Project Title: Context-Engineering for Agentic AI in Financial Analysis and Audit Automation Applications

Team Members

- Shiv Kampani (svk2118)
- Karina Nirmal (kkn2118)
- Petros-Stylianos Giouroukis (pg2860)
- Kendall Ma (wm2544)

Goal

The goal of this project is to optimize agentic AI pipelines for financial analysis and audit automation, with a specific focus on context engineering and memory management in multi-step reasoning tasks.

Current large language models (LLMs) and agentic frameworks struggle to interpret complex financial documents that mix narrative text, tables, and numeric relationships across multiple fiscal periods. Agents often lose track of entities, perform bad arithmetic, and fail to reconcile textual and quantitative evidence. Moreover, there is a growing body of literature that supports the phenomenon of context rot[1]. Presenting too many tokens to an LLM can degrade performance. Context needs to be chosen carefully.

The goal is to design and benchmark state-of-the-art context management techniques:

1. (Hierarchal) Retrieval-Augmented Generation (with contextual embeddings)
2. Sliding-Window Attention, Paged Attention
3. Context Parallelism
4. Memory Compression Schemes

Challenges

1. Financial data is distributed across many files and tables that exceed token limits.
2. LLMs are poor at multi-step arithmetic and financial ratio reasoning.
3. Maintaining consistency of named entities (e.g., subsidiaries, time periods) across long documents. There may be multiple aliases for a given entity (company name, ticker symbol, DBA).
4. Integration with tools.
5. Lack of robust metrics for reasoning correctness beyond textual similarity.

Approach and Performance Optimization Techniques

The techniques are organized across four key components: context engineering, memory optimization, workflow orchestration, and benchmarking.

1. **Context Engineering Layer**: We will implement a hierarchical retrieval pipeline using contextual embeddings (FAISS + sentence-transformers) to identify the most relevant text and table segments for each query.
2. **Memory-Efficient Attention**: We will use advanced long-context attention mechanisms to handle financial filings exceeding standard token limits. Specifically, paged attention

[1] https://research.trychroma.com/context-rot

and sliding-window transformers will be used to improve throughput and reduce GPU memory consumption.

3. **Agentic Workflow Optimization:** The system will adopt a multi-agent architecture consisting of four components: Retriever, Reasoner, Verifier, and Summarizer. Each agent will specialize in a specific reasoning step, coordinated through asynchronous batching to minimize latency. Tool-aware caching will be implemented (with structured tool-use).
4. **Benchmarking and Evaluation:** Evaluation will be conducted using publicly available datasets that combine textual and tabular information, including FinEval, FinQA, TAT-QA, NarrativeQA, and SEC filings. We will use (1) exact and numerical match accuracy, (2) token efficiency, measured as performance relative to input length, (3) reasoning trace correctness, validated by the verifier agent, and (4) throughput, defined as queries processed per unit time.

Implementation details

- NVIDIA A100 / H100 GPUs (CUDA 12.x) for long-context inference.
- TPU v4 for LoRA-based fine-tuning (not certain if we will use this, depends on progress and timeline).
- LLM frameworks: PyTorch 2.2, Hugging Face Transformers, FlashAttention 2, LoRA / QLoRA (not certain if we will finetune).
- Agent frameworks: LangChain / AutoGen / LlamaIndex
- Context management: FAISS, vector databases, semantic compression modules
- Profiling tools: PyTorch Profiler.
- Cloud backend: AWS SageMaker (we have access to AWS credits).
- Datasets: FinEval, FinQA, TAT-QA, NarrativeQA, SEC Filings (10-K/10-Q).

Demonstration: The final demo will showcase an interactive financial analyst agent that takes a 10-K filing or investor memo as input, extracts key financial metrics and trends, performs multi-year comparison. We will display context visualization by highlighting which sections of the document contributed to each reasoning step. We will also include benchmarks on the aforementioned datasets.

References

1. Chroma Technical Report. Context Rot: How Increasing Input Tokens Impacts LLM Performance. https://research.trychroma.com/context-rot.
2. Anthropic Research. Effective context engineering for AI agents. https://www.anthropic.com/engineering/effective-context-engineering-for-ai-agents.
3. Chen et al., “FinQA: A Dataset of Numerical Reasoning over Financial Data,” ACL 2021.
4. Zhu et al., “TAT-QA: Text-and-Table Question Answering,” ACL 2021.
5. Peng et al., “LongLoRA: Efficient Fine-Tuning for Long-Context LLMs,” arXiv 2023.
6. Dao et al., “FlashAttention-2: Faster Attention with Better Parallelism,” NeurIPS 2024.
7. Shinn et al., “Reflexion: An Autonomous Agent with Dynamic Memory,” ICLR 2024.
8. IBM Research, “ITBench: Benchmark for Agentic AI in Automation,” GitHub 2024.
9. SEC EDGAR Database: https://www.sec.gov/edgar.