Lumen: A Princeton Course Search and Recommendation Engine

COS435: Information Retrieval, Discovery, and Delivery

Christopher Hay Junhan Chen

<https://lumen-princeton.herokuapp.com>

<https://github.com/haychris/class-recs>

**I. Introduction**

Princeton offers myriad courses each semester across a multitude of departments, and for many students, choosing which courses to enroll in during course enrollment period can often be distressing. Not knowing what courses they want to take, need to take, or may be interested in, many students end up inefficiently browsing the course listings site (currently maintained by the Registrar’s Office) as if aimlessly browsing a catalog, while those who have some sense of what their interests are constrained to using basic query searches on fields such as course title or department.

Through Lumen, we aim to solve this problem by helping students quickly and easily search for relevant courses they would want to enroll in. Named after the standard measure of light, our product vision was to help students discover interesting and relevant courses they did not know existed in an easy and efficient manner, to inspire them to learn more and try out new fields by taking those courses, and to ultimately *enlighten* them. Lumen features a fast query search option that improves upon the Registrar’s Office basic search features, allowing students to search for classes using queries such as “interesting LA” or “engaging lectures” and quickly obtain relevant results. An alternative personalized search option instead utilizes data on courses the student has previously taken, then returns tailored course recommendations based on how much the student has liked or disliked his or her past courses.

The remainder of this paper is organized as follows: *II. Goals & Measurement* defines Lumen’s goals and metrics to measure success, *III*. *Product* describes Lumen’s features, use cases, and component implementations, while *IV. Results* presents and discusses Lumen’s ability to achieve its goal as measured via metrics and user studies.

**II. Goals & Measurement**

Lumen’s main goal is to enlighten student users about courses that interest them by allowing the user to easily perform a search and efficiently receive relevant results. To measure success, we will ask selected users to rank all retrieved results as not relevant (score 0), slightly relevant (score 1) or relevant (score 5), and give one overall rating of the user experience. Using the result scores, we will then calculate precision, reciprocal rank, and expected reciprocal rank (ERR) to arrive at a total of four metrics to measure success. User experience will be rated on a 1 to 5 scale, while precision will be measured by taking the number of relevant results divided by the total number of retrieved results. Reciprocal rank is defined as 1 divided by the rank of the first relevant result, and ERR will be calculated using the following equation:

**III. Product**

Our product is hosted at <https://lumen-princeton.herokuapp.com>, while the code repository can be found at <https://github.com/haychris/class-recs>.

The index page provides a very brief pitch of Lumen and aims to give the user a first impression that consists of a very clean and consistent chalkboard-themed user experience. From the index page, the user can then either do a fast query search for a course if they already have an idea in mind, or allow us to recommend them courses by clicking on the “Enlighten me” button on the bottom. A workflow diagram is presented below:

Index Page

Query is processed

Has the user already inputted courses?

Course history data is processed

User inputs course history and ratings

Present query search results

Present personalized recommendations

Query search

Personalized recommendations

Yes

No

Query search <BACKEND DETAILS>. The top twenty results are then presented to the user, ranked in order of descending relevance. The left-hand component of each result was styled to be reminiscent of a chalkboard and presents information about the course’s department, number, title, and professor, while the right-hand component presents the overall rating of the course from the most recent semester it was offered, as well as a sample selection of student comments regarding the course. Clicking on the left-hand component of each score reveals a more detailed breakdown of each course’s ratings from the most recent offering into Classes, Coursework & Exams, Readings, Classes, and Overall, with the individual ratings presented in a chalk-colored bar graph format. A link to the course’s Registrar’s page is also provided for each result, allowing the user to quickly and conveniently view the course’s official page if the user wishes to do so.

If the user instead wants us to give personalized recommendations based on their course history instead (“Enlighten me”), then we check for a cookie that indicates the user has already provided their course history. If such a cookie does exist, then the user is taken directly to the results page. Otherwise, if no such cookie exists, then the user is redirected to an input page where they are asked to enter past courses and ratings (scale of 1-5 or “Hate” to “Amazing”) for each course. By default, this input page starts by showing freshman year with four courses per semester; users can then choose to add additional courses for each semester (max seven courses per semester) or add/remove years (max junior year) as needed. The user can also select their major and certificate if they wish to do so. Once the user finishes and clicks on the “Inspire” button, the inputted course data is then saved to a cookie and the top twenty personalized recommendations are presented to the user. These results are styled identically to the query search results so as to maintain a consistent user experience, but if a course happens to fulfill a major or certificate requirement for the user, then an additional notification will appear on the left-hand side of that result. If the user wishes to edit or update their course history, then they are also able to do so from the results page by clicking on the prompt near the top of the page. Course and ratings data will automatically be read and parsed from the cookie then loaded so as to save the user the hassle inputting everything again.

For both types of searches, all results with a department designation of WRI or FRS (i.e. all writing seminars or freshman seminars) are filtered out and discarded. As course offerings for both type of courses are very volatile and often change annually (e.g. FRS101 from one year is unlikely to have the same content as FRS101 the following year), we concluded that these courses will very rarely be relevant to the user and instead clutter the results page if returned. Thus we felt that the benefits of omitting all WRI and FRS courses outweighed the costs.

Personalized recommendations are generated by <BACKEND DETAILS>. Major and certificate requirements are currently limited to ECO, COS, and ANT and Applications of Computing, Creative Writing Certificate, and Certificate in Linguistics respectively, as this feature was meant to be a proof of concept at the time of writing.

WRI/FRS are filtered out

**IV. Results**

User studies were conducted on five users using both query search and personalized recommendations, and the four metrics as calculated from each user’s relevancy assessment for each search method are presented below:

Table A. Measurement of Results for Query Search

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Metric** | User 1 | User 2 | User 3 | User 4 | **Avg** |
| Query | “Easy PDF” |  |  |  |  |
| User Experience | 5 |  |  |  |  |
| Precision | 0.7 |  |  |  |  |
| Reciprocal Rank | 1 |  |  |  |  |
| ERR | 0.98 |  |  |  |  |

Table B. Measurement of Results for Personalized Recommendations

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Metric** | User 1 | User 2 | User 3 | User 4 | **Avg** |
| User Experience | 5 |  |  |  |  |
| Precision | 0.5 |  |  |  |  |
| Reciprocal Rank | 1 |  |  |  |  |
| ERR | 0.98 |  |  |  |  |

<discuss>

**V. Conclusion**

Lumen <success/room to be improved upon>.

Ideas for further exploration include testing the effectiveness of peer recommendation methods such as user- or course-based matrix completion (requires a critical mass of user course data), while additional features such as automatically retrieving and loading course history based on the user’s netID (assuming CAS support is added) would further improve the user experience.

Filtering courses that have been taken and are equivalent (e.g. ECO300 and ECO310)