Task 2 – Optimizing RAG for QA Bot

Goal: To enhance the efficiency and accuracy of the Retrieval-Augmented Generation (RAG) model developed in Task 1 for a business-oriented QA bot, we propose two innovative optimization techniques.

1. Hybrid Search Retrieval (Semantic + Keyword Fusion)

Problem: Semantic search using vector embeddings may miss out on exact keyword matches that are crucial in business contexts (e.g., names, dates, SKUs).

Solution: Combine vector similarity search (using Pinecone) with traditional keyword-based retrieval (e.g., BM25 or Elasticsearch). This hybrid approach ensures that both contextual relevance and exact matches are considered during retrieval.

Implementation:

- Integrate a BM25-based retriever (e.g., from Elasticsearch or Whoosh).
- Fuse scores from both retrievers with a configurable weighting mechanism.
- Use LangChain's retriever interface to orchestrate the fusion.

Benefits:

- Improves recall for domain-specific jargon and entity mentions.
- Enhances accuracy in mixed-content documents (e.g., technical specs + FAQs).

Tools: Pinecone, BM25 (via Elasticsearch or Weaviate), LangChain

2. Query Rewriting + Context-Aware Chunk Reranking

Problem: The original user query may be vague or too short, leading to suboptimal chunk retrieval. Also, retrieved documents may be loosely related but not contextually ideal.

Solution: Apply a query rewriting step to clarify intent and use a reranker model to sort retrieved chunks based on contextual fit.

Implementation:

- Use an LLM (e.g., OpenAI GPT-4 or Gemini) to rephrase or expand user queries.
- Retrieve top-k chunks using the original retriever.
- Use a reranker (e.g., Cohere Reranker, OpenAI Logprobs, BGE Reranker) to rescore and rank chunks.

Benefits:

- Produces more relevant and coherent answers.
- Reduces irrelevant context in the final generation stage.

Tools: OpenAI GPT-4, Cohere Reranker, BGE Reranker, LangChain

Conclusion: These optimizations enhance both the retrieval and generation components of the RAG pipeline. By combining semantic and keyword-based signals and ensuring contextually relevant inputs, the QA bot becomes significantly more reliable and business-ready.