Justin Feinfield (justinf6@illinois.edu)
Ryan DeStefano (ryanfd2@illinois.edu)
CS 410: Text Information Systems
November 19, 2023

Project Progress Report: Intelligent Browsing with Advanced Search

I. Completed Tasks

Of the 13 tasks proposed in the original project proposal, 4 have been completed in full. These are:

1. Create initial extension - 2 hours
    a. Create a very basic extension, with minimal UI and backend, This will allow both group members to work on different tasks in parallel
2. Backend to parse web page content - 4 hours
    a. Parse the text of a web page. Potential challenges include parsing texts in headers, footers, and menu bars
3. Create UI element to display the number of relevant documents - 2 hours
    a. This should display the number of relevant documents, as well as the current one being shown, such as "1 of 5".
4. Create search bar UI - 6 hours
    a. Create the search bar UI using JavaScript and CSS for styling. The text typed into the search bar needs to be read in and used as input to the ranking function.

With the completion of these tasks, the current Chrome extension behaves similar to the typical search functionality in a web browser, in that it can count the number of times the search text exactly matches text in the web page.

II. Pending Tasks

The pending tasks will take the current extension and grow it into the intelligent search that was proposed. These tasks will enable matching paragraphs in the web page that are relevant to the search, but not exact matches. There are 10 pending tasks to be completed over the next 3 weeks:

1. Create set of web pages to benchmark performance - 4 hours
    a. Collect a set of 10-15 web pages that can be used to test the extension. Having a fixed set of web pages, with expected search queries and results, will allow benchmarking results when fine tuning the algorithm
2. Break up content into "documents" - 4 hours
    a. Break the parsed web page content into chunks, or documents, to be searched by the BM25 algorithm

3. Implement initial BM25 ranking algorithm - 2 hours
    a. Implement the initial ranking algorithm, which can be finetuned in future tasks
4. Create UI element to navigate between the relevant documents - 2 hours
    a. Add up and down arrows to the search bar to allow the user to view all relevant documents
5. Highlight the relevant document in the web page - 4 hours
    a. Highlight the currently selected relevant document in the web page itself so that it can be easily viewed by the user
6. Scroll to the relevant document in the web page - 4 hours
    a. The web page needs to scroll to center the relevant document in the center of the page
7. Fine tune the BM25 algorithm for better results - 4 hours
    a. Improve the results of the BM25 algorithm by testing different parameters. The results should be compared across the testing set of web pages to choose the best set of parameters
8. Fine tune minimum relevance - 4 hours
    a. Fine tune what score for a document should be the minimum for that document to be shown to the user as matching the query. If this is too low, the user will get results that are not actually relevant, but if it is too high, they will miss some potentially valuable results
9. Fine tune the chunk size of the web page contents - 2 hours
    a. The size of the documents that web pages are broken into will have an impact on the performance of the search results.

These tasks can be divided into 3 sections with the following remaining time estimated:

1. Evaluation data and fine tuning - 14 hours
2. Document ranking algorithm and the backend to support it - 6 hours
3. UI improvements to enhance user experience - 10 hours

III. Challenges

A few challenges that we've encountered so far:

- The main challenge has been learning how to develop a Chrome extension. Since it is an unfamiliar development experience for either team member, reading through documentation and trying to implement the extension such that it builds successfully and is compliant with Chrome's Content Security Policy (CSP) took a larger portion of time than expected. Now that this challenge has been overcome, however, development for the extension should be much smoother moving forward.

- Learning how to interact with the content of a web page from within the extension was a slight challenge as well. The actual UI (referred to as the "popup") of the extension does not naturally have context of the currently selected tab, so we need to develop what are known as content scripts to serve as the communication between the page and the popup. This script enables the popup to gain context of the contents of the current tab and thus enables us to perform a search based on the user's input.