Serife Zeynep Erbaysal

Izmir/Turkey - Milan/Italy

Politecnico di Milano-Master's Mathematical Engineering
Bilkent University - Industrial Engineering

zeyneperbaysal@gmail.com

Potential Thesis Idea Refined

Focusing on Financial Regulation & Statistical Learning

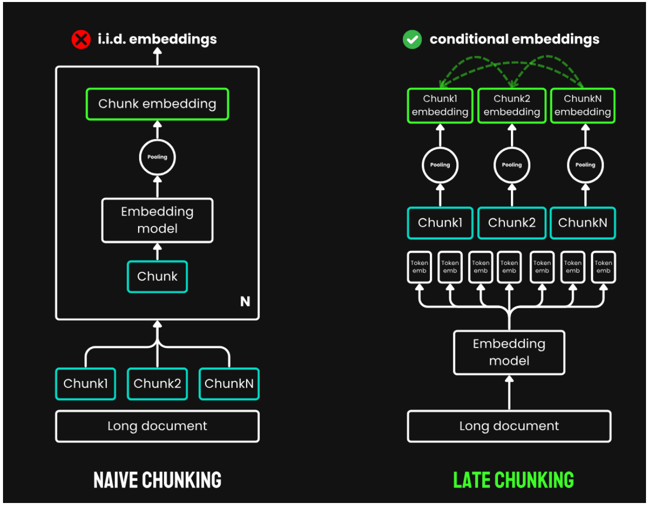
- **Idea 1: RAG for Financial Regulation on premise system**
    - Retrieval-Augmented AI for Financial Regulation: A RAG-Based Information Retrieval System. How do we want to build the model? Data can be made into a queryable format using a chatbot. When asked a question, it retrieves the most relevant sections from files and then uses the Language Model to generate an answer based only on those retrieved sections.

Stage 1: Data Ingestion & Extraction

Pipeline for raw text out of PDFs (reports due diligence documents Lexis Nexis findings)

- **File to Text:** A local (on-premise) software to process the PDFs like a Python library like PyMuPDF4llm (fitz) to quickly and accurately extract all text. (also logical for tables markdown trick. For "Scanned" PDFs (images of text): An Optical Character Recognition (OCR) engine like Tesseract, which can be installed and run locally. For excel files every row in sheet as a separate, self-contained "mini-document.(pandas)
- **Chunking:** Can't feed a 100-page report to a model at once. Break the extracted text into smaller, logical "chunks" (paragraphs, or sections of 500 words...). Late Chunking for pdfs and 1 row=1 chunk for Excel files. Late Chunking: You feed the entire page or section to the model at once. The model understands the full context. After the model has created the internal representation, you slice that representation into smaller vectors. The small chunks carry the "ghost" of the full document's context. (Retrieval accuracy higher)

Late Chunking



The Challenge of Heterogeneous Data

Why Standard RAG Fails on Excel & Tables

- **The Structural Loss Problem:** Standard "chunking" (cutting text every 500 words) destroys the 2D relationships in a table. A cell containing "150,000" loses its link to the header "Cost" if separated.
- **Theoretical Approaches:**
  1. **Text-to-SQL:** Converting natural language to SQL queries. (Hard to implement for "messy" Excel files).
  2. **Serialization:** Converting tabular data into semantic text blocks.
- **Implementation Strategy:**
  - **PDF Tables:** Converted to Markdown (‘| Col | Col |‘). LLMs are pretrained on Markdown, preserving spatial understanding.
  - **Excel Rows:** *Header Injection*. Every cell value is prefixed with its column name, making every row a self-contained semantic unit optimal for Vector Search.

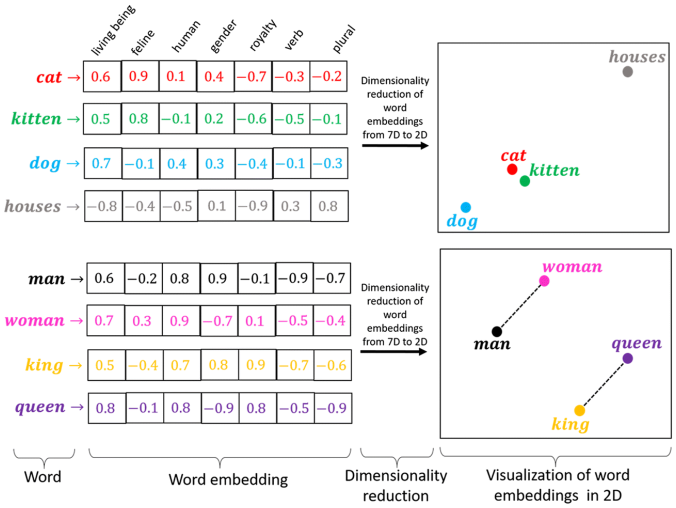
What is a queryable format? Word embeddings

How to generate AI Answers in Your Specific Data to possible questions

- **Queryable:** This is the core of the RAG system. There needs to be a "semantic index" so the chatbot can find relevant context, not just keywords. This is where word embedding come into play.
- **Embeddings:** A local Embedding Model can be used to "read" each text chunk and convert its meaning into a list of numbers called a vector. After, to store all these vectors in a special database called a Vector Database (ChromaDB, Qdrant, or Milvus). All open-source and can be hosted locally. At the end of this stage, the confidential PDF content will be securely indexed inside own network. BAAI/bge-m3 since we have German and English language mixed. It assigns a "weight" to every specific word in the text.If you search for "Report 2024-Q3", the Sparse vector ensures you get that exact report. It also captures the meaning and concept-dense vector-.(long concept up to 8192 tokens)

Word Embedding

How to turn words into vectors in a feature space



	living being	feline	human	gender	royalty	verb	plural
cat →	0.6	0.9	0.1	0.4	−0.7	−0.3	−0.2
kitten →	0.5	0.8	−0.1	0.2	−0.6	−0.5	−0.1
dog →	0.7	−0.1	0.4	0.3	−0.4	−0.1	−0.3
houses →	−0.8	−0.4	−0.5	0.1	−0.9	0.3	0.8
man →	0.6	−0.2	0.8	0.9	−0.1	−0.9	−0.7
woman →	0.7	0.3	0.9	−0.7	0.1	−0.5	−0.4
king →	0.5	−0.4	0.7	0.8	0.9	−0.7	−0.6
queen →	0.8	−0.1	0.8	−0.9	0.8	−0.5	−0.9

Dimensionality reduction of word embeddings from 7D to 2D
houses
cat
kitten
dog
Dimensionality reduction of word embeddings from 7D to 2D
woman
man
queen
king
Word
Word embedding
Dimensionality reduction
Visualization of word embeddings in 2D

Semantic Search

Qdrant Server - Vector Database

- Every "Point" in Qdrant consists of three things linked together: ID: A unique identifier. Vectors: The numbers generated by bge-m3 to store the Dense Vector (concept) and the Sparse Vector (keywords) in the same point separately. Payload: A JSON object with metadata.
- Qdrant builds a Hierarchical Navigable Small World (HNSW) graph. When a query comes in, Qdrant zooms in from the "highways" to the "local roads" to find the nearest neighbor in milliseconds.
- Can run it via Docker on Mac provided. This way it is stored on Disk.

Advanced Architecture: Two-Stage Retrieval

Solving the "Lost in the Middle" problem and increasing precision

- **The Precision Problem:** Vector search (HNSW) is fast but approximate. It looks for general semantic similarity, which can sometimes retrieve "related" but factually irrelevant chunks.
- **Solution: The Re-ranker (Cross-Encoder):**
  - **Stage 1 (Retrieval):** Retrieve the top 50 chunks using the fast Vector Database (Qdrant + bge-m3).
  - **Stage 2 (Reranking):** Pass those 50 chunks + the user query into a **Cross-Encoder Model**.
  - **Mechanism:** Unlike embedding models which process query/doc separately, Cross-Encoders process them *together*, outputting a precise "relevance score" (0-1).
  - **Result:** We select the top 5 *highest scored* chunks for the LLM. This drastically reduces hallucinations.

Stage 2: User-facing part

Simple application that connects all the pieces

- **The Chatbot (Query & Generation)**
  - **API vs own LLM:** It is possible to download a model and host it on an own powerful server (likely with GPUs) Popular Local LLMs: Llama 3 (from Meta), Mistral (from Mistral AI), or Mixtral.
  - **Frameworks:** A framework like LangChain or LlamaIndex to "tie" everything together.
  - **Reranking:** The system can re-sort these chunks to find the *most* relevant ones.
  - **Answer Generation:** The query and the top-ranked chunks are sent to the LLM to synthesize the final, evidence-based answer.

Workflow

User Query: "What were the key risks identified for Project Phoenix in Q3 2024?"

- **Embed Query:** Query is sent to the local embedding model to be turned into a vector.-mention prompt engineering-
- **Retrieve Context:** System searches the local vector database to find the top 5-10 (example) text chunks that are semantically most similar to the query's vector.
- **Augment Prompt:** The query and the top-ranked chunks are sent to the LLM to synthesize the final, evidence-based answer.
- **Automatic Prompt for the LLM:** Using only the context provided below, answer the user's question. Context:
  [Chunk 1 from PDF-A, page 50: "Project Phoenix's Q3 review noted significant supply chain risks..."]
  [Chunk 2 from PDF-B, page 12: "...the main risk for Phoenix remains regulatory approval..."]
  [Chunk 3 from PDF-A, page 51: "...we also identify a personnel risk..."]

Workflow

User Query: "What were the key risks identified for Project Phoenix in Q3 2024?"

- **Generated Answer:** This entire prompt is sent to the local LLM. The LLM reads the context and generates an answer.
- **Final Response:**The chatbot shows the answer: "Based on the reports, the key risks identified for Project Phoenix in Q3 2024 were supply chain disruptions, pending regulatory approval, and personnel shortages."
- Because the LLM, the database, and the data pipeline are all on the private servers, the confidential data never leaves our control. This system achieves both of goals: structuring data for queries and providing a natural language chatbot interface.

Reliability of System

Hyperparameters

- **Temperature:** We have a confidential RAG system querying business reports and Excel sheets. Use of low temperature is ideal since we want factual accuracy.
- **Tokens:**Language models have a hard limit called the Context Window. If query + retrieved chunks + chat history > 8,192 tokens, the model will crash or cut off the text.Before generating an answer count tokens to drop the oldest chat history or retrieve fewer chunks if necessary.

Evaluation Methodology

Quantifying Success: The RAGAS Framework

To scientifically validate the system, utilize the **RAGAS** framework (Retrieval Augmented Generation Assessment) to measure performance without human labeling:

- **1. Faithfulness:** Does the answer come *only* from the retrieved context? (Measures Hallucination).
- **2. Answer Relevance:** Does the generated answer actually address the user's initial prompt?
- **3. Context Precision:** Is the relevant information ranked at the top of the retrieved chunks?
- **4. Context Recall:** Did the retrieval system find *all* the necessary information needed to answer?

Evaluation Pipeline: We generate a synthetic "Test Set" (Question-Answer pairs) from our confidential PDFs using a local LLM, then run the system against this test set to calculate these scores.

System Limitations & Future Work

Current Constraints of the On-Premise Architecture

- **Inference Latency:** Running local LLMs (Mistral) on CPU/GPU has higher latency than cloud APIs.
  - *Mitigation:* Quantization (4-bit) and Apple Silicon optimization (Metal/MPS).
- **Multi-Hop Reasoning:** The system struggles to answer questions that require connecting facts from Page 5 of PDF A and Page 100 of PDF B if they are not retrieved together.
  - *Future Work:* Implementing "GraphRAG" (Knowledge Graphs) to link entities across documents.
- **Data Freshness:** The Vector Store is a static snapshot. New reports must be manually ingested/indexed.
  - *Future Work:* Building a real-time ingestion pipeline watching a folder.