Rony Verch

Mark Volfson

Vidit Chhabra

CPS 842

December 2nd, 2020

Project 1

Link to website: <http://159.203.6.204/>

Implemented a search engine on the BBC website to search for any news, article on the website

Using Laravel, a Web Application Framework that uses PHP to host the website and to connect the backend logic with the frontend. We also used Digital ocean as a cloud provider for a server to host our files and run the website

We crawled the BBC website using a custom PHP script to extract over 4000 links alongside article texts and stored them in a MySQL database within the Laravel infrastructure.

At first, we decided to do an index of the entire 4000 links, but encountering speed constraints, we had to rewrite our application to make it more efficient. This was done with the use of MySQL and only retrieving entries from MySQL that has any words with the queries and not all 4000 links to make the search faster. Once we retrieve the entries from the database that has at least one of the query words, we then work with that set of data to look through the dictionary and postings file and a generate a vector space model number to find out the similarity between a particular article and the query. After finding the vector space model (from the articles we got from the database), we then get page rank using an API.

We use OpenPageRankAPI to get the page rank for it with the particular API as second metric for similarity. We are trying to retrieve the relevant document in order of relevancy not only on the Vector space model implemented by us but also on page rank that we get.

Once we get page rank and vector space model, we add the numbers together by assigning a weight of 0.5 on both numbers. Using the formula given on the project description and after careful analysis, best result has been when the weights has been 0.5 for both metrics. 0.5 \* vsm + 0.5\*page rank and that is the score we base our relevancy on. *Formula below:*

***score(d, q) = w1\*cos-score(d, q) + w2\*pagerank(d) where w1+w2=1***

*w1 = 0.5 and w2 = 0.5 where w1+w2 = 0.5*

Once we get the result and we output the result in a formatted way on our front end using Laravel that displays in a descending order of relevancy. Using top-k algorithm, we only display top 10 relevant documents.

Citations

OpenPageRank, https://openpagerank.com/api/v1.0/