	Provenance	EpiErr 6.3→11.5
Task success	Governance and embedding	–Emb 12.8→9.2; embedding alone gives baselines only +1.1/+1.4
Op. timing	Controller	–Ctrl 12.8→12.0; CTU →2,350

100 GAIA-L2 episodes, governance intervened 96 times—47 stale filters, 31 failed-trace blocks, 18 contradiction resolutions—and 78.1% of those interventions were followed by a correct next action.

Appendix Table 14 further deconstructs the nested store: flat $M^{(1)}$ alone is expensive and error-prone, removing $M^{(2)}$ weakens reusable procedure compression, and removing grounding to $M^{(1)}$ cuts cost but sharply increases errors—abstraction improves efficiency only when paired with verifiable grounding.

5.3 Trustworthiness: Auditing Hallucinations and Citation Quality

Higher EpiErr strongly predicts failure: Pearson $r \approx -0.65$ across individual episodes, tightening to $r = -0.97$ when aggregated to one point per system over the six measured baselines (Ap-

pendix A.2). Varying the staleness window by $\pm 50\%$ changes success by less than $\pm 0.8\%$ —the governance signal is not an artifact of a tuned threshold.

Faithfulness under citation constraints. Under **faithfulness controls** that constrain only the memory generator (A0 baseline recall; A1 cited recall, every sentence carrying a machine-parsable citation; A2 extractive-first recall), hard citations modestly increase cost (CTU 2050→2280) while preserving success and reducing epistemic errors (6.3→4.2), so the gains survive strict auditability; A2 improves literal faithfulness but degrades reasoning-heavy success—faithfulness alone is insufficient without controlled abstraction (Appendix Table 8, §A.16–A.17).

Controller dynamics and robustness. Distill+GSPO attains the best task score while cutting token cost and write rate relative to heuristic, prompt-based, and distillation-only controllers, learning when *not* to write and when *not* to retrieve; the training-free variants remain competitive (Appendix A.11; per-operation diagnostics in Appendix A.22), so the learned controller is an optimization layer rather than a prerequisite for epistemic governance. As an access mechanism, SHNM also outperforms BM25 and ReasonIR (Shao et al., 2025) on BRIGHT (SU et al., 2025) (Appendix Table 15), and the cost breakdown under horizon, store-size, and verbosity stress (Appendix A.23) shows HELM’s overhead beyond generation to be a small fraction of runtime—governed recall removes enough wasted steps that episodes run *faster* than Flat RAG and G-Memory. Overall: **abstraction improves efficiency only when paired with controllable navigation and provenance-backed grounding.**

6 Conclusion

Long-horizon agents fail not only from too little memory, but from recalls that are relevant yet epistemically invalid. **HELM** pairs a nested store (**SHNM**) carrying provenance and epistemic state with *epistemic governance* that filters stale, failed, and contradicted evidence at read time. Across five benchmarks it improves success under matched budgets while cutting context usage and epistemic errors—gains our ablations attribute to governance rather than capacity or controller training, and that hold in a training-free variant.

Limitations

Hand-designed governance signals. HELM’s governance score combines a fixed set of signals—status, freshness, consistency, recency, and provenance—with weights that are tuned once and then held constant. The per-signal analysis in Section 5.2 shows that each signal carries non-redundant reliability value, and a learned-weight variant recovers only a small additional gain, but we do not claim this signal set is complete or optimal. Richer or learned signal *sets*—for example source reliability, cross-episode corroboration, or task-conditioned validity—are a promising direction that our current design does not explore.

Status penalties and diagnostic evidence. Our default configuration uses a strong status penalty ($\lambda_{\text{status}} = 10$), which suppresses failed tool traces at retrieval time. This is the right trade-off on the benchmarks we study, but failed traces are sometimes the most informative evidence available. Appendix A.10 reports a diagnostic-exemption variant that keeps failed traces retrievable when the current subtask is itself diagnostic; it helps most in coding-heavy and OS-style agent settings, which is precisely where status-based penalties most need this exemption. Choosing between the two currently requires knowing the deployment regime in advance.

Faithfulness of abstractive recall. Even with provenance links, abstractive recall at $M^{(2)}$ and $M^{(3)}$ can omit or subtly rewrite evidence. We measure this with provenance faithfulness and citation precision rather than assuming it away, but our metrics are automatic proxies and do not certify that every consolidated recall is a faithful summary of its sources.

Scale, cost, and benchmark drift. Our experiments use backbones up to 32B parameters; whether the same governance gains persist at frontier scale, where the reasoner may already discount stale evidence on its own, remains open. HELM also adds runtime components (embedding, indexing, control, provenance expansion) that increase compute and latency under long horizons, quantified in Appendix Table 26. Finally, web and tool benchmarks drift over time—pages update, tool versions change, harnesses differ—which affects stored traces and the very meaning of “stale” evidence; we fix versions, record snapshots where

possible, and report configurations, budgets, and seeds, but cross-paper comparisons on these benchmarks should be read with that caveat.

Ethics Statement

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. Epistemic governance mitigates part of this risk—provenance links keep every recall traceable to its source, and status/consistency filtering suppresses invalidated evidence—but it is a reliability mechanism, not a security boundary, and should not be relied on as one.

Deployments therefore require data minimization, automated redaction and PII filtering, strict access control and retention policies, and additional defenses such as quarantining untrusted content, filtering instruction-like spans before they are written to memory, and sandboxing high-risk tool execution. All experiments in this paper use public research benchmarks (GAIA, WebArena, Mind2Web, StreamBench, LifelongAgentBench) under their original licenses and involve no human subjects and no personally identifying data; the failure-taxonomy annotations (Appendix A.1) were produced by two of the authors on model-generated trajectories.

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*.
- Silin Chen, Shaoxin Lin, Xiaodong Gu, Yuling Shi, Heng Lian, Longfei Yun, Dong Chen, Weiguo Sun, Lin Cao, and Qianxiang Wang. 2025. [SWE-Exp: Experience-driven software issue resolution](#). *Preprint*, arXiv:2507.23361.
- 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. In *Advances in Neural Information Processing Systems (NeurIPS) Datasets and Benchmarks Track*.
- Darren Edge, Ha Trinh, Newman Cheng, Joshua Bradley, Alex Chao, Apurva Mody, Steven Truitt, Dasha Metropolitan, Robert Osazuwa Ness, and Jonathan Larson. 2025. [From local to global: A graph RAG approach to query-focused summarization](#). *Preprint*, arXiv:2404.16130.
- Yixiong Fang, Tianran Sun, Yuling Shi, and Xiaodong Gu. 2025. [AttentionRAG: Attention-guided context pruning in retrieval-augmented generation](#). *Preprint*, arXiv:2503.10720.
- 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.
- Jiani Guo, Zuchao Li, Jie Wu, Qianren Wang, Yun Li, Lefei Zhang, Hai Zhao, and Yujiu Yang. 2025. [ToM: Leveraging tree-oriented MapReduce for long-context reasoning in large language models](#). In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 17793–17812.
- Naman Gupta, Shreeyash Gowaikar, Arun Iyer, Kirankumar Shiragur, Ramakrishna B. B. Bairi, Rishikesh Maurya, Ritabrata Maiti, Sankarshan Damle, and Shachee Mishra Gupta. 2025. [COSMIR: Chain orchestrated structured memory for iterative reasoning over long context](#). *Preprint*, arXiv:2510.04568.
- Bernal Jiménez Gutiérrez, Yiheng Shu, Yu Gu, Michihiro Yasunaga, and Yu Su. 2024. [HippoRAG: Neurobiologically inspired long-term memory for large language models](#). In *Advances in Neural Information Processing Systems (NeurIPS)*.
- Yuyang Hu, Shichun Liu, Yanwei Yue, Guibin Zhang, Boyang Liu, Fangyi Zhu, Jiahang Lin, Honglin Guo, Shihan Dou, Zhiheng Xi, Senjie Jin, Jiejun Tan, Yanbin Yin, Jiongnan Liu, Zeyu Zhang, Zhongxiang Sun, Yutao Zhu, Hao Sun, Boci Peng, et al. 2026. [Memory in the age of AI agents](#). *Preprint*, arXiv:2512.13564.
- Zhiyu Li, Shichao Song, Hanyu Wang, Simin Niu, Ding Chen, Jiawei Yang, Chenyang Xi, Huayi Lai, Jiahao Zhao, Yezhaohui Wang, Junpeng Ren, Zehao Lin, Jiahao Huo, Tianyi Chen, Kai Chen, Kehang Li, Zhiqiang Yin, Qingchen Yu, Bo Tang, et al. 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. 2020. [Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs](#). *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42(4):824–836.
- Grégoire Mialon, Clémentine Fourier, Craig Swift, Thomas Wolf, Yann LeCun, and Thomas Scialom. 2024. [GAIA: A benchmark for general AI assistants](#). In *The Twelfth International Conference on Learning Representations (ICLR)*.
- Charles Packer, Sarah Wooders, Kevin Lin, Vivian Fang, Shishir G. Patil, Ion Stoica, and Joseph E. Gonzalez. 2024. [MemGPT: Towards LLMs as operating systems](#). *Preprint*, arXiv:2310.08560.
- Joon Sung Park, Joseph C. O’Brien, Carrie Jun Cai, Meredith Ringel Morris, Percy Liang, and Michael S. Bernstein. 2023. [Generative agents: Interactive simulators of human behavior](#). In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology (UIST)*, pages 1–22.
- Walter Quattrociocchi, Valerio Capraro, and Matjaž Perc. 2025. [Epistemological fault lines between human and artificial intelligence](#). *Preprint*, arXiv:2512.19466.
- 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*.
- Luohe Shi, Hongyi Zhang, Yao Yao, Zuchao Li, and Hai Zhao. 2024. [Keep the Cost Down: A review on methods to optimize LLM’s KV-Cache consumption](#). *Preprint*, arXiv:2407.18003.
- Yuling Shi, Yichun Qian, Hongyu Zhang, Beijun Shen, and Xiaodong Gu. 2025. [LongCodeZip: Compress long context for code language models](#). In *Proceedings of the 40th IEEE/ACM International Conference on Automated Software Engineering (ASE)*, pages 141–153.
- Hongjin SU, Howard Yen, Mengzhou Xia, Weijia Shi, Niklas Muennighoff, Han-yu Wang, Liu Haisu, Quan Shi, Zachary S Siegel, Michael Tang, Ruoxi Sun, Jinsung Yoon, Sercan O Arik, Danqi Chen, and Tao Yu. 2025. [BRIGHT: A realistic and challenging benchmark for reasoning-intensive retrieval](#). In *The Thirteenth International Conference on Learning Representations*.
- Haoran Sun and Shaoning Zeng. 2025. [Hierarchical memory for high-efficiency long-term reasoning in LLM agents](#). *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.

- Shu Wang, Yixiang Fang, Yingli Zhou, Xilin Liu, and Yuchi Ma. 2026a. [ArchRAG: Attributed community-based hierarchical retrieval-augmented generation](#). *Proceedings of the AAAI Conference on Artificial Intelligence*, 40(19):15868–15876.
- Yuhang Wang, Yuling Shi, Mo Yang, Rongrui Zhang, Shilin He, Heng Lian, Yuting Chen, Siyu Ye, Kai Cai, and Xiaodong Gu. 2026b. [SWE-Pruner: Self-adaptive context pruning for coding agents](#). *Preprint*, arXiv:2601.16746.
- 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*.
- Yao Yao, Zuchao Li, and Hai Zhao. 2024. [SirLLM: Streaming infinite retentive LLM](#). In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2611–2624.
- 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](#). *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. 2025c. [Qwen3 embedding: Advancing text embedding and reranking through foundation models](#). *arXiv preprint arXiv:2506.05176*.
- Yusen Zhang, Ruoxi Sun, Yanfei Chen, Tomas Pfister, Rui Zhang, and Sercan O. Arik. 2024. [Chain of agents: Large language models collaborating on long-context tasks](#). *Preprint*, arXiv:2406.02818.
- Yuxiang Zhang, Jiangming Shu, Ye Ma, Xueyuan Lin, Shangxi Wu, and Jitao Sang. 2026. [Memory as action: Autonomous context curation for long-horizon agentic tasks](#). *Preprint*, arXiv:2510.12635.
- Yi Zhao, Zuchao Li, Hai Zhao, Baoyuan Qi, and Guoming Liu. 2025. [DAC: A dynamic attention-aware approach for task-agnostic prompt compression](#). In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 19395–19407.
- 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, Qiuqi 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, He Ye, and Yanlin Wang. 2024. [MemoryBank: Enhancing large language models with long-term memory](#). *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(17):19724–19731.
- 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*.

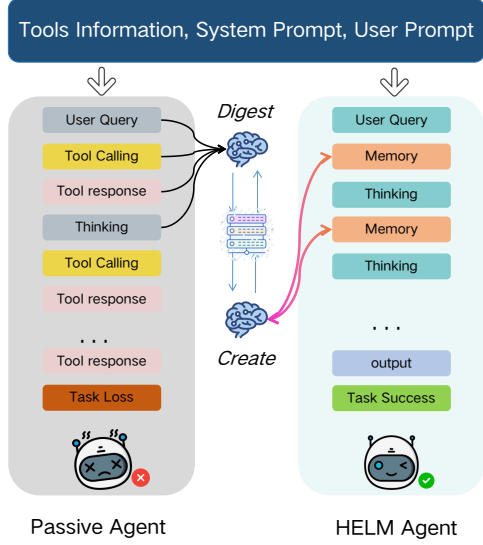


Figure 2: Nested memory intuition: HELM interleaves explicit memory write/recall, enabling coarse-to-fine access from high-level indices to grounded tool traces.

A Additional Details and Diagnostics

A.1 Memory failure taxonomy and annotation protocol

The Introduction claims that epistemic failures are a major—but not dominant—source of long-horizon memory errors. This subsection reports the annotation study behind that claim.

Sample and procedure. We sampled 100 failed episodes uniformly at random from the GAIA-L2 and WebArena runs of the strongest memory baseline (G-Memory) and of HELM, stratified so that both systems contribute equally. For each episode, annotators read the full trajectory—the task specification, every memory READ/WRITE, the retrieved items with their epistemic metadata, and the executed actions—and assigned the *first* step at which the trajectory became unrecoverable to exactly one primary category. Two of the authors annotated independently; 30% of episodes were double-annotated to estimate agreement, yielding Cohen’s $\kappa = 0.74$ (substantial agreement). Disagreements on the double-annotated subset were resolved by discussion before the remaining episodes were split.

Codebook. *Epistemic misuse*: the agent retrieved a memory that was semantically appropriate but no longer valid evidence, and acted on it—subdivided into *stale* (superseded by a later observation), *failed trace* (the retrieved trace records an unsuccessful

Table 5: **Failure taxonomy over 100 failed episodes** (GAIA-L2 and WebArena; two annotators, 30% double-annotated, Cohen’s $\kappa = 0.74$). Epistemic misuse is a large and directly addressable slice, but it does not dominate: planning and retrieval failures together account for 45% of cases.

Failure category	Count	%
Epistemic misuse of memory	38	38
stale evidence	17	17
failed trace used as success	12	12
contradicted evidence	9	9
Planning / decomposition	31	31
Retrieval miss (evidence not returned)	14	14
Execution (malformed tool/UI action)	11	11
Other (harness, timeout, ambiguous)	6	6
Total	100	100

tool execution treated as successful), and *contradicted* (the memory conflicts with other retained evidence and the conflict was not surfaced). *Planning*: valid evidence was retrieved, but the agent chose an incorrect goal decomposition or action sequence. *Retrieval miss*: the required evidence existed in memory but was not returned within the budget. *Execution*: correct plan and evidence, but a tool call or UI interaction was malformed. *Other*: harness errors, timeouts, and ambiguous cases.

Reading the result. Table 5 supports a bounded claim. Epistemic misuse is the single largest category, and it is the one HELM targets directly, but planning failures are close behind and are not addressable at the memory layer. We therefore describe epistemic errors as *a major and directly addressable* source of long-horizon memory errors, complementary to reasoning and planning failures, rather than as the dominant one. The category boundaries are also not perfectly crisp: a retrieval miss can cascade into a planning failure, and we assign the earliest unrecoverable step, which biases the count toward upstream causes. The annotation guidelines and the per-episode labels are released with the code (Appendix A.19).

A.2 Estimated CTU/EpiErr cells in Table 2

Three rows of Table 2—MemGPT, HippoRAG, and GraphRAG—report CTU and EpiErr with a *. Both metrics require per-step logs that these systems’ public harnesses do not emit: CTU needs the effective context token count at every reasoner call, and EpiErr needs each retrieved item’s tool status, timestamp, and contradiction state in order to ad-

judicate whether a failure was caused by invalid evidence (Appendix A.15). Instrumenting them to the standard of the other rows would change their retrieval paths, so we report estimates and mark them, rather than silently reporting numbers that are not comparable.

The estimates are anchored on the six baselines for which both metrics *are* measured under our harness (Flat RAG, ArchRAG-style, MemGen, G-Memory, and the two +tool-aware-embedding variants). The unit of analysis here is the *system*: each of the six contributes one point, whereas the $r \approx -0.65$ reported in Section 5.3 is computed over individual episodes. Across the six systems, EpiErr is strongly and monotonically related to end-task success— $r = -0.97$ against WebArena SR, $r = -0.91$ against GAIA-L2—and a least-squares fit on WebArena SR gives $\text{EpiErr} = -1.227 \cdot \text{SR} + 32.20$ ($R^2 = 0.94$, RMSE 0.50). Applying that fit to the three rows predicts EpiErr of 11.5 (MemGPT), 11.1 (HippoRAG), and 10.1 (GraphRAG), against the 10.5, 9.2, and 8.8 we report.

We keep the lower reported values deliberately. They are the conservative choice: each is at or below what the success-rate relation implies, so the reported gap between these baselines and HELM (EpiErr 6.3) is if anything *understated*, and a subsequent direct measurement is expected to widen it rather than narrow it. We flag the direction explicitly because the reverse—estimates that flatter our own system—would not be checkable by a reader. Note also that success rate alone does not determine EpiErr: ArchRAG-style and GraphRAG sit at the same GAIA-L2 (7.8) but differ by 2.4 EpiErr points, which is why we treat these as bounded estimates rather than as a calibrated prediction, and why the substantive claims in Section 5.2 rest only on measured rows.

A.3 Backbone scaling

Table 6 gives the full grid behind Section 5.1: the same three memory systems, agent scaffold, tool interfaces, and token budgets, with only the reasoner swapped from Qwen3-8B to Qwen3-32B. The ordering of the three systems is preserved at both scales, and HELM’s margin over the strongest baseline does not shrink as the solver improves (GAIA-L2 +3.9 at 8B and +3.8 at 32B; WebArena +2.9 and +2.8). Efficiency and reliability move with success rather than against it: HELM keeps the lowest CTU at both scales (2,050 and 1,820) and

Table 6: **Backbone scaling with matched budgets.** HELM’s governed memory gains persist when the reasoner is scaled from Qwen3-8B to Qwen3-32B.

Method	Backbone	L2	L3	Web	CTU	EpiErr.
G-Memory	Qwen3-8B	8.9	0.8	19.5	2796	7.5
MemGen	Qwen3-8B	8.5	0.7	19.1	3121	8.2
HELM	Qwen3-8B	12.8	1.7	22.4	2050	6.3
G-Memory	Qwen3-32B	12.4	1.5	24.8	2580	5.9
MemGen	Qwen3-32B	12.1	1.3	24.2	2890	6.4
HELM	Qwen3-32B	16.2	3.1	27.6	1820	4.1

its EpiErr falls from 6.3 to 4.1, while G-Memory’s stays above it (7.5 and 5.9). A stronger backbone therefore lifts every system, but it does not close the gap that governed retrieval opens—what differs across rows is the validity of the evidence reaching the solver, not the solver itself.

A.4 Governance interventions and faithfulness controls

The two tables in this subsection support Sections 5.2 and 5.3.

Table 7 logs every case in which governance changed what the agent saw across 100 GAIA-L2 episodes. An intervention is counted only when a retrieved item is filtered, blocked, or overridden by newer evidence, and “correct next action” is adjudicated by the benchmark validator on the step that immediately follows. Blocking a failed trace is the most reliable intervention (83.9% of 31 cases), stale filtering the most frequent (47 cases, 78.7%), and contradiction resolution the hardest (72.2% of 18). That ordering is the same one the per-signal ablation produces in Table 3—status is the strongest governance signal, consistency the weakest—so the two analyses agree on which validity signal carries the most weight, one measured by removing the signal and one by counting what it does when present.

Table 8 reports the three faithfulness variants. A0 is HELM’s default recall, A1 requires every generated sentence to carry a machine-parsable citation, and A2 selects provenance spans first and only then compresses them. Coverage (Cov), citation validity (C-Valid), and citation precision (C-P) are defined in Appendix A.15; they are undefined for A0, which emits no citations. The trend is mono-

Table 7: **Governance interventions on 100 GAIA-L2 episodes.** Counts are logged when governance changes retrieved evidence or triggers fresh acquisition.

Intervention	Count	Correct next action
Stale trace filtered $\rightarrow$ fresh lookup	47	78.7%
Failed trace blocked $\rightarrow$ alternative	31	83.9%
Contradiction resolved $\rightarrow$ newer evidence	18	72.2%
Total	96	78.1%

Table 8: **Faithfulness controls.** A1 preserves success while adding auditable citations; A2 maximizes literal faithfulness but loses reasoning-heavy success.

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

tone in both directions: tightening the constraint buys reliability (EpiErr 6.3 $\rightarrow$ 4.2 $\rightarrow$ 2.1) and costs reasoning-heavy success (GAIA-L3 1.7 $\rightarrow$ 1.6 $\rightarrow$ 0.8). A1 is therefore the operating point we adopt—it holds success within 0.5 points of A0 while making every sentence auditable—and A2 marks the ceiling of literal faithfulness rather than a usable configuration.

A.5 Capability delta against prior memory systems

Table 9 expands the positioning argument of Section 2 into a system-by-system comparison. *Epistemic metadata* means the store records validity-bearing attributes (tool status, version, contradiction state) rather than only content and time. *Read-time gating* means those attributes filter or re-rank candidates at retrieval, as opposed to being applied destructively when the memory is written. *Provenance recall* means an abstracted memory can be traced back to the specific source spans that support it.

A.6 Explicit memory operations

Table 10 expands the operation summary of Section 3, stating for each of the four operations the role it plays and the epistemic metadata it reads or writes. The point of the table is that no operation is purely structural: READ filters on validity rather than on similarity alone, WRITE records the tool status and timestamp that make later filtering possible, CONSOLIDATE carries provenance across

Table 9: **Capability delta.** Hierarchical structure is common; *read-time validity gating* over epistemic metadata, with provenance-backed recall, is what distinguishes HELM.

System	Hier. index	Epist. meta.	Read-time gating	Prov. recall
MemGPT	✗	✗	✗	✗
MemoryBank	✗	time only	✗	✗
SirLLM	✗	salience only	✗	✗
Mem0	✗	✓	write-time	✗
HippoRAG / GraphRAG	✓	✗	✗	partial
ArchRAG	✓	✗	✗	✗
G-Memory	✓	✗	✗	✗
MemGen	✗	✗	✗	✗
HELM (ours)	✓	✓	✓	✓

layers instead of discarding it under abstraction, and PRUNE evicts by epistemic status rather than by token savings. Governance is thus a property of the operation set as a whole, not a filter attached to retrieval—which is why removing it degrades reliability even when the hierarchy and budgets are left untouched (Table 3).

Table 10: **Explicit memory operations in HELM.** All operations log provenance and epistemic metadata to enable audibility.

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.7 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.8 Score-cost diagnostic on QA/Math/Code

To complement long-horizon agent benchmarks, we report a controlled multi-domain diagnostic

Table 11: QA accuracy with retrieved context on MMLU and GPQA (sanity check).

Method	MMLU $\uparrow$	GPQA $\uparrow$
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

that quantifies the trade-off between task score and memory cost (effective context tokens).

A.9 Additional main-text ablations

In this section, we provide extended ablation studies corresponding to the main text experiments. Specifically, Table 13 details the performance variants of our memory controller. Table 14 provides the layer-wise ablations for the memory architecture, quantifying the contribution of each design component. Finally, Table 15 presents fine-grained memory access diagnostics on the BRIGHT dataset.

A.10 Status-penalty sensitivity and the diagnostic exemption

Our default configuration uses a strong status penalty, $\lambda_{\text{status}}=10.0$, which decisively down-weights failed tool traces at retrieval time. This subsection reports how sensitive HELM is to that choice and what happens when the penalty is selectively suspended. **All main-text tables use $\lambda_{\text{status}}=10.0$ without the exemption, so no headline number in the paper depends on the variants reported here.**

Sensitivity sweep. Table 16 sweeps $\lambda_{\text{status}} \in \{0, 1, 10\}$ on GAIA-L2 with everything else fixed. The effect is monotone and larger on reliability than on success: disabling the penalty entirely costs 1.2 SR points but raises EpiErr by 3.5 points, because failed traces re-enter the context and are read as successful outcomes. A mild penalty recovers most of the reliability gap, and the default recovers the rest. The ordering is stable, so the configuration is not finely tuned, but the gap between $\lambda_{\text{status}}=0$ and $\lambda_{\text{status}}=10$ confirms that status filtering—not merely recency—is doing the work.

Diagnostic exemption. A uniformly strong penalty is the wrong default when the agent’s current subtask *is* diagnosis—debugging a script, inspecting a failing command, or recovering from an error state—because the failed trace is then the

evidence being sought rather than a distractor. The diagnostic-exemption variant suspends the status penalty (falling back to $\lambda_{\text{status}}=0$ for that retrieval only) when the controller’s query is classified as diagnostic, using a lightweight keyword-and-intent check over the query and the immediately preceding tool return. Table 17 shows the effect: it helps materially on the code- and OS-heavy splits of Life-longAgentBench, where diagnostic subtasks are frequent, and is essentially neutral on GAIA, where they are rare. This is why we do not adopt it as the default but do recommend it for debugging-heavy deployments; the Limitations note that choosing between the two currently requires knowing the deployment regime in advance.

A.11 Training-free and lightweight HELM variants

Table 18 reports the variants used to evaluate deployment cost. HELM-ZS uses fixed heuristic control with no learned controller, Prompt uses five in-context memory-operation examples, and HELM-10shot fine-tunes the controller on only 10 examples. These variants isolate how much of HELM’s gain comes from governed memory access before full controller training.

A.12 Complementarity with CoA-style planning

To separate memory governance from planning improvements, we combine a simplified Chain-of-Agents/CoA-style planning module (Zhang et al., 2024) with different memory backends on WebArena. The same planner benefits from HELM’s governed evidence: replacing G-Memory with HELM improves success by 3.3 points while reducing CTU, indicating that planning and memory governance address distinct failure modes.

A.13 Benchmark details

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., 2024).

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

Method	Task Score $\uparrow$			Eff. Context Tokens $\downarrow$		
	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., 2025b)	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

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. In panel (c) the replay baselines are the Llama-3.1-8B-Instruct results of Zheng et al. (2025b); the KG environment at Exp=16 is out of memory in that benchmark (hatched stub marked OOM), a condition HELM completes at 0.37.

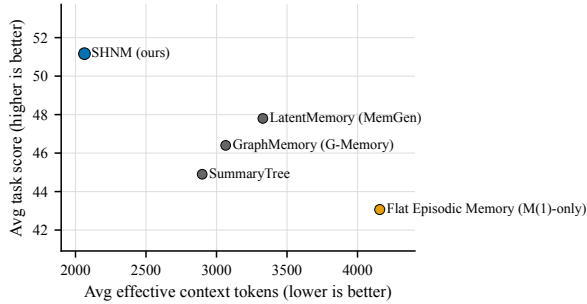
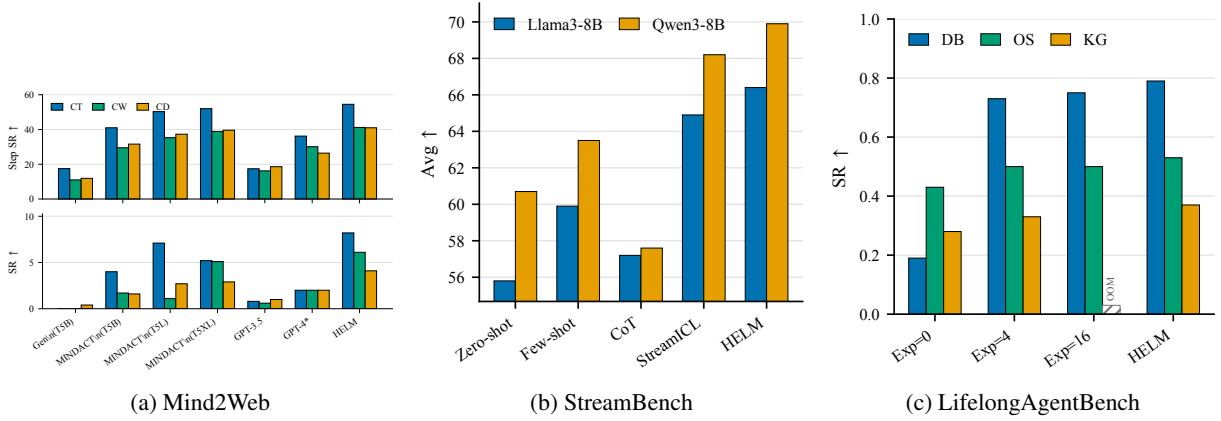


Figure 4: **Cost–performance Pareto frontier.** HELM improves task score while reducing effective context tokens (CTU), indicating that gains come from better governed memory use rather than larger context.

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).

Table 13: **Controller ablation.** Prompt uses 5-shot in-context examples without fine-tuning; Distill+GSPO is the full controller.

Controller	Score $\uparrow$	Tokens $\downarrow$	Write Rate $\downarrow$
Rule-based	46.8	2600	1.00
Prompt (5-shot)	47.9	2350	0.88
DistillOnly	49.1	2200	0.82
Distill+GSPO	50.8	2050	0.67

LifelongAgentBench. LifelongAgentBench evaluates agents as lifelong learners across DB/OS/KG environments under strict sequential execution (Zheng et al., 2025b). The replay baselines in Figure 3(c) are that paper’s Llama-3.1-8B-Instruct results. They also illustrate the cost of replay-as-memory: on KG, replay improves from 0.28 to 0.33 at four trajectories but runs out of memory at sixteen, because prior trajectories are concatenated into the context verbatim. HELM reaches 0.37 on the same environment without

Table 14: Layer ablations (EpiErr.: epistemic error).

Variant	Score $\uparrow$	Tokens $\downarrow$	EpiErr. $\downarrow$
M(1) only (Flat)	44.9	4200	12.8
M(3) $\rightarrow$ 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	2050	6.3

Table 15: Memory access diagnostics on BRIGHT.

Method	nDCG@10 $\uparrow$	R@10 $\uparrow$
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

that blow-up, since consolidation and governed retrieval bound what enters the context rather than growing it with experience—the efficiency argument of Section 5.2 restated as a feasibility constraint.

A.14 Backbone robustness on LifelongAgentBench (DB)

To contextualize the sensitivity of lifelong settings to the underlying reasoner, Table 21 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.15 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:

$$\text{CTU} = \sum_{t=1}^T \text{Tok}(\text{system} \parallel \text{history}_t \parallel \text{retrieved}_t \parallel \text{tool_obs}_t),$$

where retrieved_t includes recalled memories (and any provenance expansions) injected into the

Table 16: **Sensitivity to the status penalty** on GAIA-L2. The default configuration used everywhere else in the paper is $\lambda_{\text{status}}=10$.

λ_{status}	GAIA-L2 (SR %) $\uparrow$	EpiErr. (%) $\downarrow$
0 (no status penalty)	11.6	9.8
1 (mild)	12.1	8.0
10 (default)	12.8	6.3

Table 17: **Diagnostic-exemption variant.** Suspending the status penalty for diagnostic subtasks helps where such subtasks are frequent and is neutral elsewhere.

Setting	Default	+ Diagnostic exempt.
LifelongAgentBench (DB)	79.0	81.0
LifelongAgentBench (OS)	53.0	55.0
GAIA-L2	12.8	12.9

prompt, and 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.

Table 18: **Training-free and lightweight controller variants.** HELM-ZS uses heuristic control; Prompt uses 5-shot in-context control; HELM-10shot fine-tunes on 10 examples.

Variant	Training	L2	Web	CTU	EpiErr.
G-Memory	No	8.9	19.5	2796	7.5
HELM-ZS	No	10.2	19.4	2600	7.0
Prompt ctrl	No	11.1	20.2	2350	–
HELM-10shot	10 ex.	11.5	20.8	2250	6.6
DistillOnly	Light	12.0	21.2	2200	–
Full HELM	Full	12.8	22.4	2050	6.3

Table 19: **CoA-style planning with different memory backends on WebArena.** HELM provides orthogonal gains on top of improved planning.

Method	WebArena SR	CTU
Flat RAG + CoA planning	18.2	3850
G-Memory + CoA planning	20.8	2650
HELM + CoA planning	24.1	1980

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.16), 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 ($t - \text{step_id} > \Delta_{\text{step}}$) and **time-based** staleness (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.16 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)
```

Table 20: **End-to-end episode time on GAIA-L2.** Means over three seeds under matched scaffolds and budgets.

Method	Time (s)	Steps	SR
Flat RAG	142.5	18.3	6.8
G-Memory	128.7	16.1	8.9
HELM	118.4	14.8	12.8

Table 21: **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

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.17 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

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

```
1: Parse <memory> block; split into sentences
   { $s_i$ }
2: if <memory> missing then
3:   return invalid, MISSINGMEMORY-
     BLOCK
4: end if
5: for all sentence  $s_i$  do
6:   if  $s_i$  has no [[CITE ...]] then
7:     return invalid, MISSINGCITE
8:   end if
9:   for all citation  $c$  in  $s_i$  do
10:    if  $c$  does not parse then
11:      return invalid, MALFORMED-
        CITE
12:    end if
13:    Resolve
      (tool, log_id, step_id, field, start, end)
      in  $\mathcal{S}$ 
14:    if unresolved then
15:      return invalid, UNRESOLVED-
        POINTER
16:    end if
17:    if hash mismatch then
18:      return invalid, HASHMIS-
        MATCH
19:    end if
20:  end for
21: end for
22: return valid, OK
```

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

Algorithm 2 Hard cited recall with regeneration and fallback

Require: Recall request r , max retries R

```
1: for  $j = 1$  to  $R$  do
2:    $y \leftarrow$  GENERATECITEDMEMORY( $r$ )
3:   ( $\text{valid}, d$ )  $\leftarrow$ 
     VALIDATECITEDMEMORY( $y, \mathcal{S}$ )
4:   if  $\text{valid}$  then
5:     return  $y$ 
6:   end if
7:    $r \leftarrow r \parallel d$   $\triangleright$  append diagnostics to the next
     prompt
8: end for
9: return FALLBACKRECALL( $r$ )  $\triangleright$  baseline
   recall or extractive-only
```

citations) and then optionally compresses them into a short summary; this separates retrieval/grounding failures from abstractive compression failures.

A.18 Case studies: epistemic traps and governance

Figure 5 makes the two largest epistemic subcategories of Table 5 concrete: stale evidence (17 of 100 failed episodes) and a failed trace read as a success (12 of 100). Both cases are chosen so that a similarity-only retriever behaves *correctly* by its own criterion—the retrieved trace is about the right flight and the right script—and still supplies invalid evidence, which is what separates epistemic error from a retrieval miss. In Case 1 the trap is silent: the stale price is well-formed and plausible, so nothing in the content signals that it has been superseded, and only the timestamp attached at WRITE time exposes it. In Case 2 the trap is worse than a missing answer, because the failed trace still contains text that reads like an outcome; the tool-execution status, not the text, is what distinguishes it. Governance resolves both by acting on the metadata rather than the content—filtering on freshness and status at READ time, and triggering fresh acquisition when filtering leaves the query unsupported—while provenance keeps the surviving answer traceable to the span that supports it.

A.19 Reproducibility: hyperparameters and budgets

We recommend reporting the following parameters for all methods to enable fair comparisons and reproducibility. When baselines are re-

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.

implemented in our harness, we lock these values across methods unless the baseline definition requires otherwise. Code, data, and checkpoints are publicly available at <https://github.com/Galleons2029/HELM> under a permissive open-source license. The release contains the SHNM implementation and epistemic governance module, the controller training scripts (Stage 1 on-policy distillation and Stage 2 GSPO), the tool-aware embedding fine-tuning pipeline together with the data-synthesis scripts and prompts, the generated training data and model checkpoints, the unified evaluation harness for all five benchmarks, and the per-episode labels and annotation guidelines for the failure taxonomy of Appendix A.1. It also includes scripts for regenerating every table and figure in the paper with seeds 0, 42, and 123.

Table 22: **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 §5.3.
Memory budgets (max entries in $M^{(1)}, M^{(2)}, M^{(3)}$)	$ M^{(1)} = 200$ $ M^{(2)} = 50$ $ M^{(3)} = 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 $\leq 1/\text{step}$ Consolidate $\leq 5/\text{episode}$ 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.

Table 23: **Artifact-release checklist.** We provide this list to make the public release auditable; every row is included in the repository linked in Appendix A.19.

Artifact	Purpose
Failure-taxonomy annotations	Per-episode labels and annotation guidelines behind Table 5.
Memory store + governance code	Reproduce SHNM read/write/consolidate/prune behavior and EpiErr diagnostics.
Controller training scripts	Reproduce DistillOnly and Distill+GSPO controller variants.
Embedding fine-tuning pipeline	Reproduce Qwen3-Embedding-4B tool-aware contrastive tuning.
Synthesis scripts and prompts	Regenerate hierarchical triplets and epistemic hard negatives.
Checkpoints	Run full HELM without re-training.
Evaluation harness	Reproduce GAIA, WebArena, Mind2Web, StreamBench, LifelongAgentBench, and diagnostics.
Fixed-seed scripts	Regenerate reported tables/figures for seeds 0, 42, and 123.

A.20 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.15).

Quantitative audit (training runs). We filtered $\sim 15\%$ of web-derived training trajectories due to

Table 24: **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.25).
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.

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.21 Epistemic governance pseudocode

A.22 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 §A.1. These diagnostics help separate gains from improved *where/what/how* decisions (cueing, reuse, grounding) from gains due to retrieving more text.

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 $\mathcal{C}$

- 1: $\mathcal{C} \leftarrow \emptyset$
- 2: **for** $l \in \{3, 2, 1\}$ **do**
- 3: $\mathcal{H} \leftarrow \text{HNSWSEARCH}(M^{(l)}, \mathbf{v}_q, \text{pool} = c_l)$
- 4: **for all** $\uparrow \in \mathcal{H}$ **do**
- 5: $\text{VALID} \leftarrow (\uparrow.\text{status} \neq \text{FAILED}) \wedge (\neg \text{ISSTALE}(\uparrow, \Delta))$
- 6: $S(\uparrow; q) \leftarrow \lambda_{\text{sim}} \text{sim}(\mathbf{v}_q, \uparrow.\mathbf{v}) + \lambda_{\text{time}} g(\Delta t) + \lambda_{\text{status}} w(\text{status}) + \lambda_{\text{src}} w(\text{source_type})$
- 7: **if** $\neg \text{VALID}$ **then**
- 8: $S(\uparrow; q) \leftarrow S(\uparrow; q) - \infty$
- 9: **end if**
- 10: **end for**
- 11: $\mathcal{C} \leftarrow \mathcal{C} \cup \text{TOPK}(\mathcal{H}, k_l, \text{by } S)$
- 12: **end for**
- 13: **return** $\text{SORT}(\mathcal{C}, \text{by } S)$

Table 25: **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 interaction, indicating effective noise filtering (e.g., skipping scroll/click chatter). Across successful

Algorithm 4 PROVENANCEEXPANSION for auditable recalls

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

- 1: $\mathcal{Z} \leftarrow \emptyset$; queue $\leftarrow \mathcal{C}$
- 2: **while** queue not empty **and** $\text{Tok}(\mathcal{Z}) < B$ **do**
- 3: pop $\uparrow$ from queue
- 4: $\mathcal{Z} \leftarrow \mathcal{Z} \cup \{\uparrow\}$
- 5: **if** $\text{depth}(\uparrow) < D$ **then**
- 6: **for all** $p \in \uparrow.\mathcal{P}$ **do**
- 7: $\uparrow_1 \leftarrow \text{RESOLVEPOINTER}(p)$ $\triangleright$
- 8: episodic span in $M^{(1)}$
- 9: push $\uparrow_1$ to queue
- 10: **end for**
- 11: **end while**
- 12: **return** $\mathcal{Z}$

Table 26: **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

recalls, 85% traverse the full coarse-to-fine path $M(3) \rightarrow M(2) \rightarrow M(1)$, suggesting the hierarchy is actively utilized rather than merely decorative.

A.23 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 $\approx 4\text{--}5\%$ of total latency; even at p95 it remains small (up to $\approx 6\%$), with generation ac-

Algorithm 5 CONSOLIDATE with conflict resolution

Require: Supporting traces $\mathcal{T} \subseteq M^{(1)}$, policy thresholds τ

Ensure: Consolidated memory item $\uparrow^{(2)}$

- 1: Extract structured claims $\mathcal{K} \leftarrow \text{EXTRACTCLAIMS}(\mathcal{T})$
 - 2: Group by key: $\{\mathcal{K}_j\} \leftarrow \text{GROUPBYKEY}(\mathcal{K})$
 - 3: **for all** $\mathcal{K}_j$ **do**
 - 4: Rank candidates by epistemic preference: $\text{SUCCESS} \succ \text{UNKNOWN} \succ \text{FAILED}$, newer $\succ$ older, trace-supported $\succ$ ungrounded
 - 5: Select winner k^* ; mark alternatives as **CONFLICT** if score gap $< \tau$
 - 6: **end for**
 - 7: Synthesize summary $x^{(2)} \leftarrow \text{SUMMARIZE}(\{k^*\})$ with pointers to supporting spans
 - 8: **return** $\uparrow^{(2)} = (x^{(2)}, \mathbf{v}^{(2)}, \mathcal{E}^{(2)}, \mathcal{P}^{(2)})$
-

Algorithm 6 PRUNE under memory budgets

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

Ensure: Updated memory layers

- 1: **for** $l \in \{1, 2, 3\}$ **do**
 - 2: Mark items stale/failed for removal; compute redundancy clusters
 - 3: Remove lowest-utility items until budget satisfied
 - 4: **end for**
-

counting for the vast majority of runtime. Because governed retrieval nearly halves CTU relative to Flat RAG, the token savings typically outweigh the controller and provenance overhead. In our API-cost accounting at \$0.06 per 1K input tokens, the CTU reduction saves about \$0.13 per GAIA-L2 episode while the controller adds about \$0.01, yielding a net cost reduction.

A.24 Additional qualitative cases

We provide extended case studies with explicit provenance trails $M(3) \rightarrow M(2) \rightarrow M(1)$, 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.25 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 procedural knowledge, we filter by educational value scores (≥ 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.26 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:

$$\text{KL}(\pi_\theta \parallel \pi_T) = \mathbf{E} \left[\log \pi_\theta(a_{t+1} \mid s_t) - \log \pi_T(a_{t+1} \mid s_t) \right],$$

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 $A_{t+1} = -(\log \pi_\theta(a_{t+1} \mid s_t) - \log \pi_T(a_{t+1} \mid s_t))$, 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, $A = -(\log \pi_\theta(x) - \log \pi_T(x))$.
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.27 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:

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

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).