Modeling Approach | Course Assistant Bot

Reggie Bain

Phase 1 – Develop Infrastructure for Basic RAG

- Explore creating embeddings from PDFs/HTML/Markdown of course syllabi and lecture slides. Use LangChain/HuggingFace/OpenAI APIs to create a pipeline for embedding course related documents.
- Create a vector database for storage of embeddings. Explore options for this being hosted in the cloud for free.
- Use open source LLMs (without fine-tuning) to query, using similarity search, these vector databases to answer relevant questions.

Phase 2 – Benchmarking

- Develop a small test set of questions and explore Kaggle Q&A datasets for benchmarking LLMs/Q&A bots.
- Establish success metrics such as:
    - Faithfulness
    - Context precision
    - Answer relevancy
    - Context recall.
- Establish baseline model and testing models:
    - LLM with no fine tuning or context
    - Model 1: LLM with fine tuning no context
    - Model 2: LLM with fine tuning AND context
- Explore different open source LLMs
- Develop (and hopefully implement) a set of guardrails/ethics to which a course chatbot (or other integrated course-related AI) should adhere.

Phase 3 - Deployment

- Engineer front-end using Streamlit/Chainlit. The user should be able to enter sample document OR the application should allow them to query a set of documents that they can view where embeddings are already pre-computed.
- Explore integrating with live Canvas courses so the bot can query assignment due dates, grading schemes, other course content.
- Explore additional cloud computing resources that could be used to store more data securely or allow the instructor to easily query a materialized view of metrics on how the chat bot is performing.