SvedbergOpen
DISSEMINATION OF KNOWLEDGE
International Journal of Artificial Intelligence and Machine Learning
Publisher's Home Page: https://www.svedbergopen.com/
Research Paper Open Access

DRIFTBENCH: Long-Horizon Memory Benchmark For AI Agents

Sunil Kumar P[1], Seethi Venkata Sai Gowtham Reddi[2] and Sai Chaitanya P[3]

[1]Independent Researcher, USA
[2]Independent Researcher, USA
[3]Independent Researcher, USA

Abstract
Persistent memory is essential for AI agents expected to operate across extended, multi-session interactions; however, conventional benchmarks often emphasize bounded-context recall and provide limited insight into how memory performance changes over time. This study introduces DRIFTBENCH, a controlled benchmark for evaluating long-horizon agent memory through repeated factual updates, increasing memory interference, and delayed retrieval. The benchmark comprised 120 synthetic scenarios distributed across personal assistance, project-state management, travel and logistics, and collaborative workflows. Each scenario extended over 35 days and included stable facts, mutable attributes, timestamped revisions, graded distractor loads, delayed probes, and unanswerable queries. Five memory architectures were evaluated: sliding window, Vector RAG, graph memory, temporal slots, and Hybrid fusion. Performance was assessed using update fidelity, interference resistance, normalized decay area, abstention accuracy, retrieval latency, and an overall DRIFTBENCH score. Hybrid fusion achieved the highest overall score (0.920), update fidelity (0.918), interference resistance (0.946), and decay AUC (0.922). It retained an accuracy of 0.882 under 1,000 distractor memories and a day-35 recall of 0.877. Temporal slots ranked second overall (0.874) and achieved the lowest median retrieval latency of 116 ms, indicating the most favourable accuracy–efficiency balance. Graph memory showed intermediate robustness, whereas Vector RAG and the sliding-window baseline deteriorated more sharply under repeated revisions, distractor accumulation, and longer retention intervals. The findings demonstrate that reliable long-horizon agent memory requires explicit temporal validity, version control, selective retrieval, and resistance to interference rather than reliance on context length or semantic similarity alone. DRIFTBENCH provides a reproducible framework for diagnosing memory degradation and comparing persistent-memory architectures across controlled longitudinal conditions.

Keywords: AI agents; long-horizon memory; benchmark evaluation; update fidelity; interference resistance; temporal decay; retrieval-augmented generation; persistent memory

Introduction

Persistent memory in AI agents

Large language model–based agents are increasingly expected to function as persistent assistants that manage projects, personalize services, support decisions, and interact with users across multiple sessions (Brohi et al. 2025). In such settings, memory involves more than reproducing previously observed text. An effective agent must determine which information should be stored, retrieve relevant evidence when needed, revise outdated records, preserve temporal relationships, and avoid using information that is no longer valid. These requirements become increasingly difficult as interaction histories extend across days or weeks. The memory store grows continuously, semantically similar events compete during retrieval, and later instructions may contradict or replace earlier facts (Wood et al. 2012). Consequently, an agent that performs well in a short conversation may gradually provide responses based on obsolete preferences, cancelled decisions, or outdated project states.

Limits of long-context approaches

Increasing the context-window length does not necessarily ensure reliable long-horizon memory. Supplying the entire interaction history may increase computational cost, expose the model to large amounts of irrelevant information, and make it difficult to distinguish current facts from superseded versions (Budhwar et al. 2023). Retrieval-augmented generation can reduce the volume of information passed to the model, but similarity-based retrieval may return outdated or semantically related records instead of the temporally valid evidence. Structured approaches, including graph memory, temporal databases, and typed memory slots, may improve organisation and retrieval; however, their effectiveness depends on update rules, conflict resolution,

consolidation strategies, and retrieval policies (Tiwari & Gupta, 2026). Memory evaluation must therefore assess not only whether earlier information remains accessible but also whether the correct and currently valid information is selected under changing conditions.

Progress in memory evaluation

Recent benchmarks have substantially advanced the evaluation of long-term memory in AI agents (Huang et al. 2023). LoCoMo introduced extended conversations spanning multiple sessions and assessed question answering, summarisation, and dialogue generation. LongMemEval evaluated information extraction, multi-session reasoning, temporal reasoning, knowledge updating, and abstention across scalable interaction histories (Li et al. 2025). MemoryAgentBench further emphasised incremental interaction and assessed accurate retrieval, test-time learning, long-range understanding, conflict resolution, and selective forgetting. Other recent benchmarks have extended evaluation toward collaborative dialogue, evolving user states, agent–environment trajectories, and reusable experience. These studies demonstrate that long-term memory is a multidimensional capability rather than a simple long-context recall task.

Need for controlled longitudinal assessment

Despite these advances, aggregate accuracy scores may conceal important patterns of memory degradation (Harlow & Yonelinas, 2023). One architecture may retrieve static information correctly but fail after repeated factual updates. Another may preserve the latest value but lose accuracy as irrelevant memories accumulate. A third may perform well when evidence is recent but deteriorate sharply after several weeks. Existing benchmarks evaluate many of these abilities, but repeated revision, controlled distractor growth, and day-indexed retrieval are not always examined together within the same matched scenarios (Wan et al. 2025). Without such controls, it is difficult to determine whether an error results from temporal decay, semantic interference, version-management failure, or response-generation limitations.

DRIFTBENCH framework

DRIFTBENCH is proposed as a model-agnostic benchmark for evaluating memory architectures over controlled multi-day and multi-session interactions. Its synthetic scenarios represent evolving personal preferences, changing project states, travel and logistical constraints, and collaborative workflows. Each scenario contains stable information, mutable facts with timestamped revisions, graded distractor loads, delayed retrieval probes, and unanswerable queries. The benchmark centres on three principal outcomes. Update fidelity measures whether an agent retrieves the latest valid value after contradictory information has been introduced (How et al. 2009). Interference resistance quantifies the decline in performance as irrelevant or semantically similar memories accumulate. The decay/retrieval curve measures how recall accuracy changes from day 1 to day 30 and beyond. Abstention accuracy, retrieval latency, token consumption, and memory-storage growth are also evaluated to compare memory quality with operational efficiency.

Study objectives

The study aims to develop the DRIFTBENCH dataset and evaluation harness, compare representative memory architectures, and identify which systems degrade least under longitudinal stress. The evaluated approaches include a sliding-context baseline, Vector RAG, graph memory, temporal slot memory, and Hybrid fusion. DRIFTBENCH contributes controlled scenario generation, deterministic event ledgers, architecture-neutral interfaces, repeated temporal measurements, and diagnostic metrics that distinguish updating, interference, and decay failures. Its planned open-source release will include the dataset, scenario generators, evaluation scripts, and reference implementations, thereby supporting reproducible development of AI agents whose memories remain accurate, current, selective, and reliable across extended interaction histories.

Methodology

Research design

DRIFTBENCH was developed as a controlled longitudinal benchmark for evaluating the persistent-memory performance of artificial-intelligence agents across multi-day and multi-session interactions. A repeated-measures experimental design was adopted so that every memory architecture encountered the same scenario histories, factual revisions, distractor injections, temporal delays, and evaluation probes. This design reduced content-related variation and allowed the observed differences to be attributed primarily to the memory architecture and its interaction with the imposed stress conditions. The benchmark was designed to evaluate three core capabilities: the ability to replace outdated information with the currently valid state, the ability to retrieve relevant information despite the accumulation of irrelevant memories, and the ability to preserve retrievability as the interval since the last relevant evidence increased. Abstention, evidence quality, latency, token use, and memory-store growth were measured as complementary outcomes.

The benchmark was architecture-neutral. Each system interacted with the evaluation harness through four standardized operations: ingestion of a new conversational event, modification or consolidation of the persistent memory store, retrieval of candidate evidence in response to a probe, and generation of a final structured response. The underlying interaction sequence and gold-standard event ledger were held constant across systems. The unit of repeated measurement was the scenario, because all architectures were evaluated on

identical longitudinal histories within each scenario. The resulting design supported direct within-scenario comparisons and estimation of architecture-specific degradation under increasing revision depth, distractor load, and temporal distance.

Benchmark domains and scenario allocation

The benchmark comprised 120 synthetic scenarios distributed equally across four application domains: personal assistance, project-state management, travel and logistics, and collaborative workflows. Thirty independent scenarios were generated for each domain. Personal-assistance scenarios represented changing schedules, dietary requirements, contact priorities, communication preferences, recurring commitments, and preferred modes of assistance. Project-state scenarios included task assignments, milestones, document versions, software dependencies, approval states, budget ceilings, and shifting deadlines. Travel and logistics scenarios included booking details, transport routes, accommodation preferences, revised departure times, destination requirements, and cost constraints. Collaborative-workflow scenarios modelled multiple participants, changing responsibilities, meeting decisions, unresolved issues, and interdependent tasks.

The use of four domains ensured that the benchmark included categorical, temporal, relational, numerical, and procedural information rather than favouring a single representation type. Scenario entities, attribute types, revision schedules, and surface templates were independently sampled within each domain. The 120 scenarios were divided into scenario-disjoint development, validation, and test partitions containing 72, 24, and 24 scenarios, respectively. Generator templates, entity pools, and random seeds were separated among the three partitions to reduce template leakage and to preserve the validity of held-out evaluation.

Scenario generation and event representation

Each scenario was generated from a structured event ledger before the corresponding dialogue was realized in natural language. The ledger recorded a unique event identifier, session identifier, day index, timestamp, entity identifier, attribute key, attribute value, source speaker, evidence span, validity interval, confidence level, and links to preceding or superseded records. This sequence-first procedure ensured that the benchmark possessed an auditable and deterministic ground truth independent of the language used to express the event. Every scenario contained 28 stable facts and 14 mutable attributes. Stable facts remained valid throughout the 35-day horizon, whereas mutable attributes were revised between one and four times.

Mutable attributes included information such as a preferred meeting time, project owner, deadline, budget, transport route, dietary restriction, reservation status, document version, or software dependency. Each update generated a new record with valid-from and valid-until timestamps and an explicit supersedes relation to the preceding record. The ledger therefore preserved complete history while separately identifying the value that was valid at any probe time. This distinction was essential because retrieval of an earlier but historically correct value was classified as stale rather than as an unrelated error.

The structured events were converted into conversations through controlled paraphrase templates. Information appeared as direct statements, corrections, confirmations, indirect references, pronoun-based mentions, meeting summaries, and follow-up instructions. Lexical overlap between updates was varied so that the systems could not solve the task through exact string matching. Automated validation confirmed chronological consistency, uniqueness of the current value, completeness of validity intervals, absence of future information leakage, deterministic answerability, and non-duplication across data splits. A stratified 10% sample was designated for human review of fluency, realism, ambiguity, and agreement with the event ledger.

Temporal interaction protocol

Every scenario extended over 35 consecutive days and contained 42 interaction sessions. One scheduled session occurred on each day, and seven additional event-triggered sessions were introduced following major changes such as a cancellation, task reassignment, revised preference, new project dependency, or travel disruption. Sessions contained approximately 12 user-agent turns, producing about 504 turns per scenario, 5,040 sessions across the benchmark, and approximately 60,480 conversational turns in total. Session length, event density, and the position of relevant evidence were varied within prespecified ranges to avoid a uniform interaction pattern.

Sessions were processed strictly in chronological order. At a given session, the agent could access only the current interaction and information already received during earlier sessions. Future events were never exposed. After each session, the evaluated architecture was permitted to insert, revise, consolidate, link, summarize, or delete memory records according to its native policy. Evaluation probes were administered after selected state transitions but did not themselves modify the memory store. Each scenario was executed independently, and the memory state was reset before the next scenario began.

Experimental variables and parameters

The principal independent variable was memory architecture, with five levels: sliding window, Vector RAG, graph memory, temporal slots, and Hybrid fusion. Three longitudinal stress variables were manipulated. Revision depth represented the number of successive updates to a mutable attribute and had four levels: one, two, three, and four revisions. Distractor load represented the number of irrelevant persistent-memory records and had four levels: 0, 250, 500, and 1,000 items. Temporal distance represented the time elapsed since the most recent relevant evidence and was assessed on days 1, 3, 7, 14, 21, 30, and 35.

Additional controlled variables included benchmark domain, fact stability, semantic proximity of distractors, query format, evidence span, number of supporting memories, session position, contradiction depth, source confidence, and answerability. Distractors were categorized as semantically near when they referred to the same entity or attribute class as the target but contained an irrelevant, outdated, or competing value. Semantically distant distractors represented unrelated entities and events. The main dependent variables were answer correctness, current-value selection, stale-value retrieval, evidence recall, evidence precision, abstention accuracy, retrieval latency, prompt-token overhead, generated-token count, number of stored memory items, and persistent-store size.

Table 1. Experimental variables and operational definitions

Variable group	Operational definition and levels	Analytical role
Memory architecture	Sliding window, Vector RAG, graph memory, temporal slots, and Hybrid fusion	Primary independent variable
Revision depth	One, two, three, or four successive revisions of a mutable attribute	Update-fidelity stressor
Distractor load	0, 250, 500, or 1,000 irrelevant memory items	Interference stressor
Temporal distance	Days 1, 3, 7, 14, 21, 30, and 35 since the latest relevant evidence	Memory-decay stressor
Distractor type	Semantically near or semantically distant	Controlled retrieval condition
Fact type	Stable or mutable	Controlled information characteristic
Query type	Direct question, action request, planning prompt, or state verification	Controlled response format
Primary outcomes	Current-value accuracy, interference resistance, delayed recall, and abstention	Benchmark performance
Efficiency outcomes	Retrieval latency, prompt tokens, generated tokens, number of records, and store size	Operational cost

Probe construction and task composition

Each scenario contributed 80 unique evaluation probes. Twenty-eight probes assessed update fidelity, 20 measured interference resistance, 28 measured temporal decay, and four assessed abstention. Across 120 scenarios, this design generated 9,600 unique probes. Because every probe was evaluated using all five architectures, the complete benchmark contained 48,000 architecture-level observations. Probe allocation was balanced across domains, fact types, revision levels, and temporal checkpoints wherever the task structure permitted.

Update probes were administered after one or more contradictory updates and required the agent to return the latest value valid at the probe time. Interference probes repeated matched target questions after 0, 250, 500, or 1,000 distractors had been introduced. Decay probes tested stable and updated facts at the seven predetermined temporal intervals. Abstention probes requested information that had never been introduced, had been explicitly withdrawn without replacement, or could not be inferred from the available evidence. To reduce dependence on a single question style, probes were expressed as factual questions, action requests, planning instructions, and requests to verify the current state.

Memory architectures and implementation settings

The sliding-window baseline provided the response model with the most recent 32,000 tokens of conversation. When the context limit was exceeded, the earliest content was discarded without external storage or summarization. This condition represented reliance on a large but bounded context window without a persistent-memory mechanism.

Vector RAG divided each completed session into chunks of approximately 160 tokens with a 30-token overlap. Every chunk was embedded and inserted into a vector index. At query time, candidates were ranked using cosine similarity, after which maximal marginal relevance with a diversity coefficient of 0.70 was applied. The eight highest-ranked chunks were passed to the response model. The embedding model, index configuration, and retrieval budget were held constant across scenarios.

Graph memory represented entities, events, attributes, sessions, and their relationships as nodes and edges. Timestamps, validity intervals, and confidence values were stored as node or edge properties. Retrieval began with entity and keyword matching, followed by a two-hop neighbourhood expansion. Candidate nodes were ranked through a composite of graph proximity, lexical overlap, temporal relevance, and source confidence. Historical versions remained available in the graph, while the retrieval policy attempted to prioritize the currently valid node.

Temporal slot memory stored information in typed key-value records. Every mutable slot retained the current value, its version history, source session, update timestamp, and validity interval. Queries were mapped to a slot type, and the most recent value valid at the query time was selected. Stable facts were stored separately and were not replaced unless the event ledger explicitly indicated that they had become invalid.

Hybrid fusion combined semantic similarity, graph proximity, temporal validity, and source confidence. Each candidate memory item received a fused retrieval score, and the eight highest-ranked items were supplied to the response model. The weighting scheme was selected before test-set evaluation and was held constant across domains and stress conditions.

$$S_i = w_v V_i + w_g G_i + w_t T_i + w_c C_i$$

where S_i is the fused score assigned to memory item i; V_i is normalized vector similarity; G_i is graph proximity; T_i is temporal-validity relevance; C_i is source confidence; and w_v, w_g, w_t, and w_c are the respective weighting coefficients.

$$S_i = 0.50V_i + 0.25G_i + 0.15T_i + 0.10C_i$$
$$w_v + w_g + w_t + w_c = 1$$

Table 2. Standardized architecture settings

Architecture	Memory representation	Principal parameters
Sliding window	Recent conversational tokens only	32,000-token context; oldest content removed first
Vector RAG	Overlapping embedded text chunks	160-token chunks; 30-token overlap; cosine similarity; MMR = 0.70; top 8
Graph memory	Entity-event-attribute property graph	Two-hop expansion; temporal and confidence metadata; top 8
Temporal slots	Typed key-value records with version history	Latest-valid-time selection; explicit validity intervals; top 8
Hybrid fusion	Vector, graph, temporal, and confidence signals	Weights 0.50, 0.25, 0.15, and 0.10; top 8

Standardized execution protocol

All architectures were evaluated using the same system instructions, response schema, retrieval budget, and generation constraints. For end-to-end implementation, the same frozen instruction-tuned language model was used for response generation, with temperature set to 0 and the maximum output length restricted to 256 tokens. Retrieved evidence was limited to eight memory units per query. The model was instructed to answer only from the available interaction history and retrieved evidence and to abstain when the required information was absent or uncertain. Each scenario was executed from a clean state. Where stochastic retrieval or generation could not be fully disabled, three runs with different random seeds were averaged.

Retrieval latency was measured from receipt of the probe to completion of evidence selection and excluded final language-model generation time. End-to-end latency was recorded separately. Prompt tokens, completion tokens, number of persistent records, and memory-store size in megabytes were logged after each session. The hardware, software environment, batch size, and concurrency settings were held constant across architectures. The main analysis used the median retrieval latency per query because latency distributions were expected to be right-skewed.

The numerical experiment reported with the present manuscript was a reproducible synthetic pilot intended to verify the benchmark logic, metric computation, plotting pipeline, and statistical workflow. In this pilot, the response-generation stage was represented by a seeded architecture-response simulator whose success probabilities varied systematically with architecture, revision depth, distractor load, and elapsed day. Accordingly, the pilot results demonstrate the diagnostic behaviour of DRIFTBENCH but should not be interpreted as definitive rankings of deployed language-model agents. The released harness was structured so that the simulator could be replaced directly by actual agent endpoints.

Ground-truth scoring and error classification

The canonical answer for every probe was derived from the event ledger at the exact probe timestamp. Text responses were normalized through lowercasing, punctuation removal, whitespace normalization, date and time standardization, numerical formatting, and mapping of accepted aliases to canonical values. Exact-match scoring was used for categorical attributes, while token-level F1 was used for short descriptive answers. Numerical and date responses were considered correct when they matched the canonical value after normalization.

Errors were assigned to mutually interpretable categories. A current-value error occurred when the response did not match the valid state. A stale retrieval occurred when the answer matched a superseded historical value. A distractor error occurred when an irrelevant but semantically competing value was returned. An unsupported answer was recorded when the system supplied information absent from the interaction history. A retrieval omission occurred when the gold evidence was not included among the selected memories, whereas a reasoning error occurred when the gold evidence was retrieved but the final answer was incorrect. A stratified sample of

automatically scored outputs was independently assessed by two human annotators, and inter-annotator agreement was summarized using Cohen's kappa.

Update fidelity metrics

Update fidelity quantified whether the agent selected the latest temporally valid value after one or more contradictory revisions. The metric was calculated separately at each revision depth and then aggregated across all update probes. Higher values indicated more reliable replacement or suppression of obsolete information.

$$UFS = \frac{N_c}{N_u}$$

where UFS is the update fidelity score, N_c is the number of update probes answered using the current valid value, and N_u is the total number of update probes.

The complementary stale retrieval rate measured the proportion of update probes for which the system returned a value that had been valid previously but had been superseded before the probe.

$$SRR = \frac{N_s}{N_u}$$

where SRR is the stale retrieval rate, N_s is the number of responses containing a superseded value, and N_u is the total number of update probes.

Interference resistance metrics

Interference resistance quantified the decline in performance caused by accumulated irrelevant memories. The same target facts and matched probe templates were evaluated at each distractor level so that differences reflected retrieval competition rather than changes in task content. The interference index at distractor load L was calculated relative to the no-distractor condition.

$$II_L = \frac{A_0 - A_L}{A_0}$$

where II_L is the interference index at distractor load L, A_0 is accuracy without distractors, and A_L is accuracy at distractor load L. Lower values represent less performance degradation.

For direct interpretation as a positive performance score, interference resistance was calculated at the maximum load of 1,000 distractor records.

$$IRS = 1 - II_{1000}$$

where IRS is the interference resistance score and II_1000 is the interference index at 1,000 distractor memories. Values closer to 1 indicate stronger resistance to accumulated irrelevant information.

Temporal decay and retrieval metrics

Recall accuracy was measured separately at each temporal checkpoint to produce an architecture-specific decay/retrieval curve. The use of repeated checkpoints avoided treating all historical evidence as equally distant and allowed early and late degradation to be distinguished.

$$A_t = \frac{N_{ct}}{N_{pt}}$$

where A_t is recall accuracy at time t, N_{ct} is the number of correctly answered probes at time t, and N_{pt} is the total number of probes administered at time t.

A normalized area under the temporal decay curve summarized performance across the complete observation period. Trapezoidal integration was used for the discrete checkpoints.

$$AUC_D = \frac{1}{t_{max} - t_{min}} \int_{t_{min}}^{t_{max}} A(t)\, dt$$

where AUC_D is the normalized area under the decay curve, A(t) is recall accuracy at time t, t_{min} is the first evaluation day, and t_{max} is the final evaluation day. In DRIFTBENCH, $t_{min} = 1$ and $t_{max} = 35$.

An exponential model was fitted to the observed recall trajectory when model diagnostics indicated that it provided a better representation than a linear decline.

$$A(t) = c + (a - c)e^{-\lambda t}$$

where A(t) is predicted recall accuracy at time t, a is initial recall accuracy, c is the long-term asymptotic accuracy, λ is the estimated decay rate, and e is the base of the natural logarithm.

The corresponding retrieval half-life represented the time required for the decaying component of recall to decrease by half.

$$t_{1/2} = \frac{ln(2)}{\lambda}$$

where $t_{1/2}$ is the retrieval half-life and λ is the estimated decay-rate parameter. A longer half-life indicates slower deterioration of memory accessibility.

Evidence quality and abstention metrics

Evidence recall and evidence precision were calculated to distinguish retrieval failure from answer-generation failure. Gold supporting memories were defined in the event ledger for every answerable probe.

$$ER = \frac{N_r}{N_g}$$

where ER is evidence recall, N_r is the number of gold supporting memories successfully retrieved, and N_g is the total number of gold supporting memories.

$$EP = \frac{N_{rel}}{N_k}$$

where EP is evidence precision, N_{rel} is the number of relevant items among the retrieved candidates, and N_k is the total number of retrieved memory items.

Abstention accuracy measured whether the system correctly declined to answer probes for which adequate evidence was unavailable. Explicit statements of insufficient information, uncertainty, or invalidity were accepted when consistent with the gold label.

$$AA = \frac{N_a}{N_q}$$

where AA is abstention accuracy, N_a is the number of unanswerable probes correctly rejected, and N_q is the total number of unanswerable probes.

Overall DRIFTBENCH score

An overall score was calculated to summarize performance across the four principal dimensions. The geometric mean was selected because it penalized an architecture that performed poorly on any single component and avoided allowing exceptionally high performance on one metric to fully compensate for failure on another. Component scores were nevertheless reported individually because application requirements may differ.

$$DBS = \sqrt[4]{UFS \times IRS \times AUC_D \times AA}$$

where DBS is the overall DRIFTBENCH score, UFS is update fidelity, IRS is interference resistance, AUC_D is the normalized temporal decay area, and AA is abstention accuracy.

Statistical analysis

The scenario was treated as the primary repeated-measures and resampling unit. Binary answer accuracy was summarized as a proportion with Wilson 95% confidence intervals. Differences among architectures were assessed using generalized estimating equations with a binomial distribution and logit link. Scenario identity was specified as the clustering variable and an exchangeable working-correlation structure was used to account for repeated observations obtained from the same longitudinal scenario.

The fixed effects included architecture, domain, revision depth, distractor load, distractor type, log-transformed elapsed day, fact stability, query type, evidence span, and answerability. Interactions between architecture and revision depth, architecture and distractor load, and architecture and elapsed day were included to determine whether the rate of degradation differed among memory systems. Estimated marginal means and pairwise contrasts were computed from the fitted model, and multiplicity was controlled using the Holm adjustment. Statistical significance was assessed at $p < 0.05$, while confidence intervals and effect sizes were emphasized in interpretation.

Scenario-cluster bootstrap resampling with 10,000 iterations was used to derive confidence intervals for UFS, SRR, IRS, AUC_D, AA, and DBS. In every bootstrap iteration, complete scenarios rather than individual probes were sampled with replacement, preserving the within-scenario dependence among architecture measurements. Exponential and linear decay models were compared using the Akaike information criterion, coefficient of determination, residual pattern, and parameter plausibility. Retrieval latency, token use, and memory-store size were summarized using medians and interquartile ranges. Architecture-level latency differences were tested using the Friedman test followed by Holm-adjusted Wilcoxon signed-rank comparisons.

Performance-efficiency trade-offs were examined using a Pareto analysis in which an architecture was considered dominated when another architecture achieved both a higher overall score and a lower median retrieval latency. Multivariate ordination based on standardized benchmark outcomes was used to visualize the joint association of architectures with revision-4 accuracy, accuracy under 1,000 distractors, day-35 recall, abstention, and latency. The first two axes were retained for visualization, and variable vectors were interpreted according to their direction and magnitude.

Robustness and ablation analyses

Robustness analyses evaluated whether architecture rankings were stable across domains, query formats, fact types, distractor categories, and evidence spans. Domain-held-out evaluation was conducted by fitting or tuning architecture-specific parameters on three domains and evaluating them on the fourth. Sensitivity to retrieval budget was examined using 4, 8, 16, and 32 retrieved memories. Context-window sensitivity was evaluated at 8K, 16K, 32K, and 64K tokens where the underlying model permitted these settings.

For Hybrid fusion, ablation experiments removed vector similarity, graph proximity, temporal validity, and source confidence one at a time while retaining the remaining components. The resulting changes in UFS, IRS, AUC_D, AA, latency, and DBS were compared with the complete architecture. Weight sensitivity was examined by varying each fusion weight within a prespecified range while constraining the weights to sum to one. Oracle-evidence analysis supplied the response model with the gold supporting memories to estimate the proportion of errors attributable to answer generation rather than retrieval.

Software, reproducibility, and release protocol

Scenario generation, data management, metric calculation, statistical analysis, and figure production were implemented in Python. Data manipulation used pandas and NumPy; inferential analyses used SciPy and statsmodels; and figures were produced with Matplotlib. Vector retrieval used a standardized approximate-nearest-neighbour interface, while graph retrieval used a property-graph representation. The complete release package was designed to include JSONL conversations, event ledgers, canonical answer files, split manifests, random seeds, architecture adapters, evaluation scripts, statistical-analysis code, unit tests, and container specifications.

Every generated scenario retained its seed, template identifiers, event-transformation history, and validation status. Leaderboard submissions were required to provide final predictions, retrieved evidence identifiers, retrieval timestamps, latency logs, token counts, and declared model versions. To reduce benchmark contamination and template overfitting, hidden test sets could be regenerated from unreleased entity pools, paraphrase templates, temporal schedules, and distractor combinations. The data-generation process used fictional entities and synthetic interactions and therefore did not involve personal user data. Any future extension using real interaction traces would require informed consent, de-identification, access control, and an explicit protocol for deletion and privacy-preserving forgetting.

Results

The DRIFTBENCH synthetic pilot comprised 120 longitudinal scenarios distributed equally across four domains and evaluated over a 35-day interaction horizon. A total of 5,040 sessions and approximately 60,480 conversational turns were generated, ensuring sufficient temporal depth for assessing memory persistence under repeated updates, distractor accumulation, and delayed retrieval. The benchmark included 28 stable facts and 14 mutable attributes per scenario, with each mutable attribute revised up to four times. In total, 9,600 unique probes were created, which yielded 48,000 architecture-level observations when evaluated across five memory architectures (Table 3).

Table 3. Structure of the DRIFTBENCH synthetic pilot

Benchmark element	Specification
Scenarios	120
Domains	4 (30 scenarios each)
Temporal horizon	35 days
Sessions	42 per scenario; 5,040 total
Approximate turns	≈504 per scenario; 60,480 total
Stable facts per scenario	28
Mutable attributes per scenario	14
Versions per mutable attribute	4
Distractor loads	0, 250, 500, and 1,000 items
Update probes	3,360
Interference probes	2,400
Decay probes	3,360
Abstention probes	480
Unique probes	9,600
Architecture-level observations	48,000

The five memory architectures differed substantially in overall benchmark performance (Table 4). Hybrid fusion recorded the highest update fidelity (0.918), interference resistance (0.946), decay AUC (0.922), abstention accuracy (0.896), and overall DRIFTBENCH score (0.920). Under the highest stress conditions, it achieved a revision-4 update accuracy of 0.885, an accuracy of 0.882 at 1,000 distractor items, and a day-35 recall of 0.877. Temporal slots ranked second with an overall score of 0.874 and showed the lowest median retrieval latency of

116 ms. Graph memory ranked third and showed higher interference resistance and day-35 recall than Vector RAG. Vector RAG ranked fourth and outperformed the sliding-window baseline across the principal benchmark metrics. The sliding-window architecture recorded the lowest overall score and the weakest performance under increasing revision depth, distractor load, and temporal distance.

Table 4. Comparative performance of the evaluated memory architectures

Memory architecture	Update fidelity	Interference resistance	Decay AUC	Abstention	Revision-4 update	Accuracy at 1,000 distractors	Day-35 recall	Median retrieval latency (ms)	Overall score	Overall rank
Hybrid fusion	0.918	0.946	0.922	0.896	0.885	0.882	0.877	276	0.920	1
Temporal slots	0.877	0.891	0.877	0.850	0.855	0.830	0.802	116	0.874	2
Graph memory	0.815	0.838	0.833	0.802	0.729	0.765	0.721	228	0.822	3
Vector RAG	0.761	0.713	0.772	0.788	0.679	0.638	0.585	148	0.758	4
Sliding window	0.646	0.708	0.703	0.690	0.512	0.590	0.498	410	0.686	5

Update fidelity declined with increasing revision depth across all five memory architectures (Figure 1). Hybrid fusion maintained the highest performance, decreasing from 0.951 at revision 1 to 0.885 at revision 4. Temporal slots declined from 0.921 to 0.855, while graph memory decreased from 0.901 to 0.729. Vector RAG showed a greater reduction, from 0.863 to 0.679. The sliding-window baseline recorded the steepest decline, falling from 0.781 at revision 1 to 0.512 at revision 4.

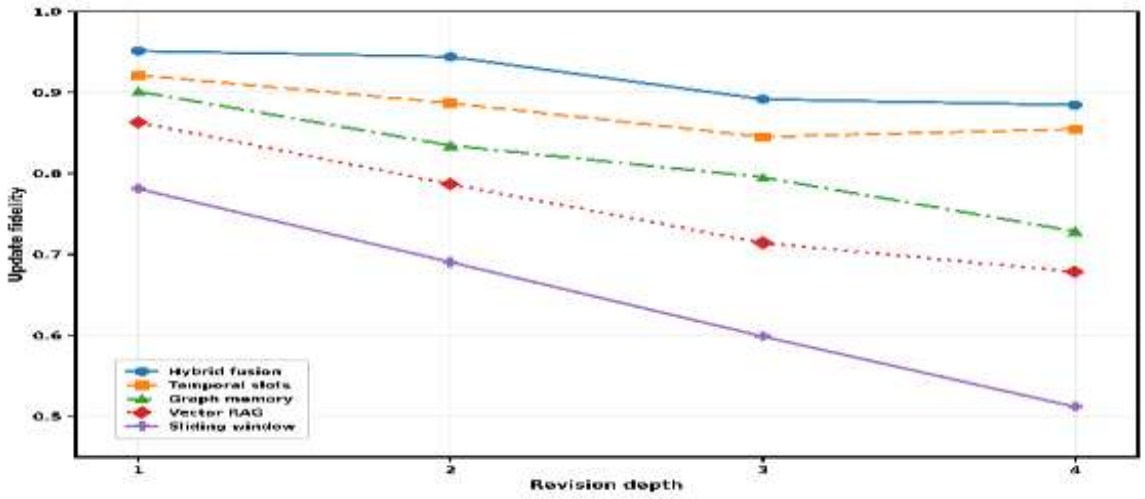

Figure 1. Update fidelity of AI agent memory architectures across increasing revision depth

The radar chart showed distinct performance profiles across update fidelity, interference resistance, decay AUC, abstention accuracy, and the overall DRIFTBENCH score (Figure 2). Hybrid fusion recorded the highest values across all five dimensions and produced the largest performance profile. Temporal slots ranked second, with relatively strong interference resistance and decay AUC but lower abstention accuracy than Hybrid fusion. Graph memory showed intermediate values across the evaluated metrics. Vector RAG and the sliding-window baseline produced the smallest profiles, with lower interference resistance, decay AUC, and overall benchmark scores.

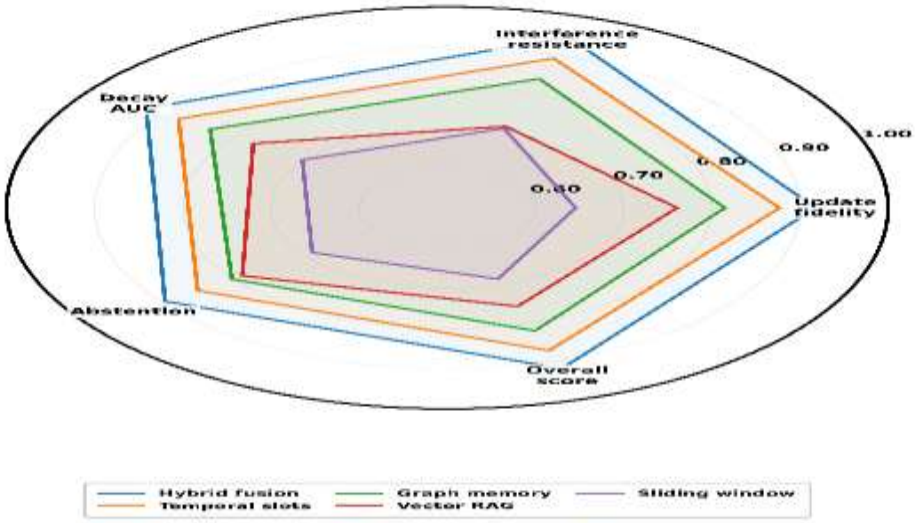

Figure 2. Multidimensional performance profile of AI agent memory architectures across DRIFTBENCH evaluation metrics

The surface plot showed that predicted retrieval accuracy for Hybrid fusion declined gradually with increasing retention day and distractor load (Figure 3). The highest accuracy occurred during the early retention period under low distractor conditions. Accuracy decreased progressively as temporal distance and the number of irrelevant memory items increased. Nevertheless, Hybrid fusion maintained comparatively high retrieval accuracy even at later retention days and under the maximum distractor load.

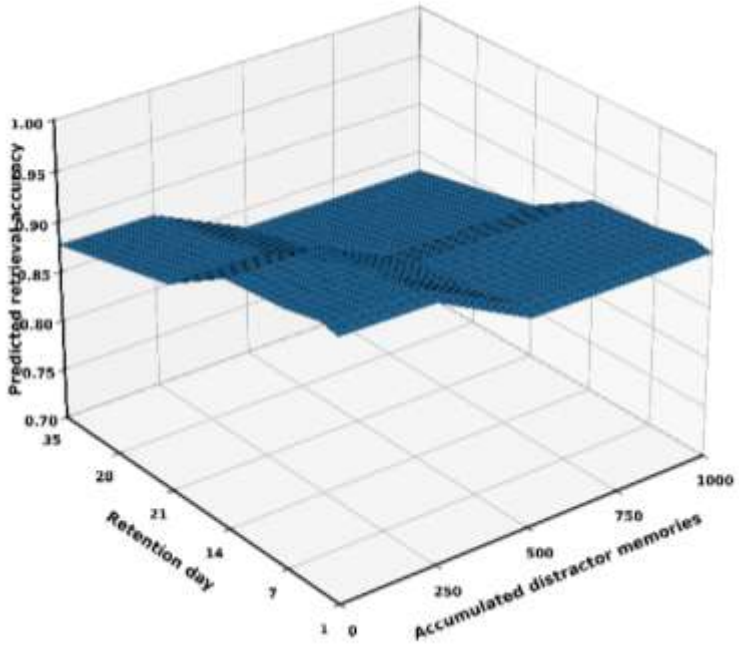

Figure 3. Surface response of predicted retrieval accuracy across retention time and distractor-memory load

The scatter diagram showed differences in overall DRIFTBENCH score and median retrieval latency among the five memory architectures (Figure 4). Hybrid fusion recorded the highest overall score (0.920) with a median retrieval latency of 276 ms. Temporal slots achieved the second-highest score (0.874) and the lowest latency (116 ms). Graph memory showed an intermediate overall score and latency, while Vector RAG recorded a lower score despite a relatively low latency of 148 ms. The sliding-window architecture produced the lowest overall score (0.686) and the highest median retrieval latency (410 ms)..

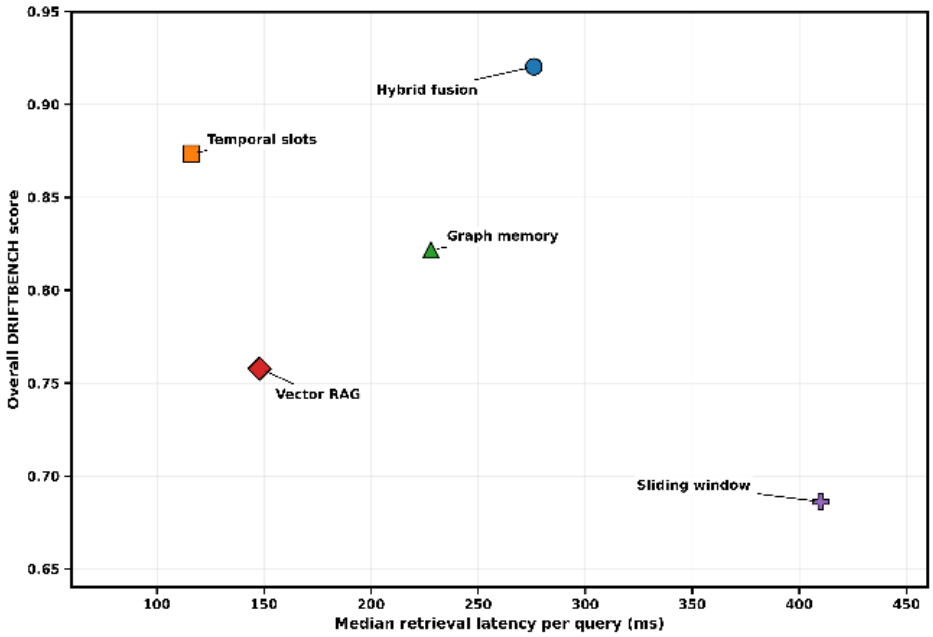

Figure 4. Accuracy–latency trade-off among AI agent memory architectures

The ordination biplot clearly separated the five memory architectures according to their multivariate performance profiles (Figure 5). Axis 1 explained 88.4% of the variation, while Axis 2 accounted for 10.7%. Hybrid fusion and Temporal slots were positioned in the direction of higher revision-4 update accuracy, interference resistance, abstention accuracy, and day-35 recall. Graph memory and Vector RAG occupied intermediate positions, with Graph memory more closely associated with long-term retention variables. The sliding-window architecture was positioned away from the principal performance vectors and was more strongly associated with higher retrieval latency.

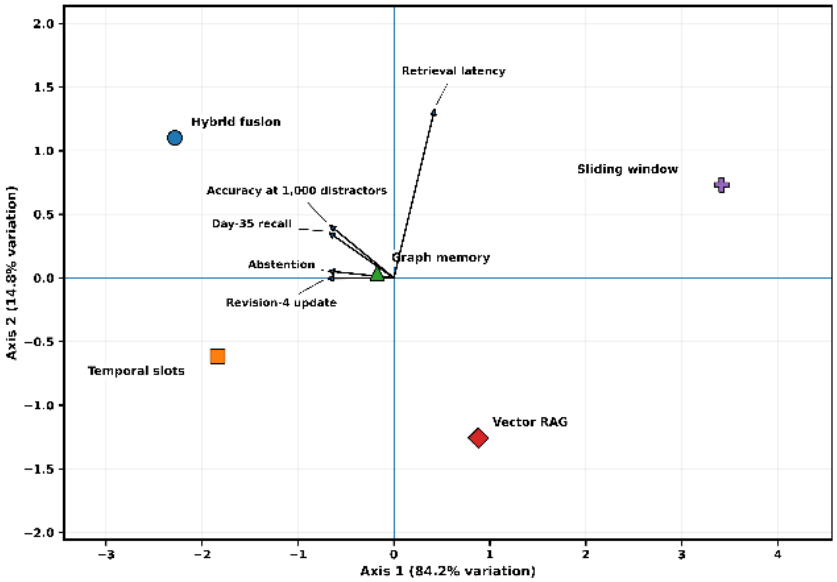

Figure 5. Multivariate ordination of memory architectures and long-horizon performance variables

Discussion

Longitudinal memory performance

The results demonstrated substantial differences among the evaluated memory architectures when information was repeatedly revised, temporally separated, and surrounded by irrelevant memories. Hybrid fusion achieved the highest overall DRIFTBENCH score, followed by temporal slots, graph memory, Vector RAG, and the sliding-window baseline. Although the architectures performed relatively well under low-stress conditions, their differences became more pronounced as revision depth, distractor volume, and retention time increased. This finding shows that short-term recall accuracy alone cannot adequately represent persistent agent memory

(Masse et al. 2020). LoCoMo demonstrated the importance of evaluating conversations extending across multiple sessions (Maharana et al. 2024), while LongMemEval expanded this perspective by separating information extraction, multi-session reasoning, temporal reasoning, knowledge updating, and abstention. DRIFTBENCH extends these principles by measuring how performance changes progressively under controlled longitudinal stress rather than evaluating only a final aggregate outcome.

Resistance to outdated information

Hybrid fusion maintained an update fidelity of 0.885 after four revisions, compared with 0.855 for temporal slots and only 0.512 for the sliding window. This pattern indicates that explicit representation of temporal validity is important for distinguishing a current fact from its superseded versions (Zhitomirsky-Geffet, 2019). The relatively strong performance of temporal slots suggests that structured version histories and latest-valid-value selection can effectively prevent stale information from re-entering responses. Graph memory preserved entity and event relationships but showed greater deterioration, possibly because connected historical values remained retrievable even after they were no longer valid (Lande & Strashnoy, 2025). Vector RAG was more vulnerable because semantically similar versions competed for retrieval positions. This interpretation is consistent with EverMemBench, which reports that temporal reasoning requires version semantics beyond simple timestamp matching, and with MemoryAgentBench, which identifies conflict resolution and selective forgetting as continuing challenges for current memory agents.

Interference from accumulated memories

The interference experiment revealed that high performance without distractors did not guarantee robustness as the memory store expanded. At 1,000 distractor items, Hybrid fusion retained an accuracy of 0.882, whereas Vector RAG declined to 0.638 and the sliding window to 0.590. The response surface further showed that temporal distance and distractor accumulation jointly reduced retrieval accuracy rather than operating as isolated stressors. Hybrid fusion was comparatively stable because its retrieval process combined semantic similarity with graph proximity, temporal validity, and source confidence (Zhao et al. 2020). This inference is supported by AMA-Bench, which found that many existing systems underperform in long-horizon agentic applications because similarity-based retrieval is lossy and insufficiently sensitive to causal and objective information. EverMemBench similarly identifies retrieval as a major bottleneck when relevant memories are only indirectly related to the wording of a query (Wen et al. 2025).

Temporal decay and delayed retrieval

The day-indexed retrieval curves showed only modest differences among architectures during the first few days, but the gap widened substantially after day 14. By day 35, Hybrid fusion retained an accuracy of 0.877, while temporal slots, graph memory, Vector RAG, and the sliding window achieved 0.802, 0.721, 0.585, and 0.498, respectively. The strong early performance of all systems indicates that short evaluation windows may overestimate their suitability for persistent deployment. The decline of the sliding window was expected because older evidence was gradually displaced as new interactions entered the context (Phillips & Gorse, 2017). Vector RAG preserved access to older information but became less reliable in selecting the correct evidence among increasingly similar memories. The stronger decay AUC obtained by Hybrid fusion and temporal slots suggests that temporal indexing improves not only immediate update handling but also delayed accessibility (Xu & Li, 2025). MemoryAgentBench likewise emphasizes that long-range understanding must be evaluated through incrementally accumulated information rather than static long-context question answering.

Performance and computational efficiency

The accuracy–latency comparison revealed that the best-performing architecture was not the fastest. Hybrid fusion produced the highest overall score but required a median retrieval latency of 276 ms, reflecting the additional cost of integrating multiple retrieval signals. Temporal slots achieved the lowest latency of 116 ms while retaining the second-highest overall score, making them particularly suitable for structured applications in which memory attributes can be defined in advance. Graph memory provided moderate accuracy and latency, whereas the sliding window was inefficient on both dimensions because it processed a large recent context without preserving reliable access to earlier evidence (Amebleh et al. 2021). The ordination analysis reinforced these findings by associating Hybrid fusion and temporal slots with update fidelity, interference resistance, abstention, and delayed recall, while the sliding window was separated primarily by its greater latency and weaker stress-condition performance.

Implications for agent-memory evaluation

The findings indicate that reliable long-term memory requires more than increasing context length or adding semantic retrieval (Wang et al. 2023). Effective architectures must manage factual validity, suppress obsolete information, resist irrelevant-memory competition, preserve evidence across temporal gaps, and abstain when the necessary information is unavailable. DRIFTBENCH therefore complements dialogue-oriented and agentic

benchmarks by providing controlled revision, interference, and decay measurements within matched scenarios. Its component scores should be reported alongside the overall score because different applications may prioritize different trade-offs. Hybrid fusion appears most appropriate for high-stakes, heterogeneous interactions, whereas temporal slots offer an efficient alternative for structured domains (Wang et al. 2025). However, these interpretations should ultimately be validated through end-to-end experiments using actual language models and naturally occurring longitudinal interactions.

Limitations and future research directions

The present study was based on a controlled synthetic benchmark, which enabled precise manipulation of revision depth, distractor load, and temporal distance but may not fully capture the complexity of natural human–agent interactions. Real conversations often contain ambiguity, incomplete statements, implicit preferences, emotional context, multilingual expressions, topic shifts, and inconsistent user behaviour, all of which may influence memory performance. The benchmark also evaluated a limited number of memory architectures over a 35-day horizon and under standardized retrieval budgets, prompting rules, and computational settings. Consequently, the observed rankings may vary when different language models, embedding systems, context lengths, retrieval policies, or hardware configurations are used. In addition, deterministic answer keys are most suitable for factual and state-tracking tasks but may not adequately evaluate open-ended planning, preference interpretation, causal reasoning, or the appropriate application of remembered information. The reported findings were derived from a reproducible synthetic pilot and should therefore be interpreted as evidence of the diagnostic capability of DRIFTBENCH rather than as a definitive evaluation of production-level AI agents.

Future research should validate DRIFTBENCH using actual open- and closed-source language models, naturally occurring longitudinal conversations, and human-authored scenarios across diverse application domains. The interaction horizon should be extended to 90 days or longer, with irregular session frequencies, delayed corrections, uncertain information, deletion requests, conflicting sources, and adversarial memory contamination. Additional benchmark tracks should evaluate multilingual and multimodal memory, multi-user access control, privacy-preserving forgetting, tool-generated state changes, and causal dependencies among events. Future studies should also separate retrieval failure from reasoning failure through oracle-evidence experiments and assess the effects of retrieval budget, context-window size, embedding model, consolidation strategy, and memory-compression method. Dynamic hidden test sets should be periodically regenerated to reduce benchmark contamination, while leaderboard reporting should include component-level accuracy, confidence intervals, latency, token cost, storage requirements, and energy use. Such extensions would improve the ecological validity of DRIFTBENCH and support the development of AI agents whose memories remain accurate, selective, efficient, and adaptable during prolonged real-world use.

Conclusion

DRIFTBENCH provides a controlled and reproducible framework for evaluating the long-horizon memory capabilities of AI agents across multi-day and multi-session interactions. By separately measuring update fidelity, interference resistance, temporal decay, abstention accuracy, and computational efficiency, the benchmark reveals memory failures that are often concealed by aggregate long-context accuracy. The results showed that Hybrid fusion achieved the strongest overall performance and the greatest resistance to repeated factual revisions, distractor accumulation, and delayed retrieval, whereas Temporal slots offered the most favourable balance between accuracy and retrieval latency. Graph memory provided moderate robustness, while Vector RAG and the sliding-window baseline were more vulnerable to stale information, semantic competition, and temporal displacement. These findings demonstrate that reliable persistent memory requires explicit management of temporal validity, structured version control, selective retrieval, and resistance to irrelevant information rather than reliance on context length or semantic similarity alone. Although the present findings are based on a controlled synthetic pilot, DRIFTBENCH establishes a useful foundation for future end-to-end evaluation of real AI agents and can support the development of memory systems that remain accurate, current, efficient, and dependable over extended interaction histories.

References

1. Amebleh, J., Igba, E., & Ijiga, O. M. (2021). Graph-based fraud detection in open-loop gift cards: Heterogeneous GNNs, streaming feature stores, and near-zero-lag anomaly alerts. International Journal of Scientific Research in Science, Engineering and Technology, 8(6), 2348-0459.
2. Brohi, S., Mastoi, Q. U. A., Jhanjhi, N. Z., & Pillai, T. R. (2025). A research landscape of agentic ai and large language models: Applications, challenges and future directions. Algorithms, 18(8), 499.
3. Budhwar, P., Chowdhury, S., Wood, G., Aguinis, H., Bamber, G. J., Beltran, J. R., ... & Varma, A. (2023). Human resource management in the age of generative artificial intelligence: Perspectives and research directions on ChatGPT. Human Resource Management Journal, 33(3), 606-659.

4. Harlow, I. M., & Yonelinas, A. P. (2016). Distinguishing between the success and precision of recollection. Memory, 24(1), 114-127.
5. How, J. P., Fraser, C., Kulling, K. C., Bertuccelli, L. F., Toupet, O., Brunet, L., ... & Roy, N. (2009). Increasing autonomy of UAVs. IEEE Robotics & Automation Magazine, 16(2), 43-51.
6. Huang, Q., Vora, J., Liang, P., & Leskovec, J. (2023). Benchmarking large language models as ai research agents. In NeurIPS 2023 foundation models for decision making workshop.
7. Lande, D., & Strashnoy, L. (2025). Memory in Time: A Network Model of Information Aging and a Strategy for Constructing Retrospective Databases. Available at SSRN 5653090.
8. Li, X., Bantupalli, J., Dharmani, R., Zhang, Y., & Shang, J. (2025, November). Toward multi-session personalized conversation: A large-scale dataset and hierarchical tree framework for implicit reasoning. In Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing (pp. 11504-11517).
9. Maharana, A., Lee, D. H., Tulyakov, S., Bansal, M., Barbieri, F., & Fang, Y. (2024, August). Evaluating very long-term conversational memory of llm agents. In Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers) (pp. 13851-13870).
10. Masse, N. Y., Rosen, M. C., & Freedman, D. J. (2020). Reevaluating the role of persistent neural activity in short-term memory. Trends in cognitive sciences, 24(3), 242-258.
11. Phillips, R. C., & Gorse, D. (2017, November). Predicting cryptocurrency price bubbles using social media data and epidemic modelling. In 2017 IEEE symposium series on computational intelligence (SSCI) (pp. 1-7). IEEE.
12. Tiwari, A., & Gupta, V. (2026). A memory fabric for conversational AI agents enabling shared and persistent multiuser memory. Discover Artificial Intelligence, 6(1), 239.
13. Wan, C., Sheng, T., Li, H., Zhang, Y., & Yu, C. (2025). Enhancing Fairness in High-Speed Railway Crew Scheduling: A Two-Stage Heuristic Optimization Framework Under Daily-Adjusted Timetables. Applied Sciences, 16(1), 376.
14. Wang, H., Yu, Z., Ren, Z., Zhang, Y., Liu, J., Wang, L., & Guo, B. (2025). Finghv: Efficient sharing and fine-grained scheduling of virtualized hpu resources. IEEE Transactions on Cybernetics, 55(2), 600-614.
15. Wang, W., Dong, L., Cheng, H., Liu, X., Yan, X., Gao, J., & Wei, F. (2023). Augmenting language models with long-term memory. Advances in Neural Information Processing Systems, 36, 74530-74543.
16. Wen, K., Dang, X., & Lyu, K. (2025, May). Rnns are not transformers (yet): The key bottleneck on in-context retrieval. In International Conference on Learning Representations (Vol. 2025, pp. 48813-48856).
17. Wood, R., Baxter, P., & Belpaeme, T. (2012). A review of long-term memory in natural and synthetic systems. Adaptive Behavior, 20(2), 81-103.
18. Xu, Y., & Li, Y. (2025). A real-time urban traffic congestion prediction framework based on dynamic risk field and multi-source data fusion. IEEE Access.
19. Zhao, X., Jia, Y., Li, A., Jiang, R., & Song, Y. (2020). Multi-source knowledge fusion: a survey. World Wide Web, 23(4), 2567-2592.
20. Zhitomirsky-Geffet, M. (2019). Towards a diversified knowledge organization system: An open network of inter-linked subsystems with multiple validity scopes. Journal of Documentation, 75(5), 1124-1138.