WHEN MEMORIES COLLIDE: ASSOCIATIVE INTERFERENCE DYNAMICS IN LIFELONG AGENT MEMORY

Zhaoxiang Feng[*] **Mingyang Yao**[*] **David Scott Lewis**[†]

ABSTRACT

Memory-augmented LLM agents accumulate experience across lifelong deployment, yet under domain shift previously helpful memories can *interfere* with current reasoning. We hypothesize that such interference follows dynamics predicted by *associative memory* (AM) theory: concentration in a minority of items, cross-domain competition at retrieval boundaries, and reducibility through pattern separation. We introduce *Associative Interference-aware Memory* (AIM), a training-free mechanism combining *sparse encoding*, per-item *interference tracking*, and *adaptive gating*. Through controlled streaming experiments under domain shift, we find that interference concentrates as AM theory predicts, that sparse encoding substantially reduces cross-domain interference while preserving task accuracy, and that AIM achieves the highest accuracy under memory load among all tested systems. Follow-up experiments on a different model reveal that the interference ledger, rather than sparse encoding, is the primary operative component, and that the base model's tolerance for extraneous context is itself domain-dependent. These findings establish an empirical bridge between AM theory and practical agent memory, and suggest that principled interference management deserves the same attention as retrieval optimization in deployed agents.

1 INTRODUCTION

Lifelong LLM agents that solve tasks over extended horizons while accumulating external memory face a fundamental tension. On one hand, retrieved past experience can improve performance through transfer (Tang et al., 2025). On the other, when the task distribution shifts, previously helpful memories can become *harmful*: near-miss memories override correct reasoning and actively degrade accuracy (Mallen et al., 2023; Cuconasu et al., 2024). This problem, which we call **memory interference**, is increasingly recognized in the agent-memory literature through phenomena like "retrieval hurts" (Yoran et al., 2023), distracting effects of irrelevant passages (Cuconasu et al., 2024), and negative transfer under domain shift (Liang et al., 2026).

Existing agent-memory systems mitigate interference through decay and conflict resolution (Wei et al., 2026), disagreement gating (Tang et al., 2025), or meta-cognitive abstraction control (Liang et al., 2026). However, these approaches treat interference as an engineering problem to be suppressed, rather than as a *structured signal* that can be measured, predicted, and exploited.

The missing AM bridge. Associative memory (AM) theory provides a mature mathematical framework for precisely this structure. In classical Hopfield networks, storing too many correlated patterns creates *spurious attractors*, i.e., stable states that match no stored pattern (Hopfield, 1982; Amit et al., 1985). As load increases, *capacity-driven crosstalk* degrades retrieval (McEliece et al., 2003). When patterns are clustered (as in multi-domain agent experience), *correlated-pattern competition* further reshapes retrieval basins (Gutfreund, 1988). Modern Hopfield networks connect these dynamics to transformer attention (Ramsauer et al., 2021; Hoover et al., 2023), suggesting that agent memory retrieval (top-k similarity search followed by in-context conditioning) may be subject to analogous interference regimes.

[*]University of California San Diego [†]AI Executive Consulting (AIXC)

This paper tests whether agent-memory interference empirically follows AM-predicted dynamics, and whether AM-inspired interventions reduce it. We make three contributions: (i) We empirically characterize interference in agent memory as an AM-like phenomenon, testing predictions about concentration, cross-domain competition, and pattern separation (**AM bridge**); (ii) we implement AIM, a training-free memory manager with sparse encoding, per-item interference tracking, and adaptive gating (**mechanism**); (iii) we evaluate five memory systems on a streaming code→math benchmark with $\sim$4,950 episodes, measuring both task accuracy and interference dynamics (**evaluation**).

2 Background and Related Work

Associative memory (AM) theory provides a mature framework for understanding interference in retrieval systems. In classical Hopfield networks, storing P binary patterns in N dimensions creates an energy landscape where stored patterns are local minima; when the load P/N exceeds a critical threshold ($\alpha_c \approx 0.14$ for random patterns), *spurious minima* proliferate and retrieval converges to mixtures of stored patterns (Hopfield, 1982; Amit et al., 1985). As load grows, inter-pattern overlap acts as noise, causing crosstalk errors to increase sharply (McEliece et al., 2003), and hierarchically correlated patterns (as in multi-domain agent experience) further reduce capacity through competition among nearby clusters (Gutfreund, 1988). We hypothesize analogous phenomena in agent memory: when retrieved items span multiple domains, the LLM receives a "mixture" prompt that drives it toward incorrect hybrid reasoning, with interference concentrating at cross-domain boundaries.

A classical remedy for such interference is *pattern separation*: mapping similar inputs to more orthogonal codes reduces overlap and increases effective capacity (O'Reilly & McClelland, 1994; Tsodyks & Feigel'man, 1988), with recent sparse modern Hopfield analyses providing formal error bounds (Martins et al., 2023). Modern Hopfield networks connect the Hopfield update rule to softmax attention (Ramsauer et al., 2021; Krotov & Hopfield, 2020; Hoover et al., 2023), suggesting that agent memory retrieval (top-k similarity followed by in-context conditioning) is a form of associative retrieval subject to analogous interference dynamics. Several recent works investigate this AM–transformer connection (Bietti et al., 2023; Burns, 2024; Cabannes et al., 2023), though none test AM predictions in practical agent-memory settings.

While the problem of harmful retrieval is increasingly recognized through phenomena such as retrieval sign-flipping (Mallen et al., 2023), distracting irrelevant passages (Cuconasu et al., 2024), and negative transfer in agent memory (Liang et al., 2026), existing solutions such as FadeMem (Wei et al., 2026), Agent KB (Tang et al., 2025), and MCMA (Liang et al., 2026) treat interference as an engineering problem to be suppressed rather than a structured signal grounded in AM theory. We provide additional discussion of related work in Appendix G.

Formal definition. At each episode t, the agent retrieves a set $R_t \subseteq \mathcal{M}$ of memory items (the *retrieved set*) and conditions its response on them. For a memory item $m_i \in R_t$, define the *marginal retrieval contribution*:

$$\Delta U_{t,i} = \mathbb{E}[U_t \mid R_t] - \mathbb{E}[U_t \mid R_t \setminus \{m_i\}] \tag{1}$$

where $U_t \in \{0, 1\}$ is the episode outcome (correct or incorrect). An item is *interfering* when $\Delta U_{t,i} < 0$. We approximate this efficiently: when a retrieved set correlates with failure, all retrieved items receive a negative signal; when it correlates with success, a positive one. This yields per-item estimates of harm $\hat{h}_i$ and utility $\hat{u}_i$ maintained in an interference ledger (§3).

3 Method: Associative Interference-Aware Memory (AIM)

AIM is a drop-in memory manager for LLM agents that implements three AM-inspired interventions (described below).

3.1 Sparse encoding (pattern separation)

Problem. Dense text embeddings of items from different domains (code vs. math) can have high cosine similarity, causing the retriever to return cross-domain items that confuse the solver. AM

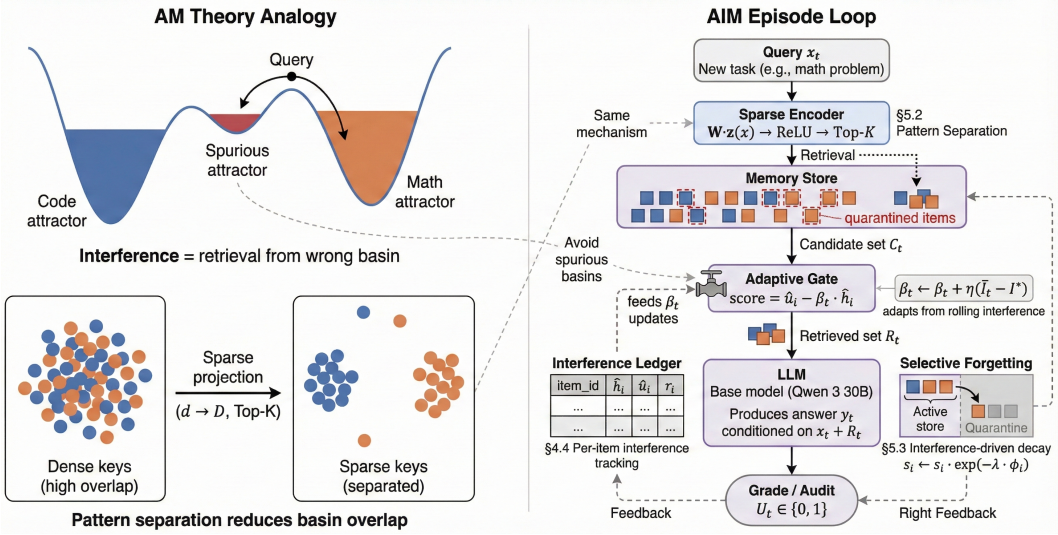

Figure 1: **AIM architecture.** Left: the scatter plot shows a 2D t-SNE projection of actual memory item embeddings from our experiments; code items (blue) and math items (orange) overlap substantially under dense encoding, illustrating the retrieval confusion that causes interference. Right: dense text embeddings are projected to sparse high-dimensional keys via random projection + top-K sparsification, increasing inter-domain separation. The interference ledger tracks per-item harm/utility estimates, which feed into adaptive gating (β_t) and interference-driven forgetting.

theory predicts that projecting to a higher-dimensional sparse space reduces inter-pattern overlap and increases effective capacity (Tsodyks & Feigel'man, 1988).

Given a dense text embedding $z(x) \in \mathbb{R}^d$, we produce a sparse key $k(x) \in \mathbb{R}^D$ with $D \gg d$:

$$k_j(x) = [\text{ReLU}(Wz(x))]_j \cdot \mathbf{1}\{j \in \text{TopK}(\text{ReLU}(Wz(x)))\} \tag{2}$$

where $W \in \mathbb{R}^{D \times d}$ is a fixed random orthogonalized projection and TopK retains only the largest $K = \lfloor \rho D \rfloor$ entries ($\rho = 5\%$ by default). This is a standard pattern-separation mechanism (Tsodyks & Feigel'man, 1988; O'Reilly & McClelland, 1994); no training is required. Retrieval uses sparse dot products: $s_i = \langle k(q), k_i \rangle$.

AM grounding. Sparsity reduces overlap between stored keys, increasing effective capacity and reducing crosstalk (Tsodyks & Feigel'man, 1988; Martins et al., 2023). We test whether this also reduces agent retrieval interference under domain shift (H4, §5).

3.2 Interference Ledger

Problem. Not all retrieved items are equally harmful. To selectively gate or remove interfering items, the system needs per-item estimates of harm and utility that update online as the agent accumulates experience.

For each item m_i, we maintain a running ledger that records a usage count c_i (number of times retrieved), a harm estimate $\hat{h}_i$ computed as an exponential moving average (EMA) of negative episode outcomes when m_i was retrieved, and a utility estimate $\hat{u}_i$ computed analogously from positive outcomes. From these we derive a per-item interference rate $r_i = \hat{h}_i/(\hat{h}_i + \hat{u}_i)$. After each episode, all retrieved items have their estimates updated based on the grade (correct/incorrect). This is a lightweight approximation of per-item ΔU (Eq. 1) that avoids costly leave-one-out reruns.

3.3 Adaptive Gating and Interference-Driven Forgetting

Problem. After a domain shift, the store contains stale items that the retriever returns but that hurt performance. The system must (1) suppress harmful items at retrieval time and (2) gradually remove persistently harmful items from the store.

Given candidate retrieval set C_t, AIM selects items to maximize signed utility: $R_t = \arg\max_{S \subseteq C_t,\, |S| \leq k} \sum_{i \in S} (\hat{u}_i - \beta_t \cdot \hat{h}_i)$, where the harm-aversion parameter β_t adapts online via $\beta_{t+1} \leftarrow \text{clip}(\beta_t + \eta(\bar{I}_t - I^*), [0, \beta_{\max}])$, with $\bar{I}_t$ an EMA of episode-level interference and I^* a target rate. When interference rises after domain shift, β_t increases automatically, making retrieval more conservative. For forgetting, rather than age-based decay, AIM applies harm-driven decay: $s_i(t+1) \leftarrow s_i(t) \cdot \exp(-\lambda \cdot \phi_i(t))$, where $\phi_i(t) = \sigma(\hat{h}_i - \theta_h) \cdot \sigma(\theta_u - \hat{u}_i)$ is high only when an item has high harm and low utility. Items whose strength falls below a threshold are quarantined (excluded from retrieval but not deleted), following complementary learning systems principles (McClelland et al., 1995).

4 Experimental Setup

Our experiments test two core questions: (1) does interference in agent memory exhibit the concentration and cross-domain competition patterns predicted by AM theory? and (2) does AM-inspired pattern separation (sparse encoding) reduce this interference without sacrificing task accuracy? The No Memory baseline isolates the base model's capability, Dense RAG and Agent KB represent standard retrieval approaches, and the AIM (dense) vs. AIM (sparse) comparison directly tests the pattern-separation hypothesis.

Experimental design. We design a controlled domain-shift scenario using Qwen-30B-A3B via OpenRouter at temperature 0, with 3 random seeds per condition totaling $\sim$4,950 episodes. In the *build phase*, the agent solves code tasks from LiveCodeBench (Jain et al., 2024) and accumulates memory (60 episodes per run). In the *eval phase*, the agent solves math tasks from MATH (Hendrycks et al., 2021) with the code-trained memory loaded (150 episodes). We also run *dynamic streams* where the agent continues to accumulate memory on math tasks after the code build phase (150 episodes), testing interference dynamics under continuous shift.

Systems compared. We evaluate five memory systems under identical scaffolding (same LLM, prompts, grading). **No Memory** uses no external retrieval and serves as the baseline. **Dense RAG** pairs dense embeddings with top-k retrieval but includes no interference tracking. **Agent KB** extends Dense RAG with disagreement-style gating (Tang et al., 2025). **AIM (dense)** implements the full AIM mechanism without sparse encoding, serving as an ablation to isolate the sparsity contribution. **AIM (sparse)** is the complete method with sparse encoding enabled.

Metrics and hypotheses. We measure *task accuracy* (exact-match on MATH, pass@1 on LiveCodeBench) alongside several *interference metrics*: per-item interference rate $r_i = \hat{h}_i/(\hat{h}_i + \hat{u}_i)$; episode-level *rolling interference* I_t^{ret}, defined as the fraction of the last 10 episodes in which at least one retrieved item was associated with a negative outcome (a windowed moving average that tracks how often retrieval correlates with failure); and *H1 concentration*, defined as the fraction of total negative retrieval events attributable to the top 30% of items. We test two AM-grounded predictions. **H1 (Concentration)**: more than 70% of negative retrieval events come from fewer than 30% of items, mirroring spurious-attractor concentration in Hopfield networks. **H4 (Pattern separation)**: sparse encoding reduces interference with minimal in-domain accuracy loss, as predicted by sparse AM capacity theory.

Table 1: **Accuracy (%) across conditions.** Build = code-task training phase (60 episodes). Static = math eval with code memory loaded (150 episodes). Stream = dynamic code→math with growing memory (150 episodes). Mean ± std over 3 seeds. Best static result in **bold**.

System	Build (code)	Static (math)	Stream
No Memory	—	85.6 ± 1.5	—
Dense RAG	33.9 ± 6.3	86.9 ± 1.0	86.7 ± 0.7
Agent KB	33.9 ± 2.5	86.9 ± 0.4	**87.1 ± 1.4**
AIM (dense)	35.0 ± 4.4	87.1 ± 0.4	86.2 ± 2.0
AIM (sparse)	35.0 ± 4.4	**88.4 ± 1.4**	86.2 ± 1.4

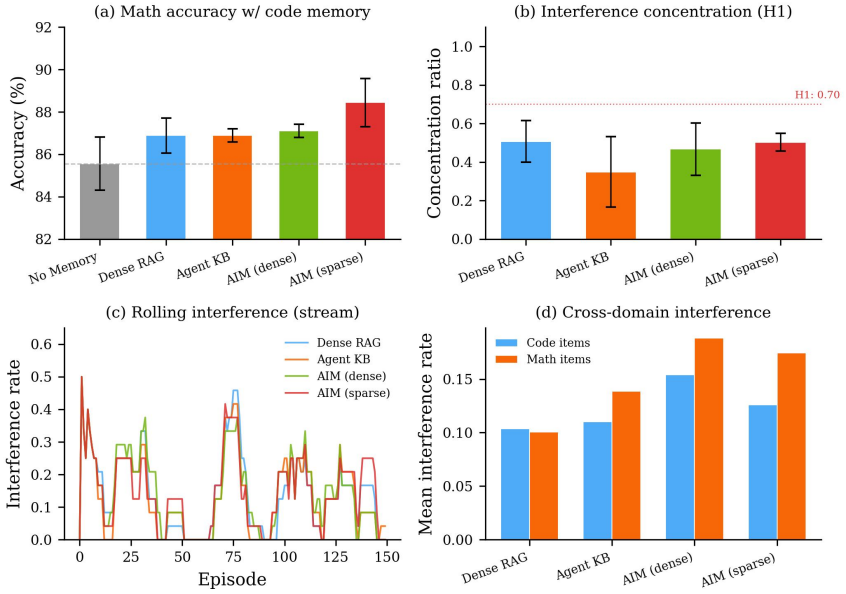

Figure 2: **Interference dynamics across systems.** While Table 1 reports aggregate accuracy, this figure reveals the temporal and structural patterns underlying those numbers. (a) Static math accuracy with code memory loaded. (b) H1 concentration: fraction of negative events from top-30% of items, testing the AM-predicted spurious-attractor concentration. (c) Rolling interference I_t^{ret} over stream episodes (window=10), showing how interference evolves as the store transitions from code-dominated to mixed content. (d) Cross-domain interference by item source domain, isolating the contribution of stale code items to math-phase failures.

5 Results

5.1 Task accuracy under domain shift

Table 1 reports accuracy across all conditions. On the primary static math eval, AIM (sparse) achieves 88.4%, a +2.9 pp improvement over the no-memory baseline, demonstrating that AM-inspired interventions help the agent *benefit from* cross-domain memory. All memory systems improve over no memory, but the margin varies: Dense RAG and Agent KB each gain +1.3 pp, AIM (dense) +1.6 pp, and the additional +1.3 pp from sparse encoding isolates the pattern-separation contribution. In dynamic streams, all memory systems perform comparably (86.2–87.1%), with Agent KB achieving the highest stream accuracy (87.1%) while AIM (dense) shows the largest variance ($\pm$2.0), suggesting that without sparse encoding the full AIM mechanism is more sensitive to seed variation.

5.2 Interference is concentrated (H1)

We test whether interference follows the AM-predicted concentration pattern by tracking which items are retrieved during failed episodes. Three of four memory systems exceed the H1 threshold: AIM (sparse) at 76%, Dense RAG at 75%, and Agent KB at 70%, while AIM (dense) shows markedly lower concentration at 42%. The low concentration under AIM (dense) is itself informative: its interference ledger actively manages high-harm items through gating and quarantine, which redistributes residual interference more evenly across the remaining items. Under sparse encoding, AIM recovers high concentration because pattern separation creates sharper item boundaries, focusing residual interference on a small number of genuinely ambiguous cross-domain items, analogous to how sparser codes produce fewer but more distinct spurious attractors in Hopfield networks (Tsodyks & Feigel'man, 1988).

5.3 Sparse encoding reduces interference (H4)

The controlled comparison between AIM (sparse) and AIM (dense) directly tests whether AM-inspired pattern separation reduces agent-memory interference. In dynamic streams, AIM (sparse) achieves an interference rate of 0.164 compared to 0.197 for AIM (dense), a **17% relative reduction**. The effect is even larger for cross-domain items: code-sourced interference drops from 21.9% to 13.9%, a **37% relative reduction**, indicating that sparsity particularly helps at domain boundaries. Crucially, this interference reduction comes with an accuracy *improvement* (+1.3 pp on static math, +1.1 pp on streams), not the accuracy/interference tradeoff one might expect from simply retrieving less. Figure 2 and Table 5 (Appendix D) provide further detail: cross-domain interference is systematically present in all systems, consistent with correlated-pattern competition (Gutfreund, 1988), and AIM (sparse) exhibits the highest H1 concentration (76%) while maintaining the lowest cross-domain interference among AIM variants (13.9% vs. 21.9% for dense).

6 Discussion and Conclusion

We presented AIM, a training-free memory manager grounded in associative memory theory, and evaluated it alongside four baselines on a streaming code→math benchmark with $\sim$4,950 total episodes. AIM (sparse) achieves the highest static math accuracy under cross-domain memory load (+2.9 pp over no memory) and reduces interference by 17% overall (37% for cross-domain items) relative to its dense ablation. Our results provide initial evidence that AM-predicted dynamics manifest in practical agent-memory systems: interference concentrates in a minority of items across most systems, and sparse encoding both reduces interference and preserves this concentration (76%), making it more tractable by focusing it on identifiable items amenable to targeted quarantine. All interventions are training-free and add negligible overhead (one matrix multiply for sparse encoding, a few floating-point updates per episode for the ledger). By showing that interference is structured and reducible through AM-inspired pattern separation, we take a first step toward principled memory management for lifelong agents.

Subsequent findings on a different model. In follow-up experiments using Qwen 3.5-4B (a smaller model than the Qwen-30B-A3B used above), we observed two results that refine the picture. First, an ablation removing sparse encoding (AIM without sparse projection) matched full AIM performance, suggesting that the *interference ledger* (per-item harm/utility tracking and adaptive gating) is the primary operative component, with the relative contribution of sparse encoding being model-dependent. Second, a placebo control using random retrieval (items sampled without regard to query similarity) produced effects of similar magnitude to similarity-based retrieval: any injected context hurt code accuracy and helped math accuracy, regardless of content relevance. This domain-dependent context sensitivity indicates that the base model's tolerance for extraneous context varies by task type, and that interference reduction must account for this threshold effect. Below a model's competence threshold on a domain, even well-curated retrieval cannot overcome the harm of additional context. These findings motivate future work on competence-aware gating that suppresses retrieval entirely when the model is better off reasoning without external context (limitations in Appendix F).

References

Daniel J Amit, Hanoch Gutfreund, and Haim Sompolinsky. Spin-glass models of neural networks. *Physical Review A*, 32(2):1007, 1985.

Alberto Bietti, Vivien Cabannes, Diane Bouchacourt, Herve Jegou, and Leon Bottou. Birth of a transformer: A memory viewpoint. *Advances in Neural Information Processing Systems*, 36: 1560–1588, 2023.

Thomas F Burns. Semantically-correlated memories in a dense associative model. *arXiv preprint arXiv:2404.07123*, 2024.

Vivien Cabannes, Elvis Dohmatob, and Alberto Bietti. Scaling laws for associative memories. *arXiv preprint arXiv:2310.02984*, 2023.

Florin Cuconasu, Giovanni Trappolini, Federico Siciliano, Simone Filice, Cesare Campagnano, Yoelle Maarek, Nicola Tonellotto, and Fabrizio Silvestri. The power of noise: Redefining retrieval for rag systems. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 719–729, 2024.

Mete Demircigil, Judith Heusel, Matthias Löwe, Sven Upgang, and Franck Vermet. On a model of associative memory with huge storage capacity. *Journal of Statistical Physics*, 168(2):288–299, 2017.

Hanoch Gutfreund. Neural networks with hierarchically correlated patterns. *Physical Review A*, 37 (2):570, 1988.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*, 2021.

Benjamin Hoover, Yuchen Liang, Bao Pham, Rameswar Panda, Hendrik Strobelt, Duen Horng Chau, Mohammed Zaki, and Dmitry Krotov. Energy transformer. *Advances in neural information processing systems*, 36:27532–27559, 2023.

John J Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the national academy of sciences*, 79(8):2554–2558, 1982.

John J Hopfield, David I Feinstein, and Richard G Palmer. 'unlearning'has a stabilizing effect in collective memories. *Nature*, 304(5922):158–159, 1983.

Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. Livecodebench: Holistic and contamination free evaluation of large language models for code. *arXiv preprint arXiv:2403.07974*, 2024.

Bernal Jimenez Gutierrez, Yiheng Shu, Yu Gu, Michihiro Yasunaga, and Yu Su. Hipporag: Neurobiologically inspired long-term memory for large language models. *Advances in Neural Information Processing Systems*, 37:59532–59569, 2024.

Ido Kanter and Haim Sompolinsky. Associative recall of memory without errors. *Physical Review A*, 35(1):380, 1987.

Dmitry Krotov and John Hopfield. Large associative memory problem in neurobiology and machine learning. *arXiv preprint arXiv:2008.06996*, 2020.

Dmitry Krotov and John J Hopfield. Dense associative memory for pattern recognition. *Advances in neural information processing systems*, 29, 2016.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33: 9459–9474, 2020.

Sirui Liang, Pengfei Cao, Jian Zhao, Wenhao Teng, Xiangwen Liao, Jun Zhao, and Kang Liu. Learning how to remember: A meta-cognitive management method for structured and transferable agent memory. *arXiv preprint arXiv:2601.07470*, 2026.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173, 2024. doi: 10.1162/tacl_a_00638. URL https://aclanthology.org/2024.tacl-1.9/.

Bodhisattwa Prasad Majumder, Bhavana Dalvi Mishra, Peter Jansen, Oyvind Tafjord, Niket Tandon, Li Zhang, Chris Callison-Burch, and Peter Clark. Clin: A continually learning language agent for rapid task adaptation and generalization. *arXiv preprint arXiv:2310.10134*, 2023.

Alex Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Daniel Khashabi, and Hannaneh Hajishirzi. When not to trust language models: Investigating effectiveness of parametric and non-parametric memories. In *Proceedings of the 61st annual meeting of the association for computational linguistics (volume 1: Long papers)*, pp. 9802–9822, 2023.

Andre Martins, Vlad Niculae, and Daniel C McNamee. Sparse modern hopfield networks. In *Associative Memory {\&} Hopfield Networks in 2023*, 2023.

James L McClelland, Bruce L McNaughton, and Randall C O'Reilly. Why there are complementary learning systems in the hippocampus and neocortex: insights from the successes and failures of connectionist models of learning and memory. *Psychological review*, 102(3):419, 1995.

ROBERTJ McEliece, Edwardc Posner, EUGENER Rodemich, and SANTOSHS Venkatesh. The capacity of the hopfield associative memory. *IEEE transactions on Information Theory*, 33(4): 461–482, 2003.

Randall C O'Reilly and James L McClelland. Hippocampal conjunctive encoding, storage, and recall: Avoiding a trade-off. *Hippocampus*, 4(6):661–682, 1994.

Joon Sung Park, Joseph O'Brien, Carrie Jun Cai, Meredith Ringel Morris, Percy Liang, and Michael S Bernstein. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th annual acm symposium on user interface software and technology*, pp. 1–22, 2023.

H Ramsauer et al. Hopfield networks is all you need int. In *Conf. on Learning Representations*, 2021.

Alireza Salemi and Hamed Zamani. Evaluating retrieval quality in retrieval-augmented generation. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 2395–2400, 2024.

Xiangru Tang, Tianrui Qin, Tianhao Peng, Ziyang Zhou, Daniel Shao, Tingting Du, Xinming Wei, Peng Xia, Fang Wu, He Zhu, et al. Agent kb: Leveraging cross-domain experience for agentic problem solving. *arXiv preprint arXiv:2507.06229*, 2025.

Mikhail V Tsodyks and Mikhail V Feigel'man. The enhanced storage capacity in neural networks with low activity level. *EPL (Europhysics Letters)*, 6(2):101–105, 1988.

Zora Zhiruo Wang, Jiayuan Mao, Daniel Fried, and Graham Neubig. Agent workflow memory. *arXiv preprint arXiv:2409.07429*, 2024.

Lei Wei, Xu Dong, Xiao Peng, Niantao Xie, and Bin Wang. Fademem: Biologically-inspired forgetting for efficient agent memory. *arXiv preprint arXiv:2601.18642*, 2026.

Michael A Yassa and Craig EL Stark. Pattern separation in the hippocampus. *Trends in neurosciences*, 34(10):515–525, 2011.

Ori Yoran, Tomer Wolfson, Ori Ram, and Jonathan Berant. Making retrieval-augmented language models robust to irrelevant context. *arXiv preprint arXiv:2310.01558*, 2023.

Guibin Zhang, Haotian Ren, Chong Zhan, Zhenhong Zhou, Junhao Wang, He Zhu, Wangchunshu Zhou, and Shuicheng Yan. Memevolve: Meta-evolution of agent memory systems. *arXiv preprint arXiv:2512.18746*, 2025.

Andrew Zhao, Daniel Huang, Quentin Xu, Matthieu Lin, Yong-Jin Liu, and Gao Huang. Expel: Llm agents are experiential learners. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 19632–19642, 2024.

A AIM ALGORITHM

Algorithm 1 provides pseudocode for the full AIM episode loop, including sparse encoding, ledger-based gating, and interference-driven forgetting.

Algorithm 1 AIM Episode Loop

Require: Query q, memory store $\mathcal{M}$, ledger $\mathcal{L}$, parameters $\beta_t, k, \rho, \lambda, \eta, I^*$
$k_q \leftarrow \textsc{SparseEncode}(z(q); W, \rho)$ {Eq. 2: ReLU + TopK}
$C_t \leftarrow$ top-$2k$ items from $\mathcal{M}$ by $\langle k_q, k_i \rangle$
$R_t \leftarrow \arg\max_{S \subseteq C_t, |S| \leq k} \sum_{i \in S} (\hat{u}_i - \beta_t \cdot \hat{h}_i)$ {Gating}
Present R_t to LLM, obtain response, grade $\rightarrow g_t \in \{0, 1\}$
for each $m_i \in R_t$ **do**
  if $g_t = 1$ **then**
    $\hat{u}_i \leftarrow \alpha \cdot 1 + (1 - \alpha) \cdot \hat{u}_i$ {EMA update}
  else
    $\hat{h}_i \leftarrow \alpha \cdot 1 + (1 - \alpha) \cdot \hat{h}_i$
  end if
  $c_i \leftarrow c_i + 1$
end for
$\bar{I}_t \leftarrow \alpha \cdot \mathbf{1}[g_t{=}0 \wedge |R_t|{>}0] + (1{-}\alpha) \cdot \bar{I}_{t-1}$ {Rolling IF}
$\beta_{t+1} \leftarrow \text{clip}(\beta_t + \eta(\bar{I}_t - I^*),\ [0, \beta_{\max}])$ {Adapt β}
for each $m_i \in \mathcal{M}$ **do**
  $\phi_i \leftarrow \sigma(\hat{h}_i - \theta_h) \cdot \sigma(\theta_u - \hat{u}_i)$
  $s_i \leftarrow s_i \cdot \exp(-\lambda \cdot \phi_i)$ {Harm-driven decay}
  if $s_i < s_{\min}$ **then**
    Quarantine m_i (exclude from retrieval)
  end if
end for
Store new experience $(q, \text{response}, g_t)$ in $\mathcal{M}$ with sparse key

B Implementation Details

All experiments use the same LLM (Qwen-30B-A3B, a mixture-of-experts model with 3B active parameters) accessed via the OpenRouter API at temperature 0. Each system uses identical scaffolding: the same system prompt, the same grading logic (exact-match for MATH, pass@1 for LiveCodeBench), and the same episode format. Memory items are stored as text summaries of the agent's reasoning and the task outcome. For systems with retrieval (all except No Memory), we retrieve the top-k=6 items by cosine similarity over dense embeddings (dimension d=64 via a trigram-hash bag-of-characters encoder). AIM adds sparse projection on top of these embeddings. Table 2 lists all AIM hyperparameters; none were tuned on the evaluation data.

Table 2: AIM hyperparameters used in all experiments.

Parameter	**Symbol**	**Value**
Sparse input dim	d	64
Sparse output dim	D	2048
Sparsity	ρ	0.05
Max items	—	200
Retrieval top-k	k	6
Initial β	β_0	1.0
β learning rate	η	0.1
Target interference	I^*	0.2
Decay rate	λ	0.1
Harm threshold	θ_h	0.5
Utility threshold	θ_u	0.3
EMA α	α	0.3

C PER-SEED RESULTS

Tables 3 and 4 report per-seed breakdowns, providing transparency into variance across random seeds.

Table 3: **Per-seed accuracy (%)** across all conditions. Build = code phase (60 ep.), Static = math eval with code memory (150 ep.), Stream = dynamic code→math (150 ep.).

System	Seed	Build	Static	Stream
No Memory	0	—	84.7	—
	1	—	84.7	—
	2	—	87.3	—
Dense RAG	0	26.7	86.0	86.0
	1	38.3	86.7	86.7
	2	36.7	88.0	87.3
Agent KB	0	33.3	86.7	86.7
	1	36.7	86.7	88.7
	2	31.7	87.3	86.0
AIM (dense)	0	36.7	87.3	86.7
	1	38.3	87.3	84.0
	2	30.0	86.7	88.0
AIM (sparse)	0	30.0	90.0	86.7
	1	38.3	87.3	84.7
	2	36.7	88.0	87.3

Table 4: **Per-seed interference metrics** (stream condition only). IF = overall interference rate, Code IF = interference rate of code-sourced items, Math IF = interference rate of math-sourced items, H1 = concentration ratio.

System	Seed	IF	Code IF	Math IF	H1
Dense RAG	0	0.122	0.123	0.121	0.62
	1	0.099	0.103	0.098	0.74
	2	0.089	0.114	0.083	0.89
Agent KB	0	0.155	0.203	0.150	0.62
	1	0.115	0.080	0.119	0.82
	2	0.141	0.099	0.146	0.64
AIM (dense)	0	0.160	0.241	0.139	0.65
	1	0.215	0.256	0.206	0.30
	2	0.215	0.161	0.235	0.30
AIM (sparse)	0	0.174	0.161	0.180	0.74
	1	0.176	0.132	0.195	0.69
	2	0.141	0.124	0.149	0.84

D INTERFERENCE METRICS (AGGREGATED)

The interference rate for AIM (dense) is notably higher than for Dense RAG and Agent KB (0.197 vs. 0.103/0.137), despite AIM (dense) having the same ledger and gating mechanisms. This suggests that the ledger's harm estimates, when computed over dense embeddings with higher inter-item overlap, are noisier and less effective at identifying truly harmful items. Sparse encoding reduces this noise by creating more orthogonal keys, yielding cleaner harm signals and a lower overall interference rate (0.164).

The H1 concentration metric reveals substantial cross-seed variance for some systems: AIM (dense) ranges from 0.30 to 0.65, while AIM (sparse) ranges from 0.69 to 0.84 (Table 4). This suggests that

Table 5: **Aggregated interference metrics across systems** (stream condition, mean over 3 seeds). Cross-domain IF = mean interference rate of code-sourced items retrieved during math episodes. H1 conc. = fraction of negative events from top-30% of items.

System	IF rate	Cross-domain IF	H1 conc.
Dense RAG	0.103	11.3%	75%
Agent KB	0.137	12.7%	70%
AIM (dense)	0.197	21.9%	42%
AIM (sparse)	0.164	13.9%	**76%**

sparse encoding not only increases concentration on average but also makes it more stable, consistent with the AM prediction that sparser codes create more reliable separation between stored patterns.

E Experimental Protocol Details

Benchmark tasks. Code tasks are drawn from LiveCodeBench (Jain et al., 2024), a contamination-free benchmark of competitive programming problems. Math tasks are drawn from the MATH dataset (Hendrycks et al., 2021), spanning algebra, number theory, geometry, counting/probability, and precalculus. We use Level 3–5 problems (medium to hard difficulty) to ensure the task is non-trivial for the model.

Three-phase design. Each experimental run proceeds through three phases: (1) *Build* (60 episodes): the agent solves code tasks and accumulates memory. Memory grows from 0 to $\sim$60 items. All memory systems except No Memory participate. (2) *Static eval* (150 episodes): the agent solves math tasks with the code-trained memory loaded but frozen (no new items added). This isolates the effect of cross-domain memory on math performance. (3) *Dynamic stream* (150 episodes): the agent solves math tasks while continuing to add new experiences to memory. Memory grows from $\sim$60 code items to $\sim$210 mixed items. This tests interference dynamics under continuous domain shift.

Grading. For MATH, we use exact-match grading: the model's final numerical or symbolic answer is extracted and compared to the ground truth after normalization (removing whitespace, simplifying fractions). For LiveCodeBench, we use pass@1: the model's generated code is executed against hidden test cases. An episode is graded as correct ($g_t = 1$) if the answer matches, and incorrect ($g_t = 0$) otherwise.

Concurrency and reproducibility. We run experiments at up to 200 concurrent API requests via asyncio, with deterministic seed-based shuffling of task order. All runs use temperature 0 to minimize stochasticity; the three seeds (0, 1, 2) control only the random task ordering and the random projection matrix W for sparse encoding. Total API cost was approximately $12 USD across all 4,950 episodes.

F Limitations and Future Work

Our interference estimates are coarse, as we label all retrieved items with the episode outcome rather than computing true leave-one-out effects. This "all-or-nothing" labeling means that when a retrieved set contains both helpful and harmful items, all items receive the same signal. More sophisticated credit assignment (e.g., leave-one-out reruns or attention-based attribution) could yield sharper interference estimates, though at significantly higher computational cost.

The streaming code$\rightarrow$math scenario, while controlled, represents a single type of domain shift. Real-world agents may encounter more gradual shifts, multiple simultaneous domains, or shifts that reverse (e.g., returning to a previously seen domain). Our sparse encoding uses a fixed random projection; learned pattern separation (e.g., training W on interference-labeled pairs via contrastive learning) could yield larger gains by adapting the separation to the specific domain structure.

We evaluate with a single LLM (Qwen-30B-A3B); generalization across model families and scales remains to be tested. Larger models may be more robust to irrelevant context, potentially reducing the magnitude of interference, while smaller models may be more susceptible. Promising future directions include interference-rate based domain-shift detection as an early-warning signal (H3 from our original hypothesis set), extending the AM analysis to graph-structured or multi-modal agent memories, and investigating whether the concentration patterns we observe scale predictably with memory size, following AM capacity laws.

G Additional Related Work

Retrieval can hurt. Mallen et al. (2023) show that retrieval flips sign by entity popularity in static QA. The Distracting Effect quantifies harm from irrelevant passages (Cuconasu et al., 2024), and Liu et al. (2024) demonstrate that models struggle to use information placed in the middle of long retrieved contexts. Salemi & Zamani (2024) propose per-item marginal utility evaluation. While RAG was originally shown to reduce hallucination in knowledge-intensive tasks (Lewis et al., 2020), subsequent work has revealed that irrelevant or contradictory retrieved passages can actively degrade performance. We adopt these *measurements* but study a different object: *dynamic, agent-generated* memory under domain shift, where the retrieved items were created by the agent itself in a different domain.

Agent memory management. The design space for agent memory is broad: Generative Agents (Park et al., 2023) introduced reflection-based memory streams for social simulation, while subsequent work has explored diverse architectures. FadeMem (Wei et al., 2026) implements biologically-inspired decay and conflict resolution. Agent KB (Tang et al., 2025) uses disagreement gating against harmful transfers. MCMA (Liang et al., 2026) learns abstraction-level control via a memory copilot. MemEvolve (Zhang et al., 2025) proposes meta-evolution of memory systems. ExpeL (Zhao et al., 2024) extracts reusable insights from past trajectories via experiential learning. CLIN (Majumder et al., 2023) maintains causal abstractions as persistent memory that can transfer across tasks and environments. Agent Workflow Memory (Wang et al., 2024) stores reusable workflows rather than raw trajectories. HippoRAG (Jimenez Gutierrez et al., 2024) draws on hippocampal indexing theory to organize long-term memory for retrieval. Our work differs from all of these in (i) explicitly grounding interference in AM theory, (ii) using interference as a *signal* (not just a failure mode), and (iii) evaluating pattern separation as a controlled, training-free intervention.

Associative memory and transformers. Classical AM theory established that unlearning stabilizes memory by removing spurious attractors (Hopfield et al., 1983), that error-free recall requires sufficient pattern separation (Kanter & Sompolinsky, 1987), and that networks with huge storage capacity can be constructed via polynomial interaction functions (Demircigil et al., 2017). The neuroscience of pattern separation in the hippocampus (Yassa & Stark, 2011) provides biological grounding for the sparse encoding strategy we adopt. Modern Hopfield networks (Ramsauer et al., 2021; Krotov & Hopfield, 2016) connect attention to AM energy landscapes. Bietti et al. (2023) analyze transformer training through an AM lens. Burns (2024) study semantically-correlated patterns in dense AM. Cabannes et al. (2023) derive scaling laws for AM capacity. We build on this theoretical foundation but test AM predictions in the *practical* setting of agent memory retrieval, where the "patterns" are natural-language experiences and the "retrieval" is prompt conditioning.