Project Overview

Louis Gomez, CS600

This readme serves as the general overview of my final project. In this document, I will give a high-level overview of the main sections of the project. The webpages I used are from a collection of different pages on my research advisor website. The entire program was written in python. The four main sections of the project were:

1. Web crawling
2. Word Processing
3. Compressed Trie Creation
4. Page Ranking

Web Crawling

To crawl across the websites, I used a python library called beautiful soup. I go through all the possible word tags on each HTML page and extract the information embedded within. I did this process iterating through all the 10 webpages I used for this project.

Word Processing

To process the extracted information I received from the web crawling operation, I used a python library popular in the natural language processing space called NLTK. Using the library and its associated packages, I segmented each sentence structure into token words, removed stop words and also eliminate punctuations. During this process, I also keep track of words in each URL and their occurrences in an occurrence list. I use a tuple structure that contains both the document Id for fast retrieval and word frequency.

Compressed Trie

After web crawling and web processing, I move on to storing the new information into the compressed trie. To implement the compressed trie, I model a similar structure to a normal trie but instead of storing each individual character, I store the word prefixes instead. This reduces the storage space necessary to store all the words. I deeper implementation will be in the video demonstration attached to my submission. When a word is searched, we traverse down the trie in a depth-first search manner matching prefixes as we go along until we get to the word. I then return the occurrence list associated with it. This information is passed to the page ranking algorithm.

Page Ranking

To rank the pages in our engine, we first unpack the occurrence list. To rank pages, I used a two criteria approach. The first ranking done is based on which pages had the most hits. This means that which page had the majority of words appear in them. After this process, my sorted from most to least hits. After this sorting, a second-round is done, but this round focusses on only total word frequency in each webpage. Based on the ranking results, I output the top four webpages back to the user.