Project Proposal

Xiaohan Tian (xtian13@illinois.edu)

1. Overview

For the CS 410 Text Information Systems course project, I would like to build a personalized search engine and recommendation system working as a Chrome Extension under the “Intelligent Browsing” theme.

1.1 Team Members

I will work as a single-developer team.

Name: Xiaohan Tian

NetID: xtian13

Email: xtian13@illinois.edu

1.2 Repository Information

As directed, all the documents and source code will be submitted to Github.

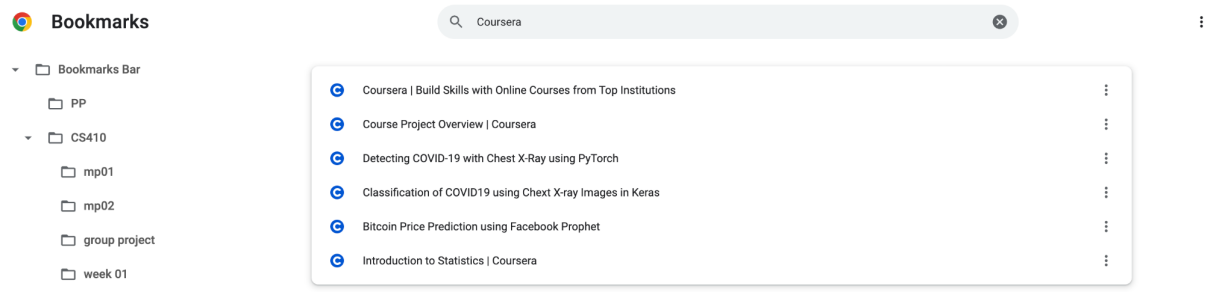
URL: https://github.com/xtian-uiuc/CS410-CourseProject/

2. Topic

2.1 Motivation: Why Choose This Topic and Why It Is a Problem?

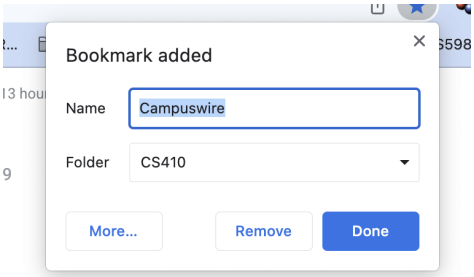
The motivation for this project comes from one of my own challenges in real life. Nowadays, the Internet and Browsers are essential parts of many different people. When browsing the Internet, we often bookmark those pages we would like to save to read them later or study the content when needed.

A big challenge is that all major browsers on the market (Google Chrome, Firefox, Safari, and Microsoft Edge) only support searching the pages in your bookmarks by their page titles and with strictly exact string match methods.



(Figure 1. Exact matching based query methods in Chrome Bookmark Manager)

This becomes very inconvenient when trying to query against the content of those bookmarked pages since most of the time, the page title is not even an abstract, it is merely a short sentence, and sometimes it is a simple word that cannot even describe the page content.



(Figure 2. Page title not always represents page content)

To resolve this problem, I would like to build a Chrome Extension which can be your search engine that can index and perform queries against the content of your saved page; additionally, it should be able to provide you with possible similar pages in your bookmarks according to the page you are visiting.

2.2 Relationship with the Theme and the Class

The form of the project will be a Chrome Extension, working to help users directly improve their browsing experience. Hopefully, after completing the project, this extension will help users retrieve bookmarked pages efficiently and accurately and avoid being confused by those page titles in the bookmark list.

Additionally, by completing the project, users can take advantage of the recommending system of the extension, can find all the related pages they saved easily, and avoid manually going through them one by one.

I believe this is one step towards intelligent browsing. Inside the project, the search engine part is a typical text retrieval system, which was covered in the first three weeks of the course. The

recommendation system belongs to the text filtering system, which was covered in the fourth to sixth weeks of the course.

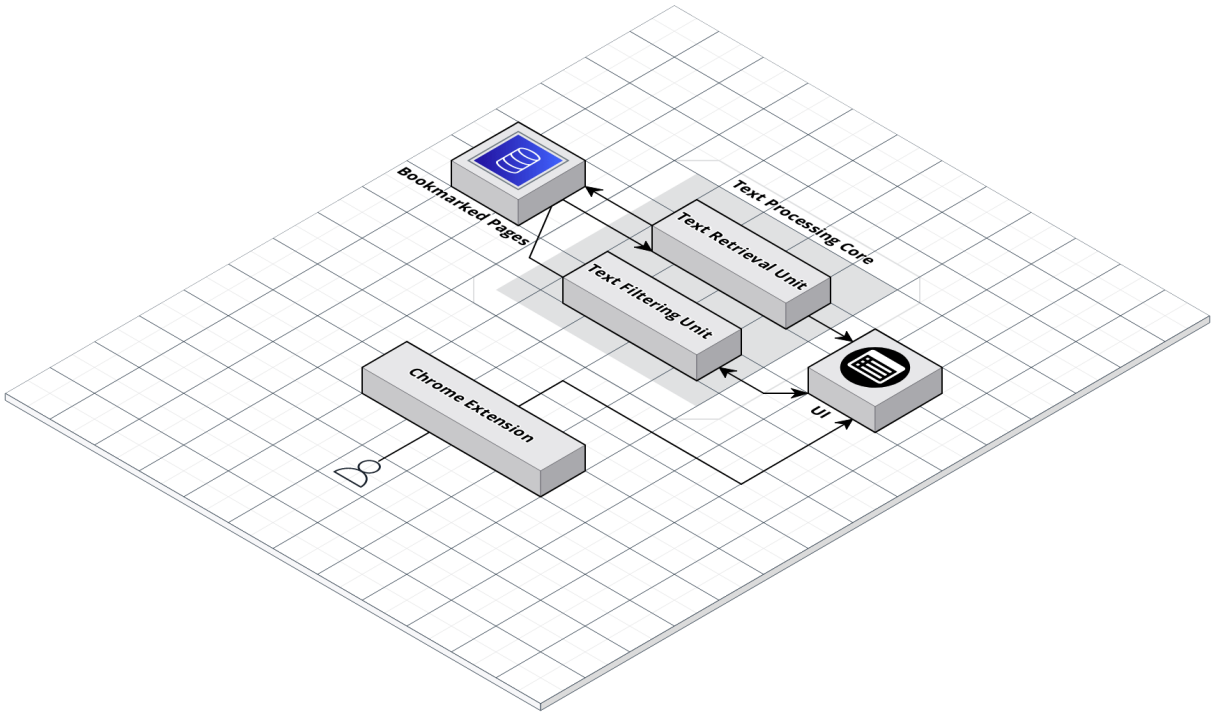
3. Dataset and Techniques

3.1 Data Source and Dataset Construction Method

For this project, I plan to use my personal bookmarked pages as the data source, which in real life, the data source should be the user's own bookmarked pages. To quantify the evaluation result, I will manually label those pages by taking Cranfield Evaluation Methodology and compare it with search results. Different evaluation figures (including F1 and MAP) will be used.

Regarding the text filtering system part, similarly, I will manually label related pages and compare the similarity between the recommendation results and the actual labeled pages.

3.2 Techniques



(Figure 3. High-Level System Architecture)

As mentioned above, this project will be in a Chrome Extension form, leveraging HTML/CSS/JS.

The text processing core will be in charge of handling the query requests and providing recommendations. I will try to avoid "reinventing the wheel"; my main target is to find if there are

any suitable 3rd party JavaScript-based (ideally browser friendly, no other heavy NodeJS library dependencies) text processing toolkits. My second choice is to manually port some core parts of modern search engines (e.g., ElasticSearch or MeTA) into JavaScript, and this is challenging due to different limitations; it is complicated to port everything to JavaScript; thus, only selected core parts will be ported, all unnecessary and additional features will be ignored. My last choice is to code everything from scratch.

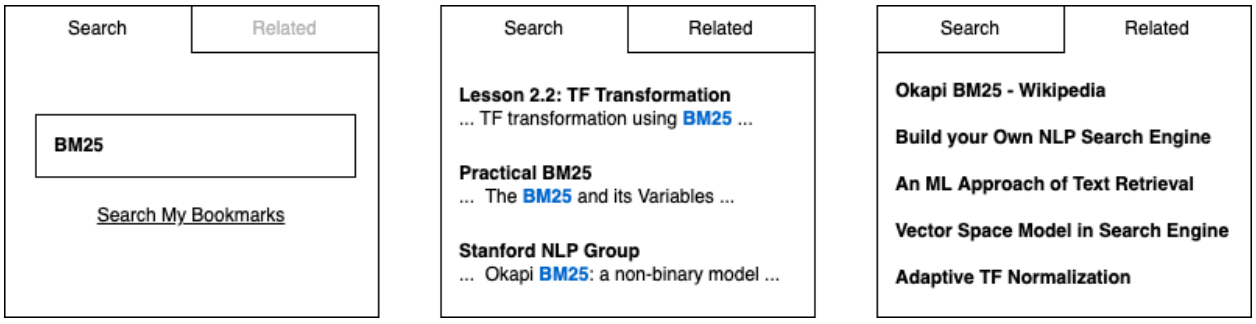
4. Expected Result

4.1 How will it be Demonstrated?

This project will be demonstrated by using the real Chrome browser with this extension. Also, the dataset will be 100% from real life.

I will provide a step-by-step walkthrough video at the end of the project.

4.2 UI Prototype



(Figure 4. Design of the UI Prototype)

5. Programming Languages and Frameworks

5.1 Programming Language

The main programming language will be JavaScript for this project. For the UI part, HTML and CSS will also be used.

5.2 Frameworks

Frameworks will be determined after research. As mentioned in section 3.2, my primary preference will be building this project upon a suitable JavaScript-based text processing toolkit.

Additionally, in order to keep the extension lightweight, I will try to avoid using too many 3rd party libraries.

6. Workload Analysis

Research suitable JS-based Text Processing Toolkit	2 hours.
Embedding and Enhancing the chosen toolkit. Or, write one from scratch.	3-10 hours
Build extension UI.	5 hours
Implement the text retrieval unit.	7 hours
Implement the text filtering unit.	7 hours
Testing and evaluation.	3 hours

7. References

- Google extension developer guide
    - https://developer.chrome.com/docs/extensions/
- CS 410 Course Materials
    - https://www.coursera.org/learn/cs-410/
- HTML/CSS Guides from W3School
    - https://www.w3schools.com/html/
- A simple implementation of an in-browser search engine (pure JS implementation)
    - https://javascript.plainenglish.io/building-a-simple-in-browser-search-engine-d87c86ac3261
- “Build a simple search engine with HTML, CSS and JS”
    - https://www.youtube.com/watch?v=o2aAaVsC1Oo
- “Create Product Search Engine/Bar/Filter in JavaScript”
    - https://www.youtube.com/watch?v=ZFUOC-y4i0s