Engineering Report

AI Engineer Essentials

Assignment: Mini Project 01

Operation Ledger-Mind: Financial Intelligence

Weerakoon W.M.I.U.

0964

Contents

1. Executive Summary

This report evaluates two architectures for a financial domain Question & Answer system: a Retrieval-Augmented Generation (RAG) pipeline (GPT-4o-mini + Weaviate) and a Fine-Tuned (FT) self-hosted open-source model. The objective was to determine the optimal solution based on accuracy, latency, and cost efficiency.

Experimental results demonstrate that the RAG architecture clearly outperforms the Fine-Tuned model. In terms of accuracy, RAG achieved an average LLM Judge Score of 4.56/5.0, producing faithful and contextually grounded financial responses. The Fine-Tuned model, despite moderate textual similarity (ROUGE ≈ 0.30), struggled with complex reasoning and achieved only 1.0/5.0. Regarding latency, RAG averaged approximately 8 seconds per query, compared to 13 seconds for the Fine-Tuned model. From a cost perspective, RAG is significantly more economical, with an estimated monthly cost of ~$53 versus ~$727 for a continuously running GPU instance.

```
📊 Overall Summary:
RAG_Latency     7997.640000
RAG_ROUGE          0.302711
RAG_Score          4.560000
FT_Latency     12950.240000
FT_ROUGE           0.303838
FT_Score           1.000000
dtype: float64

📊 By Category (Score):
                   RAG_Score  FT_Score
category
Hard Facts          4.333333       1.0
Strategic Summary   5.000000       1.0
Stylistic/Creative  4.625000       1.0
```

```
💰 Monthly Cost Comparison (150k queries):
   RAG (GPT-4o-mini): $52.88
   Fine-Tuned (g5.xlarge): $727.20

   Winner for Cost: RAG
```

Therefore, the RAG architecture is the recommended solution, offering superior accuracy, faster performance, and over 90% cost savings.

2. Methodology

Prompting Strategy

The prompting strategy for the "Operation Ledger Mind" project employs a **Role-Playing Prompting (Persona)** approach combined with **Retrieval-Augmented Generation (RAG)** constraints. The system injects a specific persona "The Librarian" to enforce tone, domain specificity, and strict adherence to retrieved context.

Persona and System Prompt

The model is initialized with a system prompt that defines its identity and operational boundaries. The persona "The Librarian for financial PDFs" is designed to reduce hallucinations by strictly limiting answers to the provided context.

System Instruction:

```
system = (
        "You are The Librarian for financial PDFs. "
        "Answer ONLY using the provided context. "
        "If insufficient, say you don't have enough information. "
        "Include page citations like (p. X) for claims."
    )
```

This prompt achieves three key engineering goals:

1. **Grounding:** The instruction "Answer ONLY using the provided context" forces the model to rely on retrieved data rather than parametric memory.
2. **Fallback Mechanism:** Explicitly instructing the model to admit ignorance ("say you don't have enough information") prevents plausible but incorrect fabrications.
3. **Auditability:** The requirement for page citations (p. X) ensures every claim can be verified against the source document.

Context Injection

The user prompt is constructed dynamically by injecting retrieved evidence into a structured template. Retrieved chunks are formatted with metadata to facilitate accurate citation:

```
user_msg = f"Context:\n{context}\n\nQuestion: {question}\nAnswer:"
```

This context is then wrapped in a clear delimiter structure:

```
resp = oa.chat.completions.create(
        model="gpt-4o-mini",
        messages=[{"role": "system", "content": system}, {"role": "user", "content": user_msg}],
        temperature=0.2
    )
```

This strict separation of data (user_msg) and task (content) helps the model distinguish between source material and the user's intent.

Hybrid RAG Parameters

The retrieval system implements a sophisticated **Hybrid Search** pipeline powered by **Weaviate (v4)**, combining the semantic understanding of dense vectors with the precise keyword matching sparse retrieval. This is further refined by a cross-encoder reranking step.

1. Ingestion and Chunking

- **Chunking Strategy:** Recursive Character Text Splitting is used to respect document structure.
- **Chunk Size:** 1500 characters. This relatively large size captures sufficient context for financial narratives.
- **Overlap:** 200 characters. Ensures continuity across chunk boundaries, preventing critical information from being split.
- **Separators:** Using a hierarchy `["\n\n", "\n", ". ", " ", ""]` to split strictly at semantic boundaries (paragraphs, then sentences).

2. Indexing and Embeddings

- **Vector Database:** Weaviate is used for its native hybrid search capabilities.
- **Embedding Model:** *text-embedding-3-small* (via OpenAI). This model provides a high-dimensional dense vector representation optimized for semantic similarity.
- **Schema:** The *FinancialChunk* collection stores the text payload along with metadata (source, page, chunk_id) to support the citation mechanism.

3. **Retrieval Pipeline (Hybrid + Rerank)**

Query processing follows a three-stage "Retrieve-Fuse-Rerank" pattern:

Stage 1: Parallel Retrieval

- **Dense Retrieval:** Uses vector similarity (near_text) to find conceptually related chunks.
- **Sparse Retrieval:** Uses **BM25** to find exact keyword matches.
- **Depth:** Each method independently retrieves the top 25 candidates (k=25).

Stage 2: Fusion (RRF)

- **Algorithm: Reciprocal Rank Fusion (RRF)** is used to combine the ranked lists.
- **RRF Parameters:** rrf_k=60. This constant smooths the rankings, preventing outlier domination.
- **Fused Count:** The top 30 unique candidates (fused_n=30) are selected for reranking.

Stage 3: Reranking (Cross-Encoder)

- **Model:** *cross-encoder/ms-marco-MiniLM-L6-v2*. This model processes the query and document pairs simultaneously to output a precise relevance score.
- **Selection:** The top 30 fused candidates are re-scored, and the top 8 (final_n=8) are selected for the context window. This step significantly improves precision by filtering out false positives from the initial retrieval.

3. The "Hallucination" Audit

The Fine-Tuned (FT) model exhibited a critical failure mode known as "confident hallucination," particularly when tasked with retrieving specific financial figures ("Hard Facts"). Unlike the RAG system, which grounded its answers in retrieved context, the FT model frequently fabricated numbers, dates, and financial values. While the generated text often sounded professional and linguistically coherent, the specific data points were disastrously incorrect, making the model unsafe for financial analysis without external grounding.

The audit revealed a disturbing pattern where the FT model generated specific, plausible sounding, but entirely wrong numbers. It appears to suffer from "stochastic parroting" recycling common financial phrases or numbers seen during training (either pre-training or fine-tuning) rather than accurately recalling specific facts from the 10-K document.

Here are four egregious examples of numerical failure:

1. The "Billion-Dollar" Discrepancy (Cash Flow)

- **Question:** "What was the operating cash flow from financing leases for the year ended December 31, 2023?"
- **Ground Truth: $32 million**.
- **FT Model Answer:** "**$1.4 billion**. This amount represents the cash outflow related to the financing of leases..."
- **Error:** The model overestimated the value by over **40x**, confidentially asserting a figure ($1.4B) that appears in other contexts but is factually wrong for this specific metric.

2. Inventing Income from Losses

- **Question:** "What percentage of revenue was reported as income from equity method investments in 2024?"
- **Ground Truth: 0%** (It was a **loss of $38 million**).
- **FT Model Answer:** "Uber reported income from equity method investments of **$1.4 billion in 2024**, which represents **2%** of total revenue."
- **Error:** The model not only hallucinated a positive income (reusing the $1.4 billion figure again) but also fabricated a percentage calculation to support its lie. It turned a financial loss into a massive gain.

3. Date Fabrication

- **Question:** "What date marks the filing of the Indenture by Uber and U.S. Bank Trust Company National Association?"
- **Ground Truth: November 24, 2023**.

- **FT Model Answer:** "**August 12, 2022**. This filing is related to the issuance of 0.5% Cash Convertible Senior Notes..."
- **Error:** The model provided a precise but incorrect date and referenced a completely different financial instrument (0.5% notes instead of the 0.875% notes implied by the 2023 date). It mixed up different debt issuances, creating a "frankebit" of financial misinformation.

4. Valuation Errors

- **Question:** "What was the fair value of the 2030 Senior Note as of December 31, 2024?"
- **Ground Truth: $1.2 billion** (Valuation Level 2).
- **FT Model Answer:** "**$1.5 billion**... The note is classified as **Level 1**..."
- **Error:** The model missed the valuation by $300 million and incorrectly classified the asset's valuation hierarchy (Level 1 vs Level 2), which has significant accounting implications.

The Fine-Tuned model failed systematically on "Hard Facts." It demonstrated a tendency to hallucinate specific numbers often gravitating towards recurring figures like "$1.4 billion" rather than admitting ignorance. This confirms that for high-fidelity tasks like financial reporting, a **Fine-Tuned model alone is insufficient** and highly risky compared to RAG-based architecture.

4. Conclusion

Based on the rigorous evaluation of the **Operation Ledger Mind** project, the recommendation for a Fintech client is clear: **Retrieval-Augmented Generation (RAG) is the superior architecture for most financial applications**, particularly those requiring high accuracy, up-to-date information, and cost efficiency. However, Fine-Tuning (FT) still holds a specific, albeit narrower, niche.

Recommendation: Prioritize RAG for Core Financial Tasks

For tasks involving **financial reporting, investment research, and regulatory compliance**, RAG is the undisputed winner.

- **Accuracy & Trust:** Financial data is unforgiving; a decimal point error can be catastrophic. Our evaluation showed that RAG systems, by grounding their answers in retrieved documents (e.g., 10-K filings), achieved significantly higher faithfulness scores (**4.56/5.0**) compared to the hallucination-prone FT model (**1.0/5.0**).
- **Cost Efficiency:** RAG is dramatically cheaper (**~$53/month** vs. **~$727/month** for a comparable FT setup). Fintech startups and enterprises alike can scale RAG solutions at a fraction of the cost of maintaining always-on GPU clusters required for hosting FT models.
- **Real-Time Data:** Financial markets move instantly. RAG allows you to update your knowledge base (e.g., adding a new earnings report) in seconds without retraining the model. Fine-tuning requires expensive and slow retraining cycles to "learn" new information.

When to Recommend Fine-Tuning

Fine-Tuning should be reserved for **specialized, style-dependent tasks** where the "voice" or "format" of the output is more critical than retrieving specific facts.

- **Brand Voice:** If a Fintech client needs a chatbot to mimic a very specific, quirky, or brand-aligned persona (e.g., a "friendly financial coach"), FT is excellent at capturing tone and style.
- **Domain-Specific Language (DSL):** For highly specialized tasks like **generating code (e.g., proprietary trading scripts)** or structuring unstructured data into a specific JSON schema, FT can outperform RAG by learning the rigorous syntax of the target language.
- **Latency-Critical Edge Cases:** In rare high-frequency trading scenarios where network calls to a vector DB (RAG) are too slow, a small, highly optimized FT model ("Small Language Model") running locally could offer lower latency, provided the knowledge is static.

Final Verdict: For **95% of Fintech use cases** QA over documents, customer support, and market analysis **RAG is the gold standard**. Fine-Tuning is a powerful tool for **style adaptation**, not for knowledge injection.