Retrieval Is Not Memory: A Cognitively-Inspired Architecture for Production AI Agent Memory Systems

Nicholas Zinner Beacon Bot

Future Shock (`future-shock.ai`)

March 2026

Abstract

AI agents deployed in sustained production environments face memory challenges absent from standard benchmarks. We present a cognitively-inspired three-layer memory architecture grounded in Complementary Learning Systems (CLS) theory, implemented and validated in a production AI newsroom agent over 30 days of continuous operation. The system comprises an episodic activity log (629 entries), a semantic knowledge base (108 extracted rules and procedures), and a co-occurrence graph (4,082 edges) supporting spreading activation retrieval. We evaluate the architecture with 80 human-designed test cases across four difficulty tiers, from standard lookup to multi-hop reasoning. We include embedding-based search (qmd vsearch) as a baseline: it scores worst at 23% top-1 accuracy with a 50% false positive rate (on the Original 30 benchmark), confirming that semantic similarity is not operational relevance. Legacy BM25 search reaches 40% with 37% false positives. Unified retrieval across all layers reaches 77% top-1 accuracy on standard queries and 55% overall (44/80), while cutting false positives to ˜10%. However, retrieval alone is insufficient: on queries requiring inference, contradiction resolution, or multi-document synthesis, retrieval accuracy drops to 20–30%. Adding an LLM reasoning layer reaches 81% overall (65/80), a +26 percentage point gain across all suites. The LLM delta grows with query difficulty: +13pp on standard queries, +15pp on medium queries, +45pp on hard queries, and +50pp on extreme multi-hop queries. We identify this retrieval-reasoning boundary as the architectural analog of the cognitive science distinction between hippocampal pattern completion and prefrontal reconstruction, and argue that the boundary—not the retrieval system—is the critical design surface for production agent memory.

1 Introduction

On March 15, 2026, an AI agent operating an automated newsroom was shown a screenshot of that morning's newsletter, a newsletter the agent itself had published seven hours earlier and promoted on social media at 8:30 AM. The agent did not recognize its own work. It offered to write a new article about the topic.

This failure occurred in a production system with 49 automated cron jobs, 629 logged activities across 30 days, and a semantic search index over 84 workspace files. The agent had published the newsletter, logged the publication, and promoted it—yet when queried about the topic, its memory system returned results for a *different* topic with high confidence. The semantic embedding search scored 0.5 on the relevant query, just below the threshold where an agent treats a result as a match.

The incident was not a retrieval bug in the conventional sense. The information existed in the agent's memory stores. The search system returned results. The failure was that the results

were *wrong*, and the system provided no signal that they were wrong. The agent exhibited what we term *false confidence*: a system behavior indistinguishable from confident certainty—returning results without signaling their potential incorrectness. In information retrieval terminology, this maps to low precision—the system returns results that appear relevant but are not. We use "false confidence" to emphasize the operational consequence: unlike a research benchmark where low precision merely reduces a score, in a production agent it leads to action on incorrect information.

Follow-up investigation revealed a 37% false positive rate in the legacy retrieval system (more than one in three queries returned confident but incorrect results). This rate is not anomalous; it is a structural property of embedding-based search over heterogeneous document collections. Embeddings capture semantic similarity but not operational relevance. A document about "Anthropic's context window expansion" is semantically similar to one about "Anthropic's government contracts," but operationally they are unrelated stories requiring distinct coverage decisions.

This paper presents a memory architecture designed to address these failures, grounded in cognitive science and validated in production. Our contributions are:

1. A three-layer architecture inspired by Complementary Learning Systems theory [1], separating episodic logging, semantic knowledge, and associative retrieval.
2. A unified retrieval system that searches across all layers simultaneously, with query intent classification, source authority weighting, document binding, and temporal filtering.
3. An 80-test evaluation suite across four difficulty tiers, designed by a human domain expert and executed blind against the system.
4. Identification of the **retrieval-reasoning boundary**: the point at which programmatic retrieval reaches its ceiling and LLM inference becomes necessary. We find this boundary at approximately 77% accuracy on standard queries and 30% on hard reasoning queries, with LLM reasoning adding +45–50 percentage points on the hardest queries and +26pp overall across all 80 tests.

2 Related Work

2.1 Cognitive Science Foundations

Our architecture draws on six established theories from cognitive science and neuroscience.

Complementary Learning Systems. McClelland, McNaughton, and O'Reilly [1] demonstrated that biological memory requires two systems with different learning rates: a hippocampal system for rapid, episodic encoding and a neocortical system for slow, statistical extraction of regularities. Kumaran, Hassabis, and McClelland [2] extended CLS explicitly to intelligent agents, arguing that any system requiring both rapid adaptation and stable long-term knowledge must maintain complementary fast and slow learning. CLS predicts that a monolithic memory store will exhibit catastrophic interference: new information overwrites old. Our three-layer architecture directly implements this separation.

Episodic and Semantic Memory. Tulving's [4] distinction between episodic memory (specific events bound to spatiotemporal context) and semantic memory (general knowledge stripped of context) maps onto the difference between an agent's activity log ("published newsletter at 5:30 AM on March 15") and its extracted knowledge ("always use `?source=html` when publishing to Ghost API"). The conversion from episodic to semantic is the consolidation process CLS describes.

Spreading Activation. Collins and Loftus [3] proposed that concepts are stored in a semantic network where activation spreads along weighted edges. When one concept is activated, related

concepts receive residual activation proportional to edge weight and inversely proportional to distance. This provides the theoretical basis for our co-occurrence graph: when a query activates the topic "Ghost," related topics like "newsletter," "publishing," and "slug" receive activation, broadening retrieval without requiring exact keyword matches.

Encoding Specificity. Tulving and Thomson [5] showed that retrieval succeeds when cues at retrieval match cues present at encoding. A memory encoded under "debugging the newsletter pipeline" may be inaccessible when searched via "email delivery issues," even though both describe the same event. This argues for multiple retrieval paths (keyword, semantic, temporal, and topical) rather than reliance on a single embedding space.

Levels of Processing. Craik and Lockhart [6] demonstrated that deeper processing at encoding produces more durable, retrievable memory traces. For agents, this implies that actively processing events into structured lessons ("the Ghost API rate-limits after 50 requests/minute") produces stronger memory than simply logging raw events ("API returned 429"). Our architecture implements this through the knowledge extraction layer, which transforms episodic entries into structured rules.

Forgetting Curve. Ebbinghaus [7] established that memory retention follows exponential decay without rehearsal. In agent memory, unbounded accumulation degrades retrieval quality as the store becomes noisy. Our nightly consolidation process applies decay (0.95/night with a 0.1 floor) to co-occurrence edge weights, ensuring that recent associations are prioritized while old associations fade gracefully rather than disappearing.

2.2 AI Agent Memory Systems

The evolution of AI memory architectures shows a field converging toward cognitively-inspired designs, often independently rediscovering principles from the cognitive science literature.

Park et al. [8] introduced Generative Agents with a memory architecture comprising observation, reflection, and planning streams. Their retrieval function combines recency, importance, and relevance scores, a design that parallels CLS's dual-system approach, with observations serving as episodic traces and reflections as consolidated semantic knowledge. However, their evaluation was limited to a simulated environment ("Smallville"), with no production validation.

Packer et al. [9] addressed the context window limitation directly with MemGPT, which gives agents explicit control over memory management through function calls that move information between a main context (working memory) and external storage (long-term memory). This matters because it makes the working-memory/long-term-memory boundary programmable. However, MemGPT's evaluation focused on document QA and conversational tasks, not sustained operational deployment.

Shinn et al. [10] introduced a self-reflective architecture where agents maintain an episodic memory buffer of past trajectories and generate verbal reflections to guide future behavior. Their key insight, that natural language reflections outperform raw trajectory replay, aligns with levels of processing theory: deeper, more elaborated encoding produces more useful memories.

Lewis et al. [11] established Retrieval-Augmented Generation (RAG) as the dominant paradigm for connecting language models to external knowledge. While RAG addresses the knowledge staleness problem, standard RAG implementations treat retrieval as a solved problem: retrieve relevant documents and inject them into the prompt. Our findings suggest this assumption is flawed: retrieval returns results with high confidence regardless of correctness, creating the false confidence problem that motivated this work.

Sumers et al. [13] proposed CoALA (Cognitive Architectures for Language Agents), a framework that explicitly maps cognitive science constructs—episodic memory, semantic memory, and proce-

dural knowledge—onto agent architectural components. CoALA provides a theoretical taxonomy for classifying agent memory designs. Our work complements CoALA by providing production-validated empirical data on how these cognitive layers perform under sustained deployment, including failure rates and the retrieval-reasoning boundary that CoALA's framework predicts but does not quantify.

2.3 The Production Gap

What these systems share is the absence of production validation. Generative Agents were evaluated in simulation. MemGPT was benchmarked on document QA. Reflexion was tested on coding and reasoning benchmarks. None have been evaluated under conditions of sustained deployment where failure modes compound over weeks, where the agent must retrieve information it generated hours ago, and where false positive retrieval has real operational consequences (publishing wrong information, duplicating work, missing coverage). Our work addresses this gap.

3 Architecture

3.1 Design Overview

The architecture comprises three layers, each corresponding to a component of CLS theory, plus a unified retrieval system and an LLM reasoning layer (Figure 1).

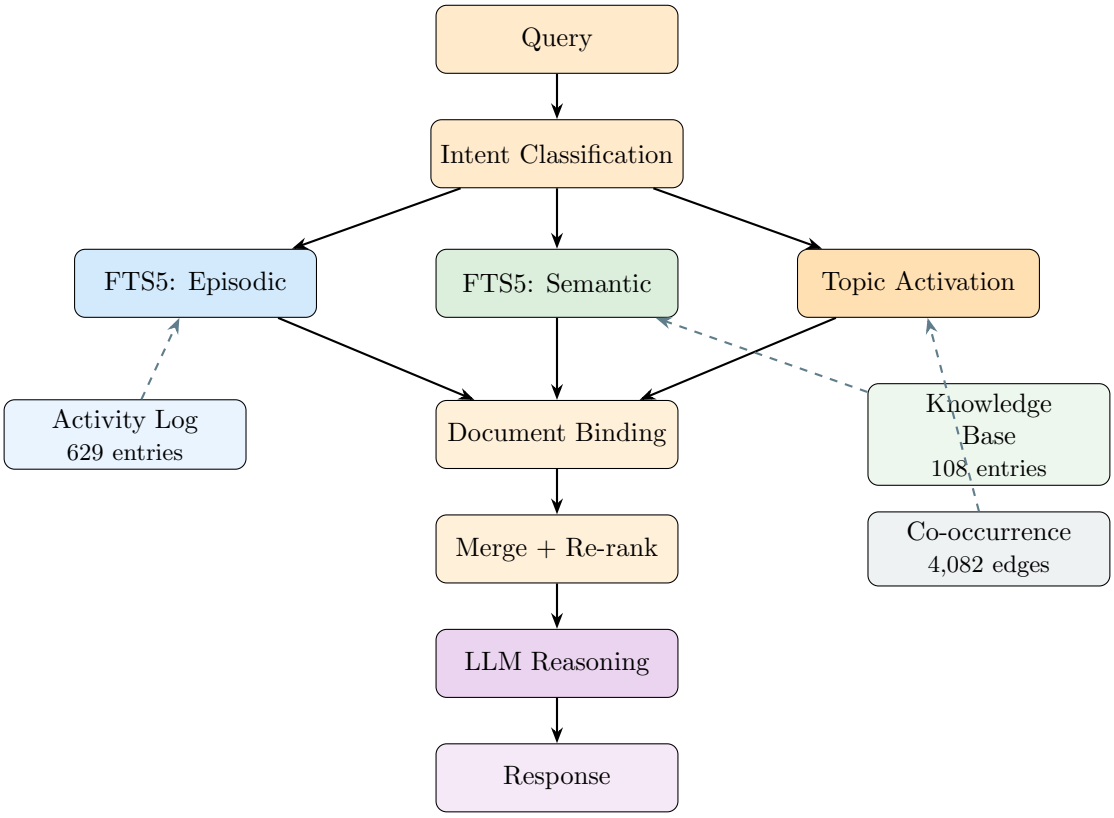

Figure 1: Unified retrieval architecture. Queries are classified by intent, routed to parallel FTS5 indexes over episodic and semantic stores plus topic activation on the co-occurrence graph, then merged via document binding and re-ranked before passing to the LLM reasoning layer.

3.2 Layer 1: Episodic Memory (Activity Log)

The episodic layer is a SQLite database with FTS5 full-text indexing, storing 629 timestamped activity entries accumulated over 30 days of continuous operation. Each entry records an action type (e.g., `publish_newsletter`, `ingest_news`, `draft_created`), a natural-language description, associated topic tags, and optional artifact URLs. This layer corresponds to hippocampal fast encoding in CLS: every significant agent action is recorded immediately with full spatiotemporal context.

Entries are created automatically by the agent's operational crons and by manual logging during interactive sessions. The action type taxonomy comprises 13 categories covering publishing, social media, ingestion, drafting, and system operations.

3.3 Layer 2: Associative Network (Co-occurrence Graph)

The co-occurrence graph captures relationships between 362 canonical topics derived from a normalization layer that maps raw terms to standardized forms (e.g., "GPT-4," "gpt4," "GPT4" $\rightarrow$ `gpt-4`). When two topics co-occur in the same activity entry, their co-occurrence edge weight is incremented. The resulting graph contains 4,082 edges.

Retrieval via spreading activation uses a recursive Common Table Expression (CTE) in SQLite, propagating activation from seed topics outward to a maximum depth of 3 hops with a minimum activation threshold of 0.05. This implements Collins and Loftus's [3] spreading activation model within the constraints of a relational database. This implementation captures only the excitatory component of Collins and Loftus's model; the original theory also includes inhibitory links between competing concepts. In our architecture, inhibitory control is delegated to the LLM reasoning layer (Section 5).

Edge weights decay nightly at a rate of 0.95 with a floor of 0.1, approximating the decay dynamics described by Ebbinghaus [7]. This ensures that recently co-occurring topics are prioritized while historical associations remain accessible at reduced strength.

3.4 Layer 3: Semantic Memory (Knowledge Base)

The semantic layer stores 108 structured knowledge entries extracted from three sources: operational lessons (71 entries from `lessons.md`, each containing a mistake description and a preventive rule), tool documentation (20 entries from `TOOLS.md`), and skill descriptions (17 entries from skill definition files). Each entry is indexed with FTS5 for full-text retrieval.

This layer corresponds to neocortical slow learning in CLS. Knowledge entries are not raw events but processed, abstracted rules (the product of consolidation). A lesson like "Ghost API requires `?source=html` parameter" is the semantic residue of an episodic memory ("newsletter published with empty body on March 12").

3.5 Consolidation Process

A nightly consolidation cron (3:00 AM) performs three operations: (1) decay of co-occurrence edge weights, (2) regeneration of topic summaries from updated activity data, and (3) FTS5 index rebuild. This directly implements the CLS prediction that consolidation is an active process: hippocampal traces are replayed to update neocortical representations. In biological terms, this is the sleep replay that Diekelmann and Born [12] showed is essential for memory stability.

3.6 Unified Retrieval

The unified retrieval system queries all three layers simultaneously through a single interface:

1. **Query intent classification**: Simple keyword matching routes queries to appropriate source weights. Queries containing "rule/should/policy/never/always" upweight the knowledge layer 2×; queries containing "when/did we/today/recent" upweight the activity log 2× with recency boost.
2. **Parallel FTS5 search**: Both the activity log and knowledge base are queried via FTS5, returning ranked results with SQLite's built-in BM25 scoring.
3. **Topic activation**: Query terms are matched against the topic normalization layer. If canonical topics are found, spreading activation retrieves associated topics and their summaries.
4. **Document binding**: A document co-occurrence table (4,533 pairs) links entries that share 2+ topics. When a top-5 result is found, related entries with bind strength ≥ 3 are added to the candidate set, tagged as `[bound]` results.
5. **Date-range filtering**: Temporal terms in queries ("last 7 days," "since March 1") are parsed and applied as SQL WHERE clauses before ranking.
6. **Merge and re-rank**: Results from all sources are scored by:

$$\text{score}_{\text{final}} = |\text{FTS5\_rank}| \times w_{\text{source}} \times r_{\text{recency}} \times a_{\text{authority}} \times t_{\text{tag}} \tag{1}$$

where |FTS5_rank| is the absolute value of SQLite's BM25 output (FTS5 returns negative scores where more negative indicates higher relevance), w_{source} is the intent-derived source weight (1.0 or 2.0), $r_{\text{recency}} = 1.0 + \min(0.3, 0.3 \times d_{\text{fresh}}/7)$ where $d_{\text{fresh}} = \max(0, 7 - d_{\text{age}})$ and d_{age} is the number of days since the entry was created (a document created today ($d_{\text{age}} = 0$) receives the maximum boost ($r_{\text{recency}} = 1.3$); a document older than 7 days receives no boost ($r_{\text{recency}} = 1.0$)), $a_{\text{authority}}$ reflects source type (lessons: 1.5, tools: 1.3, skills: 1.2, activity: 1.0), and t_{tag} is a tag-match boost (2.0 if detected tag filters match result tags, 1.0 otherwise).

Source and authority weights were set by manual heuristic based on the operational observation that lessons and tool documentation are more frequently the correct answer for rule-based queries than raw activity entries. These weights were not tuned on the evaluation test suites.

3.7 Implementation

The entire system is implemented in 639 lines of Node.js with no external dependencies, using SQLite via the command-line interface. All data resides in a single database file (activity-log.db). The co-occurrence graph, spreading activation, topic normalization, and document binding all operate through SQL queries, including recursive CTEs for graph traversal. Average retrieval latency is 60–70ms.

The LLM reasoning layer uses Claude Sonnet 4 (Anthropic) for production queries. Evaluation of the retrieval+LLM condition also used Claude Sonnet 4 as the reasoning model. The agent's primary operational model is Claude Opus 4. Results may vary with different LLMs; the reasoning layer's effectiveness likely depends on the model's instruction-following capability and context window size.

4 Evaluation

4.1 Methodology

We evaluate the architecture with 80 test cases across four suites of increasing difficulty, all designed by a human domain expert (first author) who operates the production system daily. Tests were designed to target specific failure modes observed in production, not to maximize scores.

Table 1: Evaluation suite summary

Suite	Tests	Design Phase	Difficulty
Original	30	Pre-build	Standard lookup
Blind v1	20	Post-build, unseen	Medium (ambiguity, ranking)
Blind v2	20	Post-fixes, unseen	Hard (reasoning required)
Extreme	10	Full-stack test	Extreme (multi-hop, synthesis)

Each query is evaluated on two primary metrics: top-1 accuracy (correct result ranked first) and false positive rate (confident but incorrect result returned).

The **Original 30** tests were designed before the unified retrieval system was built, targeting known embedding failures from production incidents (e.g., the newsletter recognition failure). The **Blind v1** and **Blind v2** tests were designed after the system was built but never shown to the system or used in development. The **Extreme 10** tests evaluate the full stack including LLM reasoning.

4.2 Results

Table 2: Top-1 accuracy and false positive rates across systems and test suites

System	Original 30	Blind v1	Blind v2	Extreme 10	All 80
Top-1 Accuracy					
Embeddings (vsearch)	23%	—	—	—	—
Legacy (BM25)	40%	—	—	—	—
Unified Retrieval	77% [59,89]	65% [43,82]	30% [14,52]	20% [6,51]	55%
Retrieval + LLM	**90%** [74,97]	**80%**	**75%**	**70%**	**81%** [71,89]
False Positive Rate					
Embeddings (vsearch)	50%	—	—	—	—
Legacy (BM25)	37%	—	—	—	—
Unified Retrieval	10%	10%	20%	—	˜10%
Retrieval + LLM	**0%**	**0%**	**0%**	**0%**	**˜0%**

95% Wilson binomial confidence intervals shown in brackets for key comparisons. The primary comparison (Retrieval+LLM 65/80 vs. Unified 44/80) is significant ($\chi^2 = 9.5$, $p < 0.01$).

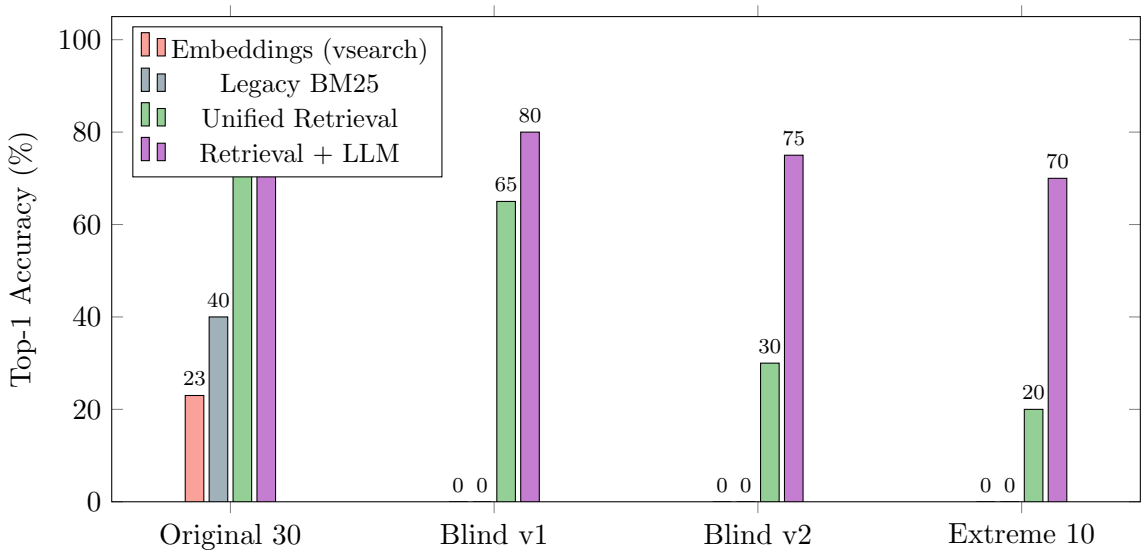

Figure 2: Top-1 accuracy across evaluation suites and systems. Embedding-based search (red) scores worst at 23% on standard queries and was not evaluated on blind suites. Legacy BM25 (gray) reaches 40% but degrades. Unified retrieval (green) improves substantially but drops sharply on reasoning-intensive queries. Adding an LLM reasoning layer (purple) improves all suites, with the largest gains on hard queries. Zero-height bars indicate systems not tested on that suite.

Three findings emerge from the evaluation:

Finding 1: Unified multi-layer retrieval substantially outperforms single-source search. On the Original 30 tests, top-1 accuracy improved from 40% (BM25) to 77% (23/30) and false positives dropped from 37% to 10%. Embedding-based search (qmd vsearch) scored worst at 23% top-1 with a 50% false positive rate, confirming that semantic similarity is not operational relevance. A document about "prompt injection attacks" scores nearly identically to a document about "prompt injection screening tools," but only the second is the correct answer to a query about our content screener. The improvement from BM25 to unified retrieval comes from searching episodic and semantic stores simultaneously: the activity log answers "what did we do?" while the knowledge base answers "what's the rule?" Neither alone is sufficient.

Finding 2: The architecture generalizes to unseen queries with reasonable degradation. Blind v1 (65%) represents a 12-point drop from the Original 30 (77%), indicating that the system was not heavily overfit to the test cases it was designed around. This generalization gap is within normal bounds for information retrieval systems.

Finding 3: The LLM reasoning layer improves all suites, with gains that grow with query difficulty. Adding an LLM reasoning layer on top of retrieval improves accuracy across every test suite: +13pp on Original 30 (77% $\rightarrow$ 90%), +15pp on Blind v1 (65% $\rightarrow$ 80%), +45pp on Blind v2 (30% $\rightarrow$ 75%), and +50pp on Extreme (20% $\rightarrow$ 70%). This is not a specialized fix for hard queries—it is a consistent improvement at every difficulty level. The delta growing with difficulty reflects the nature of what retrieval cannot do: contradiction detection, multi-hop reasoning, negation handling, and state transition inference all scale with query complexity. Blind v2 accuracy without LLM (30%) represents a 35-point drop from Blind v1 (65%), marking the point where retrieval alone reaches its structural ceiling.

4.3 Failure Taxonomy

We classify failures across all 80 tests into four categories:

Table 3: Failure taxonomy across all evaluation suites

Category	Count	%	Description
Content gap	12	27%	Answer exists but not indexed (research files, external knowledge)
Ranking issue	5	11%	Correct result in set but not at rank 1
Capability gap	8	18%	Feature not built (event fingerprinting, approval tracking)
Reasoning gap	**20**	**44%**	Requires inference across documents

Note: Failure modes are classified across all unified retrieval-only tests. Some tests exhibit multiple failure modes; totals exceed the unique failure count of 36.

Reasoning gaps dominate at 44%. These are not failures of indexing, ranking, or coverage. They cannot be resolved by improving the retrieval system; they require a different kind of processing entirely.

For instance, a content gap failure: the query "Yann LeCun current role" returns no indexed result because external personnel changes are not tracked in the activity log. A reasoning gap failure: the query "lesson prevent both duplicate sends AND partial corrections" requires connecting two separate knowledge entries (the triple-send incident and the Atlassian correction) into a unified rule—each entry exists but the connection requires cross-document inference.

5 The Retrieval-Reasoning Boundary

The evaluation reveals a sharp boundary between what programmatic retrieval can solve and what requires LLM inference. We characterize this boundary through the cognitive science lens that motivated the architecture.

5.1 What Retrieval Handles

Programmatic retrieval (FTS5 full-text search, query intent classification, source authority weighting, document binding, and temporal filtering) handles queries where the answer exists in a single document or can be assembled from co-occurring documents with matching keywords. FTS5 full-text search performs pattern matching—it requires lexical overlap between query terms and indexed content. The co-occurrence spreading activation is closer to what Collins and Loftus [3] describe as pattern completion: a partial cue activates associated patterns through the semantic network, broadening retrieval beyond exact lexical matches.

Retrieval excels at:

- **Single-document lookup**: "What's the Git author email format?" → exact match in knowledge base.
- **Temporal filtering**: "What changed in the last 7 days?" → date-range SQL filter.
- **Rule retrieval**: "Where should I post this tech link?" → surfaces the r/technology ban rule.
- **Confidence estimation**: Low-scoring results are flagged, preventing false confidence.

5.2 What Embeddings Get Wrong

Before characterizing what LLM reasoning handles, it is worth examining why embedding-based search performed worst in our evaluation (23% top-1, 50% false positives). The failure is not a calibration problem or a threshold problem—it is structural.

Embedding models encode semantic proximity: documents that discuss similar topics cluster in the same region of the vector space. This works well for open-domain question answering over a homogeneous corpus. It breaks down when the query is operationally specific: the agent needs a tool, a procedure, or a rule, not a document that discusses the same topic.

In our evaluation, queries like "find our prompt injection screening tool" return articles about prompt injection attacks with cosine similarity scores of 0.60–0.73—close enough that the system treats them as matches. The scores for the correct document (the screener's own documentation) fall in the same range. Embeddings provide no separation between the relevant result and the related-but-wrong result, because both are genuinely semantically proximate.

A second failure mode is directional: "Anthropic's context window expansion" and "Anthropic's government contracts" both embed near the cluster for Anthropic news. A query about coverage decisions for one will surface the other with high confidence. BM25 keyword search at least requires lexical overlap; embeddings require only topical proximity.

The 50% false positive rate in embedding search means that half of all confident results were wrong. An agent operating on those results would act on wrong information half the time it trusted the retrieval system. This is worse than returning no results, because no results signal uncertainty while a confident wrong answer signals certainty.

The embedding model used here (300M parameters) is relatively small, and larger models might score better on some queries. However, the structural argument—that semantic proximity is not operational relevance—would likely hold regardless of model size, since it reflects a property of the task rather than of the encoder.

5.3 What Requires LLM Reasoning

The LLM reasoning layer[1] handles five classes of queries that retrieval cannot:

Inhibition (suppressing irrelevant activations). When a query asks for "MoltBook verifier tool, NOT the verification challenge," retrieval activates all documents matching "MoltBook" + "verifier" + "verification." It cannot suppress the challenge-related results, because FTS5 has no concept of negation context. The LLM reads the results and ignores the irrelevant ones, performing the top-down inhibitory control that cognitive science attributes to the prefrontal cortex.

Binding (connecting separate traces). Multi-hop queries like "What lesson prevents both duplicate sends and partial corrections?" require connecting two separate knowledge entries. Document binding ensures both entries appear in the candidate set (by following co-occurrence edges), but the *connection* (the inference that both share a common pattern: "verify before acting") requires the LLM. In biological memory, binding is a hippocampal function coordinated by theta-rhythm oscillations; in our architecture, it is an LLM function operating over retrieval results.

Schema-driven reconstruction. Queries about workflows ("What changes when a draft moves to publish-ready?") require assembling a coherent sequence from fragments scattered across multiple lessons. No single document contains the complete workflow. The LLM reconstructs the

[1]We acknowledge a structural irony: the solution to the retrieval system's memory limitations is a larger, more opaque memory system (the LLM's parametric knowledge). This mirrors the biological architecture—prefrontal cortex draws on its own distributed representations to control hippocampal retrieval—but the parallel should not obscure the fact that we have deferred, rather than solved, the hardest memory problems.

schema from retrieved pieces, paralleling the cognitive process where humans use general knowledge structures to fill gaps in episodic recall.

Metamemory (knowing what you don't know). When retrieval returns low-confidence results for "Which documents are marked internal-only?" the LLM correctly concludes: "Our system has no access control metadata—this capability doesn't exist." Retrieval can signal low confidence; only the LLM can distinguish between "the answer isn't indexed" and "the capability doesn't exist."

Contradiction resolution. When two rules conflict (an old retry policy and a newer one that supersedes it), retrieval may return both without indicating which is current. The LLM examines dates and content to determine precedence.

5.4 The Cognitive Science Mapping

This boundary maps precisely to the distinction between two cognitive systems:

Table 4: Mapping between cognitive processes and architectural layers. Mappings are heuristic guides for system design, not claims of functional equivalence between biological and computational systems.

Cognitive Process	Brain Region	Our Architecture
Fast pattern completion	Hippocampus	FTS5 + topic activation
Associative priming	Lateral temporal cortex	Co-occurrence graph
Memory consolidation	Sleep replay	Nightly cron (3 AM)
Forgetting/decay	Synaptic weakening	Weight decay (0.95/night)
Inhibitory control	*Prefrontal cortex*	*LLM reasoning*
Episodic binding	*Hippocampus (theta)*	*LLM reasoning*
Schema reconstruction	*Prefrontal cortex*	*LLM reasoning*
Metamemory	*Frontal lobe*	*LLM reasoning*

Human memory is as much about *filtering and suppressing* as it is about *storing and retrieving*. Our retrieval system implements the excitatory side of memory: activation, association, priming. The LLM provides the inhibitory side: suppression, selection, focus. Both are necessary; neither alone is sufficient.

6 Lessons from Production

Thirty days of continuous operation revealed failure modes absent from standard benchmarks.

6.1 False Confidence Is the Operational Killer

Embedding-based retrieval doesn't just fail—it fails *confidently*. A search that returns no results is safe; a search that returns plausible but wrong results leads to action on incorrect information. Our legacy system's 37% false positive rate meant the agent was acting on wrong information more than once in every three queries. The unified retrieval system reduced this to 10%, primarily through source authority weighting and confidence thresholds that flag low-scoring results.

6.2 Silent Failures Compound Over Weeks

In sustained deployment, a single retrieval failure compounds: the agent publishes duplicate coverage, which creates noise in the activity log, which degrades future retrieval quality. We observed this pattern with newsletter content: the agent published a topic, failed to retrieve the publication record, and offered to write about the same topic again. Without the triggering incident that exposed this cycle, it would have continued indefinitely.

6.3 The Simplest Interventions Are the Most Effective

A flat markdown file (`lessons.md`) with structured entries (each containing a mistake description and a preventive rule), read by the agent on startup, prevented more repeat failures than any architectural improvement. This is consistent with levels of processing theory [6]: the act of explicitly writing "what went wrong" and "the rule that prevents this" constitutes deep, elaborative encoding that produces highly retrievable memories.

6.4 Nightly Consolidation as Biological Sleep

The 3 AM consolidation cron (decay, summary generation, index rebuild) proved essential for retrieval quality. Without decay, the co-occurrence graph accumulated noise from transient topic co-occurrences, diluting the signal from meaningful associations. Without summary regeneration, topic contexts became stale. This is consistent with CLS's prediction that consolidation is not optional maintenance but an architectural necessity.

7 Limitations and Future Work

Single agent, single domain. Our evaluation is based on one agent operating in one domain (AI newsroom). The architecture's generalizability to other domains (coding assistants, customer service, research agents) requires further validation.

Human-designed test suites. All 80 test cases were designed by the system's operator. While the blind suites were not used during development, they were still designed by the system's operator, someone who knows the system's strengths well. This introduces evaluator bias: the test suite likely over-represents failure modes the operator anticipated and under-represents surprising ones. Adversarial or automatically generated test suites could reveal additional weaknesses.

Embedding model size. The embedding baseline uses a 300M-parameter model (qmd vsearch). Larger embedding models might achieve higher retrieval scores on some queries. The structural argument—that semantic similarity is not operational relevance—would likely hold regardless of model scale, since it reflects the nature of the task rather than encoder capacity. But this remains an open empirical question.

SQLite scale limitations. The current implementation handles $\sim$700 episodic entries and $\sim$100 knowledge entries efficiently. At 10,000+ entries, the document binding step (which computes pairwise co-occurrence) would require optimization or migration to a more scalable store.

Future work. Three architectural additions could address the remaining 30% failure rate on extreme queries: (1) a programmatic inhibition layer that parses negation from queries and demotes matching results before LLM processing; (2) a schema template library for common workflows (publishing, correction, deployment) that structures retrieval results before LLM synthesis; and (3) automated test generation to continuously probe the retrieval-reasoning boundary as the system evolves.

7.1 Threats to Validity

Three methodological limitations constrain these findings. First, all 80 test cases were designed by the system's operator. While the blind suites were not used during development, evaluator familiarity with the system may systematically bias the test distribution toward anticipated failure modes. Second, the embedding baseline uses a single 300M-parameter model; larger embedding models may narrow the performance gap, though the structural argument (semantic proximity $\neq$ operational relevance) would likely persist. Third, the evaluation uses a single human judge for correctness assessment; inter-rater reliability is unknown.

8 Conclusion

Retrieval is not memory. Search finds documents; memory reconstructs knowledge. This distinction, well-established in cognitive science, has been largely overlooked in AI agent memory design.

We presented a three-layer architecture grounded in Complementary Learning Systems theory that separates episodic logging, semantic knowledge extraction, and associative retrieval, unified through a multi-source retrieval system with document binding and temporal filtering. Evaluated across 80 human-designed test cases, the architecture improves retrieval accuracy from 23% (embeddings, 50% false positives) and 40% (BM25, 37% false positives) to 77% on standard queries and 55% overall. Embeddings performing worst in this evaluation is not a surprising result—it is the expected consequence of optimizing for semantic proximity when the task requires operational relevance.

The more important finding is the **retrieval-reasoning boundary**: the point at which programmatic search reaches its structural ceiling and LLM inference becomes necessary. On reasoning-intensive queries, retrieval accuracy drops to 20–30%. Adding an LLM reasoning layer (which provides inhibitory control, cross-document binding, schema reconstruction, and metamemory) reaches 81% overall (65/80), a +26 percentage point improvement across all suites and +45–50pp on the hardest queries.

This boundary maps to the cognitive science distinction between hippocampal pattern completion (fast, associative, content-addressable) and prefrontal reconstruction (slow, deliberate, schema-driven). Human memory depends on both; so does effective agent memory. The 23% embedding baseline with 50% false positives shows the problem is not "we need better search"—it is structural. Investing exclusively in better retrieval systems yields diminishing returns. The design surface that matters is the interface between retrieval and reasoning: ensuring that the retrieval layer provides the right candidate set for the reasoning layer to operate on.

Production deployment is arguably the most stringent test of agent memory systems. Benchmarks miss the failure modes that matter most: false confidence, silent compounding errors, weeks of slow drift. The field would benefit from moving toward production-validated architectures alongside simulated evaluations.

References

[1] McClelland, J. L., McNaughton, B. L., & O'Reilly, R. C. (1995). 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), 419–457. DOI: 10.1037/0033-295X.102.3.419

[2] Kumaran, D., Hassabis, D., & McClelland, J. L. (2016). What learning systems do intelligent agents need? Complementary learning systems theory updated. *Trends in Cognitive Sciences*, 20(7), 512–534. DOI: 10.1016/j.tics.2016.05.004

[3] Collins, A. M., & Loftus, E. F. (1975). A spreading-activation theory of semantic processing. *Psychological Review*, 82(6), 407–428. DOI: 10.1037/0033-295X.82.6.407

[4] Tulving, E. (1972). Episodic and semantic memory. In E. Tulving & W. Donaldson (Eds.), *Organization of Memory* (pp. 381–403). Academic Press.

[5] Tulving, E., & Thomson, D. M. (1973). Encoding specificity and retrieval processes in episodic memory. *Psychological Review*, 80(5), 352–373. DOI: 10.1037/h0020071

[6] Craik, F. I. M., & Lockhart, R. S. (1972). Levels of processing: A framework for memory research. *Journal of Verbal Learning and Verbal Behavior*, 11(6), 671–684. DOI: 10.1016/S0022-5371(72)80001-X

[7] Ebbinghaus, H. (1885/1913). *Memory: A Contribution to Experimental Psychology.* Trans. H. A. Ruger & C. E. Bussenius. Teachers College, Columbia University.

[8] Park, J. S., O'Brien, J. C., Cai, C. J., Morris, M. R., Liang, P., & Bernstein, M. S. (2023). Generative agents: Interactive simulacra of human behavior. *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology (UIST '23).* DOI: 10.1145/3586183.3606763

[9] Packer, C., Wooders, S., Lin, K., Fang, V., Patil, S. G., Stoica, I., & Gonzalez, J. E. (2023). MemGPT: Towards LLMs as operating systems. *Advances in Neural Information Processing Systems (NeurIPS '23).*

[10] Shinn, N., Cassano, F., Gopinath, A., Narasimhan, K., & Yao, S. (2023). Reflexion: Language agents with verbal reinforcement learning. *Advances in Neural Information Processing Systems (NeurIPS '23).*

[11] Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W., Rocktäschel, T., Riedel, S., & Kiela, D. (2020). Retrieval-augmented generation for knowledge-intensive NLP tasks. *Advances in Neural Information Processing Systems (NeurIPS '20).*

[12] Diekelmann, S., & Born, J. (2010). The memory function of sleep. *Nature Reviews Neuroscience*, 11(2), 114–126. DOI: 10.1038/nrn2762

[13] Sumers, T. R., Yao, S., Narasimhan, K., & Griffiths, T. L. (2023). Cognitive architectures for language agents. *arXiv preprint arXiv:2309.02427.*