Course Topic Explorer

CS 410 Project Proposal (Free Topic)

OVERVIEW

The aim of this project is to build a comprehensive search system for students looking for the different courses that US universities are offering. We intend to implement a search engine with some recommendation features built based on the users' recent search history. The major component of this project is the search feature, which the students can use to get the courses relevant to the keywords they enter. An advanced feature that we intend to build is a simple recommendation system that will recommend suitable programs that the student can apply to get knowledge on the topics they have been searching for.

GOALS

The goals of the project are:

- Crawl multiple university websites' course catalogs.
- Scrape the text data from those websites and store it in a structured format.
- Enable smart search for students looking to take courses from top universities.
- Implement some advanced search and recommendation features, including recommending users programs (for example, Master's in Information Sciences at XYZ university) based on their recent topic search history.

TEAM

The team name is HREC. This team consists of **Harita Reddy (haritar2)** and **Eric Crawford (ecraw3)**. The team captain will be **Harita Reddy**.

DESCRIPTION

Motivation

Currently, there is no easy way to search for courses offered by universities that cover a topic a student may be interested in. Most universities allow you to do small text searches about their courses, but those searches are not aware of offerings at other universities. Searching across multiple universities will give students a little more information to help them plan what school to attend and what courses to take at those universities. We wish to make it easier for students by getting all the information in one place, without spending hours browsing to get what they need.

Approach

The first step in our project is to scrape the data from university course catalog websites. This is a time-taking step because the course catalog websites of different universities are organized and formatted differently. We will need to study each site to understand how they tag the relevant information we are looking for in HTML. This information includes things such as course number, class location, teacher, and description of course. We need to arrive at a common format and store the required data from the scraped websites into this format.

After having the required data, we plan to implement a **smart search** for users using the application. This work will have two components (1) Search and Ranking implementation in the backend, and (2) User Interface in the frontend.

After implementing the basic search features, we intend to implement some advanced search and recommendation features. When a student searches for a course, recommend an online or an offline program offered by a university that caters to what the student wants to learn based on the user's recent search history. Some other advanced features include linking the MOOC if the course is being offered online.

Milestone 1

Crawl and scrape the course catalog websites of selected universities. We are limiting our crawler to 10 universities due to time constraints, universities having multiple colleges, and because the course catalogs and program information websites of different universities are organized in different ways and can be quite unstructured. This is expected to take 10-15 hours.

We will also start setting up the backend server to store the crawled and scraped data. The server can be used to also index and rank each course, then exposed to the frontend via restful APIs.

Milestone 2

Work on the **search features** and build the **user interface** for the search either through a web or a desktop application. Implement basic search features on the data, including searching the most relevant courses based on the keywords. This is expected to take 20 hours because we will also be evaluating different similarity and ranking measures to obtain the best-suited algorithms for the job.

Milestone 3

We intend to implement some advanced search and recommendation features in this phase. Some of the features that we will try to implement include:

- When a student searches for a course, recommend an online or an offline program offered by a university that caters to what the student wants to learn. An interesting problem to explore would be if we can leverage item-based similarity to find programs that might be interesting to the student based on their recent search history.
- If the course is actually being offered online as a MOOC via a platform such as Coursera, give a link to the MOOC.
- Optional feature: Display the department (school within a university) that is offering the course and rank the departments based on the number of courses that the department is offering related to the topic the student is looking for. The user will also be able to drill down into the provided results to view the list of courses under that department that offers the course.

This phase is expected to take 20 hours.

Work Distribution

This project is expected to take between 50-60 hours to complete and tasked out as follows:

- Frontend Web/Desktop code -Eric
- Backend Server infrastructure - Harita/Eric
- Web scraping - Harita
- Rank and Search implementation - Harita/Eric

Tools, Systems, and Datasets

We plan to use the following tools for our project:

- User Interface:
  - Desktop/Web App using Kotlin Compose Desktop
- Backend Server
  - Django
- Web Scraping:

    - BeautifulSoup
    - Selenium
- Search and Ranking:
    - MeTAPy
    - Apache Lucene (Optional)

Dataset will be generated from the scraped data.

Programming Languages

The programming languages that we intend to use include Python and Kotlin. Kotlin will be mainly used for the App, whereas Python will be used for all the backend work such as scrawling, scaping, search, ranking, and recommendation.

Expected Outcome and Evaluation

The expected outcome of the project is a working UI that allows users to enter keywords to search for courses offered by different universities. The UI should display the relevant courses in the order of relevance. If the user has searched multiple times with keywords, the system should recommend a program offered by the US university based on the user's search history.

We plan to evaluate the system by generating test cases for search. We will manually generate test cases containing the search keywords and the expected list of results. Using metrics such as Precision@10 documents, Average Precision, and Normalized Discounted Cumulative Gain, we will evaluate the search results.