Ethan Hu, David Chiang, Brandon Ta

INFO 498: Search and Recommender Systems

Project Report

6/05/2022

- Introduction - what is this project about and what it does/serves. (1/2 to 1 page) [5 points]
    - Design details (may include figures). Explain your decisions behind certain design choices (think about stemming, stopwords, retrieval models). (1-2 pages) [15 points]
    - Usage scenario (sample queries, may include screenshots). (1-3 pages) [15 points]
    - Known issues and future work. (1-2 pages) [10 points]
    - License. An appropriate Creative Commons License (Links to an external site.) is recommended. [5 points]

**Introduction**

The purpose of our project is to develop a web interface with search functionality that returns links to food recipes that users can use for their own cooking interests. Our collection consists of food recipes with roughly 1000 documents crawled from seriouseats.com, a site for people to find a broad range of cooking recipes. The results returned from the search query are in the form of a ranked list sorted by relevance. A core feature of our project is a user-generated recommendation function that recommends queries to try based on the last five search results. Our site is meant to be accessible to people with varying degrees of cooking experiences and use

case scenarios. It serves as a quick and informative way for people to find recipes for an ingredient such as an egg to people seeking information for a dish such as chicken noodle soup. For users that are just browsing through our website, they can use the last five search results from past users as inspiration for possible queries. With our search engine dedicated to food recipes, we hope that more people can easily find recipes that are more relevant to their cooking needs.

**Design Details**

The corpus consists of Serious Eats Recipes due to a more selective wget process. Running a recursive wget function over the entire website adds articles and content that aren't recipes - equipment reviews, retrospectives, and the requisite files to support them. This inefficiency means that in order to get a usable body of recipes, we would need to operate on an unacceptably large corpus.

Instead, we ran multiple wget functions over different categories of recipes, and put them in the same file. While each wget had a very small amount of recursion, adding them together is relatively easy since each of the files are dumped into the same folder, and exact duplicates are not stored. As a result, the corpus envelops a wide range of recipes while remaining relatively lightweight.

Retrieval uses the PyTerrier FilesIndexer function, which inherently includes PorterStemmer and nltk stopword removal. The functions do their tasks well - retrieval models no longer have to go through common stopwords, and there is a coherent body of tokens to do indexing on. The all-in-one nature of FilesIndexer makes it a convenient choice for indexing.

Once FileIndexer creates an index for the corpus, we used TF-IDF to create rankings for queries. Simpler than BM20 and language modeling, TF-IDF is perhaps the most understandable of retrieval ranking systems. Along that vein, having 5 results for the queue is a good balance between readability and utility, since processing later results is uncommon for many people during retrieval.

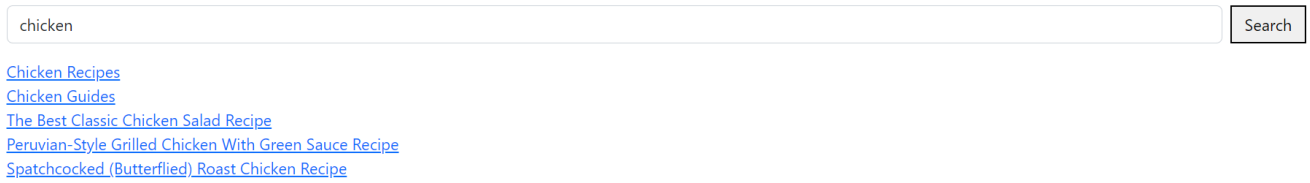
Our recommendation system is fairly unique in that it spits out the five most recent searches. The benefit of such a system is that it helps users remember previous queries when returning to the webpage. This is especially useful when leaving and returning from different recipe webpages.

**Usage Scenario**

The Serious Eats browser provides users a quick way to search for documents from Serious Eats, the majority of which are recipes but there are a few guides and other documents available as well. There is a short intro description at the top of the browser to help guide users. There are 3 main usage scenarios that users that we have envisioned:

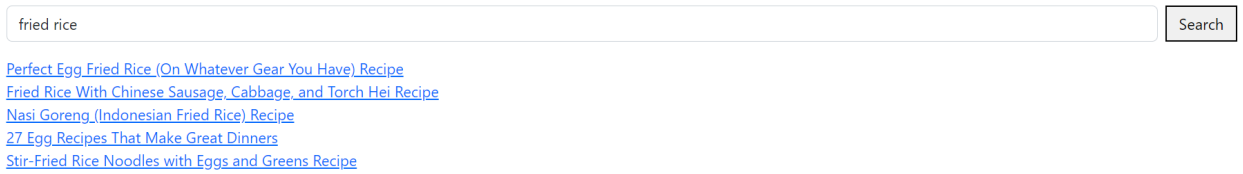
1) Ingredient-based Search

Users can search by an ingredient. In this scenario, the user may have a particular ingredient lying around that they want to make a dish out of. For example, a user might have leftover chicken that is going to expire soon. The user can use our browser and search for chicken to find a suitable recipe.



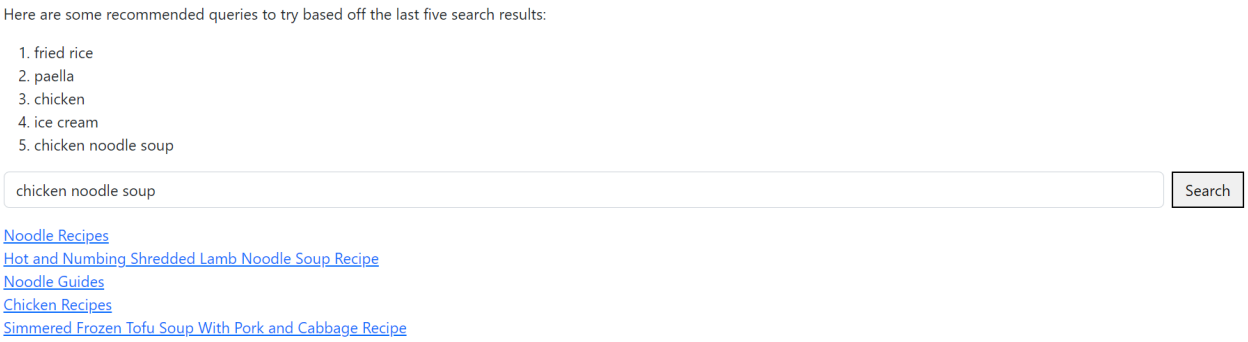

## 2) Dish-based Search

Users can search for a particular dish. In this scenario, the user may be craving a certain dish and wish to find a recipe for said dish. The user can enter the dish they are craving instead of an ingredient and find a suitable recipe for that dish.




## 3) Inspiration

If users are looking for some inspiration for what to cook, they can look at the 5 most recent queries for potential ingredients to use or dishes to cook. In this scenario, can find inspiration by looking at the recent search list and searching for the one they like.



**Known Issues and Future Work**

The wget functions used to expand the corpus aren't perfect - there are still some recipes that are missing, and many non-recipe pages included in the corpus. Manual filtering could try to crop out irrelevant results from the files directly, or filtered with a Pyterrier function. Corpus scope can be further adjusted by including recipes from other websites, such as those containing more ethnically niche dishes.

The website itself, while functional, is still visually barebones - meaning that it struggles to differentiate itself from the onsite Serious Eats' searching system. In addition, the recommendations and history only show the query, which can be self-defeating in some aspects if the user already knows about what they searched earlier.

Fleshing out the results and recommendations can be a good way to make this browser more unique and usable. Displaying pictures and a preview of the page body not only makes it more appealing, but also useful, as users will not have to click on the hyperlinks to access essential page data. However, a limitation of Pyterrier's indexing functionality is that it doesn't particularly index non .html files or pre-formatted dataframes. We will have to search for a library that does process this data in order to add this functionality.

**License**

(Also linked on our website: http://searchrec.ischool.uw.edu/dchiang0/search.php)