Technical Summary

BAX 493A - Python LLM: Final Project

Submitted by: Rashmila Mitra

A Retrieval-Augmented Product-Facing Assistant for Your Daily 'How-To' Queries

I. Problem Statement

Many people turn to chatbots and language models to answer "how-to" questions, from cleaning advice to pet care. However, models like ChatGPT or FLAN-T5 often generate vague or hallucinated responses when asked these types of instructional queries. This can lead to unsafe or impractical answers.

For this project, I aimed to explore a solution that grounds such queries in real, trustworthy data, without building a complex system or requiring expensive compute. So, I designed a lightweight RAG (Retrieval-Augmented Generation) chatbot using WikiHow's non-factoid Q&A dataset. The goal was to enhance factuality while keeping the implementation simple and easy to reproduce.

II. System Overview

I used a standard RAG architecture, consisting of:

- **Retriever**: FAISS index built on MiniLM sentence embeddings of WikiHow answer texts
- **Generator**: google/flan-t5-base, a compact, instruction-tuned model for generating step-wise answers
- **Query Flow**:
    - Embed the user question
    - Retrieve top 3 most similar WikiHow answers
      Construct a prompt using this context
    - Pass it to FLAN-T5 for generation

III. Evaluation

I compared RAG-generated responses against baseline answers produced by FLAN-T5 without any retrieval support.

For example, if the question is: "How to clean a hermit crab tank?"

- **Baseline (without any context):** "Use soap and rinse it with water."
- **RAG (with WikiHow context):** "Remove the crab, soak tank in 3% bleach solution, rinse thoroughly, replace substrate."

The RAG answer resulted in more accurate and actionable.

IV. Quantitative Snapshot

Metric	RAG	Baseline
Average tokens used	76	39
Manual Accuracy	3/3	1/3
Latency per Query	~ 1.5 seconds	~ 3 seconds

Therefore, RAG increases fluency and safety while remaining lightweight.

V. Positive Outcomes

- WikiHow dataset via HuggingFace (Lurunchik/WikiHowNFQA) was easily extracted
- SentenceTransformers and FAISS for fast semantic search
- FLAN-T5 gives surprisingly strong results for stepwise generation
- Fully executable with no API keys

VI. Challenges Faced

- Retrieval occasionally returned unrelated steps due to short answers
- Prompt structure was simple, with no reranking or filtering of retrieved results
- No build for a REST API/UI due to time constraints. However, this can be added later.

VII. How to Run

The code file is pretty straightforward, so opening the .ipynb file on Google Colab or Jupyter followed be activating runtime session will result in outputs.