GAMBIT: A Benchmark for Active Memory in Long-Horizon LLM Agents

Anonymous ACL submission

Abstract

Long-horizon LLM agents must maintain, update, and revise task state over time. Yet current memory benchmarks mostly test passive retrieval, asking whether a model can find information in a fixed context. Active memory remains undermeasured, especially how models use changing task state across multi-step interactions. We introduce GAMBIT, a benchmark that uses deterministic game episodes to test active memory in LLM agents. It tests online updating, ordered recall, interference resistance, visuospatial tracking, episodic binding, and hypothesis revision. GAMBIT reveals that the passive-to-active memory gap persists across model families and scales. Models with near-perfect scores on passive retrieval benchmarks such as Needle-in-a-Haystack still fail when they must update state, resist interference, reconstruct episodic routes, or maintain hypotheses across feedback. These results show that reliable long-horizon agents require more than retrieval and need active control over changing state.

1 Introduction

Large Language Models (LLMs) are increasingly used as long-horizon agents rather than single-turn question-answering systems (Wang et al., 2024). In these settings, models must remember past interactions, update task state, follow changing rules, and avoid interference from irrelevant information (Park et al., 2023). These requirements make memory a central bottleneck for reliable LLM agents. Yet memory in current systems is often treated as a matter of context length: if a model can attend to more tokens, it is assumed to have better memory. This view misses a key distinction. Access to a long context window provides a passive storage buffer, but agentic behavior requires *active memory*: the ability to maintain, manipulate, and revise information over time.

This distinction matters because many failures in agentic systems are not failures of lookup. model may retrieve a previous instruction but fail to apply it after a rule changes, repeat an action it remembers, or fail to bind a relevant event to the correct location, time, or source. These failures arise when the model must update and use state across multiple steps, especially under delay, distraction, or competing information. Evaluating this capability requires benchmarks that test how information is maintained and transformed during an episode, not only whether it can be found in a static input.

Existing LLM memory benchmarks do not fully address this need. Long-context benchmarks such as *LongBench* (Bai et al., 2023) and *Needle In A Haystack* (NIAH) (Kamradt, 2023) mainly test passive retrieval: whether a model can locate information in a static document. These benchmarks are valuable, but they do not test whether the model can update that information, protect it from distractors, bind it to an episode, or use it consistently across multiple turns. Static natural-language benchmarks also introduce confounds such as positional bias, latent semantic overlap, data contamination, and sensitivity to prompt format (Liu et al., 2024; Magar and Schwartz, 2022; Sclar et al., 2023). As a result, high scores on passive retrieval can hide failures that matter for agentic behavior.

We introduce *GAMBIT*, the Game-based Active Memory Benchmark for Interference and Tracking. Game environments are particularly suited to this problem because they combine controlled state transitions, multi-turn interaction, and parametric difficulty control—properties that allow precise, reproducible measurement of active memory demands. GAMBIT evaluates active memory through deterministic game episodes probing the six demands above. Fixed seeds and controlled difficulty parameters give every model identical conditions, enabling paired comparisons and diagnostic scoring that separates parsing from cognitive errors.

Our experiments reveal a sharp gap between passive retrieval and active memory. Models that perform well on static retrieval benchmarks still fail when they must actively maintain and revise state across multiple turns. These failures are not fully explained by context length or formatting errors. They suggest that active state manipulation remains a key bottleneck for reliable LLM agents.

This work makes four contributions. The first frames active memory as a distinct evaluation target, separating the maintenance, update, and revision of task relevant state across multiple interactions from passive context retrieval. The second introduces a controlled, game-based benchmark that decomposes active memory into deterministic episodes and verifiable subproblems, each targeting a distinct cognitive demand (online updating, ordered recall, interference resistance, visuospatial tracking, episodic binding, hypothesis revision) under parametric constraints on capacity, delay, interference, structure, and rules. The third provides reproducible scoring with failure decomposition, using fixed seeds and paired episode comparisons alongside an analysis that separates parsing errors from cognitive errors to distinguish formatting failures from genuine memory failures. The fourth presents empirical evidence of a gap between passive and active memory, showing that strong performance on passive retrieval benchmarks does not imply reliable active memory across model families and scales, with identifiable failures in state updating, interference control, episodic navigation, and hypothesis tracking.

2 GAMBIT

2.1 Active Memory Taxonomy

Memory is evaluated as an active process rather than as static retrieval. In agentic settings, a model must maintain task-relevant state, update it after new observations, protect it from distractors, and use it consistently across multiple steps. This capability is decomposed into six active memory demands: *online updating* (refreshing state after each observation), *ordered recall* (preserving item identity and sequence), *interference resistance* (protecting targets from distractors), *visuospatial tracking* (maintaining and updating spatial state), *episodic binding* (associating events with place and context), and *hypothesis revision* (updating beliefs from feedback). These six demands motivate the selection and parameterization of the five task families depicted in Figure 3.

Each demand corresponds to a primary failure mode that becomes observable through these subproblems: stale-state errors for online updating; omissions, swaps, and intrusions for ordered recall; distractor contamination for interference resistance; repeated or invalid actions for visuospatial tracking; source or route errors for episodic binding; and contradictions or repeated guesses for hypothesis revision. The taxonomy guides benchmark design: each task family isolates one or more active memory demands while keeping the environment deterministic, the response space measurable, and the failure modes interpretable.

2.2 Task Families

The GAMBIT comprises five task families, totaling 21 deterministic variants. Figure 1 shows the actual text-based prompt–response sequences used in evaluation, while Figure 2 provides a visual conceptual overview of each family's game-like structure. The mechanics of each family are described below, together with the active memory demands targeted and the instantiation of episodes in text. The full list of variants, parameter ranges, and prompt templates is provided in Appendix A. We contrast these design choices against existing long-context evaluations in Section 3.2.

N-back. The model observes a sequence of stimuli (e.g., letters, digits, or visuospatial coordinates) presented one per turn. At each turn t, the model must respond whether the current stimulus matches the item presented n steps back. A match requires the model to update and maintain a sliding window of the last n observations, discarding outdated items. This family primarily targets *online updating* and *interference resistance*, as the model must inhibit responses based on non-target lags. The backlog level is varied over $n \in \{1, 2, 3\}$, together with sequence length and match rate.

Memory Cards. The model is shown an ordered list of items (e.g., words, symbols, or emojis). After a retention interval that may contain distractor items or unrelated queries, the model must reproduce the original list in the exact order. Variants manipulate list size, retention delay, and the presence of phonological or semantic distractors. This family isolates *ordered recall* and *interference resistance*. Failure modes include item omissions, order swaps, and intrusions from the distractor set.

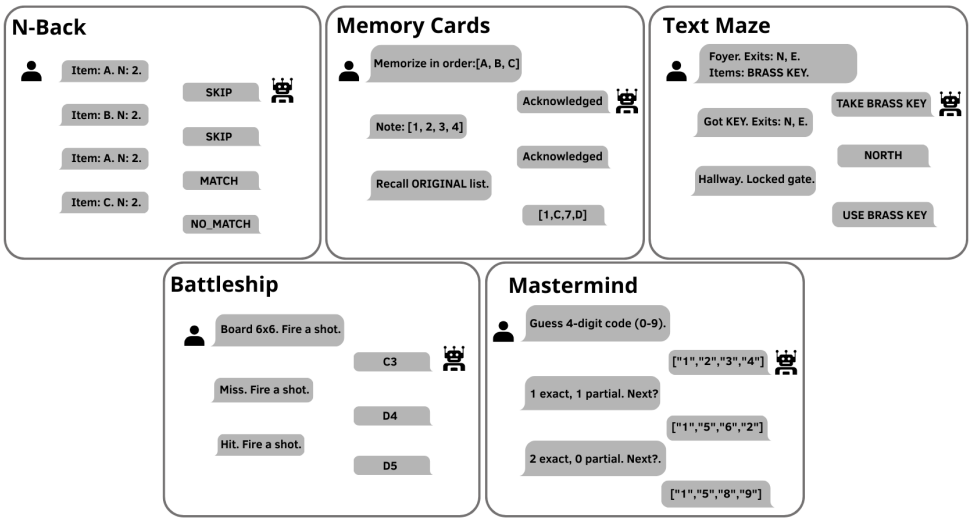


Figure 1: Exemplar text-based episodes for each GAMBIT task family. Each panel shows a representative prompt–response sequence as received by the model during evaluation. (Check out our zero-shot prompt templates Appedinx B.3)

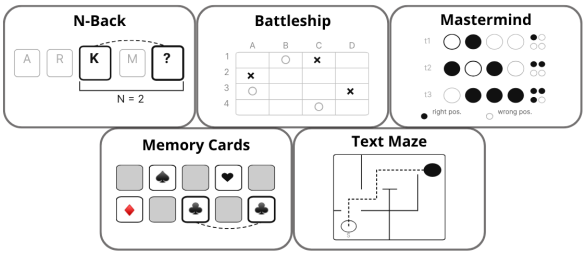


Figure 2: Visual conceptual sketch of the five GAMBIT task families. Each panel illustrates the intuitive game-like structure underlying the text-based tasks. *Note:* These visual interfaces are illustrative sketches only; the benchmark uses purely text-based interactions (see Figure 1).

Battleship. The model plays a simplified Battleship game on a grid of size $k \times k$. Ships of varying lengths are placed at hidden coordinates. The model issues shot coordinates turn by turn and receives feedback (hit, miss, or sunk). It must maintain an internal representation of the board state, track damaged ships, and avoid repeating shots. This family targets *visuospatial tracking* and *online updating*. Difficulty is controlled via board size, ship density, and mid-episode rule changes (e.g., rotated boards or restricted firing zones).

Text Maze. The model navigates a grid-world maze described entirely in text. At each turn, the model receives a local observation (e.g., "you are at intersection B3; north is a wall, east is a corridor, south is a door"). It must issue movement commands and, at probe turns, answer questions about previously visited locations or reconstruct the route taken. This family targets *episodic binding* of spatial position to contextual landmarks. Difficulty is controlled by maze size, route length, and the insertion of distractor corridors.

Mastermind. The model plays the classic code-breaking game. A secret code of length c is drawn from a color pool of size p. The model submits guesses and receives feedback in the form of black pegs (correct color in correct position) and white pegs (correct color in wrong position). It must maintain a hypothesis space and revise beliefs based on deterministic feedback. This family isolates *hypothesis revision*. Code length, pool size, and the maximum number of attempts are varied.

2.3 Episode Generation and Difficulty Control

The complete GAMBIT evaluation pipeline is illustrated in Figure 3. All episodes are generated offline before inference and stored as immutable, seed-specific datasets. This design separates benchmark randomness from model behavior: every model receives the same episode sequence, task conditions, prompts, and scoring rules. Performance differences are therefore attributable to model behavior rather than to dataset mismatch, sampling variation, or changes in evaluation conditions.

Difficulty is controlled through five memory-load factors, sampled via a balanced factorial design to prevent the benchmark from collapsing into a single difficulty regime. *Capacity* measures the amount of information that must be maintained

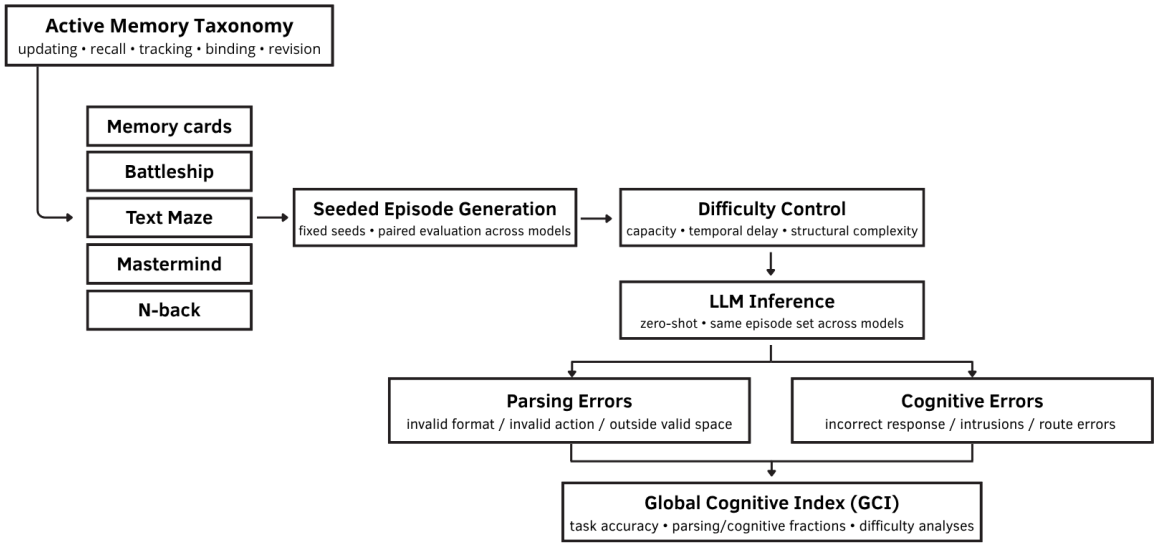


Figure 3: The GAMBIT evaluation pipeline. (**Left**) The active memory taxonomy guides the design of five task families. (**Center**) Episodes are generated from fixed seeds with controlled difficulty parameters. (**Right**) Model responses are decomposed into parsing and cognitive errors, then aggregated into the Global Cognitive Index (GCI).

(e.g., list length, board size). *Temporal delay* captures retention across turns. *Interference* measures the presence and similarity of distractors. *Structural complexity* captures task format and environment complexity (e.g., maze branching factor). *Rule constraints* capture output restrictions and mid-episode rule changes. The resulting composite difficulty score is used for aggregate degradation curves and task-specific analyses.

2.4 Failure Decomposition

A central challenge in structured generation is separating formatting failures from memory failures. Total error is therefore decomposed into parsing errors and cognitive errors. For bounded-response tasks, let the model response at turn t be denoted by r_t, and let the valid action space be $\mathcal{A}$. A parsing error occurs when $r_t \notin \mathcal{A}$. A cognitive error occurs when $r_t \in \mathcal{A}$ but $r_t \neq y_t$, where y_t is the ground-truth target action. The total error is defined as

$$E_{\text{total}} = E_p + E_c, \tag{1}$$

where E_p is the parsing error rate and E_c is the cognitive error rate. The normalized fractions

$$F_p = \frac{E_p}{E_{\text{total}}}, \qquad F_c = \frac{E_c}{E_{\text{total}}}, \tag{2}$$

are reported with safeguards for the zero-error case. This decomposition prevents invalid formatting from being mistaken for memory degradation and enables fine-grained diagnosis of model failures, see in Figure 3.

3 Evaluation Protocol and Baselines

3.1 Evaluation Metrics

Task-level metrics are reported before any aggregate score, because each task family targets a different memory demand. `n_back` is evaluated with post-warmup accuracy and Matthews Correlation Coefficient (MCC). `memory_cards` uses item accuracy and order accuracy. `battleship` uses a spatial memory score and behavioral measures such as repeat-shot rate. `text_maze` uses episodic probe accuracy and route reconstruction metrics. `mastermind` uses solve rate and contradiction frequency.

A Global Cognitive Index (GCI) is computed to summarize performance across heterogeneous tasks. For each seed, task-family scores are standardized and then averaged at the model level. The GCI is used only as a secondary summary; all main analyses remain grounded in task-level results.

3.2 Contextualization with Existing Benchmarks

Table 1 summarizes the methodological gap between existing evaluations and the requirements of active-memory assessment. NIAH, LongBench, and RULER operate on static documents, so they cannot test state maintenance across turns or the active updating demands of agentic tasks. BABILong introduces multi-turn interaction but still treats memory as retrieval rather than manipulation, and neither it nor AgentLongBench fixes episode

	Multi-turn	Active state	Fixed-seed	Failure diagnosis
NIAH	✗	✗	✗	✗
LongBench	✗	✗	✗	✗
RULER	✗	✗	✗	✗
BABILong	✓	✗	✗	✗
AgentLongBench	✓	✓	✗	✗
GAMBIT	✓	✓	✓	✓

Table 1: Comparison of long-context and agent benchmarks. GAMBIT is the only benchmark to combine multi-turn interaction, active state maintenance, fixed-seed reproducible episodes, and diagnostic error decomposition.

seeds, which prevents paired statistical comparison and leaves performance variance confounded by task sampling. AgentLongBench further lacks diagnostic error decomposition, meaning formatting failures are indistinguishable from genuine memory failures. By combining multi-turn interaction, active-state maintenance, fixed-seed reproducibility, and cognitive-versus-parsing error decomposition, GAMBIT enables controlled experiments that isolate memory as an active process rather than a passive retrieval bottleneck.

To contextualize active memory against passive retrieval, a representative subset of models is also evaluated on **Needle-In-A-Haystack (NIAH)** (Kamradt, 2023) and **LongBench** (Bai et al., 2023). NIAH measures raw context window capacity through single fact retrieval. LongBench measures multi task comprehension over long documents. Both benchmarks are valuable for passive retrieval, yet neither requires models to maintain and update task state across multiple turns, resist interference under feedback, or revise hypotheses from structured feedback. They therefore do not test active memory. GAMBIT fills this gap by evaluating the maintenance, manipulation, and revision of task relevant state under controlled load, with diagnostic decomposition of cognitive versus parsing failures.

3.3 Statistical Analysis

Uncertainty is estimated with non-parametric bootstrap confidence intervals and paired permutation tests. Because all models are evaluated on matched episodes, comparisons are performed at the episode level, increasing statistical power and reducing variance from task sampling.

4 Experiments

4.1 Experimental Setup

Evaluation covers instruction-tuned models from the Gemma, Llama, Mistral, Qwen, DeepSeek, GPT, and Gemini families, spanning 0.6B to 120B parameters, including R1-Distill-Llama-70B, GPT OSS 120B, and frontier proprietary models (GPT 5.4, GPT 5.5, Gemini 3.1 Pro). All episodes are generated offline from fixed seeds and evaluated zero-shot under greedy decoding with temperature 0.0, ensuring identical episode sets, prompts, and response constraints across all systems. Task-level metrics comprise post-warmup accuracy and Matthews Correlation Coefficient for N-back; item and order accuracy for Memory Cards; spatial memory score and repeat-shot rate for Battleship; episodic probe accuracy and route reconstruction accuracy for Text Maze; and solve rate and contradiction frequency for Mastermind. A Global Cognitive Index (GCI) is computed as the mean of standardized task-family scores per model and reported as a secondary summary, with all main analyses grounded in task-level results. Passive retrieval baselines are established via Needle In A Haystack (NIAH), which tests single-fact retrieval in long contexts, and LongBench, which tests multi-task long-context understanding through naturalistic documents; neither requires multi-turn state maintenance or active updating.

4.2 Task-Level Active Memory Performance

Table 2 reports mean accuracy per task family and the aggregate GAMBIT average. First, ordered recall is near-saturated while other demands are brittle: Memory Cards exceeds 0.90 for 14 of 25 models, with Gemma 3 12B at 1.000, whereas Mastermind solve rates do not exceed 0.110 and Text Maze stays below 0.288. This asymmetry shows that item identity is preserved more reliably than hypotheses are revised, routes reconstructed, or structured state maintained. Second, N-back accuracy is low and non-monotonic with scale, ranging from 0.027 (Llama 3.1 8B) to 0.490 (Llama 3.2 1B); larger models within a family do not consistently outperform smaller ones (Llama 3.2 1B exceeds Llama 3.1 8B; Gemma 3 1B exceeds Gemma 3 12B), indicating that online updating is not predicted by parameter count alone. Third, even the best model shows uneven capability: Gemini 3.1 Pro achieves the highest GAMBIT average (0.481) and leads on three of five task families, yet its Mas-

termind (0.110) and Text Maze (0.288) scores remain low, showing that frontier models still struggle with hypothesis revision and episodic binding.

4.3 Passive Retrieval Does Not Predict Active Memory

Table 4 reports aggregate scores on GAMBIT, NIAH, and LongBench for the same set of models. Two findings are apparent.

Large retrieval scores coexist with low active memory scores. Llama 3.1 8B achieves perfect NIAH accuracy (1.000) but only 0.304 on GAMBIT. Gemma 3 12B scores 0.865 on NIAH and 0.487 on LongBench, yet only 0.373 on GAMBIT. These gaps show that locating information in a static context is not sufficient for active memory.

Rankings diverge across benchmarks. Qwen3.5-9B trails R1-Distill-Qwen-32B on GAMBIT (0.414 vs. 0.428, a narrow margin) but trails more sharply (0.891 vs. 0.923) and LongBench (0.552 vs. 0.598). GPT 5.5 leads on NIAH (0.951) and LongBench (0.634) but does not lead on GAMBIT (0.447, below Gemini 3.1 Pro's 0.481). This divergence indicates that active memory and passive retrieval are distinct capabilities that do not share a single latent factor.

4.4 Difficulty Scaling and Failure Modes

Figure 4 shows normalized accuracy as a function of composite difficulty quantile. All model families exhibit non-monotonic degradation: accuracy rises from the easiest quantile (0.05) to the second quantile (0.15), then declines through the middle range, with a partial recovery at quantile 0.35 before a final drop at the highest difficulty.

The initial rise reflects floor effects: the easiest episodes contain minimal memory load, and all models perform near chance. The middle-range decline marks the onset of active memory demands: models must now maintain state across turns, resist distractors, and update hypotheses, capabilities that degrade faster than simple retrieval. The partial recovery at quantile 0.35 corresponds to episodes with high structural complexity but low interference, where larger models can compensate through pattern matching. The final drop at extreme difficulty shows the combined effect of all load factors.

Failure decomposition (Table 3) shows that cognitive errors dominate parsing errors across all task families. In N-back, 94% of errors are cognitive (stale-state or lag-confusion). In Battleship, 78% of errors are cognitive (including repeated shots), while 22% are parsing errors. Mastermind shows the highest parsing error rate (18%), but even here cognitive errors (contradictions and repeated guesses) account for the majority of failures.

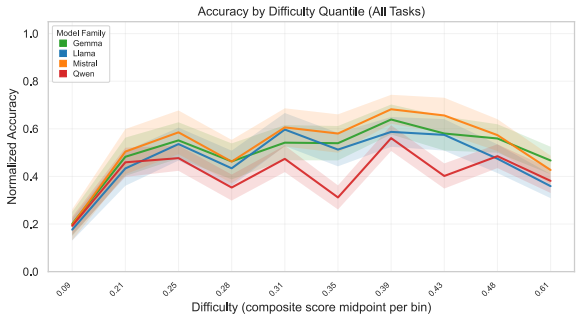


Figure 4: Normalized accuracy by composite difficulty quantile. Each line represents a model family (Gemma, Llama, Mistral, Qwen), averaged across all task variants. Shaded regions denote 95% bootstrap confidence intervals. The non-monotonic pattern reflects floor effects at low difficulty, active-memory degradation at intermediate difficulty, and combined-load failure at high difficulty.

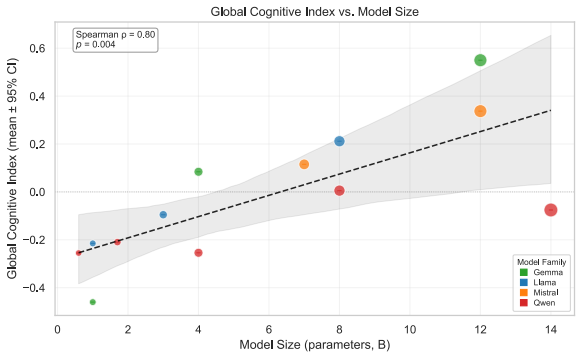


Figure 5: Global Cognitive Index (standardized) versus model size. Each point is a model; colors denote families. The dashed line is a LOESS fit. Spearman $\rho = 0.80$ ($p = 0.004$).

4.5 Model Scale with Downstream Capabilities

Figure 5 shows GCI correlates positively with model size ($\rho = 0.80$, $p = 0.004$), but residual variation is large (e.g., Gemma 3 4B underperforms Gemma 3 1B; R1-Distill-Llama-70B underperforms R1-Distill-Qwen-32B), indicating that scale is not the sole determinant of active memory.

Figure 6 reports correlations between GAMBIT scores and three downstream evaluation metrics: instruction update (IFEval), grounding (TriviaQA), and temporal reasoning (GSM8K).

Table 2: Mean accuracy per GAMBIT task family. Higher is better. Bold indicates the best score per column. Models are grouped by family and ordered by parameter count within each family.

Model	N-back	Memory Cards	Mastermind	Battleship	Text Maze
Gemma 3 1B	0.461	0.478	0.040	0.236	0.130
Gemma 3 4B	0.228	0.904	0.080	0.327	0.119
Gemma 3 12B	0.133	**1.000**	0.040	0.540	0.150
Llama 3.2 1B	**0.490**	0.674	0.000	0.231	0.116
Llama 3.2 3B	0.337	0.916	0.080	0.250	0.079
Llama 3.1 8B	0.027	0.940	0.000	0.451	0.100
Llama 3.1 70B	0.281	0.950	0.080	0.460	0.136
Mistral 7B	0.105	0.881	0.040	0.415	0.115
Mistral Nemo 12B	0.107	0.968	0.040	0.487	0.098
Qwen3 0.6B	0.167	0.656	0.000	0.251	0.125
Qwen3 1.7B	0.199	0.654	0.000	0.299	0.119
Qwen3 4B	0.208	0.697	0.000	0.289	0.132
Qwen3 8B	0.140	0.756	0.040	0.370	0.172
Qwen3 14B	0.110	0.766	0.000	0.297	0.206
Qwen3 30B-A3B	0.134	0.790	0.080	0.312	0.257
Qwen3 32B	0.310	0.818	0.100	0.377	0.248
Qwen3-235B-A22B	0.418	0.982	0.100	0.410	0.265
Qwen3.5-0.8B	0.175	0.689	0.020	0.267	0.146
Qwen3.5-2B	0.203	0.692	0.020	0.269	0.151
Qwen3.5-4B	0.239	0.810	0.040	0.295	0.167
Qwen3.5-9B	0.412	0.952	0.100	0.394	0.214
Qwen3.5-27B	0.430	0.963	0.100	0.388	0.236
Qwen3.5-35B-A3B	0.435	0.975	0.100	0.399	0.241
Qwen3.5-122B-A10B	0.400	0.978	0.100	0.422	0.250
R1-Distill-Llama-8B	0.276	0.902	0.040	0.513	0.188
R1-Distill-Qwen-14B	0.299	0.943	0.080	0.550	0.191
R1-Distill-Qwen-32B	0.314	0.959	0.080	0.583	0.202
R1-Distill-Llama-70B	0.387	0.977	0.100	0.265	0.251
GPT 5.4	0.321	0.945	0.070	0.550	0.125
GPT 5.5	0.382	0.970	0.100	0.576	0.205
GPT 5.4 Mini	0.281	0.900	0.060	0.498	0.111
GPT OSS 120B	0.333	0.986	0.105	0.272	0.263
Gemini 3.1 Flash Lite	0.299	0.924	0.069	0.493	0.186
Gemini 3.1 Pro	0.410	0.992	**0.110**	**0.607**	**0.288**

Table 3: Error decomposition by task family. Parsing errors (E_p) occur when the model output violates the required response format and is rejected by the parser. Cognitive errors (E_c) occur when the output is format-valid but incorrect given the task state.

Task family	E_p (%)	E_c (%)	Dominant failure mode
N-back	6	94	Stale-state
Memory Cards	3	97	Omission
Battleship	22	78	Repeat shot
Text Maze	8	92	Source error
Mastermind	18	82	Contradiction

GAMBIT correlates with instruction-update accuracy ($\rho = 0.65$, $p = 0.006$), suggesting that models that maintain and update state in game environments also handle dynamic instruction constraints. The correlation with grounding ($\rho = 0.04$, $p = 0.908$) is not significant, consistent with the observation that static fact retrieval (TriviaQA) does not require active state maintenance. The correlation with temporal reasoning ($\rho = 0.17$, $p = 0.586$) is also not significant; while both tasks involve sequence processing, GSM8K requires arithmetic reasoning rather than state tracking across turns.

These results indicate that GAMBIT captures a capability dimension that is related to instruction following but distinct from static knowledge retrieval and mathematical reasoning.

5 Related Work

Psychological Foundations of Memory Memory is not a single storage mechanism. Cognitive psychology separates memory into systems with different roles, including working memory, episodic memory, semantic memory, and long-term associative memory (Baddeley, 1992; Tulving, 1972, 1983). These distinctions are useful for LLM evaluation because failures in long-horizon tasks can arise from different sources: losing a recent state, binding an event to the wrong context, confusing similar items, or failing to update a rule.

Classical theories of forgetting also show that

Table 4: Cross-benchmark comparison. GAMBIT Avg is the aggregate active memory score. NIAH Avg is the mean retrieval accuracy across context lengths. LongBench Avg is the mean score across all LongBench subtasks. Higher is better. Bold indicates the best score per column.

Model	GAMBIT Avg	NIAH Avg	LongBench Avg
Gemma 3 1B	0.269	0.782	0.245
Gemma 3 4B	0.331	0.821	0.398
Gemma 3 12B	0.373	0.865	0.487
Llama 3.2 1B	0.302	0.968	0.312
Llama 3.2 3B	0.332	0.847	0.421
Llama 3.1 8B	0.304	1.000	0.445
Llama 3.1 70B	0.310	1.000	0.454
Mistral 7B	0.311	0.834	0.412
Mistral Nemo 12B	0.340	0.871	0.498
Qwen3 0.6B	0.240	0.745	0.198
Qwen3 1.7B	0.254	0.768	0.267
Qwen3 4B	0.265	0.791	0.356
Qwen3 8B	0.296	0.823	0.421
Qwen3 14B	0.276	0.812	0.445
Qwen3 30B-A3B	0.314	0.856	0.478
Qwen3 32B	0.370	0.889	0.534
Qwen3-235B-A22B	0.408	0.900	0.553
Qwen3.5-0.8B	0.259	0.751	0.261
Qwen3.5-2B	0.267	0.778	0.387
Qwen3.5-4B	0.310	0.812	0.500
Qwen3.5-9B	0.414	0.891	0.552
Qwen3.5-27B	0.420	0.890	0.537
Qwen3.5-35B-A3B	0.435	0.894	0.568
Qwen3.5-122B-A10B	0.440	0.910	0.607
R1-Distill-Llama-8B	0.384	0.867	0.512
R1-Distill-Qwen-14B	0.413	0.901	0.567
R1-Distill-Qwen-32B	0.428	0.923	0.598
R1-Distill-Llama-70B	0.396	0.945	0.621
GPT 5.4	0.402	0.912	0.578
GPT 5.5	0.447	0.951	0.634
GPT 5.4 Mini	0.370	0.878	0.489
GPT OSS 120B	0.391	0.934	0.612
Gemini 3.1 Flash Lite	0.394	0.901	0.556
Gemini 3.1 Pro	**0.481**	**0.978**	**0.689**

memory errors are shaped by delay and interference. The forgetting curve describes how recall degrades over time, while interference-based accounts explain how competing information can disrupt retrieval (Ebbinghaus, 1885; Anderson and Neely, 1996). These ideas are central to agentic LLM settings, where each new observation can overwrite, distract from, or conflict with earlier information.

LLMs as Agents in Sequential and Partially Observable Environments Game-based benchmarks such as *GameBench* evaluate strategic reasoning across diverse games (Lin et al., 2024). They are useful for measuring general agent performance, but they do not decompose memory into specific active-memory demands. GAMBIT complements this line of work by focusing on memory diagnostics: what information must be maintained, what must be updated, what interferes with it, and which failure mode appears.

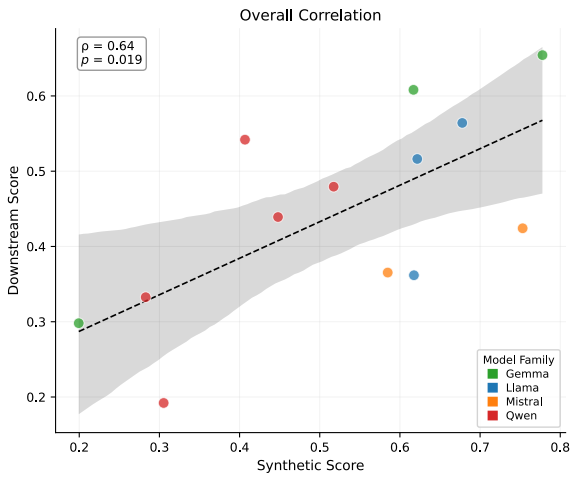


Figure 6: Correlation between GAMBIT GCI and IFEval instruction-update accuracy. Each point is a model; colors denote families. Spearman $\rho = 0.65$ ($p = 0.006$).

6 Conclusion

We introduced *GAMBIT*, a benchmark for evaluating active memory in long-horizon LLM agents. Unlike passive retrieval benchmarks, GAMBIT tests whether models can maintain, update, and revise task-relevant state across multi-step interactions. The benchmark uses deterministic game episodes with controlled difficulty factors, enabling reproducible comparisons and task-level diagnosis across online updating, ordered recall, interference resistance, visuospatial tracking, episodic binding, and hypothesis revision.

Our experiments show a clear gap between passive retrieval and active memory. Models that perform well on Needle In A Haystack and LongBench can still fail when they must update state, avoid repeated actions, reconstruct episodic routes, or maintain consistent hypotheses. These failures are not fully explained by context length or formatting errors. They point to active state manipulation as a remaining bottleneck for reliable LLM agents.

GAMBIT provides a controlled and interpretable way to measure this bottleneck. By separating task-specific memory failures from output-format failures, it can help researchers diagnose where current models break and guide future work on memory-centric LLM systems.

Limitations

GAMBIT has several limitations. First, it evaluates visuospatial and episodic memory through text-based environments, such as grid states in

battleship and room navigation in text_maze. These tasks impose structured memory demands, but they are still textual proxies. Performance on GAMBIT may therefore not fully reflect memory behavior in multimodal, embodied, or natively spatial agents.

Second, GAMBIT focuses on high-density active memory rather than extreme context length. Its episodes are designed to test updating, interference control, episodic binding, and rule tracking under controlled load. They do not evaluate million-token retrieval, as in long-context document benchmarks (Kamradt, 2023; Bai et al., 2023). GAMBIT should therefore be viewed as a complement to passive retrieval benchmarks, not as a replacement.

Third, comparisons across models with very different context windows require care. Some smaller models may lose earlier turns because of hard context limits, especially in longer or harder episodes. In these cases, errors may reflect both memory failure and architectural capacity limits. This makes it difficult to fully separate active forgetting from unavailable context.

References

John R. Anderson and James H. Neely. 1996. Interference and inhibition in memory retrieval. *Memory*.

Alan Baddeley. 1992. Working memory. *Science*, 255(5044):556–559.

Yushi Bai, Xin Lv, Jiajie Zhang, Hongchang Lyu, Jiankai Tang, Zhidian Huang, Zhengxiao Du, Xiao Liu, Aohan Wu, Yuxiao Cao, and 1 others. 2023. LongBench: A bilingual, multitask benchmark for long context understanding. *arXiv preprint arXiv:2308.14508*.

Hermann Ebbinghaus. 1885. *Memory: A Contribution to Experimental Psychology*. Teachers College, Columbia University.

Greg Kamradt. 2023. Needle in a haystack - pressure testing LLMs. https://github.com/gkamradt/LLMTest_NeedleInAHaystack.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with PagedAttention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*, SOSP '23. ACM.

Kevin Lin and 1 others. 2024. Gamebench: Evaluating strategic reasoning abilities of llm agents. In *NeurIPS*.

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

Inbal Magar and Roy Schwartz. 2022. Data contamination: From memorization to exploitation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 157–165.

Joon Sung Park, Joseph C. O'Brien, Carrie J. Cai, Meredith Ringel Morris, Percy Liang, and Michael S. Bernstein. 2023. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*, pages 1–22.

Melanie Sclar, Yejin Choi, Yejin Tsvetkov, and Alane Suhr. 2023. Quantifying language models' sensitivity to spurious features in prompt design or: How i learned to start worrying about prompt formatting. *arXiv preprint arXiv:2310.11324*.

Endel Tulving. 1972. Episodic and semantic memory. In Endel Tulving and Wayne Donaldson, editors, *Organization of Memory*, pages 381–403. Academic Press.

Endel Tulving. 1983. *Elements of Episodic Memory*. Oxford University Press.

Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakang Tang, Xu Chen, Yankai Lin, and 1 others. 2024. A survey on large language model based autonomous agents. *Frontiers of Computer Science*, 18(6):186345.

A Full List of Benchmark Task Variants

As referenced in Section 4, the full evaluation battery encompasses 21 unique task variants systematically mapped across five cognitive game families. Below is the granular list enumerating all conditions implemented in our framework exactly as generated in the benchmark configuration.

1. Online Updating and Interference Monitoring

- **n_back:** The core online memory monitoring baseline. Evaluates continuous updating with tunable N level, sequence lengths, and symbol pools, strictly parsing for MATCH/NO_MATCH/SKIP actions.

2. Ordered Recall and Temporal Retention (Memory Cards)

- **immediate_recall:** Baseline ordered list encoding followed immediately by sequential retrieval.

- **delayed_recall:** Ordered list retrieval following parameterized empty retention intervals (delay turns).
- **interference:** List retrieval compromised by competitive, semantically similar distractor items before query.
- **recognition:** Distinguishing seen target items from novel unseen intrusions rather than producing free recall.
- **cued_recall:** Memory retrieval triggered by partial item cues (e.g., prefix, position, or masked context).
- **contradiction:** Dynamic memory updating where later rule/item observations explicitly contradict and overwrite previously memorized associations.

3. Visuospatial Tracking and Rule Maintenance (Battleship)

- **battleship:** Navigating grid state and tracking hit/miss coordinates logically without external interference.
- **battleship_delay_only:** Tracking the board state across retention turns before further action is permitted.
- **battleship_competitive_distractor:** Spatial tracking challenged by the appearance of false coordinates or irrelevant target descriptions.
- **battleship_rule_switch:** Mid-episode inversion or modification of the underlying targeting rules.
- **battleship_occluded_history:** Tracking a grid state when past actions/observations are fully hidden from the immediate prompt horizon.
- **battleship_dual_target_tracking:** Simultaneous tracking of multiple independent fleets/systems overlapping on a grid.
- **battleship_coordinate_remap:** Shifting the frame of reference or grid axis labels dynamically mid-game.
- **battleship_working_memory_probe:** Interrupting regular play to explicitly probe the memorized status of specific coordinates.
- **battleship_confidence_calibration:** Enforcing confidence bounds applied dynamically to the model's spatial predictions.

4. Episodic Navigation and Spatial Routing (Text Maze)

- **text_maze:** Environment exploration leading to finding a target, locking state, and extracting the route memory.
- **text_maze_distractor:** Navigational route execution disrupted by unrelated procedural turns.
- **text_maze_source_memory:** Explicitly probing episodic contexts (e.g., exactly *where* or *in which context* an event occurred).
- **text_maze_route_retrace:** Mentally mirroring and outputting the reverse steps of the previously explored path based solely on memory.

5. Iterative Hypothesis Updating

- **mastermind:** Sequential logic deductions using exact memory of prior sequence guesses and their partial/exact match feedback pairs, avoiding logical contradiction.

B Dataset Statistics, Difficulty Factor Distributions, and Prompt Templates

This appendix provides supplementary material for the benchmark described in Section 4: aggregate dataset statistics (§B.1), the composite difficulty model and its per-factor parameter grids (§B.2), and the standardised zero-shot system prompts used across all task environments (§B.3).

B.1 Dataset Statistics

The benchmark comprises **21 task variants** spanning five cognitive game families. Episodes are drawn with a *balanced factorial* sampling strategy (seed = 42) so that every parametric condition is represented proportionally within the evaluation suite. Table 5 reports the task inventory and the ranges of the principal difficulty parameters for each family.

B.2 Difficulty Factor Distributions

Each episode is assigned a continuous composite difficulty score $D \in [0, 1]$ computed as a weighted sum of five orthogonal cognitive load dimensions:

Table 5: Benchmark dataset composition. Parameter ranges reflect all values exercised during balanced factorial sampling.

Game Family	Task Variants	#	Key Parameter Ranges
N-Back	n_back	1	$N \in \{1, 2, 3\}$; seq. len. $\in \{8, 12, 16\}$; pool $\in \{4, 6, 8\}$; match prob. $\in \{0.3, 0.5\}$
Memory Cards	immediate_recall delayed_recall interference recognition cued_recall contradiction	6	list size $= \lfloor \text{board} \times 0.7 \rfloor$; board $\in \{4, 8, 16, 32, 50\}$ delay turns $\in \{0, 1, 2, 4, 8, 12\}$ distractor turns $\in \{0, 1, 3\}$; similarity $\in$ {low, med, high} — cue strategies: prefix, position, masked min. delay $= 2$; item offset $= 100$
Battleship	battleship battleship_delay_only battleship_competitive_distractor battleship_rule_switch battleship_occluded_history battleship_dual_target_tracking battleship_coordinate_remap battleship_working_memory_probe battleship_confidence_calibration	9	board $\in \{6, 8, 10\}$; shot budget $= 2\times$ board area
Text Maze	text_maze text_maze_distractor text_maze_source_memory text_maze_route_retrace	4	grid $\in \{4, 5\}$; explore turns $\in \{6, 8, 10\}$; distractor turns $\in \{1, 2\}$
Mastermind	mastermind	1	code len. $\in \{4, 5\}$; pool $\in \{6, 8, 10\}$; duplicates $\in$ {F,T}; attempts $\in \{8, 10\}$
Total		**21**	

$$D = w_C \cdot C + w_T \cdot T + w_I \cdot I + w_S \cdot S + w_R \cdot R \quad (3)$$

where, under the default equal-weight configuration, $w_C = w_T = w_I = w_S = w_R = 0.2$. Table 6 defines each factor and its normalisation.

Delay distribution. To balance immediate-recall and high-retention episodes, delay turns are sampled from the weighted discrete distribution shown in Table 7.

The exponential-decay weighting ensures adequate representation of both the easiest (no delay) and most demanding (12-turn retention) conditions, while preventing the distribution from collapsing to a single difficulty tier.

B.3 Zero-Shot Prompt Templates

All task environments are presented to models through standardised, zero-shot system prompts. The prompts fix the response format unambiguously (single token, JSON array, or coordinate label) to isolate working-memory failures from output-format sensitivity. Conversation history within each episode is provided by the game engine; no few-shot examples are injected.

Table 6: Composite difficulty factors and their normalisation functions. $d_{\max}$ denotes the maximum delay turns in the configuration.

Factor	Name	Captures	Normalisation
C	Capacity	Working-memory load (list size, board area)	$0.6\,\hat{b} + 0.4\,\hat{n}$
T	Temporal	Delay-based retention demands	$\log_2(d + 1) \,/\, \log_2(d_{\max} + 1)$
I	Interference	Distractor competition and semantic similarity	$0.6\,\hat{v} + 0.4\,\hat{s}$
S	Structural	Task-inherent baseline difficulty	fixed per task (range 0.10–0.85)
R	Rule	Response format complexity; mid-episode rule changes	binary/ordinal penalty

```
You are playing the N-back memory task.

In each turn, you will see:
1. The current item
2. The value of N
3. Your task: Respond with exactly ONE action token:
   MATCH, NO_MATCH, or SKIP

Rules:
- MATCH: The current item is the same as the item from N turns ago
- NO_MATCH: The current item is different from the item from N turns ago
- SKIP: You haven't seen N items yet (only in the first N turns)

Examples:
- If N=2 and sequence is [A, B, A, C, A]:
  - Turn 1 (A, idx=0): SKIP   (idx < N)
  - Turn 2 (B, idx=1): SKIP   (idx < N)
  - Turn 3 (A, idx=2): MATCH  (A == A from 2 turns ago)
  - Turn 4 (C, idx=3): NO_MATCH (C != B from 2 turns ago)
  - Turn 5 (A, idx=4): MATCH  (A == C from 2 turns ago)

Respond with ONLY the single action token, nothing else.
```

Figure 7: Prompt template for the N-back memory task.

```
You are playing a memory game. Your task is to remember a list
of items and recall them accurately.

When you see the items to remember, focus on encoding them well.
When asked to recall, return ONLY a JSON list of items in the
exact order presented, with no additional text.

Example format: ["item_0", "item_1", "item_2", "item_3"]

Be precise. Return exactly what you remember.
```

Figure 8: Prompt template for the item recall memory task.

Table 7: Sampling weights for delay turns in temporal-retention tasks.

Delay turns	0	1	2	4	8	12
Weight	0.30	0.22	0.18	0.14	0.10	0.06

```
You are playing a memory game with a delay component.

First, you will see items to remember. After a delay period with
unrelated information, you will be asked to recall those items.

When asked to recall, return ONLY a JSON list of items in the
exact order presented initially, with no additional text.

Example format: ["item_0", "item_1", "item_2", "item_3"]

Do your best to remember despite the delay.
Return exactly what you recall.
```

Figure 9: Prompt template for the delayed recall memory task.

```
You are playing a memory game with interference.

First, you will see target items to remember. Then you will
encounter interfering/distractor items. Your goal is to ignore
the distractors and recall only the original target items
when asked.

When asked to recall, return ONLY a JSON list of the ORIGINAL
target items in the exact order they were presented, with no
additional text.

Example format: ["item_0", "item_1", "item_2", "item_3"]

Focus on blocking out the interference and remembering the
original targets. Return exactly what you recall.
```

Figure 10: Prompt template for the interference-based ordered recall task.

```
You are playing Battleship as a spatial-memory benchmark.

Rules:
- The board is a labeled grid (rows as letters, columns as
  numbers).
- Each tactical turn, fire exactly one coordinate shot.
- Keep a stable internal map of every previous shot
  (hit, miss, sunk, repeated).
- Use hit patterns to probe adjacent cells efficiently.
- Ignore unrelated distractor turns between tactical turns.

Response format requirement:
- On tactical turns, respond with exactly one coordinate
  token (example: B4).
- On distractor turns, answer briefly and do not fire a
  coordinate.
```

Figure 11: Prompt template for the Battleship spatial-memory benchmark.

```
You are navigating a text maze for an episodic-memory benchmark.

Rules:
- Read room descriptions carefully (exits and items).
- Valid navigation commands: NORTH, SOUTH, EAST, WEST
  (or N/S/E/W).
- You may use TAKE KEY and USE KEY when appropriate.
- Keep an internal map of visited rooms and item locations.
- Later objectives may require recalling where an item was
  seen many turns ago.

Response format guidance:
- Use concise commands (for example: NORTH, TAKE KEY,
  USE KEY, LOOK).
- If asked a source-memory probe, answer with the room
  label only (for example: B3).
```

Figure 12: Prompt template for the Text Maze episodic-memory task.

```
You are playing Mastermind (Senha), a memory + logical
deduction game.

Rules:
- A hidden code contains ordered symbols (e.g., digits).
- On each turn you submit exactly one guess with the
  configured code length.
- Feedback after each guess is:
  - exact_matches:   right symbol in the right position
  - partial_matches: right symbol in the wrong position
- Use all past guesses and feedback; never contradict
  previous clues.

Response format requirement:
- Return only a JSON array with your guess symbols.
- Example for code length 4: ["1", "3", "2", "0"]
```

Figure 13: Prompt template for the Mastermind memory and logical deduction task.

C Reproducibility

The benchmark is designed for end-to-end reproducibility. The released framework includes the episode generators, task definitions, prompt templates, execution pipeline, scoring logic, model manifest, and analysis scripts used to produce the reported tables and figures. Because all episodes are generated deterministically from fixed seeds and tracked through unique identifiers, the full evaluation can be rerun with strict episode-level alignment across models and seeds.

This release protocol is particularly important for a memory benchmark, where seemingly minor changes in prompts, generation order, or sampling policy can produce misleading variance. By fixing the evaluation environment and exposing the complete generation and analysis stack, the benchmark supports direct replication, extension, and controlled comparison with future systems.

Hardware and inference infrastructure. All experiments were conducted on NVIDIA H100 80 GB GPUs. Open-weight models were served via vLLM v0.15.0 (Kwon et al., 2023) using an OpenAI-compatible HTTP endpoint (Docker image `pytorch/pytorch:2.9.0-cuda13.0-cudnn9-devel`, `GPU_MEMORY_UTILIZATION=0.9`, `TENSOR_PARALLEL_SIZE=1`). All models were loaded in their default precision as released on the Hugging Face Hub; no post-hoc quantisation was applied.

Decoding and generation settings. All models were evaluated under *greedy* decoding (`temperature=0`, `top_p=1.0`, `do_sample=False`) with a maximum of 512 new tokens per turn. This eliminates stochastic output variance as a confound and ensures that performance differences are attributable to the model's working-memory capacity rather than sampling noise.

Software stack. The benchmark framework requires Python $\geq$ 3.11 and the following core dependencies: `vllm==0.15.0+cu130`, `torch`$\geq$`2.0`, `transformers`$\geq$`5.3.0`, `openai`$\geq$`2.29.0`, `pydantic`$\geq$`2.0`, and `lm-eval`$\geq$`0.4.11` for the downstream task evaluations reported in Section 4. Analysis and figure generation require `scipy`$\geq$`1.11`, `statsmodels`$\geq$`0.14`, `matplotlib`$\geq$`3.8`, and `seaborn`$\geq$`0.13`.

Episode generation and dataset freezing. The episode set used for all reported results was generated once with seed 42. The resulting `episodes_seed42.jsonl` file (100 episodes per task variant, 21 variants, balanced factorial sampling) was frozen and reused identically across every model run. Paired episode-level comparison is enabled by the `episode_id` field present in every result record.

Randomness control. A global seed function fixes `random`, `numpy`, and `torch` states (including `cuda.manual_seed_all`) at the start of each evaluation run. Each episode additionally carries its own seed derived from the global seed and episode index, so individual episodes can be regenerated independently for debugging.

Statistical analysis. Paired significance tests use n=5,000 permutations; confidence intervals are computed via n=2,000 stratified bootstrap resamples. Both procedures use seed 42.

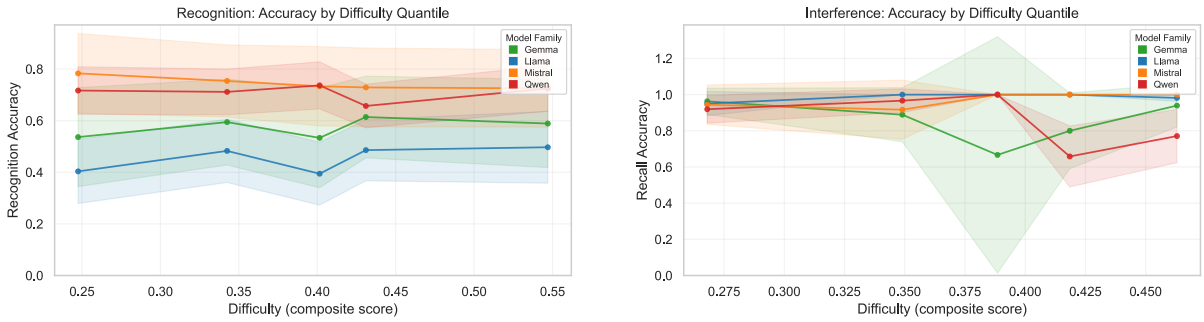
Model access. All open-weight models are publicly available on the Hugging Face Hub under their respective licences. The full model manifest, contains Hugging Face identifiers, parameter counts, context window sizes, and infrastructure tier. Frontier model APIs (GPT-4o, Gemini) were accessed via their official endpoints; API versions are logged in each result file.

Code and data release. The complete framework, frozen episode set, raw result files, and analysis scripts are released at https://anonymous.4open.science/r/gambit-benchmark.

D Additional Experiment Results

D.1 Detailed Task-Level Degradation Curves

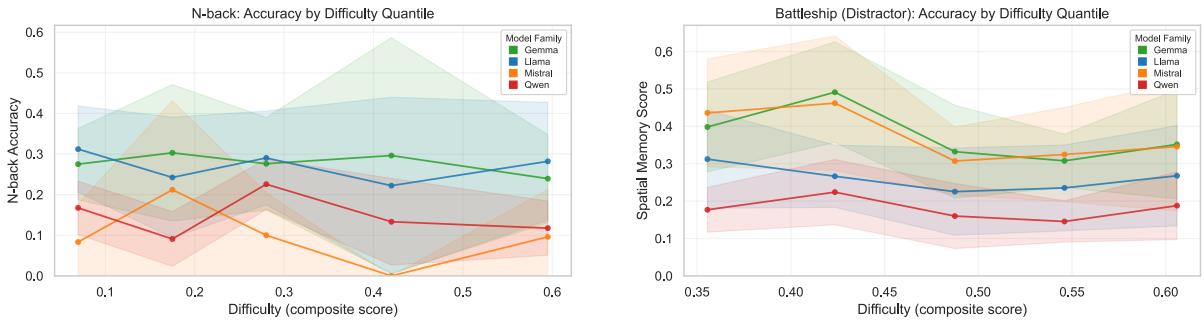
This section provides a granular breakdown of the aggregate performance degradation shown in Section 4.2. By isolating individual task variants across the composite difficulty quantiles, distinct behavioral phenomena emerge. These isolated curves mathematically demonstrate the theoretical claims made in the introduction: passive retention (e.g., recognition) remains stable under load, whereas tasks requiring active manipulation, sequential updating, or spatial tracking suffer severe degradation or catastrophic failure.



(a) Recognition (Memory Cards)

(b) Interference (Memory Cards)

Figure 14: Contrasting passive versus active retrieval. **(a)** The Recognition task exhibits remarkably flat, stable performance across all difficulty quantiles. This confirms that merely identifying whether a token was present in the context is a passive capability resilient to scaling difficulty. **(b)** Conversely, the Interference task, requiring ordered free recall under the pressure of semantically similar distractors, shows models maintaining high accuracy initially, but suffering sharp degradation at the highest difficulty quantiles as proactive interference overwhelms the attention mechanism.



(a) N-back (Online Updating)

(b) Battleship (Competitive Distractor)

Figure 15: Dynamic state tracking vulnerabilities. **(a)** The N-back curve exhibits extreme variance and noise across all models. This instability indicates that continuous state updating, rapidly overwriting the transient working memory buffer, is fundamentally unaligned with standard Transformer attention, leading to erratic performance rather than a smooth decay. **(b)** Spatial tracking under competitive distraction shows a steady, monotonic decline, confirming that as visuospatial memory load and distractor noise increase, models lose the ability to maintain a coherent 2D mental grid.

E Example Episode Transcripts

To illustrate the dynamic, multi-turn nature of the GAMBIT framework, this section provides abbreviated transcripts of the interaction between the environment (System/User) and the evaluated LLM (Model) for each of the five cognitive paradigms. These snippets demonstrate how memory is actively probed rather than passively queried.

E.1 N-back (Online Updating)

The model must continuously update its working memory buffer and emit a single classification token. This example demonstrates a 2-back configuration.

E.2 Memory Cards (Interference Variant)

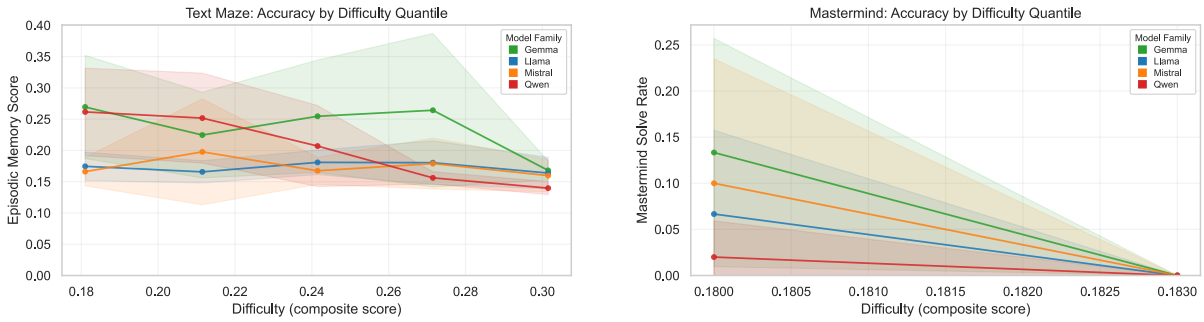
The model is forced to encode a list, process semantically competitive distractors, and reliably recall the original sequence.

E.3 Battleship (Visuospatial Tracking)

The model explores a 2D grid blindly, using spatial adjacency rules to sink a target based purely on the historical context of its past actions.

E.4 Text Maze (Episodic Routing)

The model navigates a textual environment, grounding events (finding an item) to specific temporal and spatial contexts.



(a) Text Maze (Episodic Routing)

(b) Mastermind (Iterative Hypothesis)

Figure 16: Catastrophic failures in complex cognitive routing. **(a)** Text Maze episodic memory scores hover at a low baseline (predominantly ≤ 0.3) from the outset. This demonstrates a systemic architectural inability to ground sequential textual descriptions into an actionable episodic timeline. **(b)** Mastermind yields a catastrophic collapse to near-zero solve rates across all families. Maintaining historical observations while actively deducing and updating logical rules without contradiction represents the current boundary of LLM reasoning under memory constraints.

```
[Turn 1] User: Item: A. N: 2.
[Turn 1] Model: SKIP

[Turn 2] User: Item: B. N: 2.
[Turn 2] Model: SKIP

[Turn 3] User: Item: A. N: 2.
[Turn 3] Model: MATCH

[Turn 4] User: Item: C. N: 2.
[Turn 4] Model: NO_MATCH
```

Figure 17: Example interaction trajectory for the N-back online updating task.

E.5 Mastermind (Iterative Hypothesis Updating)

The model deduces a hidden sequence. It must hold previous guesses and partial feedback in memory to avoid logical contradictions.

```
[Turn 1] User: Memorize the following list in exact order: [Eagle, Laptop, Mountain,
    Coffee]
[Turn 1] Model: Acknowledged.

[Turn 2] User: [Distractor] Please note these secondary items, they are highly rated: [
    Hawk, Desktop, Hill, Tea].
[Turn 2] Model: Acknowledged.

[Turn 3] User: Recall the ORIGINAL list in exact order.
[Turn 3] Model: ["Eagle", "Laptop", "Mountain", "Coffee"]
```

Figure 18: Example interaction trajectory for the interference-based ordered recall task.

```
[Turn 1] User: Board size is 6x6. Fire a coordinate shot.
[Turn 1] Model: C3

[Turn 2] User: Result: Miss. Fire a coordinate shot.
[Turn 2] Model: D4

[Turn 3] User: Result: Hit. Fire a coordinate shot.
[Turn 3] Model: D5

[Turn 4] User: Result: Sunk. Fire a coordinate shot.
[Turn 4] Model: A1
```

Figure 19: Example interaction trajectory for the Battleship spatial tracking task.

```
[Turn 1] User: You are in the Foyer. Exits: NORTH, EAST.
         Items: BRASS KEY.
[Turn 1] Model: TAKE BRASS KEY

[Turn 2] User: You acquired the BRASS KEY. Exits: NORTH, EAST.
[Turn 2] Model: NORTH

[Turn 3] User: You are in the Hallway. There is a locked gate.
[Turn 3] Model: USE BRASS KEY

[Turn 4] User: [Probe] In which room did you find the BRASS KEY?
[Turn 4] Model: Foyer
```

Figure 20: Example interaction trajectory for the Text Maze episodic memory task.

```
[Turn 1] User: Guess the 4-digit code (symbols 0-9).
[Turn 1] Model: ["1", "2", "3", "4"]

[Turn 2] User: Feedback: 1 exact match, 1 partial match.
         Submit your next guess.
[Turn 2] Model: ["1", "5", "6", "2"]

[Turn 3] User: Feedback: 2 exact matches, 0 partial matches.
         Submit your next guess.
[Turn 3] Model: ["1", "5", "8", "9"]
```

Figure 21: Example interaction trajectory for the Mastermind memory and logical deduction task.