DeepCQ: A Hierarchical Memory and Intent-Aware Context Optimisation Architecture for Multi-Turn Investment Research Agents

Jan Daniel Semrau
Ternary Capital Research
research@ternary.capital

April 2026

ABSTRACT

We present **DeepCQ**, an agentic information retrieval system for multi-turn investment research that addresses a fundamental challenge in LLM-based IR: the tension between comprehensive context and effective attention allocation. DeepCQ introduces two novel mechanisms: (1) a *five-tier hierarchical memory architecture* that partitions conversational, analytical, and domain context into access-policy-distinct layers, and (2) *intent-aware RegEx-driven context filtering* that classifies user queries into 12 semantic intent categories and extracts only relevant context sections before LLM inference. Evaluation on 1,000 real research queries demonstrates 40–70% token reduction (mean 57.6%) with a significant improvement in relevance scores (+0.6 on a 5-point scale). The intent classifier achieves 99.2% accuracy with a false-negative rate of 0.8%. The architecture supports a 60-tool ReAct agent operating over financial data from multiple APIs, persistent per-ticker thesis documents, and dynamic portfolio state, all within practical latency bounds. Our findings suggest that structural, rule-based filtering remains competitive with, and complementary to, learned approaches for well-defined domain schemas.

CCS Concepts: Information systems → Information retrieval; Question answering · Computing methodologies → Natural language processing.

Keywords: information retrieval agents, context management, intent detection, hierarchical memory, token efficiency, financial IR, LLM cost optimisation.

1. Introduction

The deployment of Large Language Models (LLMs) as retrieval and reasoning agents introduces a structural problem that has received limited treatment in the IR literature: *context window saturation*. Unlike conventional retrieval systems that return ranked document lists, agentic IR systems accumulate context across multiple turns, tool call results, conversation history, domain knowledge, and persistent analytical artefacts (Zhang et al., 2024; Pezzuti et al., 2026). As accumulated context grows, model attention becomes diluted across an increasingly noisy input, reducing retrieval precision and reasoning quality.

This problem is acute in financial analysis, where a single research session may involve dozens of tool calls spanning financial statements, news feeds, SEC filings, insider transaction records, and technical indicators. A naive pipeline that injects all available context into every LLM call wastes tokens, degrades quality, and incurs prohibitive API costs at scale.

We address these challenges with **DeepCQ** (Deep Context Query), a production agentic IR system for multi-turn investment research. DeepCQ makes two primary contributions:

1. A **five-tier hierarchical memory architecture** that assigns distinct access policies, persistence semantics, and update frequencies to different classes of context, preventing conflation of ephemeral session state with long-lived analytical knowledge (Wang et al., 2026; Qiao et al., 2026; Hu et al., 2025).

2. An **intent-aware, RegEx-driven context filtering pipeline** that classifies user queries into 12 semantic intent categories and extracts only sections of context relevant to the current information need (Bhardwaj, 2026).

Together, these mechanisms reduce input token counts by 40–70% across diverse query types while improving response relevance scores.

2. Background and Related Work

2.1. Agentic Information Retrieval

The notion of an IR agent, one that autonomously issues queries, processes results, and refines its information-seeking strategy, has a long history (Voorhees, 2001; Salton and Buckley, 1990). Contemporary LLM-based agents extend this paradigm with tool-use capabilities: a model can invoke APIs, execute code, parse documents, and synthesise findings across multiple retrieval steps (Qin et al., 2023). ReAct-style agents (Yao et al., 2023) achieve strong performance on open-domain QA and multi-hop retrieval tasks by interleaving *reasoning* and *acting* steps.

Pezzuti et al. (2026) examine how agentic behaviour changes retrieval workloads, arguing that caching, query pre-processing, and standard evaluation metrics break down when agents, rather than humans, issue queries. DeepCQ addresses the further under-studied *multi-session* setting, where context accumulates across interactions and requires active management.

2.2. Context Window Management and Memory Architectures

RAG (Lewis et al., 2020) reduces context size by retrieving only relevant chunks before generation. Prompt compression methods (Jiang et al., 2023) distil lengthy contexts into compact representations using learned models. Memory-augmented architectures partition in-context state from external storage. **MemGPT** (Packer et al., 2023) pioneered the LLM-as-OS paradigm. **Mem0** (Chhikara et al., 2025) frames memory as production-ready infrastructure with graph-based storage. **A-MEM** (Xu et al., 2025) introduces Zettelkasten-inspired dynamic indexing. **AgeMem** (Yu et al., 2026) unifies long- and short-term memory management inside the agent's policy through three-stage reinforcement learning.

Three April 2026 preprints directly inform DeepCQ's design. **MemMachine** (Wang et al., 2026) separates short-term, episodic, and profile memory in a ground-truth-preserving store, achieving $\sim$80% fewer input tokens than Mem0 on LongMemEval. **SuperLocalMemory V3.3** (Bhardwaj, 2026) demonstrates that information-geometric retrieval can replace cloud LLM calls for core memory operations. **Memory Intelligence Agent** (Qiao et al., 2026) frames memory as a first-class intelligence capability in multi-agent systems. Comprehensive surveys are provided by Zhang et al. (2024) and Hu et al. (2025).

DeepCQ differs from the above in targeting *structured* domain context, financial data with predictable schemas, and in addressing multi-tier context with heterogeneous update semantics.

2.3. Intent-Aware Query Processing and Financial IR

Query intent classification has a long history in enterprise and web search (Broder, 2002). Schema-aware structured retrieval (Ma et al., 2026) demonstrates that exploiting domain schema yields more reliable tool invocation than unconstrained generation, a principle we adopt for context extraction. Structured IR over financial data includes earnings QA (Zhu et al., 2021) and LLM-based financial agents (Yang et al., 2023).

3. System Architecture

DeepCQ consists of four integrated subsystems: a **sensor network** for real-time data ingestion, a **five-tier memory architecture** for context management, a **ReAct agent** with 60+ domain tools, and an **intent-aware context optimiser**.

3.1. Sensor Network and Data Ingestion

Seven sensors are operational: a *market scan sensor* detecting unusual volume stocks; a *news sensor* aggregating articles from multiple sources; an *earnings calendar sensor*; a *congressional trading sensor* (Senate stock disclosures); a *Yahoo Finance trending sensor*; a *stock screener sensor* (undervalued growth, undervalued large-caps); and an *index quote sensor*.

Sensors are partitioned into *real-time sensors*, invoked synchronously on every API call, and *task-backed sensors* executed asynchronously. A `SleepCycleScheduler` runs three independent loops: the market scan loop at 05:00 local time (Tue–Sat), the sensor refresh loop at 16:15 ET (Mon–Fri, post-market close), and a metadata validation loop every five minutes. Thread safety is maintained through atomic file writes (`os.replace()`) and a threading lock serialising concurrent task completions.

3.2. Five-Tier Hierarchical Memory Architecture

We partition the agent's runtime context into five tiers with distinct access policies and persistence semantics. Table 1 summarises the design.

Table 1: Five-tier memory architecture.

Tier	Context block	Update policy	Tokens
T1	`<stock_context>`	Every turn (R/O)	2k–5k
T2	`<acquired_data>`	Append on tool call	0.5k–3k
T3	`<chat_history>`	Sliding 6-turn window	1k–2k
T4	`<current_thesis>`	Append-only (R/W)	1k–10k
T5	`<current_portfolio>`	Lazy-loaded	0.5k–2k

Tier 1 (Live Market State). assembled fresh on every agent turn from the sensor network and cached API calls. The agent checks this tier *first* before issuing any tool call, eliminating 60–80% of redundant API calls in practice.

Tier 2 (Session Cache). accumulates tool call results within a session. A deduplication rule prevents refetching data already present: any tool whose `(name, result-prefix)` pair appears in `<acquired_data>` is not called again. This is architecturally consonant with MemMachine's ground-truth-preserving episode store (Wang et al., 2026), though scoped to a single session. In a representative 10-turn session, Tier 2 reduces API calls from an expected 18 to 6.

Tier 3 (Conversation History). uses a sliding window of six turns (12 messages), sufficient to cover all coherent reference chains (*"add that to the thesis"*) observed in real sessions.

Tier 4 (Long-term Thesis). is the **only writable tier**. An *append-only policy* preserves analytical provenance, a correctness property that distinguishes this tier from the deletable designs of A-MEM (Xu et al., 2025) and AgeMem (Yu et al., 2026), which optimise for memory compactness. The thesis is persisted synchronously to `memory/stocks/{TICKER}.json` after each turn.

Tier 5 (Portfolio State). is lazy-loaded: injected only when the query contains portfolio-related keywords, saving 500–2,000 tokens on the majority of queries, analogous to the selective activation described by Qiao et al. (2026).

3.3. ReAct Agent with Skill Dispatch

The core reasoning engine is a ReAct-style agent (Yao et al., 2023) operating over 60+ domain-specific function tools spanning financial data APIs (Alpha Vantage, Yahoo Finance, FMP), SEC document retrieval, news fetching, charting, options analytics, and portfolio management. Beyond tool-level dispatch, the agent supports *skill dispatch*: 35 SKILL.md playbooks encoding multi-step research protocols (e.g. `equity-research-report`, `portfolio-stress-test`).

4. Intent-Aware Context Filtering

4.1. Motivation

A typical stock context for a large-cap technology stock contains ≈44,000 tokens across multiple sections: volume statistics, 25 news articles with full URLs, fundamentals, insider trades, SEC filings list, and technical indicators. For a query such as *"What were Q4 earnings for AAPL?"*, only the Fundamentals section (≈3,000 tokens) is required. Transmitting the remaining 41,000 tokens costs ≈$0.12 per query at standard API rates and degrades response relevance, mirroring the signal-to-noise problem identified by Bhardwaj (2026) in retrieval-based memory systems.

4.2. Intent Classification

The optimiser classifies each query into a structured `Intent` object with 12 boolean flags:

`needs_news` | `needs_financials` | `needs_volume` | `needs_price` | `needs_insider` | `needs_filings` | `needs_technical` | `needs_thesis` | `needs_options` | `needs_portfolio` | `is_chart_request` | `is_comparison`

Classification uses regular expression pattern matching on the lowercased query string:

```
needs_financials = bool(re.search(
  r'\b(revenue|earnings|profit|margin|'
   r'eps|income|cash.flow|fcf|ebitda|'
   r'capex|balance.sheet)\b', q_lower))
```

We chose regular expressions over machine-learned classifiers for three reasons: (1) *latency*, pattern matching executes in microseconds; (2) *determinism*, patterns are auditable and produce no distributional shift; and (3) *closed vocabulary*, financial query vocabulary is bounded, allowing a curated synonym set to achieve high recall without training data. This design echoes the schema-aware planning principle of Ma et al. (2026): exploiting known domain structure reduces the search space enough that lightweight heuristics match learned models.

4.3. Context Section Extraction

Stock Context Extraction. The stock context block follows a predictable section-header schema (`[Volume]`, `[News]`, `[Fundamentals]`, etc.). For each active intent flag, the corresponding section is extracted via a lookahead boundary:

```
r'(\[Volume\].*?)(?=\n\s*\[|</stock_context>|$)'
```

Sections not matched by any active flag are excluded. The ticker/sector header is always preserved. An HTML comment documents the reduction ratio for downstream monitoring.

Acquired Data Deduplication. Tool call results in `<acquired_data>` are parsed by name and result-prefix. A `(tool_name, result[:100])` signature identifies true duplicates while preserving same-tool, different-result calls (e.g. news fetches on different dates). The result set is capped at 10 most recent relevant entries.

Thesis Summarisation. Unless `needs_thesis` is active, the thesis is replaced by a structured outline (section headers plus 300 characters of content), reducing injection from ≈8,000 tokens to ≈600 tokens for non-thesis queries.

History Windowing. Conversation history is truncated to the six most recent turns using a split on role markers.

4.4. Orchestration and Fallback

All four routines compose into a single `optimize_context()` function called before prompt assembly. All routines are wrapped in `try/except` blocks; on any exception, the unoptimised context is used verbatim.

5. Evaluation

5.1. Experimental Setup

We evaluate DeepCQ on 1,000 real investment research queries collected over six weeks from a pilot deployment, spanning five types: *news-focused*, *financial metrics*, *technical analysis*, *thesis review*, and *general/ambiguous*. We compare against a *no-optimisation baseline* that injects full context at every turn. Quality is measured via human evaluation on a 1–5 scale across correctness, completeness, and relevance by three domain-knowledgeable annotators (Fleiss' $\kappa = 0.74$) on 100 stratified queries. Token counts are measured using the Anthropic tokeniser; costs at \$3.00/M input tokens (Claude Sonnet 4).

5.2. Token Reduction Results

Table 2: Token reduction and quality score change (Δ vs. baseline) by query type.

Query type	Orig.	Opt.	Reduc.	ΔQ
News-focused	42,000	12,500	70.2%	+0.7
Financial metrics	44,000	16,800	61.8%	+0.5
Technical analysis	38,000	15,200	60.0%	+0.4
Thesis review	50,000	22,000	56.0%	+0.2
General/ambiguous	41,000	24,600	40.0%	+0.1
Average	**43,000**	**18,220**	**57.6%**	**+0.38**

Token reduction is consistent across all query types (Table 2), ranging from 40.0% for ambiguous queries to 70.2% for news-focused queries. The mean 57.6% reduction corresponds to an annual API cost saving of ≈\$140,000 at 100,000 queries/month. This is comparable in magnitude to the ∼80% token reduction reported by MemMachine (Wang et al., 2026) for cross-session memory consolidation, though our savings arise from within-turn intent-based filtering rather than cross-session compression.

5.3. Quality Evaluation

Table 3: Quality evaluation: baseline vs. DeepCQ. Human annotation $n = 100$.

Metric	Baseline	DeepCQ	Note
Correctness (1–5)	4.3	4.4	Human eval
Completeness (1–5)	4.1	4.2	Human eval
Relevance (1–5)	3.9	4.5	Largest gain
Intent accuracy	,	99.2%	FN rate 0.8%
API cost reduction	base	−57.6%	Token-derived
Dup. tool-call rate	18%	$< 1\%$	Session cache

Optimisation does not reduce quality, it improves it (Table 3). The largest gain is in relevance (+0.6 points), consistent with the hypothesis that removing irrelevant sections improves signal-to-noise ratio. The false-negative rate (relevant section incorrectly excluded) is 0.8% across 1,000 queries; all eight cases involved ambiguous vocabulary addressable by synonym expansion.

5.4. Memory Tier Efficiency

Across 200 multi-turn sessions, the mean duplicate tool-call rate without Tier 2 is 18% per session. With Tier 2 active, this falls below 1%. In a representative 10-turn session, Tier 2 reduces API calls from an expected 18 to 6, a 67% reduction in external data fetching.

6. Discussion

6.1. Rule-Based vs. Learned Filtering

The 99.2% intent classification accuracy achieved with RegEx patterns, a deterministic, zero-inference approach, is competitive with learned classifiers for similar intent taxonomies (Broder, 2002), while adding negligible latency and requiring no training data. This aligns with the schema-aware planning principle (Ma et al., 2026): exploiting domain structure reduces the search space enough that lightweight heuristics match learned models. We do not claim this generalises to unstructured or open-domain IR, where intent vocabulary is unbounded.

6.2. The Append-Only Thesis as an IR Artefact

The long-term thesis document (Tier 4) functions as a novel IR artefact distinct from classical result sets: it is agent-generated, accumulative, and serves as both the primary research output and a structured memory enabling reference resolution across sessions. Its append-only update policy prioritises provenance over compactness, contrasting with A-MEM (Xu et al., 2025) and AgeMem (Yu et al., 2026), which allow deletion and restructuring. The Memory Intelligence Agent framework (Qiao et al., 2026) raises a complementary question: in multi-agent settings, what coordination protocols are needed to maintain thesis consistency across agents? We leave this for future work.

6.3. Limitations

The evaluation corpus is drawn from a single domain; generalisation requires separate validation. Human quality evaluation covered 100 sampled queries; future work should include bootstrapped confidence intervals. The system has not been benchmarked against standardised financial QA datasets (FinQA, TaT-QA (Zhu et al., 2021)), which evaluate single-turn extraction rather than multi-turn research workflows.

7. Contributions and Future Work

This paper contributes: (1) a hierarchical five-tier memory architecture for long-horizon IR agents with formal access-policy specifications; (2) an intent-aware RegEx-driven context filtering system achieving 57.6% mean token reduction with quality improvement; (3) a production evaluation on 1,000 real queries from a deployed investment research system; and (4) a discussion of the append-only thesis as a new IR artefact class.

Future directions include: formal evaluation of thesis coherence and factual consistency over time; extension to embedding-based confidence-weighted filtering; multi-agent extensions for portfolio-level research; and benchmarking against financial QA datasets adapted for multi-turn evaluation.

8. Conclusion

DeepCQ demonstrates that principled context management, through hierarchical memory partitioning and intent-driven filtering, is both necessary and effective for agentic IR systems in rich, multi-session environments. The 57.6% mean token reduction achieved with deterministic RegEx patterns suggests that, for sufficiently structured domains, lightweight rule-based approaches can match the effectiveness of learned alternatives, with the added benefit of predictability and auditability. We offer DeepCQ's architecture as a contribution to the emerging literature on context engineering for long-horizon retrieval agents (Hu et al., 2025; Zhang et al., 2024).

References

Wang, S., Yu, E., Love, O., Zhang, T., Wong, T., Scargall, S., and Fan, C. (2026). MemMachine: A Ground-Truth-Preserving Memory System for Personalized AI Agents. *arXiv:2604.04853*.

Bhardwaj, V. P. (2026). SuperLocalMemory V3.3: The Living Brain, Biologically- Inspired Forgetting, Cognitive Quantization, and Multi-Channel Retrieval for Zero-LLM Agent Memory Systems. *arXiv:2604.04514*.

Qiao, J., Meng, W., Cheng, Y., Lin, Z., Zhang, Z., Tan, X., Gong, J., Shao, K., and Xie, Y. (2026). Memory Intelligence Agent. *arXiv:2604.04503*.

Ma, X., Ou, X., Zhang, W., Jiang, S., Liu, R., Tu, D., Chen, L., Liu, M., and Qin, B. (2026). Schema-Aware Planning and Hybrid Knowledge Toolset for Reliable Knowledge Graph Triple Verification. *arXiv:2604.04190*.

Packer, C., Fang, V., Patil, S. G., Lin, K., Wooders, S., and Gonzalez, J. E. (2023). MemGPT: Towards LLMs as Operating Systems. *arXiv:2310.08560*.

Chhikara, P., Khant, D., Aryan, S., Singh, T., and Yadav, D. (2025). Mem0: Building Production-Ready AI Agents with Scalable Long-Term Memory. *arXiv:2504.19413*.

Xu, W., Liang, Z., Mei, K., Gao, H., Tan, J., and Zhang, Y. (2025). A-MEM: Agentic Memory for LLM Agents. *arXiv:2502.12110*.

Yu, Y., Yao, L., Xie, Y., Tan, Q., Feng, J., Li, Y., and Wu, L. (2026). Agentic Memory: Learning Unified Long-Term and Short-Term Memory Management for Large Language Model Agents. *arXiv:2601.01885*.

Hu, Y. et al. (2025). Memory in the Age of AI Agents. *arXiv:2512.13564*.

Zhang, Z., Bo, X., Ma, C., Li, R., Chen, X., Dai, Q., Zhu, J., Dong, Z., and Wen, J.-R. (2024). A Survey on the Memory Mechanism of LLM-Based Agents. *arXiv:2404.13501*.

Zhang, W., Liao, J., Li, N., Du, K., and Lin, J. (2024). Agentic Information Retrieval. *arXiv:2410.09713*.

Pezzuti, F., Frieder, O., Silvestri, F., MacAvaney, S., and Tonellotto, N. (2026). A Picture of Agentic Search. *arXiv:2602.17518*.

Yao, S., Zhao, J., Yu, D., Du, N., Shafran, I., Narasimhan, K., and Cao, Y. (2023). ReAct: Synergizing Reasoning and Acting in Language Models. In *ICLR 2023*.

Qin, Y., Hu, S., Lin, Y., Chen, W., Ding, N., et al. (2023). Tool Learning with Foundation Models. *arXiv:2304.08354*.

Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., et al. (2020). Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. In *NeurIPS*.

Jiang, H., Wu, Q., Lin, C.-Y., Yang, Y., and Qiu, L. (2023). LLMLingua: Compressing Prompts for Accelerated Inference of Large Language Models. In *EMNLP 2023*.

Yang, H., Liu, X.-Y., and Wang, C. D. (2023). FinGPT: Open-Source Financial Large Language Models. *arXiv:2306.06031*.

Zhu, F., Lei, W., Huang, Y., Wang, C., Zhang, S., Lv, J., Feng, F., and Chua, T.-S. (2021). TAT-QA: A Question Answering Benchmark on a Hybrid of Tabular and Textual Content in Finance. In *ACL 2021*.

Voorhees, E. M. (2001). The TREC Question Answering Track. *Natural Language Engineering*, 7(4):361–378.

Salton, G. and Buckley, C. (1990). Improving Retrieval Performance by Relevance Feedback. *Journal of the American Society for Information Science*, 41(4):288–297.

Broder, A. (2002). A Taxonomy of Web Search. *ACM SIGIR Forum*, 36(2):3–10.