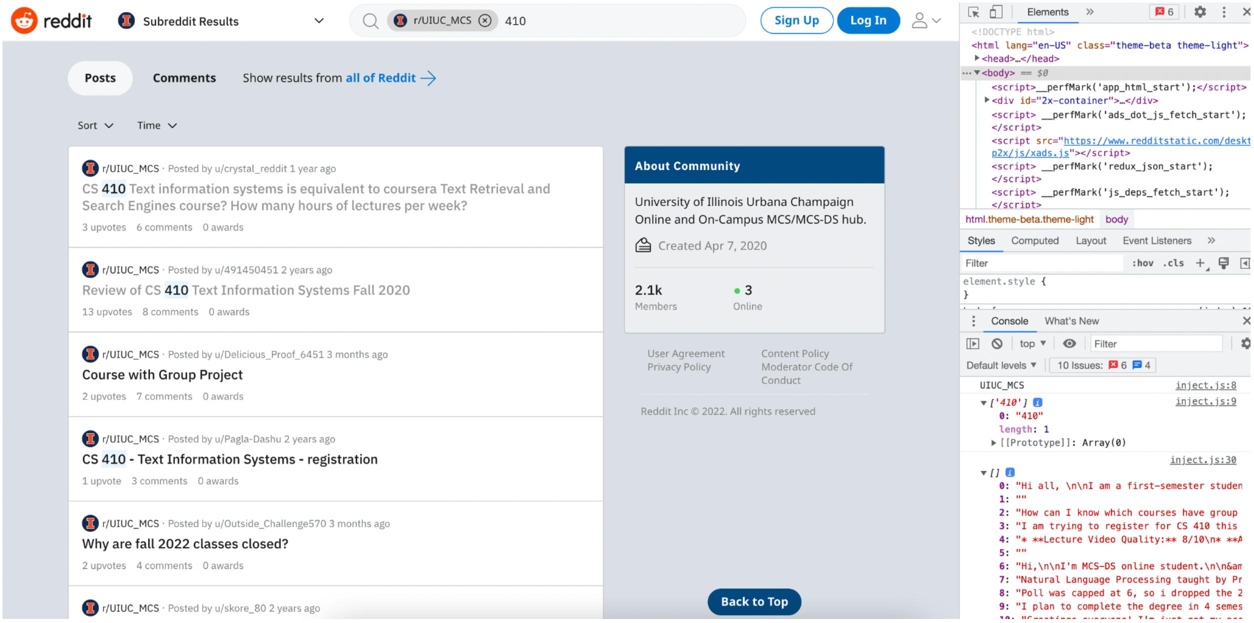
Progress Report

Team Wang

Which tasks have been completed?

We have developed a Chrome extension that listens to URL change events, retrieves search parameters, and fetches content from top search results. The screenshot below shows the logging of the subreddit “UIUC_MCS”, the search string “410”, and the content from top search results.



r/UIUC_MCS · Posted by u/crystal_reddit 1 year ago
CS 410 Text information systems is equivalent to coursera Text Retrieval and Search Engines course? How many hours of lectures per week?
3 upvotes 6 comments 0 awards

r/UIUC_MCS · Posted by u/491450451 2 years ago
Review of CS 410 Text Information Systems Fall 2020
13 upvotes 8 comments 0 awards

r/UIUC_MCS · Posted by u/Delicious_Proof_6451 3 months ago
Course with Group Project
2 upvotes 7 comments 0 awards

r/UIUC_MCS · Posted by u/Pagla-Dashu 2 years ago
CS 410 - Text Information Systems - registration
1 upvote 3 comments 0 awards

r/UIUC_MCS · Posted by u/Outside_Challenge570 3 months ago
Why are fall 2022 classes closed?
2 upvotes 4 comments 0 awards

r/UIUC_MCS · Posted by u/skore_80 2 years ago

About Community

University of Illinois Urbana Champaign Online and On-Campus MCS/MCS-DS hub.

Created Apr 7, 2020

2.1k Members · 3 Online

User Agreement · Privacy Policy · Content Policy · Moderator Code Of Conduct

Reddit Inc © 2022. All rights reserved

```
<!DOCTYPE html>
<html lang="en-US" class="theme-beta theme-light">
 ▶<head>…</head>
…▼<body> == $0
    <script>__perfMark('app_html_start');</script>
   ▶<div id="2x-container">…</div>
    <script> __perfMark('ads_dot_js_fetch_start');
    </script>
    <script src="https://www.redditstatic.com/deskt
    p2x/js/xads.js"></script>
    <script> __perfMark('redux_json_start');
    </script>
    <script> __perfMark('js_deps_fetch_start');
    </script>
html.theme-beta.theme-light  body
element.style {
}
```

```
  UIUC_MCS                                        inject.js:8
 ▼['410']                                         inject.js:9
    0: "410"
    length: 1
   ▶[[Prototype]]: Array(0)
                                                  inject.js:30
 ▼[]
    0: "Hi all, \n\nI am a first-semester studen
    1: ""
    2: "How can I know which courses have group
    3: "I am trying to register for CS 410 this
    4: "* **Lecture Video Quality:** 8/10\n* **A
    5: ""
    6: "Hi,\n\nI'm MCS-DS online student.\n\n&am
    7: "Natural Language Processing taught by Pr
    8: "Poll was capped at 6, so i dropped the 2
    9: "I plan to complete the degree in 4 semes
```

For the backend part, first we have identified a challenge in getting the word distribution of a collection model and found out a solution, which requires utilizing BERT (See more details in the Challenge section) as well as a backend Python server. Then we have set up a local Python HTTP server, which handles the HTTP requests from the Chrome extension to run the core logic for keywords recommendation. The input is a list of document urls fed by the Chrome extension in the HTTP POST body, and the output would be a list of keywords to be rendered for recommendation. Currently the implementation for web scraping from a list of document urls has been finished, and we are working on the tokenization part with lexical analysis as the preparation step for language model training.

Which tasks are pending?

For the chrome extension, the pending tasks are:

1. Integrate the server with our Chrome extension once ready
2. Add query suggestions to the UI

For backend, here is the list of pending tasks:

1. Finish tokenization implementation
2. Implement collection LM construction via BERT

3. Implement document LM construction via maximum likelihood estimator
4. Implement the recommendation process via comparing collection LM and document LM
5. Implement the validation system via paradigmatic relation analysis

Are you facing any challenges?

We faced a challenge where the top search results were logged as an empty array, even though results were present on the page. After some digging, we realized that Reddit uses dynamic loading and the extension needed to wait for the page to fully render before processing the DOM.

For the backend part, we faced a challenge where it's hard to get an existing collection LM that suited our use case, which is the capability to retrieve the probability of a word from a list of mappings between a word and its probability. We have done some research on the state-of-the-art popular LMs including BERT, BLOOM, and GPT-3, but all of those are generative LMs, so there's no easy way to directly retrieve the word distribution (the mapping between vocabulary and its probability) without context. Thus we investigated and figured out a way to approximate a collection LM without explicit training. Upon getting a fairly large collection of corpus, we can mask 15% of the tokens in the corpus and then leverage the Masked LM feature of BERT to recover those 15% hidden tokens. For each masked token during the MLM process, BERT will produce a list of possible vocabularies with corresponding probabilities, which can be used to aggregate and then approximate the collection LM, which helps us get over the challenge.