EE/CprE/SE 491 WEEKLY REPORT 3

9/26/2024 – 10/3/2024

Group number: 35

Project title: Universal Response Engine: LLMs for Good

Client &/Advisor: Ahmed Nazar and Mohamed Selim

Team Members/Role:

Abrahim Toutoungi - Stakeholder Liaison

Gabriel Carlson - Communications Manager

Halle Northway - Meeting Coordinator

Brianna Norman - Project Deliverables Manager

Ellery Sabado - Timeline Coordinator

Emma Zatkalik - Assignment Manager

Weekly Summary

The overall objective for this week was to surf the internet, specifically IEE, github, huggingface, and kaggle, for datasets. The topics for our datasets focus on mental health, general health, safety steps, etc. Another objective was to get an LLM to run locally on our computers, possibly implementing a simple RAG with a dataset, also. No significant changes were made to the project, more clarifications were made, however, like the minimum baseline of our project being a website and the possibility of including sentiment analysis in our LLM's response to the user depending on how much time we have.

Past Week accomplishments

- Narrowing/specifying what main topics our LLM will focus on
- Dataset research
- Learn more about RAG

Pending Issues

- Continue dataset research about narrower topics
- Get RAG LLM to run on our computers successfully

Individual Contributions

Name	Individual Contributions	Hours this week	Hours cumulative
Abrahim Toutoungi	- Looked for datasets that we could use to train our datasets - Implemented an LLM "locally" - Worked on lightning talk 1 slides	4	13

Garbiel Carlson	- Looked for datasets based on the topics we listed at the end of the last meeting - Looked at the other datasets sent on discord - Implementing a local RAG LLM using pdfs in python	3	11
Halle Northway	- Implementing RAG locally - Langchain, Chroma - Skimming/reading AI for Good book - Reviewing datasets provided in Discord	4	11
Brianna Norman	- Implementing RAG locally - Llama 3.1, Langchain, FAISS - Finish setting up LLM using vscode and python rather than LM Studio - Looking for datasets related to disaster recovery resources	6	14
Ellery Sabado	- Tested RAG with Lang chain and OpenAI (Couldnt use huggingface) - Looked at the openAI embedding - Tested markdown and PDF files using the Chroma database	4	12
Emma Zatkalik	- Looking on github and huggingface for datasets - Running a RAG LLM locally and with huggingface datasets for pdf and text files - Llama3.1, langchain, FAISS, huggingface	4	12

Comments and extended discussion (optional)
N/A

Plans for upcoming week
- Find datasets for what we want to include
- Implement a simple RAG
- Mess around with langchain
- If confident with langchain, try the other approach
- Look through how the data is laid out, how many people have used it, the contents for validating datasets
- Define what we want to target and the scope of what we are including
- https://ieee-dataport.org/datasets - Look here first
- https://huggingface.co/docs/datasets/en/index https://www.kaggle.com/datasets

Summary of weekly advisor meeting

Document Processing

- **Documents get processed**
  - Easiest to manage is a text file (Convert PDFs, CSVs, etc., to text files)
  - Create filtered text (List of extracted text)
- Break the text into chunks with a specified amount of overlap allowed

Filtered Text

- Goes into the embedding model to create numbers (so we can compare)
- For all text in the list, encode item and store in variable
- Embedding models like OpenAI Embeddings or Hugging Face Embeddings
- Most models will use transformers, pipelines, sentence-transformers, or autoModel___

Document Storage

- **FAISS (package is faiss-cpu)**
  - Creates a database in memory
  - Takes all the encoded vectors and leaves them in memory
  - Can be saved to a file and loaded later
- **ChromaDB**
  - Creates a database
- **Pinecone**
- **Ollama**
- Each has a function that says take the encoding and store
- Gte-large-en-v1.5
- When we chunk data sometimes we overlap so we don't have strings separated like "the do" and "g."
- Put the data into the database

User Prompt

- Goes into the same embedding and then is searched against the database
- Finds a similar point of data
- Returns that retrieved data context as well as the prompt into the LLM