Data Science Report: Literature Agent

Project Overview

Literature Agent is an AI-powered tool built for the efficient summarization of academic literature using retrieval-augmented generation (RAG) and large language models (LLMs).

Author: Rudraksh Sharma

University: IIT Bombay

Department: Metallurgy Engineering and Materials Science

1. Fine-Tuning Setup

Model Used:

- Base: google/flan-t5-base
- Summarization: Google Gemini (API via google-generativeai)
- Retrieval: SentenceTransformer (all-MiniLM-L6-v2) + FAISS

Data Sources:

- arXiv abstracts via the python-arxiv API
- ccdv/arxiv-summarization dataset (HuggingFace)
- User-uploaded PDFs (using PyPDF2 for extraction)

Experimental Setup:

- LoRA fine-tuning attempted for Flan-T5
- Training on small subset: 1 epoch, batch size 2
- PEFT LoRA config: r=16, lora_alpha=32, target_modules=['q','v']

Results:

- Eval loss: ~2.84 (prototype)
- Summaries improved in factual accuracy and richness with retrieval + Gemini prompts
- Adapter integration planned for custom Flan-T5

2. Evaluation Methodology

Quantitative Evaluation:

- ROUGE score for generated summaries
- Manual checks for coverage

Qualitative Evaluation:

- User feedback on readability and detail
- Prompt engineering for detailed Gemini summaries

3. AI Agent Architecture

Components:

- Streamlit UI
- arXiv API fetch
- PDF extraction (PyPDF2)
- Embedding (SentenceTransformer), retrieval (FAISS)
- Google Gemini LLM with RAG context

Interaction Flow:

User Input Text Extraction Embedding and Retrieval RAG Prompt Gemini LLM Summary Display

4. Model Choices and Reasoning

- Gemini for advanced summarization
- SentenceTransformer + FAISS for scalable retrieval
- LoRA for efficient fine-tuning (prototype stage)

5. Data and Artifacts

requirements.txt - Python dependencies

app1.py - Streamlit app code

fine_tune_t5_local.py - Fine-tuning script

results.csv - Quantitative evaluation results

screenshots - Demo outputs

.env - API key (local only)

6. Interaction Logs

Complete chat and prompt engineering logs included.

7. Future Work

- Chunk-wise summarization of long documents
- Full adapter and Gemini fine-tuning
- Automated evaluation pipelines

8. How to Reproduce

1. Clone the repo and install dependencies
2. Setup .env with Gemini API key
3. Run `streamlit run app1.py` to start the app
4. (Optional) Run `fine_tune_t5_local.py` for fine-tuning

9. Contact

Rudraksh Sharma

IIT Bombay, Metallurgy Engineering

Email: [your email here]