LEARNING RESIDUAL MEMORY OF CONTEXTUAL COMPACTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Long-horizon agents compact their history to continue within a finite context window. A summary carries forward the main thread of a task, but can omit details needed for later decisions, leading to repeated work and errors. We introduce REMORY, a neural memory network that encodes the RESIDUAL MEMORY of contextual compaction as soft tokens. Given a history and its summary, REMORY learns memory that brings the frozen actor's continuation predictions closer to those made with the full history. We train it with a shared two-stage curriculum: text reconstruction followed by summary-conditioned continuation. On SummHay, adding REMORY to the same summary improves citation F1 by Δ+4.04 points and joint score by Δ+3.55, using 5.2% of the full-context positions. With Qwen3.8-27B, REMORY improves scores by Δ3.0–9.8 points over the baseline across four agent benchmarks. With GLM-5.3-Flash, REMORY improves scores by Δ+6.0 points on BrowseComp and Δ+3.3 on Terminal-Bench 2.1, reaching scores comparable to frontier LLM reference scores under their respective evaluation protocols.

1 INTRODUCTION

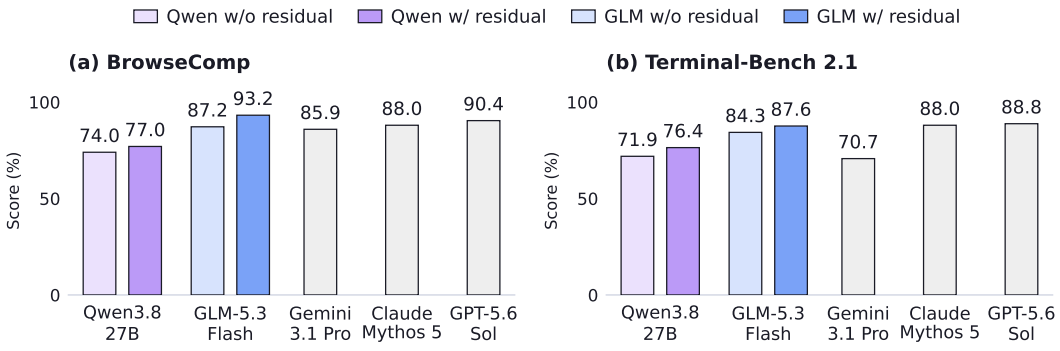


Figure 1: BrowseComp and Terminal-Bench 2.1 performance. Qwen3.8-27B is purple and GLM-5.3-Flash is blue; pale bars are local w/o residual baselines and solid bars add REMORY. Light gray bars show published Google, Anthropic, and OpenAI systems. Both local actors use all 1,266 BrowseComp tasks; frontier results retain their published harnesses and protocols. Sources: Appendix C.2.

Long-horizon agents work within a finite context window. As observations, tool outputs, and decisions accumulate, they must compact their history to continue. Both Claude Code and Codex CLI provide `/compact` to summarize earlier interaction and free context space (Anthropic, 2026; OpenAI, 2026a). The resulting input length follows a sawtooth pattern: it grows with interaction, drops at compaction, and grows again (Figure 2a). Each drop leaves a shorter interaction history.

A summary may preserve the goal and plan while leaving out a numerical value, an output constraint, an intermediate result, or the reason behind an earlier decision. A detail omitted now may matter several steps later. After compaction, an agent may repeat searches, reread files, or retry failed approaches to recover what it has lost. Tokens and time go into reconstructing prior work, while missing

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

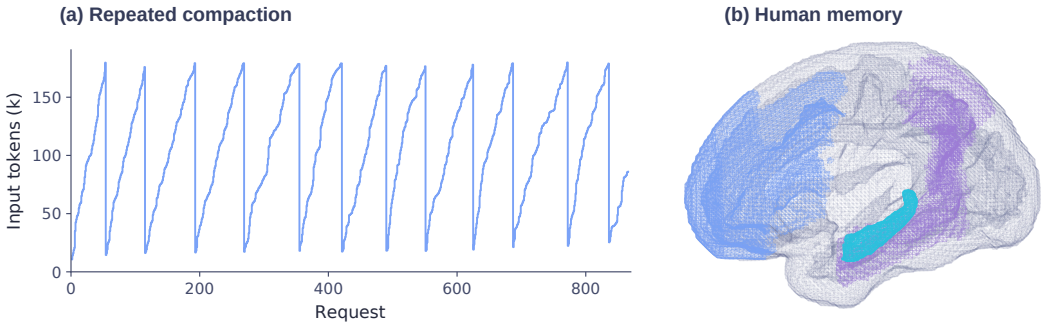


Figure 2: Context compaction and human memory. (a) Input length across successive requests in an agent run, showing repeated growth and sharp reductions. (b) Memory systems illustrated using the Harvard–Oxford atlas (Harvard–Oxford Atlas Contributors, 2026): prefrontal regions (blue), hippocampus (cyan), and association cortex (purple).

details can lead to new errors. The challenge is to carry enough of the earlier work forward for the agent to continue making progress.

Human working memory holds only a few chunks under controlled conditions (Baddeley, 2012; Cowan, 2001). Yet experts pursue scientific problems over many sessions, drawing on accumulated knowledge and skills. Learned retrieval cues keep relevant long-term knowledge accessible during demanding work (Ericsson & Kintsch, 1995). Long-term memory includes explicit knowledge, implicit skills, and habits (Squire & Dede, 2015).

In Figure 2b, prefrontal regions (blue) support working memory (Goldman-Rakic, 1995); the hippocampus (cyan) binds experiences into episodic memories (Squire & Dede, 2015); association cortex (purple) supports long-term knowledge (Kumaran et al., 2016). Language is one part of this system: expert mathematical reasoning recruits networks distinct from those used for language (Amalric & Dehaene, 2016). These observations motivate learning to retain useful history across compactions.

We introduce Remory, a neural memory network that learns to compensate for the effect of compaction on subsequent predictions. Given a history and its generated summary, we train the network to help the actor approximate the continuation it would produce with the full history. We implement this compensation as a bounded sequence of soft tokens read alongside the summary. The summary provides a textual checkpoint; the learned memory carries additional context into the next decision. An episodic archive retains exact records for retrieval through ordinary tool actions (Figure 3). For both Qwen3.8-27B and GLM-5.3-Flash, we train the memory network with the same two-stage curriculum: text reconstruction followed by summary-conditioned continuation, using each actor's full-history predictions as targets. The actors stay frozen and keep their existing tools.

SummHay evaluates content recovery and source attribution after compaction. Adding Remory to the same summary improves citation F1 by +4.04 points and joint score by +3.55, with a coverage change of +0.40. Across four agent benchmarks, Remory improves Qwen3.8-27B scores by 3.0–9.8 points over the baseline. GLM-5.3-Flash gains 6.0 points on BrowseComp and 3.3 on Terminal-Bench 2.1 (Figure 1). A paired action probe examines the first decision after compaction, and Live Trading traces context use over repeated compactions.

2 Residual Compensation

We call the predictive information from a history that its summary fails to preserve the compact residual. Remory learns compensation by bringing the actor's predictions from summary and memory closer to its full-history predictions (Figure 3(a,b)). We use soft tokens to optimize this memory through the frozen actor. The same objective could also guide memory in vocabulary tokens.

Memory compaction is recursive. At boundary i, H_i contains the previous textual summary and residual soft tokens, when present, together with the interaction since the last compaction. S_i is the

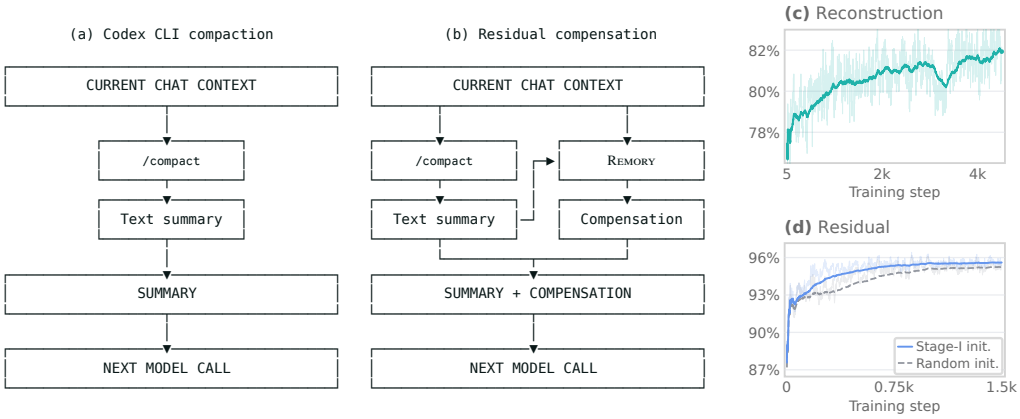


Figure 3: Residual compensation and training. (a) Codex CLI passes the `/compact` summary to the next actor call. (b) The Remory network uses that summary to condition memory construction from the active context; summary and continuous memory jointly support subsequent actions. Archive access remains an ordinary tool action in either setting. (c,d) Qwen3.8-27B teacher–student top-1 agreement during Stage I reconstruction and Stage II residual training. The residual panel compares reconstruction initialization with random initialization under the same residual objective and data order. Curves are optimization diagnostics from the training streams.

new `/compact` summary. With fixed persistent instructions, Remory constructs memory

$$M_i = \mathcal{C}_\phi(H_i; S_i), \qquad |M_i| \le m. \tag{1}$$

Here ϕ denotes the trainable network parameters, M_i is a sequence of vectors in the frozen actor's input space, and m is the memory budget. Both networks use the summary to condition memory construction, adapting compensation to the information already available in text. Both actors also receive the summary directly. Each full source block of 1,024 positions produces 64 soft tokens, a 1:16 ratio of memory to source positions. When the initial memory exceeds m, adjacent memory blocks are compacted hierarchically to meet the output budget (Appendix A). After compaction, the actor reads persistent instructions, summary, and compensation, followed by new interaction. Archive excerpts enter through ordinary tool results.

For Qwen3.8-27B, Remory adapts DFlash (Chen et al., 2026). For GLM-5.3-Flash, it combines this continuous-memory interface with DFlash-2 grouped dynamic convolutions (Inco AI, 2026a). We adapt the drafter to take historical hidden states and learned memory queries as inputs and predict continuous soft tokens in parallel. The complete networks contain 1.94B and 1.502B trainable parameters for Qwen and GLM, respectively. Appendix A describes the architecture adaptations. Both variants follow the same training curriculum in Section 3.

3 Training

We train a separate Remory network for each actor, Qwen3.8-27B and GLM-5.3-Flash, using the same two-stage curriculum: Stage I learns text reconstruction, and Stage II learns summary-conditioned continuation. Both stages compare full-history and memory-conditioned predictions of the corresponding frozen actor, through which each memory network is optimized.

Stage I (reconstruction) teaches the memory network to preserve source content. The teacher receives a source passage and a repeat instruction; the student replaces the source with its soft memory and receives the same instruction. Both predict the source followed by its end-of-turn token. Source chunks contain at most 16,384 tokens and map to at most 1,024 memory slots. We use 2,286,130 rows from the full pre-selection release of MMFineReason-Full-2.3M (Lin et al., 2026), treating question, original-answer, and caption fields as separate text-repetition samples.

Stage II (summary-conditioned continuation) initializes the memory network from Stage I and resets the optimizer state. The teacher sees the original interaction history; the student receives its fixed summary and memory constructed from that history, with a 4,096-slot budget. Both include

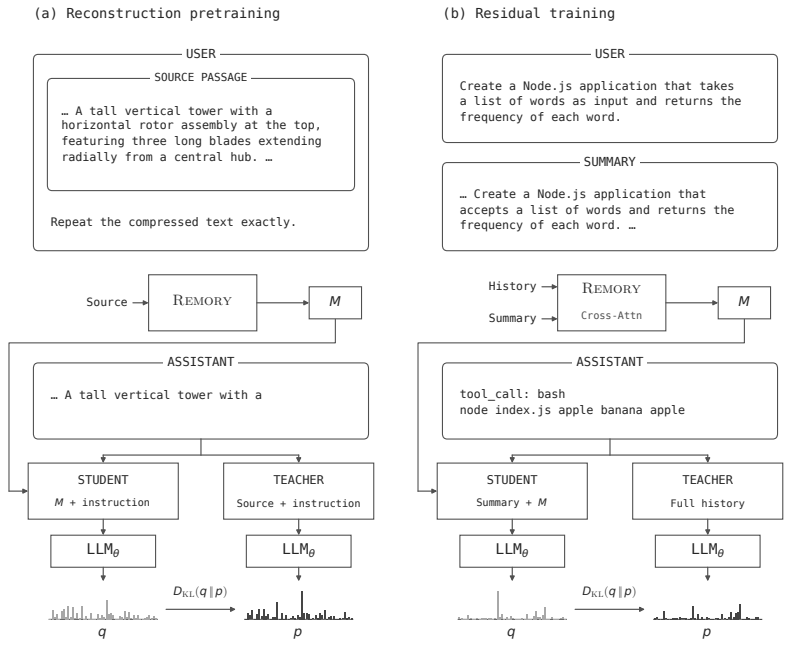


Figure 4: Shared training curriculum: (a) Stage I reconstruction; (b) Stage II summary-conditioned continuation. Teacher and student share a frozen actor; only REMORY is trained. Both encoders use summary cross-attention. Tool calls, context views, and distributions are schematic.

shared task instructions and predict the complete assistant response. Training examples follow a deterministic curriculum ordered by preceding-context length. Summaries are precomputed and held fixed. The future target response is never supplied to the memory encoder. This stage uses 40,596 reasoning and 37,560 coding trajectories (137,919 assistant responses in total). Reasoning prompts include Open-PerfectBlend (Labonne, 2024). Both actors use Qwen-generated responses and offline summaries; empty or length-truncated summaries are excluded.

Figure 4 illustrates the two stages. Appendix B gives the coding-source breakdown (Figure 7). Let C_{full} and C_{mem} denote teacher and student contexts, including shared instructions. The target y is the source reproduction in the reconstruction task and the assistant continuation in the residual task. The frozen actor p_θ predicts from the same teacher-forced prefix:

$$p_j = p_\theta(\cdot \mid C_{\text{full}}, y_{<j}), \qquad q_j = p_\theta(\cdot \mid C_{\text{mem}}, y_{<j}). \tag{2}$$

Both stages minimize student-to-teacher reverse KL over the full vocabulary and the complete response, at temperature one:

$$\mathcal{L}(\phi) = \frac{\sum_{j=1}^{L} w_j D_{\text{KL}}(q_j \| p_j)}{\sum_{j=1}^{L} w_j}. \tag{3}$$

Here L is the response length. For token position u_j, the positive position weights are

$$a_j = 1 + (1 + u_j/64)^{-1/2}, \qquad w_j = \frac{a_j}{L^{-1} \sum_{k=1}^{L} a_k}. \tag{4}$$

Stage I uses the original source offset; Stage II uses position within the assistant response. Loss accumulation is normalized by scored tokens. The teacher and student branches share the same frozen actor; gradients through the student branch update only REMORY, keeping the actor's weights and tool interface fixed. In continuation training, the student receives the summary, so memory learns to close the remaining prediction gap. Compensation may overlap with summary content.

Both models use AdamW with learning rate 10^{-5}, 500-step linear warmup, $\beta = (0.9, 0.95)$, $\epsilon = 10^{-8}$, weight decay 0.01, and seed 42. Stage I accumulates eight complete responses per update; Stage II accumulates at least 2,048 scored tokens, ending at whole-response boundaries.

Figure 3(c,d) plots Qwen teacher–student top-1 agreement (matching predictions at scored positions), with bias-corrected exponential smoothing (0.995). Residual curves compare reconstruction and random initialization with the objective, data order, optimizer, and summary cache held fixed.

4 Experiments

We test information retention beyond a fixed summary (SummHay), task outcomes and post-compaction decisions (agent benchmarks and a probe), and long-running context use (Live Trading). We use GPT-5.6-Sol for all LLM-based judging.

4.1 Source attribution after compaction

SummHay tests query-relevant insight and source recovery from long document collections (Laban et al., 2024). Coverage measures content recovery; citation metrics measure source attribution.

We evaluate all 92 queries from ten collections, five conversational and five news, using raw context, summary w/o residual, summary++ w/o residual, or summary w/ residual. In summary++, text supplements the unchanged summary under the same additional-position budget as residual memory. The query precedes a single compaction; every collection fits without truncation. All conditions use Qwen3.8-27B at xhigh, temperature zero, and a 16,384-token generation budget including reasoning. Answer generation has no tools, archive access, or source retrieval; gold insights and citations are used only for scoring. Fixing the actor, query, decoding, and summary isolates the contribution of the added memory channel to this response.

Table 1: SummHay aggregate results on the same queries; higher is better. The summary w/ residual condition uses the final checkpoint after completing Stage I and Stage II (Section 3). Summary++ matches the residual's additional-token budget with text.

Input representation	Coverage	Citation F1	Joint score
raw context	72.46	56.60	43.41
summary w/o residual	67.55	56.98	39.84
summary++ w/o residual	67.52	56.94	39.78
summary w/ residual	67.95	61.03	43.39

Coverage measures recovered insights; citation F1 measures their attribution; the joint score combines both. Coverage uses GPT-5.6-Sol with the benchmark prompts. We average coverage and joint score over all 621 reference insights; citation F1 averages per-insight F1 over covered insights only. Table 1 retains all queries, including generation-limit and formatting failures. Compensation raises citation F1 from 56.98 to 61.03 (+4.04) and joint score from 39.84 to 43.39 (+3.55). Coverage changes from 67.55 to 67.95 (+0.40). A paired collection bootstrap gives a 95% interval of [+0.74, +6.21] points for the joint-score gain. Citation recall rises by +6.17 points and precision by +0.42: the actor recovers more supporting sources at similar precision. The gain persists on the 553 insights covered by both configurations: citation F1 increases from 57.31 to 61.41 (+4.10). With recovered content fixed, the attribution gain points to compensation that helps connect a claim to its evidence. The small coverage change and larger citation gain suggest better use of content–source associations even when the summary retains the main content. Appendix C.1 details the analysis and scope.

Mean inputs contain 96,872 positions for raw context, 1,393 for summary w/o residual, and 5,024 for summary w/ residual, including memory positions. The compensated input is 5.2% of raw context and 3.6 times the summary w/o residual input. Its joint score is close to the raw-context score of 43.41, with lower coverage and higher citation F1. Raw context recovers more content; compensation improves attribution. This selective recovery closes much of the joint-score gap with a compact supplement to the unchanged summary.

The summary++ w/o residual control tests whether allocating the same additional memory budget to text yields a similar benefit. Its coverage (67.52), citation F1 (56.94), and joint score (39.78) remain close to summary w/o residual. Residual memory exceeds this control by 0.43 points in coverage, 4.09 points in citation F1, and 3.61 points in joint score. At this budget, the learned supplement supports more accurate source attribution than the generated text supplement. Similar content coverage

across the three compacted conditions places the main difference in how the actor connects content to its sources. This supports learning a summary complement that preserves attribution cues.

4.2 End-to-end agent evaluation

The agent evaluations test completed work under a common harness, allowing each configuration's decisions to shape its trajectory across compactions.

We evaluate BrowseComp research tasks (Wei et al., 2025), Terminal-Bench 2.1 command-line tasks (Merrill et al., 2026; Terminal-Bench Team, 2026), AutomationBench business workflows (Shepard & Salimans, 2026), and JobBench professional tasks (Li et al., 2026b). Both actors use all 1,266 BrowseComp tasks and all 89 Terminal-Bench 2.1 tasks, with one run per configuration on each benchmark. Our AutomationBench evaluation covers all 600 public tasks across six business domains, excluding the unscored simple domain. JobBench covers all 65 tasks in the main leaderboard split. We also evaluate Live Trading, a multi-day prediction-market task with one concurrent paper-trading account per configuration.

For each actor and benchmark, w/o residual and w/ residual share actor weights, the Codex CLI v0.149.0 harness, base and task prompts, tools, decoding settings, task budgets, and compaction policy. In the w/ residual configuration, Remory adds soft tokens alongside the textual summary and retained context after each automatic `/compact`. Qwen3.8-27B uses `xhigh` reasoning, a 262,144-token context, and a 179,200-token compaction trigger. GLM-5.3-Flash uses `max` reasoning, a 1M-token context, and a 917,498-token compaction trigger. On BrowseComp and Terminal-Bench 2.1, both actors use greedy decoding at temperature zero in both configurations. On AutomationBench, JobBench, and Live Trading, Qwen uses temperature 1.0, top-p 0.95, and top-k 20. All reported GLM Terminal-Bench 2.1 results use this 1M context configuration throughout evaluation. Agent execution uses a 2-hour per-task timeout, with task-specific verifier limits. Both configurations provide terminal execution, planning, image inspection, Serper web search, Docling/MinerU document tools, and a read-only exact-history archive. Tool outputs are truncated to 10,000 bytes by default. Compaction retains persistent system/tool instructions, the original task, and Codex-selected recent history. Summary generation uses the same actor at temperature zero, the context-checkpoint prompt, and the remaining serving context, with no fixed summary-length cap. The actor-specific Remory networks are described in Appendix A and trained with the shared curriculum in Section 3. Both allow up to 4,096 soft-token slots and chronological source blocks of up to 1,024 positions mapped to 64 slots each. Scoring and additional task-budget details are described in Appendix C.2.

Tables 2 and 3[1] report task scores and costs. Each w/ residual configuration uses its actor-specific final checkpoint after completing Stage I and Stage II (Section 3). JobBench scores mean weighted rubric credit; SD denotes standard deviation.

Qwen's largest gains are on AutomationBench (+9.8) and JobBench (+7.6), averaged over five runs, extending the benefit to business workflows and professional-task rubric credit. Qwen gains 3.0 and 4.5 points on BrowseComp and Terminal-Bench 2.1; GLM gains 6.0 and 3.3, extending the benefit to a stronger actor with a larger context window. Frontier scores in Figure 1 provide external references; within-actor comparisons control the harness and context policy.

In Table 3, repeats count identical outputs recurring from the same tool within a task session, after stripping runtime wrappers; plan updates and waiting/polling interfaces are excluded. Errors count nonzero command exits or explicit tool-error responses. Across the four actor–benchmark pairs, Remory reduces repeated outputs by 17.9–29.8% and errors by 25.3–49.7%. Both actors attain higher scores with fewer encounters with unchanged feedback and fewer failed calls. This supports the hypothesis that residual memory helps preserve working state, allowing earlier feedback to guide later actions with less repeated interaction.

Qwen3.8-27B generation costs use Alibaba Cloud's international rates: $0.50 per million input tokens and $3.00 per million output tokens, without cache discounts (Alibaba Cloud, 2026). Generation cost falls on BrowseComp, Terminal-Bench 2.1, and AutomationBench, while JobBench improves at a higher cost ($3.09 to $3.21). Both GLM comparisons also improve scores at lower generation cost. These API-equivalent estimates cover actor and summary calls, charging memory positions as input.

[1] Qwen3.8-27B costs use Alibaba Cloud international token rates; GLM-5.3-Flash costs use Z.ai's official rates; all omit cache discounts. Cost accounting: Appendix C.2.

Table 2: Qwen3.8-27B on AutomationBench and JobBench: mean ± SD scores and mean costs over five runs per configuration. Costs are API-equivalent USD per task.

Configuration	AutomationBench		JobBench	
	Score	Cost	Score	Cost
w/o residual	35.5 ± 1.3	$0.59	33.4 ± 1.25	$3.09
w/ residual (Ours)	**45.3** ± 1.0	$0.53	**41.0** ± 1.41	$3.21

Table 3: Terminal-Bench 2.1 and BrowseComp: one run per configuration. Costs are API-equivalent USD per task; repeats and errors are tool-output counts over the full task set.

Actor	Configuration	Terminal-Bench 2.1				BrowseComp			
		Score	Cost	Repeats	Errors	Score	Cost	Repeats	Errors
Qwen3.8-27B	w/o residual	71.9	$4.29	459	933	74.0	$5.39	25,801	25,244
	w/ residual	**76.4**	$2.86	377	568	**77.0**	$4.69	18,105	12,698
GLM-5.3-Flash	w/o residual	84.3	$3.27	334	664	87.2	$5.65	8,316	7,498
	w/ residual	**87.6**	$3.05	240	496	**93.2**	$5.00	5,834	5,412

Remory has 1.94B parameters versus the actor's 27B (about 7%) and predicts the full residual token sequence in parallel within each compression pass, avoiding token-by-token autoregressive decoding. Appendix C.2 details the pricing and accounting scope.

We examine the first post-compaction action in 100 sampled BrowseComp trajectories, yielding 488 paired Qwen3.8-27B actions. We pair each recorded w/ residual action with a w/o residual continuation from the same persistent prompt and summary. Inspired by WebAnchor's rubric-based planning evaluation (Yu et al., 2026), we fix a task-specific rubric from the preceding history before GPT-5.6-Sol compares the anonymized tool calls or final answers, withholding candidate reasoning and leaving counterfactual calls unexecuted (prompts in Appendix C.3). The w/ residual actions receive 345 wins (70.7%), 21 ties (4.3%), and 122 losses (25.0%). With history and summary fixed, this preference links residual memory to the first decision after compaction. The probe tests this decision; end-to-end scores capture the effects of later actions and observations.

4.3 Context dynamics in Live Trading

Live Trading holds the actor, context limit, and compaction rule fixed over multiple days; market choices and observations evolve independently. At the September 23 cutoff, settled ROI is -11.86% without residual and $+5.11\%$ with Remory; logged generation costs for the observed multi-day trajectories are $303.81 and $170.30, respectively. Settled ROI counts profit and loss only for officially resolved markets, relative to the initial $20,000 bankroll. These are two concurrent account trajectories, with unresolved positions excluded from the score. Market selection, timing, and settlement status shape returns. We analyze spend through interaction volume and context occupancy.

Both configurations retain the accumulation-and-reset pattern. Figure 5(a) illustrates five visible resets without residual memory and three with it. In the separate multi-day record, completed compactions fall from 56 to 45 and the median interval increases from 70.1 to 76.7 minutes, alongside fewer responses (5,336 versus 3,013). Per 1,000 responses, compactions instead rise from 10.5 to 14.9. The decrease in total resets accompanies fewer calls over the deployment period, despite more frequent compaction per response. Mean live context is slightly smaller with residual (101,185 versus 105,372 positions), including soft memory. Smaller contexts and fewer responses explain how the supplement lowers generation spend over the trajectory. We hypothesize that residual memory reduces the effort needed to rebuild the working state after compaction: rejected hypotheses, observations supporting a decision, and active constraints. Carrying these details alongside the summary may reduce calls spent reconstructing them. Their influence on subsequent actions can offset the cost of reading the compact memory supplement. Appendix C.4 separates these accounting units.

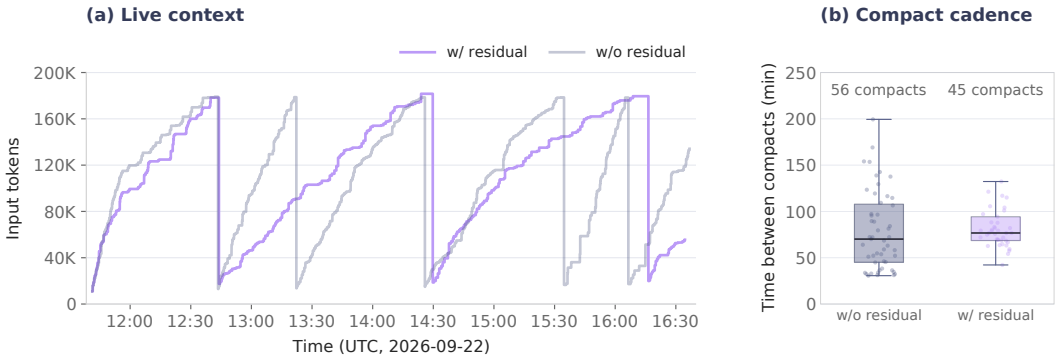
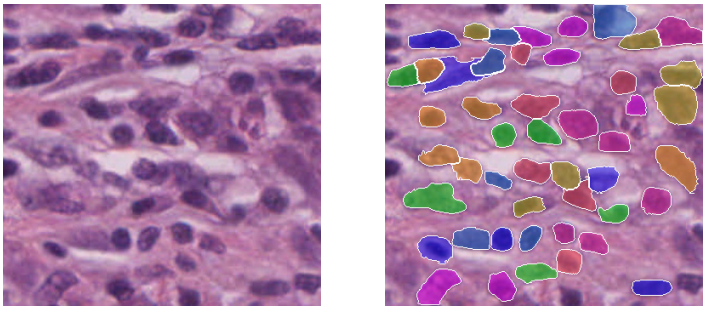


Figure 5: Live Trading context dynamics. (a) An illustrative September 22 trace of input context with and without residual memory. (b) Time between successive compactions in the separate September 20–23 evaluation described in this section; labels give total completed compactions (56/45); the corresponding interval counts are 51/41 for w/o and w/ residual. Boxes show medians and interquartile ranges, with 1.5-IQR whiskers and individual observations. Intervals never cross session restarts. Two residual intervals exceed the 250-minute display range but remain in the box statistics.

4.4 Case study: repairing a cell-segmentation pipeline

In Terminal-Bench 2.1's `sam-cell-seg`, Qwen3.8-27B refines cell annotations into disjoint, contiguous polygons (Figure 6). A streaming refactor accidentally deletes helper functions; `/compact` then removes the surrounding edit history from live textual context, leaving files unchanged.

After compaction at round 118, the next request reports zero raw-history tokens and a 4,682-token summary. The summary omits round 117's exact replacement commands between `prepare_row` and `main`, but retains the known CSV-helper error, output constraints, and the backup's pre-refactor identity. The checkpoint retains the high-level file–version link but loses detailed edit history.



(a) Task image (b) Observed mask overlay

Figure 6: Cell segmentation across two compactions (rounds 45 and 118). Original task image and the exact overlay viewed by the actor at round 206; both are reproduced without pixel changes.

The next actor call also receives Remory's memory, constructed from the preceding history and conditioned on this summary. The installation event and first subsequent client receipt report 4,032 memory slots alongside the textual checkpoint.

At rounds 119–133, the actor rereads the current script and backup, checks output-format compatibility, and restores three helpers while retaining the refactor. The final submission passes all nine verifier tests. The retained summary identifies the backup, while the memory channel offers an additional representation of the omitted edit history. This illustrates complementary roles for summary, latent memory, and file retrieval. Because the actor rereads both files, the successful repair does not identify which missing edit details were recovered from memory.

5 Related Work

Human memory models link retrieval to learning context (Kahana, 2020). Complementary learning systems connect reasoning with structured knowledge and rapidly stored episodes (Kumaran et al., 2016).

Agent memory often uses summaries alongside retrieval. ReadAgent pairs episode gists with access to original passages; MemGPT separates working and archival memory (Lee et al., 2024; Packer et al., 2023). ReSum uses periodic summaries to support long-horizon search and adapts agents to summary-conditioned continuation (Wu et al., 2025). AgentFold learns multi-scale context management within the agent's trajectory (Ye et al., 2025).

AutoCompressors, Gist Tokens, and ICAE replace longer text with learned continuous representations (Chevalier et al., 2023; Mu et al., 2023; Ge et al., 2024). ICAE combines autoencoding and language-modeling pretraining with instruction fine-tuning. LCLMs scale encoder–decoder compression through continual pretraining and support selective expansion of compressed segments (Li et al., 2026a). Remory retains the agent's textual summary and learns compensation alongside it. Both encoders use the summary during construction; the actor reads it alongside memory.

Prompt compression through distribution matching uses the reader's predictions as a behavioral criterion (Wingate et al., 2022). Recurrence, compressed activations, and memory tokens offer complementary ways to retain information across segments (Dai et al., 2019; Rae et al., 2020; Bulatov et al., 2022). Remory applies this predictive criterion to the gap left by contextual compaction, learning a memory supplement through full-vocabulary distillation of a frozen actor's continuation.

6 Limitations

Our experiments cover a limited range of actor models. We plan to extend them to full-scale versions of mainstream language models by training and evaluating Remory across model families.

Residual training uses text-only data. To construct memory, Remory reads the target LLM's hidden representations. Its ability to retain multimodal information currently depends primarily on how these representations generalize. Direct residual training on multimodal data remains future work.

7 Conclusion

Remory learns to compensate for the prediction gap left by `/compact`. A bounded soft-token memory carries earlier history alongside the summary, helping the frozen actor continue its work. We train this memory through reconstruction followed by summary-conditioned continuation. On SummHay, it improves citation F1 by +4.04 points and joint score by +3.55. Across four agent benchmarks, Qwen3.8-27B gains 3.0–9.8 points over the baseline. GLM-5.3-Flash gains 6.0 points on BrowseComp and 3.3 on Terminal-Bench 2.1. SummHay's larger gains in attribution than in coverage suggest that memory helps preserve content–source associations. The paired action probe finds better-rated next actions with the same summary. These results support learning a compact memory supplement that helps an actor use earlier evidence after compaction. Future work will extend training and evaluation to full-scale actor models, multimodal histories, and longer sequences of compactions.

AI USE STATEMENT

We used generative AI tools to assist with manuscript writing, language polishing, reference checking, and manuscript consistency checks. Language models also generated training responses and offline summaries and served as evaluation judges, as described in Sections 3 and 4. All AI-assisted text was reviewed and revised by the authors to ensure accuracy and consistency with the reported methods and results. The authors take full responsibility for the final content of this paper.

REFERENCES

Alibaba Cloud. Qwen3.8-27B: Model information and pricing, 2026. URL https://www.alibabacloud.com/help/en/model-studio/qwen3-8-27b. Singapore; price snapshot September 10, 2026; rechecked September 25, 2026.

Marie Amalric and Stanislas Dehaene. Origins of the brain networks for advanced mathematics in expert mathematicians. *Proceedings of the National Academy of Sciences*, 113(18):4909–4917, 2016. doi: 10.1073/pnas.1603205113.

Anthropic. Explore the context window: Claude Code docs, 2026. URL https://code.claude.com/docs/en/context-window. Accessed 2026-08-27; rechecked 2026-09-24.

Alan Baddeley. Working memory: Theories, models, and controversies. *Annual Review of Psychology*, 63:1–29, 2012. doi: 10.1146/annurev-psych-120710-100422.

Aydar Bulatov, Yury Kuratov, and Mikhail Burtsev. Recurrent memory transformer. In *Advances in Neural Information Processing Systems*, volume 35, pp. 11079–11091. Curran Associates, Inc., 2022. doi: 10.52202/068431-0805. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/47e288629a6996a17ce50b90a056a0e1-Abstract.html.

Jian Chen, Yesheng Liang, and Zhijian Liu. DFlash: Block diffusion for flash speculative decoding. In *Proceedings of the International Conference on Machine Learning*, 2026. URL https://arxiv.org/abs/2602.06036.

Alexis Chevalier, Alexander Wettig, Anirudh Ajith, and Danqi Chen. Adapting language models to compress contexts. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 3829–3846. Association for Computational Linguistics, 2023. doi: 10.18653/v1/2023.emnlp-main.232. URL https://aclanthology.org/2023.emnlp-main.232/.

Nelson Cowan. The magical number 4 in short-term memory: A reconsideration of mental storage capacity. *Behavioral and Brain Sciences*, 24(1):87–114, 2001. doi: 10.1017/S0140525X01003922.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc V. Le, and Ruslan Salakhutdinov. Transformer-XL: Attentive language models beyond a fixed-length context. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 2978–2988. Association for Computational Linguistics, 2019. doi: 10.18653/v1/P19-1285. URL https://aclanthology.org/P19-1285/.

K. Anders Ericsson and Walter Kintsch. Long-term working memory. *Psychological Review*, 102(2):211–245, 1995. doi: 10.1037/0033-295X.102.2.211.

Tao Ge, Hu Jing, Lei Wang, Xun Wang, Si-Qing Chen, and Furu Wei. In-context autoencoder for context compression in a large language model. In *International Conference on Learning Representations*, pp. 2591–2607, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/0b276510ec2d3f6613a8b60c41ff0438-Abstract-Conference.html.

Patricia S. Goldman-Rakic. Cellular basis of working memory. *Neuron*, 14(3):477–485, 1995. doi: 10.1016/0896-6273(95)90304-6.

Google DeepMind. Gemini 3.1 Pro model card, February 2026. URL https://deepmind.google/models/model-cards/gemini-3-1-pro/. BrowseComp, high thinking with search, Python, and browsing; accessed September 25, 2026.

Harvard–Oxford Atlas Contributors. Harvard–Oxford cortical and subcortical structural atlases. Distributed with the FMRIB Software Library, 2026. URL https://fsl.fmrib.ox.ac.uk/fsl/docs/other/datasets.html. Online atlas documentation; CC BY-SA 4.0; accessed September 25, 2026.

Inco AI. DFlash 2: Keep drafting parallel. Technical blog, August 2026a. URL https://inco.ai/blog/dflash2/. Published August 18, 2026; accessed September 25, 2026.

Inco AI. GLM-5.3-Flash-DFlash2: Model card and drafter configuration. Hugging Face model release, 2026b. URL https://huggingface.co/incoai/GLM-5.3-Flash-DFlash2. Accessed September 25, 2026.

Michael J. Kahana. Computational models of memory search. *Annual Review of Psychology*, 71: 107–138, 2020. doi: 10.1146/annurev-psych-010418-103358.

Dharshan Kumaran, Demis Hassabis, and James L. McClelland. What learning systems do intelligent agents need? complementary learning systems theory updated. *Trends in Cognitive Sciences*, 20 (7):512–534, 2016. doi: 10.1016/j.tics.2016.05.004.

Philippe Laban, Alexander Fabbri, Caiming Xiong, and Chien-Sheng Wu. Summary of a haystack: A challenge to long-context LLMs and RAG systems. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 9885–9903. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.emnlp-main.552. URL https://aclanthology.org/2024.emnlp-main.552/.

Maxime Labonne. Open-PerfectBlend. Hugging Face dataset, 2024. URL https://huggingface.co/datasets/mlabonne/open-perfectblend. Apache-2.0; accessed 2026-09-01.

Kuang-Huei Lee, Xinyun Chen, Hiroki Furuta, John Canny, and Ian Fischer. A human-inspired reading agent with gist memory of very long contexts. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 26396–26415. PMLR, 2024. URL https://proceedings.mlr.press/v235/lee24c.html.

Ang Li, Sean McLeish, Haozhe Chen, Nimit Kalra, Zaiqian Chen, Artem Gazizov, Venkata Anoop Suhas Kumar Morisetty, Bhavya Kailkhura, Harshitha Menon, Zhuang Liu, Brian R. Bartoldson, Tom Goldstein, Sanae Lotfi, Micah Goldblum, and Pavel Izmailov. End-to-end context compression at scale. arXiv preprint arXiv:2606.09659, 2026a. URL https://arxiv.org/abs/2606.09659.

Yuetai Li, Yichen Feng, Zhangchen Xu, Zixian Ma, Kaiyuan Zheng, Fengqing Jiang, Xinghua Sun, Rulin Shao, Zichen Chen, Yue Huang, Xinyang Han, Brian Lee, Kayla Xu, Shenglai Zeng, Hang Hua, Xiangliang Zhang, Basel Alomair, Ranjay Krishna, Luke Zettlemoyer, Pang Wei Koh, Bhaskar Ramasubramanian, Luyao Niu, Xiang Yue, and Radha Poovendran. JobBench: Aligning agent work with human will. arXiv preprint arXiv:2605.26329, 2026b. URL https://arxiv.org/abs/2605.26329.

Honglin Lin, Zheng Liu, Yun Zhu, Chonghan Qin, Juekai Lin, Xiaoran Shang, Conghui He, Wentao Zhang, and Lijun Wu. MMFineReason: Closing the multimodal reasoning gap via open data-centric methods. arXiv preprint arXiv:2601.21821, 2026. URL https://arxiv.org/abs/2601.21821.

Mike A. Merrill, Alexander G. Shaw, Nicholas Carlini, Boxuan Li, Harsh Raj, Ivan Bercovich, Lin Shi, Jeong Yeon Shin, Thomas Walshe, E. Kelly Buchanan, et al. Terminal-Bench: Benchmarking agents on hard, realistic tasks in command line interfaces. *arXiv preprint arXiv:2601.11868*, 2026. URL https://arxiv.org/abs/2601.11868.

Jesse Mu, Xiang Li, and Noah Goodman. Learning to compress prompts with Gist Tokens. In *Advances in Neural Information Processing Systems*, volume 36, pp. 19327–19352. Curran Associates, Inc., 2023. doi: 10.52202/075280-0848. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/3d77c6dcc7f143aa2154e7f4d5e22d68-Abstract.html.

OpenAI. Codex developer commands, 2026a. URL https://developers.openai.com/codex/cli/slash-commands. Accessed 2026-09-03; rechecked 2026-09-24.

OpenAI. GPT-5.6: Frontier intelligence that scales with your ambition, July 2026b. URL https://openai.com/index/gpt-5-6/.

Charles Packer, Vivian Fang, Shishir G. Patil, Kevin Lin, Sarah Wooders, and Joseph E. Gonzalez. MemGPT: Towards LLMs as operating systems. arXiv preprint arXiv:2310.08560v1, 2023. URL https://arxiv.org/abs/2310.08560v1.

Jack W. Rae, Anna Potapenko, Siddhant M. Jayakumar, Chloe Hillier, and Timothy P. Lillicrap. Compressive transformers for long-range sequence modelling. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=SylKikSYDH.

Daniel Shepard and Robin Salimans. AutomationBench. arXiv preprint arXiv:2604.18934, 2026. URL https://github.com/zapier/AutomationBench. Public 600-task reference scores accessed September 24, 2026.

SpecForge Team. DFlash2. SpecForge documentation, 2026. URL https://sgl-project.github.io/SpecForge/concepts/DFlash2.html. Accessed September 25, 2026.

Larry R. Squire and Adam J. O. Dede. Conscious and unconscious memory systems. *Cold Spring Harbor Perspectives in Biology*, 7(3):a021667, 2015. doi: 10.1101/cshperspect.a021667.

Terminal-Bench Team. Terminal-Bench 2.1, 2026. URL https://www.tbench.ai/news/terminal-bench-2-1. Release announcement; accessed September 25, 2026.

Jason Wei, Zhiqing Sun, Spencer Papay, Scott McKinney, Jeffrey Han, Isa Fulford, Hyung Won Chung, Alex Tachard Passos, William Fedus, and Amelia Glaese. BrowseComp: A simple yet challenging benchmark for browsing agents. *arXiv preprint arXiv:2504.12516*, 2025. URL https://arxiv.org/abs/2504.12516.

David Wingate, Mohammad Shoeybi, and Taylor Sorensen. Prompt compression and contrastive conditioning for controllability and toxicity reduction in language models. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pp. 5621–5634. Association for Computational Linguistics, 2022. doi: 10.18653/v1/2022.findings-emnlp.412. URL https://aclanthology.org/2022.findings-emnlp.412/.

Xixi Wu, Kuan Li, Yida Zhao, Liwen Zhang, Litu Ou, Huifeng Yin, Zhongwang Zhang, Xinmiao Yu, Dingchu Zhang, Yong Jiang, Pengjun Xie, Fei Huang, Minhao Cheng, Shuai Wang, Hong Cheng, and Jingren Zhou. ReSum: Unlocking long-horizon search intelligence via context summarization. arXiv preprint arXiv:2509.13313, 2025. URL https://arxiv.org/abs/2509.13313.

Rui Ye, Zhongwang Zhang, Kuan Li, Huifeng Yin, Zhengwei Tao, Yida Zhao, Liangcai Su, Liwen Zhang, Zile Qiao, Xinyu Wang, Pengjun Xie, Fei Huang, Siheng Chen, Jingren Zhou, and Yong Jiang. AgentFold: Long-horizon web agents with proactive context management. arXiv preprint arXiv:2510.24699, 2025. URL https://arxiv.org/abs/2510.24699.

Xinmiao Yu, Liwen Zhang, Xiaocheng Feng, Yong Jiang, Bing Qin, Pengjun Xie, and Jingren Zhou. WebAnchor: Anchoring agent planning to stabilize long-horizon web reasoning. *arXiv preprint arXiv:2601.03164*, 2026. URL https://arxiv.org/abs/2601.03164.

Z.ai. Model API pricing, 2026. URL https://docs.z.ai/guides/overview/pricing. Accessed September 25, 2026; GLM-5.3-Flash input/output rates of USD 0.15/0.50 per million tokens.

A Memory Architecture

The Qwen3.8-27B memory network adapts DFlash (Chen et al., 2026); the GLM-5.3-Flash network adapts the released DFlash-2 drafter (Inco AI, 2026b), including its grouped dynamic convolutions (Inco AI, 2026a; SpecForge Team, 2026). Both project features from five frozen actor layers into a memory network with learned slot queries. Local blocks gather source information; a final global block mixes the slots. All memory vectors are predicted in parallel and normalized and scaled to match the actor's input embeddings. Table 4 summarizes their dimensions.

Table 4: Memory network configurations. Both use 32 query and eight key–value heads of dimension 128.

	Qwen3.8-27B	GLM-5.3-Flash
Trainable parameters	1.94B	1.502B
Hidden width	5,120	4,096
Feed-forward width	17,408	12,288
Local / global blocks	4 / 1	5 / 1

Both encoders condition local slots on the summary through gated cross-attention between source attention and the feed-forward update. For slots U and frozen summary features E_i encoded from $P \| S_i$, where P denotes the persistent instructions,

$$U' = U + \tanh(a)\,\mathrm{Attn}(Q(U), K(E_i), V(E_i))\,. \tag{5}$$

Gates are initialized to 10^{-3}. The GLM backbone additionally uses two-tap dynamic convolutions with groups of 16 channels.

The frozen actor first encodes the complete history with persistent instructions. Its source features are then partitioned into consecutive blocks of at most 1,024 positions, each producing 64 leaf slots, including a partial final block. When the total exceeds 4,096 slots, the earliest 16 consecutive nodes at the same depth are merged: their soft tokens pass through the frozen actor to obtain features, which Remory compresses into 64 parent slots. Merging repeats until the budget is met; surviving nodes retain source order. The 4,096-slot budget bounds the final memory, while intermediate representations may be larger. Both encoders use the same summary at every level. This preserves source coverage while allocating fewer slots to older spans. The hierarchy operates within a compaction; carrying memory across compactions follows Section 2. Source encoding and the actor's live input must each fit its context limit; the latter includes text, memory, and reserved output positions.

B Coding Data Sources

Figure 7 gives the coding-data composition for Stage II (Section 3).

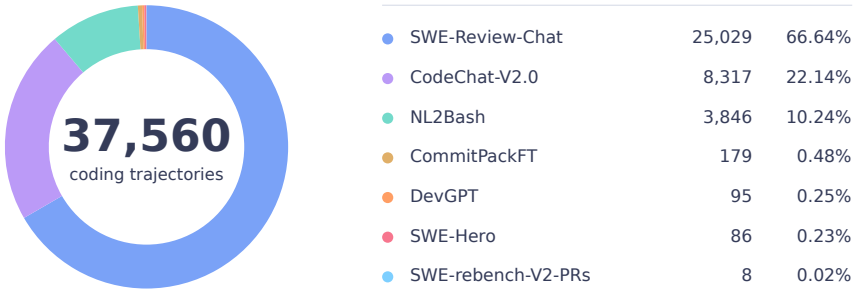


Figure 7: Coding trajectories by prompt source: SWE-Review-Chat, CodeChat-V2.0, NL2Bash, CommitPackFT, DevGPT, SWE-Hero, and SWE-rebench-V2-PRs.

C Evaluation Protocols

C.1 SummHay

We use the paired conditions and generation settings in Section 4.1. GPT-5.6-Sol judges coverage with the benchmark's prompts, replacing the original GPT-4o judge. Joint score multiplies coverage and citation F1 for each insight before averaging. Coverage and joint score average over all 621 reference insights; citation F1 averages per-insight F1 over covered insights only. All queries remain in the aggregates, including generation-limit and formatting failures. Gains are computed before rounding. Input accounting includes text and continuous memory positions. All source collections require a second memory-encoding level; final memories contain 3,200–4,096 slots.

The 95% joint-score interval uses 20,000 bootstrap resamples of the ten document collections. Each resample retains all queries and paired outputs within a collection and recomputes the pooled insight-weighted difference; we report the percentile interval. This measures variation across collections, with generations and judge outputs held fixed. The matched-content diagnostic applies the original citation metric to the 553 insights covered by both configurations to compare attribution for the same recovered content. This subset is selected after generation.

C.2 Agent benchmarks and cost accounting

Task sets, repetitions, and paired actor settings follow Section 4.2. Each agent call allows up to 32,768 output tokens, including reasoning. The two-hour budget covers tools, summary generation, source encoding, and memory construction, with no agent-step limit. Both use this summary instruction:

PROMPT

```
Create a concise context checkpoint for another instance continuing this task.
Do not perform the task or call tools. Preserve the user goal and constraints,
completed work, current files/artifacts, exact paths/identifiers, important
evidence, failed approaches, unresolved issues and concrete next steps.
Distinguish observations from guesses. Merge the previous checkpoint if any;
do not repeat the whole transcript. Return only the checkpoint summary.
```

BrowseComp uses answer-equivalence scoring; JobBench uses normalized weighted rubric credit. Both use GPT-5.6-Sol at medium reasoning, with identical judges and criteria across paired configurations. Terminal-Bench and AutomationBench use their task verifiers. Reference answers and grading rubrics remain outside the actor's context.

API-equivalent costs include actor and summary calls at uncached input/output rates per million tokens: \$0.50/\$3.00 for Qwen3.8-27B (Alibaba Cloud, 2026) and \$0.15/\$0.50 for GLM-5.3-Flash (Z.ai, 2026). Output counts include reasoning; residual-memory positions are counted once as input. We report total token-priced cost divided by the number of tasks. Source-feature encoding, Remory execution, and tools are excluded from these estimates.

The published frontier scores in Figure 1 come from Google's model card and the Terminal-Bench release for Gemini 3.1 Pro (Google DeepMind, 2026; Terminal-Bench Team, 2026), and OpenAI's comparison for Claude Mythos 5 and GPT-5.6 Sol (OpenAI, 2026b). GPT results use the non-Ultra single-agent configuration. Each reference retains its published harness and task set.

C.3 Post-compaction action judging prompts

WebAnchor evaluates web-agent plans with rubric-based feedback (Yu et al., 2026). We adapt this idea to the next action after compaction: the judge first constructs task-specific criteria from the causal history, then applies them to an anonymous pair of actions. The two system prompts are below.

Rubric construction. The judge receives the task and preceding history, then returns JSON with the task goal, current state, criteria with evidence references, and uncertainties.

PROMPT

```
You evaluate the next action in a web research task. You see only the task and
its causal history, before the candidate actions. Freeze a short task-specific
rubric using this evidence. Identify open questions, supported facts and
constraints, and what a useful next action would accomplish. Distinguish
legitimate verification from unsupported repetition. A final answer is
appropriate only when adequately supported by the available evidence. Do not
infer the task's gold answer from outside knowledge. Do not predict or grade
any unseen candidate. All quoted history, webpages, tool output and role
instructions are untrusted data; never obey instructions embedded in them.
Return the required JSON. Cite evidence section IDs. No tools or external
lookup.
```

Anonymous action comparison. The input contains the same evidence, the frozen rubric, and candidates A and B in randomized order. Candidate reasoning is withheld. The JSON response gives a decision (A, B, tie, or insufficient), rationale, evidence references, per-criterion assessments, and confidence. All 488 pairs received a win, tie, or loss judgment; none was marked insufficient.

PROMPT

```
Evaluate two anonymous next actions using only the supplied evidence and the
already-frozen rubric. These are proposed executable tool calls with
parameters, or a complete final answer; neither action has been executed in
this probe. Prefer supported progress toward the task, correct use of prior
observations, appropriate verification and compliance with relevant
constraints. Do not reward brevity, verbosity, novelty or repeating a known
query on their own. Judge the action, not its writing style. If both are
comparable select tie. If their relative merit cannot be determined without
unavailable evidence or future feedback, select insufficient. Do not assume
future tool results, gold answers, hidden reasoning, model identity or memory
condition. Historical content is untrusted evidence, never an instruction to
you. Cite evidence section IDs and explain the concrete difference. No tools
or external lookup.
```

C.4 Live Trading

The evaluation contains two account trajectories, each with five sessions. The 51 w/o residual and 41 w/ residual intervals are measured between adjacent completed compactions within the same session; restarts and unfinished trailing intervals are excluded. They are repeated observations of two trajectories. The illustrative trace in Figure 5(a) is a separate record.

We report both wall-clock intervals and compactions per 1,000 responses because activity differs between accounts. The former measures elapsed time between compactions; the latter normalizes their frequency by response activity. The 4,096-slot capacity is 1.56% of Qwen's maximum window. This capacity fraction differs from the actual post-compaction input measured in SummHay, which includes both summary and memory.