Advancing Personalized Computer Science Education: An Information Retrieval Perspective

Lois Wong
Computer Science Department
Johns Hopkins University
lwong23@jhu.edu

Amanda Ferber
Computer Science Department
Johns Hopkins University
aferber2@jhu.edu

Abstract

To support Computer Science self-learners in finding their ideal course from the extensive selection of online learning resources, we leverage web crawling and Huggingface's Sentence-Transformers to recommend open-access CS courseware from users' plain text requests.

1 Problem Statement

Navigating the changing landscape of Computer Science is more difficult than ever, for both newcomers to the field and seasoned professionals aiming to stay informed and up-to-date on trends. With a plethora of decentralized resources and generic guides to learning Artificial Intelligence or Machine Learning from scratch, finding the right course to suit your needs and help you carve out your niche in this field can be overwhelming.

As of now, self-learners are left to haphazardly browse the web and enroll in the first or most affordable course they find–a method that not only impedes their satisfaction, but also neglects to consider their unique backgrounds, interests and potential contributions to the field. We aim to solve this problem by developing a tool that aggregates resources from different open courseware sites and recommends courses to users based on a plain text prompt about what they want to learn.

2 Data & Methods

2.1 Web Crawling for Data Collection

To centralize and organize existing learning resources, this platform leverages web crawling to gather open-access Computer Science courses names and descriptions from MIT OpenCourseWare and Coursera. We consider web crawling the optimal approach for this task because it allows us to efficiently gather a wide range of up-to-date information from diverse sources, ensuring our database is comprehensive and current. Additionally, web crawling offers the advantage of automating the data collection process, saving valuable time and resources compared to manual entry methods.

2.1.1 Web Crawling Statement

We acknowledge the importance of good robot "citizenship" and are committed to ensuring that our web robot operates within the guidelines set by the websites we visit. Our web crawler adheres to the directives specified in the robots.txt files of the websites and complies with the restrictions and limitations imposed by the website administrators. We have limited the scope of our web crawling activities to only the segments of the web that are directly relevant to our research purposes. For this project, we access each course webpage from the results of the educational sites' course search filtered to "Computer Science" results and collect the course title and description.

2.2 Course Recommendation

For the task of recommending courses to users, we provide the user with two methods of searching through our course database.

The first option begins by prompting the user to enter a token, which could be a keyword or phrase of interest. Subsequently, it filters our database of courses to include only those whose title or description contains the user input. This approach differs from existing platforms like Coursera, which often suggest related courses when a keyword doesn't yield exact matches. In contrast, this search method only returns courses directly relevant to the user's query, thus preventing the user from sifting through potentially irrelevant results. By providing more precise recommendations, this method saves users valuable time and effort in finding the most suitable courses for their needs.

Our second method begins by prompting the user to input their goals and objectives for an online Computer Science course. We utilize Hugging-

Face's SentenceTransformer model, paraphrase-MiniLM-L6-v2, to compute sentence embeddings for both our scraped course descriptions and the user's stated goals. Finally, we compute the Cosine similarity between each course description in our database and the user prompt and return the courses that best match the query, sorted in descending order.

3 How to Use our Code

Users who wish to navigate the open-source CS course landscape can run open-cs-course-recommender.ipynb on Colab or Jupyter in order to engage with the database we have created. This notebook allows for three main functionalities: (1) viewing all listings in our database, (2) searching the database via string input, and (3) entering a description of one's CS goals. Each functionality occurs by running one or more code blocks grouped by functionality heading ("Public Access Computer Science Course Database", "Search the database", and "Finding courses that suit your goals").

Functionality (1) allows the user to view all courses in the database, ten courses at a time. For each course, the title, description, link, and source (either MIT OpenCourseware or Coursera) is displayed. After viewing ten courses, the user can expand the output in order to view ten more courses.

Functionality (2) allows the user to enter a token. After the token is entered, our program will search the title and description of each course for the appearance of that token. Courses containing the token in their titles or descriptions will be outputted, ten courses at a time. If the token does not appear in any course title or description, our program will output "No Matches."

Functionality (3) allows the user to enter a description of their learning goals for their CS education. After they enter this description, the course descriptions that have the highest cosine similarity score to the user-entered description will be outputted in ranked order.

4 Example Use Case and Project Functionalities

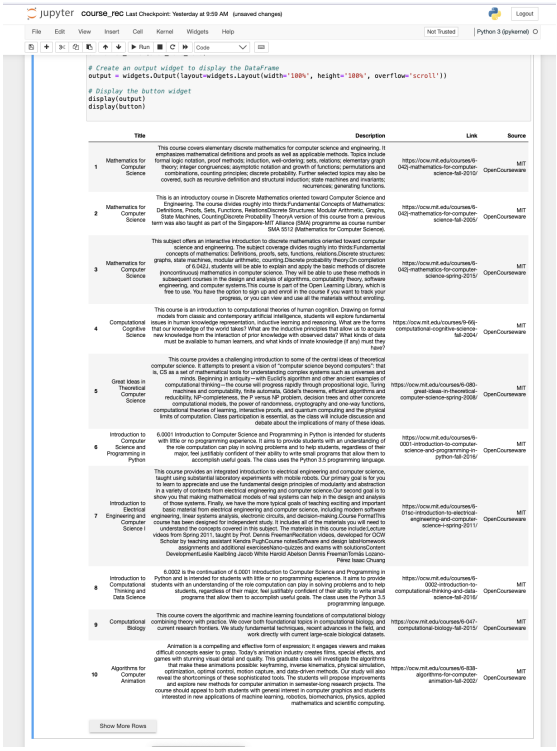

Figure 1: Our database of all Computer Science courses on MITOpenCourseware and Coursera. The title, description, link, and source of each course is displayed. The user has the option to click "Show More Rows," allows the user to see more course entries from the database.

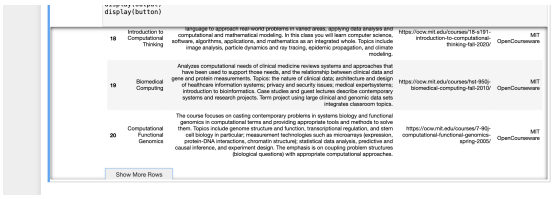

Figure 2: Upon clicking "Show More Rows" and scrolling to the bottom of the list, the user will have seen the next ten courses in the database. The user can click "Show More Rows" until all course listings have been displayed.

Figure 3: After running the code blocks that display the entire database, the user has the option to provide a token with which to search the database. If the token appears in the title or description of a course, that course will be outputted.

Figure 4: No courses contain the string "intro python" in their title or description.

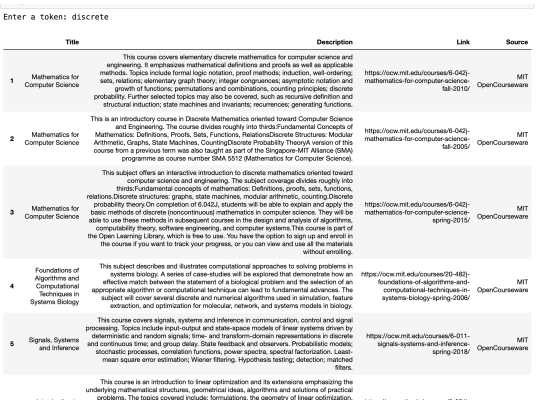

Figure 5: The above courses contain the word "discrete" in their titles, descriptions, or both. The courses are outputted in the same manner as the entire database.

Figure 6: The user has the option to click "Show More Rows" and view more courses containing the token for which they searched in the title or description.

Figure 7: After running the code blocks required to search the courses for a given token, the user has the option to enter a description of their learning goals in the field of Computer Science. Once they enter this description, courses will be outputted that match these goals.

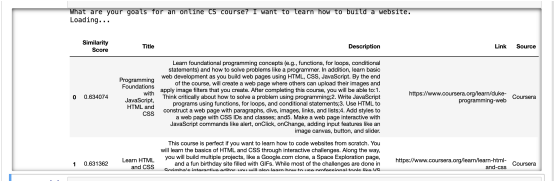

Figure 8: The above courses yield the highest similarity score to the query "I want to learn how to build a website."

5 Evaluation

To evaluate our system, we compare the course recommendation results of our second course search system with those from the original open-source learning platforms for three prompts over the top

5, top 10, and top 15 search results, and calculate precision at each level. We consider precision to be the appropriate metric for evaluating our system because it reflects our goal of ensuring users find the most relevant courses quickly and efficiently. Given that users are unlikely to look beyond the top few search results, optimizing for precision ensures that the courses recommended by our system align closely with user search intent, ultimately enhancing the user experience and increasing the likelihood of successful course discovery.

To evaluate our system, we manually tag each platform's search results as relevant if they align closely with the user search intent and provide valuable content. Precision at each level is calculated as the ratio of relevant courses retrieved by a system to the total number of courses retrieved at each level of search.

Prompts:

1. I have no coding experience and want to learn how to code
2. I want to learn about Artificial Intelligence and Machine Learning
3. I want to learn to build a ChatBot

Prompt	Precision at 5	Precision at 10	Precision at 15
1	0.0	0.0	0.07
2	1.0	0.9	0.8
3	0.0	0.0	0.0

Table 1: Amanda's Relevancy Judgement of MIT OpenCourseware

Prompt	Precision at 5	Precision at 10	Precision at 15
1	0.8	0.5	0.5
2	1.0	0.8	0.73
3	0.4	0.3	0.2

Table 2: Amanda's Relevancy Judgement of Coursera

Prompt	Precision at 5	Precision at 10	Precision at 15
1	0.8	0.7	0.6
2	1.0	1.0	1.0
3	0.4	0.5	0.6

Table 3: Amanda's Relevancy Judgement of our system

Prompt	Precision at 5	Precision at 10	Precision at 15
1	0.0	0.0	0.13
2	0.8	0.5	0.4
3	0.0	0.0	0.0

Table 4: Lois' relevancy judgement of MIT Open Courseware

Prompt	Precision at 5	Precision at 10	Precision at 15
1	0.6	0.57	0.57
2	0.6	0.4	0.27
3	0.4	0.2	0.13

Table 5: Lois' relevancy judgement of Coursera

Prompt	Precision at 5	Precision at 10	Precision at 15
1	0.8	0.8	0.87
2	0.8	0.9	0.67
3	0.4	0.2	0.13

Table 6: Lois' relevancy judgement of our system

6 Discussion

6.1 Achievements

The strengths and achievements of our projects are enumerated below:

1. Innovative Use of Plain Text Queries: Our project introduces a novel approach of allowing users to input plain text queries about what they want to learn by leveraging developing LLM technology, ultimately providing a more intuitive and flexible interface.

2. User-Friendly Interface: Encoding prompts and course information using an LLM provides the option for users to write longer, more detailed queries. This enhances the user experience and encourages deeper engagement with the system and significantly contributes to the usability and effectiveness of the tool.

3. Centralizing Open Courseware: By scraping and aggregating 1,233 courses from several open courseware sites, our project solves the problem of users having to manually search through multiple sites to find relevant courses. This not only saves time but also ensures that users have access to a comprehensive range of options.

6.2 Areas for Improvement

We have identified several areas for improvement in our project:

1. Expand Dataset: We plan to expand the number of open courseware sites we scrape to ensure a more comprehensive dataset.

2. Prompt Engineering Experimentation: We intend to experiment with prompt engineering techniques to enhance the quality of our search results.

3. Alternative Similarity Metrics Exploration: We plan to explore alternative similarity metrics beyond Cosine, as it could lead to better results.

4. User Feedback Incorporation: We hope to incorporate user feedback by conducting evaluations to help us refine our approach to ensure it satisfies user needs.

6.3 Future Work

We intend to continue this project by developing an LLM Retrieval Augmented Generation ChatBot that empowers people to achieve their goals by generating personalized learning pathways consisting of open-access courses tailored to individuals' specific backgrounds, needs, and ambitions. The system will additionally allow the user to provide feedback on the generated learning path, which can be used to further refine the recommendations. We believe that education is a transformative tool that can be used in rehabilitation and reintegration, and have ambitions to establish an EdTech social enterprise dedicated to advancing global access to education and providing personalized learning pathways that empower people to achieve their goals, both within and outside the domain of Computer Science. Ultimately, we hope to collaborate with both domestic and global humanitarian organizations to contribute to the successful integration of refugees into new communities, aiding survivors of crime and homelessness in rebuilding their lives, and creating pathways for those lacking educational opportunities to realize their full potential.