

ADVANCING PERSONALIZED COMPUTER SCIENCE EDUCATION: AN INFORMATION RETRIEVAL PERSPECTIVE

by

Lois Wong lwong23@jhu.edu

Amanda Ferber aferber2@jhu.edu

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

Baltimore, Maryland

May 2024

© 2024 Lois Wong and Amanda Ferber
All rights reserved

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 HuggingFace'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

[Link to GitHub.](#)


Users who wish to navigate the open-source CS course landscape can run `course_rec.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



course_rec Last Checkpoint: Yesterday at 9:59 AM (unsaved changes)


Logout


File Edit View Insert Cell Kernel Widgets Help

Run



Code



```

# Create an output widget to display the DataFrame
output = widgets.Output(layout=widgets.Layout(width='100%', height='100%', overflow='scroll'))

# Display the button widget
display(output)
display(button)

```

	Title	Description	Link	Source
1	Mathematics for Computer Science	This course covers elementary discrete mathematics for computer science and engineering. It emphasizes mathematical definitions and proofs as well as applicable methods. Topics include formal logic notation, proof methods; induction, well-ordering; sets, relations; elementary graph theory; integer congruences; asymptotic notation and growth of functions; permutations and combinations, counting principles; discrete probability. Further selected topics may also be covered, such as recursive definition and structural induction; state machines and invariants; recurrences; generating functions.	https://ocw.mit.edu/courses/6-042j-mathematics-for-computer-science-fall-2010/	MIT OpenCourseware
2	Mathematics for Computer Science	This is an introductory course in Discrete Mathematics oriented toward Computer Science and Engineering. The course divides roughly into thirds: Fundamental Concepts of Mathematics: Definitions, Proofs, Sets, Functions, Relations Discrete Structures: Modular Arithmetic, Graphs, State Machines, Counting Discrete Probability Theory A version of this course from a previous term was also taught as part of the Singapore-MIT Alliance (SMA) programme as course number SMA 5512 (Mathematics for Computer Science).	https://ocw.mit.edu/courses/6-042j-mathematics-for-computer-science-fall-2005/	MIT OpenCourseware
3	Mathematics for Computer Science	This subject offers an interactive introduction to discrete mathematics oriented toward computer science and engineering. The subject coverage divides roughly into thirds: Fundamental concepts of mathematics: Definitions, proofs, sets, functions, relations. Discrete structures: graphs, state machines, modular arithmetic, counting. Discrete probability theory. On completion of 6.042J, students will be able to explain and apply the basic methods of discrete (noncontinuous) mathematics in computer science. They will be able to use these methods in subsequent courses in the design and analysis of algorithms, computability theory, software engineering, and computer systems. This course is part of the Open Learning Library, which is free to use. You have the option to sign up and enroll in the course if you want to track your progress, or you can view and use all the materials without enrolling.	https://ocw.mit.edu/courses/6-042j-mathematics-for-computer-science-spring-2015/	MIT OpenCourseware
4	Computational Cognitive Science	This course is an introduction to computational theories of human cognition. Drawing on formal models from classic and contemporary artificial intelligence, students will explore fundamental issues in human knowledge representation, inductive learning and reasoning. What are the forms that our knowledge of the world takes? What are the inductive principles that allow us to acquire new knowledge from the interaction of prior knowledge with observed data? What kinds of data must be available to human learners, and what kinds of innate knowledge (if any) must they have?	https://ocw.mit.edu/courses/9-66j-computational-cognitive-science-fall-2004/	MIT OpenCourseware
5	Great Ideas in Theoretical Computer Science	This course provides a challenging introduction to some of the central ideas of theoretical computer science. It attempts to present a vision of "computer science beyond computers": that is, CS as a set of mathematical tools for understanding complex systems such as universes and minds. Beginning in antiquity—with Euclid's algorithm and other ancient examples of computational thinking—the course will progress rapidly through propositional logic, Turing machines and computability, finite automata, Gödel's theorems, efficient algorithms and reducibility, NP-completeness, the P versus NP problem, decision trees and other concrete computational models, the power of randomness, cryptography and one-way functions, computational theories of learning, interactive proofs, and quantum computing and the physical limits of computation. Class participation is essential, as the class will include discussion and debate about the implications of many of these ideas.	https://ocw.mit.edu/courses/6-080-great-ideas-in-theoretical-computer-science-spring-2008/	MIT OpenCourseware
6	Introduction to Computer Science and Programming in Python	6.0001 Introduction to Computer Science and Programming in Python is intended for students with little or no programming experience. It aims to provide students with an understanding of the role computation can play in solving problems and to help students, regardless of their major, feel justifiably confident of their ability to write small programs that allow them to accomplish useful goals. The class uses the Python 3.5 programming language.	https://ocw.mit.edu/courses/6-0001-introduction-to-computer-science-and-programming-in-python-fall-2016/	MIT OpenCourseware
7	Introduction to Electrical Engineering and Computer Science I	This course provides an integrated introduction to electrical engineering and computer science, taught using substantial laboratory experiments with mobile robots. Our primary goal is for you to learn to appreciate and use the fundamental design principles of modularity and abstraction in a variety of contexts from electrical engineering and computer science. Our second goal is to show you that making mathematical models of real systems can help in the design and analysis of those systems. Finally, we have the more typical goals of teaching exciting and important basic material from electrical engineering and computer science, including modern software engineering, linear systems analysis, electronic circuits, and decision-making. Course Format This course has been designed for independent study. It includes all of the materials you will need to understand the concepts covered in this subject. The materials in this course include: Lecture videos from Spring 2011, taught by Prof. Dennis Freeman Recitation videos, developed for OCW Scholar by teaching assistant Kendra Pugh Course notes Software and design labs Homework assignments and additional exercises Nano-quizzes and exams with solutions Content Development Leslie Kaelbling Jacob White Harold Abelson Dennis Freeman Tomás Lozano-Pérez Isaac Chuang	https://ocw.mit.edu/courses/6-01sc-introduction-to-electrical-engineering-and-computer-science-i-spring-2011/	MIT OpenCourseware
8	Introduction to Computational Thinking and Data Science	6.0002 is the continuation of 6.0001 Introduction to Computer Science and Programming in Python and is intended for students with little or no programming experience. It aims to provide students with an understanding of the role computation can play in solving problems and to help students, regardless of their major, feel justifiably confident of their ability to write small programs that allow them to accomplish useful goals. The class uses the Python 3.5 programming language.	https://ocw.mit.edu/courses/6-0002-introduction-to-computational-thinking-and-data-science-fall-2016/	MIT OpenCourseware
9	Computational Biology	This course covers the algorithmic and machine learning foundations of computational biology, combining theory with practice. We cover both foundational topics in computational biology, and current research frontiers. We study fundamental techniques, recent advances in the field, and work directly with current large-scale biological datasets.	https://ocw.mit.edu/courses/6-047-computational-biology-fall-2015/	MIT OpenCourseware
10	Algorithms for Computer Animation	Animation is a compelling and effective form of expression; it engages viewers and makes difficult concepts easier to grasp. Today's animation industry creates films, special effects, and games with stunning visual detail and quality. This graduate class will investigate the algorithms that make these animations possible: keyframing, inverse kinematics, physical simulation, optimization, optimal control, motion capture, and data-driven methods. Our study will also reveal the shortcomings of these sophisticated tools. The students will propose improvements and explore new methods for computer animation in semester-long research projects. The course should appeal to both students with general interest in computer graphics and students interested in new applications of machine learning, robotics, biomechanics, physics, applied mathematics and scientific computing.	https://ocw.mit.edu/courses/6-838-algorithms-for-computer-animation-fall-2002/	MIT OpenCourseware

Show More Rows

iv

Figure 1: Our database of all Computer Science courses on MIT OpenCourseware 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.

		display(button)		
18	Introduction to Computational Thinking	language to approach real-world problems in varied areas, applying data analysis and computational and mathematical modeling. In this class you will learn computer science, software, algorithms, applications, and mathematics as an integrated whole. Topics include image analysis, particle dynamics and ray tracing, epidemic propagation, and climate modeling.	https://ocw.mit.edu/courses/18-s191-introduction-to-computational-thinking-fall-2020/	MIT OpenCourseware
19	Biomedical Computing	Analyzes computational needs of clinical medicine reviews systems and approaches that have been used to support those needs, and the relationship between clinical data and gene and protein measurements. Topics: the nature of clinical data; architecture and design of healthcare information systems; privacy and security issues; medical expertsystems; introduction to bioinformatics. Case studies and guest lectures describe contemporary systems and research projects. Term project using large clinical and genomic data sets integrates classroom topics.	https://ocw.mit.edu/courses/hst-950j-biomedical-computing-fall-2010/	MIT OpenCourseware
20	Computational Functional Genomics	The course focuses on casting contemporary problems in systems biology and functional genomics in computational terms and providing appropriate tools and methods to solve them. Topics include genome structure and function, transcriptional regulation, and stem cell biology in particular; measurement technologies such as microarrays (expression, protein-DNA interactions, chromatin structure); statistical data analysis, predictive and causal inference, and experiment design. The emphasis is on coupling problem structures (biological questions) with appropriate computational approaches.	https://ocw.mit.edu/courses/7-90j-computational-functional-genomics-spring-2005/	MIT OpenCourseware
		Show More Rows		

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.

Enter a token:

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.

Enter a token: intro python
No matches

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

Enter a token: discrete

	Title	Description	Link	Source
1	Mathematics for Computer Science	This course covers elementary discrete mathematics for computer science and engineering. It emphasizes mathematical definitions and proofs as well as applicable methods. Topics include formal logic notation, proof methods; induction, well-ordering; sets, relations; elementary graph theory; integer congruences; asymptotic notation and growth of functions; permutations and combinations, counting principles; discrete probability. Further selected topics may also be covered, such as recursive definition and structural induction; state machines and invariants; recurrences; generating functions.	https://ocw.mit.edu/courses/6-042j-mathematics-for-computer-science-fall-2010/	MIT OpenCourseware
2	Mathematics for Computer Science	This is an introductory course in Discrete Mathematics oriented toward Computer Science and Engineering. The course divides roughly into thirds: Fundamental Concepts of Mathematics: Definitions, Proofs, Sets, Functions, Relations Discrete Structures: Modular Arithmetic, Graphs, State Machines, Counting Discrete Probability Theory A version of this course from a previous term was also taught as part of the Singapore-MIT Alliance (SMA) programme as course number SMA 5512 (Mathematics for Computer Science).	https://ocw.mit.edu/courses/6-042j-mathematics-for-computer-science-fall-2005/	MIT OpenCourseware
3	Mathematics for Computer Science	This subject offers an interactive introduction to discrete mathematics oriented toward computer science and engineering. The subject coverage divides roughly into thirds: Fundamental concepts of mathematics: Definitions, proofs, sets, functions, relations. Discrete structures: graphs, state machines, modular arithmetic, counting. Discrete probability theory. On completion of 6.042J, students will be able to explain and apply the basic methods of discrete (noncontinuous) mathematics in computer science. They will be able to use these methods in subsequent courses in the design and analysis of algorithms, computability theory, software engineering, and computer systems. This course is part of the Open Learning Library, which is free to use. You have the option to sign up and enroll in the course if you want to track your progress, or you can view and use all the materials without enrolling.	https://ocw.mit.edu/courses/6-042j-mathematics-for-computer-science-spring-2015/	MIT OpenCourseware
4	Foundations of Algorithms and Computational Techniques in Systems Biology	This subject describes and illustrates computational approaches to solving problems in systems biology. A series of case-studies will be explored that demonstrate how an effective match between the statement of a biological problem and the selection of an appropriate algorithm or computational technique can lead to fundamental advances. The subject will cover several discrete and numerical algorithms used in simulation, feature extraction, and optimization for molecular, network, and systems models in biology.	https://ocw.mit.edu/courses/20-482j-foundations-of-algorithms-and-computational-techniques-in-systems-biology-spring-2006/	MIT OpenCourseware
5	Signals, Systems and Inference	This course covers signals, systems and inference in communication, control and signal processing. Topics include input-output and state-space models of linear systems driven by deterministic and random signals; time- and transform-domain representations in discrete and continuous time; and group delay. State feedback and observers. Probabilistic models; stochastic processes, correlation functions, power spectra, spectral factorization. Least-mean square error estimation; Wiener filtering. Hypothesis testing; detection; matched filters.	https://ocw.mit.edu/courses/6-011-signals-systems-and-inference-spring-2018/	MIT OpenCourseware
		This course is an introduction to linear optimization and its extensions emphasizing the underlying mathematical structures, geometrical ideas, algorithms and solutions of practical problems. The topics covered include: formulations, the geometry of linear optimization.		

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.

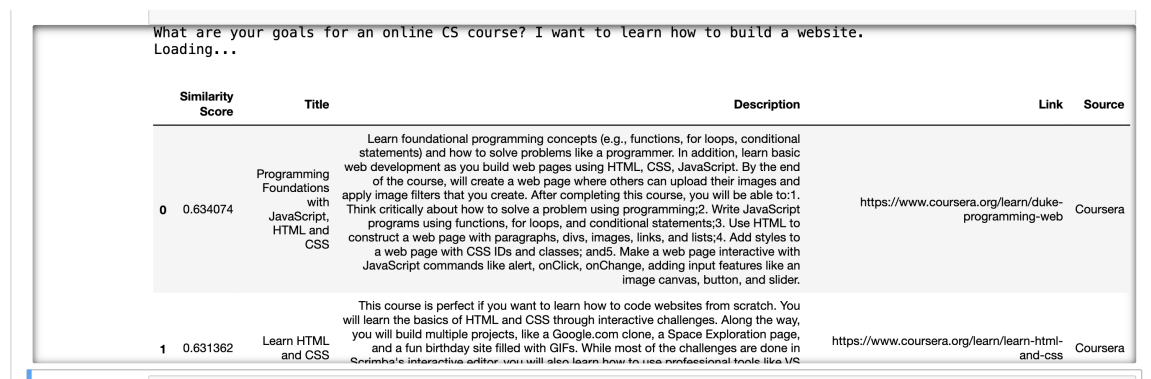
Enter a token: discrete

	Title	Description	Link	Source
1	Mathematics for Computer Science	This course covers elementary discrete mathematics for computer science and engineering. It emphasizes mathematical definitions and proofs as well as applicable methods. Topics include formal logic notation, proof methods; induction, well-ordering; sets, relations; elementary graph theory; integer congruences; asymptotic notation and growth of functions; permutations and combinations, counting principles; discrete probability. Further selected topics may also be covered, such as recursive definition and structural induction; state machines and invariants; recurrences; generating functions.	https://ocw.mit.edu/courses/6-042j-mathematics-for-computer-science-fall-2010/	MIT OpenCourseware
2	Mathematics for Computer Science	This is an introductory course in Discrete Mathematics oriented toward Computer Science and Engineering. The course divides roughly into thirds: Fundamental Concepts of Mathematics: Definitions, Proofs, Sets, Functions, Relations Discrete Structures: Modular Arithmetic, Graphs, State Machines, Counting Discrete Probability Theory A version of this course from a previous term was also taught as part of the Singapore-MIT Alliance (SMA) programme as course number SMA 5512 (Mathematics for Computer Science).	https://ocw.mit.edu/courses/6-042j-mathematics-for-computer-science-fall-2005/	MIT OpenCourseware
3	Mathematics for Computer Science	This subject offers an interactive introduction to discrete mathematics oriented toward computer science and engineering. The subject coverage divides roughly into thirds: Fundamental concepts of mathematics: Definitions, proofs, sets, functions, relations. Discrete structures: graphs, state machines, modular arithmetic, counting. Discrete probability theory. On completion of 6.042J, students will be able to explain and apply the basic methods of discrete (noncontinuous) mathematics in computer science. They will be able to use these methods in subsequent courses in the design and analysis of algorithms, computability theory, software engineering, and computer systems. This course is part of the Open Learning Library, which is free to use. You have the option to sign up and enroll in the course if you want to track your progress, or you can view and use all the materials without enrolling.	https://ocw.mit.edu/courses/6-042j-mathematics-for-computer-science-spring-2015/	MIT OpenCourseware
4	Foundations of Algorithms and Computational Techniques in Systems Biology	This subject describes and illustrates computational approaches to solving problems in systems biology. A series of case-studies will be explored that demonstrate how an effective match between the statement of a biological problem and the selection of an appropriate algorithm or computational technique can lead to fundamental advances. The subject will cover several discrete and numerical algorithms used in simulation, feature extraction, and optimization for molecular, network, and systems models in biology.	https://ocw.mit.edu/courses/20-482j-foundations-of-algorithms-and-computational-techniques-in-systems-biology-spring-2006/	MIT OpenCourseware
5	Signals, Systems and Inference	This course covers signals, systems and inference in communication, control and signal processing. Topics include input-output and state-space models of linear systems driven by deterministic and random signals; time- and transform-domain representations in discrete and continuous time; and group delay. State feedback and observers. Probabilistic models; stochastic processes, correlation functions, power spectra, spectral factorization. Least-mean square error estimation; Wiener filtering. Hypothesis testing; detection; matched filters.	https://ocw.mit.edu/courses/6-011-signals-systems-and-inference-spring-2018/	MIT OpenCourseware
6	Introduction to Mathematical Programming	This course is an introduction to linear optimization and its extensions emphasizing the underlying mathematical structures, geometrical ideas, algorithms and solutions of practical problems. The topics covered include: formulations, the geometry of linear optimization, duality theory, the simplex method, sensitivity analysis, robust optimization, large scale optimization network flows, solving problems with an exponential number of constraints and the ellipsoid method, interior point methods, semidefinite optimization, solving real world problems with computer software, discrete optimization formulations and algorithms.	https://ocw.mit.edu/courses/6-251j-introduction-to-mathematical-programming-fall-2009/	MIT OpenCourseware
7	Discrete-Time Signal Processing	This class addresses the representation, analysis, and design of discrete time signals and systems. The major concepts covered include: Discrete-time processing of continuous-time signals; decimation, interpolation, and sampling rate conversion; flowgraph structures for DT systems; time- and frequency-domain design techniques for recursive (IIR) and non-recursive (FIR) filters; linear prediction; discrete Fourier transform, FFT algorithm; short-time Fourier analysis and filter banks; multirate techniques; Hilbert transforms; Cepstral analysis and various applications. Acknowledgements I would like to express my thanks to Thomas Baran, Myung Jin Choi, and Xiaomeng Shi for compiling the lecture notes on this site from my individual lectures and handouts and their class notes during the semesters that they were students in the course. These lecture notes, the text book and included problem sets and solutions will hopefully be helpful as you learn and explore the topic of Discrete-Time Signal Processing.	https://ocw.mit.edu/courses/6-341-discrete-time-signal-processing-fall-2005/	MIT OpenCourseware
8	Optimization Methods	This course introduces the principal algorithms for linear, network, discrete, nonlinear, dynamic optimization and optimal control. Emphasis is on methodology and the underlying mathematical structures. Topics include the simplex method, network flow methods, branch and bound and cutting plane methods for discrete optimization, optimality conditions for nonlinear optimization, interior point methods for convex optimization, Newton's method, heuristic methods, and dynamic programming and optimal control methods.	https://ocw.mit.edu/courses/15-093j-optimization-methods-fall-2009/	MIT OpenCourseware
9	Introduction to Communication, Control, and Signal Processing	This course examines signals, systems and inference as unifying themes in communication, control and signal processing. Topics include input-output and state-space models of linear systems driven by deterministic and random signals; time- and transform-domain representations in discrete and continuous time; group delay; state feedback and observers; probabilistic models; stochastic processes, correlation functions, power spectra, spectral factorization; least-mean square error estimation; Wiener filtering; hypothesis testing; detection; matched filters.	https://ocw.mit.edu/courses/6-011-introduction-to-communication-control-and-signal-processing-spring-2010/	MIT OpenCourseware
10	Signals and Systems	6.003 covers the fundamentals of signal and system analysis, focusing on representations of discrete-time and continuous-time signals (singularity functions, complex exponentials and geometrics, Fourier representations, Laplace and Z transforms, sampling) and representations of linear, time-invariant systems (difference and differential equations, block diagrams, system functions, poles and zeros, convolution, impulse and step responses, frequency responses). Applications are drawn broadly from engineering and physics, including feedback and control, communications, and signal processing.	https://ocw.mit.edu/courses/6-003-signals-and-systems-fall-2011/	MIT OpenCourseware
Show More Rows				

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.

What are your goals for an online CS course?

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.



	Similarity Score	Title	Description	Link	Source
0	0.634074	Programming Foundations with JavaScript, HTML and CSS	Learn foundational programming concepts (e.g., functions, for loops, conditional statements) and how to solve problems like a programmer. In addition, learn basic web development as you build web pages using HTML, CSS, JavaScript. By the end of the course, you will create a web page where others can upload their images and apply image filters that you create. After completing this course, you will be able to:1. Think critically about how to solve a problem using programming;2. Write JavaScript programs using functions, for loops, and conditional statements;3. Use HTML to construct a web page with paragraphs, divs, images, links, and lists;4. Add styles to a web page with CSS IDs and classes; and5. Make a web page interactive with JavaScript commands like alert, onClick, onChange, adding input features like an image canvas, button, and slider.	https://www.coursera.org/learn/duke-programming-web	Coursera
1	0.631362	Learn HTML and CSS	This course is perfect if you want to learn how to code websites from scratch. You will learn the basics of HTML and CSS through interactive challenges. Along the way, you will build multiple projects, like a Google.com clone, a Space Exploration page, and a fun birthday site filled with GIFs. While most of the challenges are done in Srimba's interactive editor, you will also learn how to use professional tools like VS	https://www.coursera.org/learn/learn-html-and-css	Coursera

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. Complexity: Medium, Scope: Medium, Success: High.

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. Complexity: Medium, Scope: Medium, Success: High.
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. Complexity: Medium, Scope: Medium, Success: High.

6.2 Limitations

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

6.3 Future Work

We intend to continue this project by developing a ChatBot that empowers people to achieve their goals by generating personalized learning pathways consisting of open-access courseware that are 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.