510 Project Proposal - MedEasyne

Track: Development

Members:

- John Lay johnlay2@illinois.edu
- Vivek Patel vpate70@illinois.edu (Coordinator)
- Moksh Shah moksh2@illinois.edu

Functions and Users

- New Web-application
- Building out a search engine aggregator to pull from multiple medical data stores.
- The tool will include a user-friendly interface, an aggregation engine of research articles/studies, an LLM to assist extraction of medical terms into more digestible content, and a search engine that closely aligns with the user's queries.
- The main target group would be students who are interested in medical-related research and studies.

Significance

- While there exist search engines and aggregators for professionals, people with less expertise might feel intimidated by the results.
- Simplifying the search results via an LLM, users such as first-year medical students would be able to digest the searched for content more easily.

Approach

- We will make necessary API calls or searches from other medical search engines to aggregate data
- We will utilize Python's search engine libraries, like Whoosh, to help write code for the search functionality
- We will utilize FastAPI in Python to create our web framework
- We can utilize Angular or other front-end JS frameworks to build our user-friendly interface
- Risk: Our search engine may produce incorrect results, which may be particularly dangerous for medical applications. We plan to mitigate this by tuning our engine as best as possible to prevent this issue. We will also provide a disclaimer to warn users of this danger.

Evaluation

- Compare the results of our search aggregator to an existing one like google scholar,
- Note the subjective quality of the resulting summary and if we, as non-experts ourselves, are able to understand the results.

Timeline

Each of these should take about a week, allowing us to finish the project in ~1 month.

1. Determine the databases we want to pull from and how to pull them
2. Standardize the results from the various sources
3. Implement the search procedure to get a document ranking
4. Integrate an LLM to summarize the k best results.
5. Create the frontend website

Task division (tentative)

- Databases: John
- Backend/search: Vivek
- LLM integration: Moksh