Web Search Engine

Using Vector Space Model with Page Rank

Triparna Bhattacharya
Computer Science Department
University of Illinois at Chicago
Chicago, Illinois, USA
triparnabh@gmail.com

ABSTRACT

The project is a thorough research on specific components of how a Search Engine works. Collecting domain specific documents using focused crawlers has been considered one of most important strategies to find relevant information. While surfing the internet, it is difficult to deal with irrelevant pages and to predict which links lead to quality pages. Here I have tried to incorporate multiple components and performed comparative analysis on the results it gave.

MAIN CONCEPTS

• Web Crawler as per the Breadth First Search Algorithm • TF-IDF vectorized model • Cosine Similarity measure • Page Rank Algorithm

1 Web Crawler

A Web Crawler is a key component inside a search engine. The crawler uses Breadth First Search strategy to traverse the UIC domain, gather pages from the Web, the interlinks, create a web graph using the child URLs, extract the text to build the inverted index. Web crawlers are mainly used to create a copy of all the visited pages for later processing by a search engine that will index the downloaded pages to provide fast searches. This crawler downloads web pages relevant to the domain.

1.1 Search Technique

Breadth First Search (BFS) algorithm traverses a graph in breadth ward motion and uses a queue to remember to get the next vertex to start a search. Traverse the graph layer wise thus exploring the neighbor nodes (nodes which are directly connected to source node). You must then move towards the next-level neighbor nodes. As the name BFS suggests, you are required to traverse the graph breadthwise as follows:

I. First move horizontally and visit all the nodes of the current layer
II. Move to the next layer

For this project our we started our crawler from https://www.cs.uic.edu/ website and expand our search up to 3,000 pages, at every iteration I will insert the link and all the child URLs in the Queue after validating the link, checking HTTP status code and other criteria. After extracting all the information from a link, we are adding it to visited list and repeat the same process for all the child URLs till we encounter a new link.

2 Web Graph

Web Graph explains the relation and interlinks between pages of the web. A graph, in general, consists of several vertices, which in our case are the URLs which we are fetching and they are connected by edges child URLs. I have maintained an undirected graph for this project. This data will be later used to calculate the Page Rank score for each link present in my Web Graph based on in-links and out-links.

3 Page Rank

PageRank views the web as graph, with **inbound links being viewed as measure of the significance of a web page.** PageRank works by counting the number and quality of links to a page to determine a rough estimate of how important the website is. The underlying assumption is that more important websites are likely to receive more links from other websites. I have used the below Page Rank Formula to calculate the score for each link in the web graph for UIC domain: -

$$s(v_i) = \alpha \sum_{v_j \in Adj(v_i)} \frac{w_{ji}}{\sum_{v_k \in Adj(v_j)} w_{jk}} s(v_j) + (1-\alpha)p_i,$$

where α is a damping factor ($\alpha = 0.85$) and $p_i = \frac{1}{n}$.

3 Content Preprocessing

I have used the Python3 **Beautiful Soup** module to extract the URLs and web-page content which have been later used to construct the web-graph and inverted index. Other pre-processing steps for the inverted index used are punctuation removal, Stemming and Stop word removal using **Porter Stemmer** and **NLTK library**.

4 Inverted Index

Created an inverted index using words extracted from the URLs as keys and keeping a track in how many links(document-

frequency) that word has appeared and for how many times(term-frequency). In information retrieval, tf–idf or TFIDF, short for term frequency–inverse document frequency, is a numerical statistic that is intended to reflect how important a word is to a document in a collection or corpus. It is often used as a weighting factor in searches of information retrieval, text mining, and user modeling. This data structure keeps a track of the TF-IDF for each word corresponding to every URL they have appeared in. The same process has been repeated for every Query term provided to the Search Engine as well.

5 Query Processing & Cosine Similarity Measure

Cosine similarity is a measure of similarity between two non-zero vectors of an inner product space that measures the cosine of the angle between them. the cosine similarity equation is to solve the equation of the dot product for the $\cos\theta$

$$\vec{a} \cdot \vec{b} = \|\vec{a}\| \|\vec{b}\| \cos\theta$$

$$\cos\theta = \frac{\vec{a} \cdot \vec{b}}{\|\vec{a}\| \|\vec{b}\|}$$

I have used this measure to calculate the similarity between the query words and the inverted index terms and will return the corresponding URLs for the words which has the highest similarity measure.

6 Intelligent Component

I have combined **Page Rank scores** and **TF-IDF vectorized** model as my intelligent agent using **Harmonic Mean**. The Web Graph component keeps a score for each visited URL. After finding the match of the query words in the inverted index, I have calculated the harmonic mean of TF-IDFscore of the word and Page Rank score of the URLs corresponding to that word. Based on this score retrieve the top *k* links from the web-graph for displaying to the user.

7 Challenges

I. Since the search is domain directed, initial retrieval of URLs was faster, but as we progress the URLs being encountered were repetitive and hence the collection process gets slowers.

II. I have experimented with two approaches for this project, first being the cosine similarity measure using only TD-IDF, where the user would be displayed with relevant links as matching terms are taken into consideration between the query and the index but the URLs being fetched might not have good page rank scores(less popular), hence not reliable having less number of in-links. To overcome this challenge I combined the approach of TF-IDF and Page Rank and performed Harmonic mean of the two values to get desired results.

III. Since we are crawling around 3,000 URLs at-least, the URL request process is very expensive and time consuming, so initially it was taking around 50 - 60 minutes for the crawler to gather the information. I implemented multi-threading to overcome this challenge, the performance has increased, and it takes 15 – 20 minutes.

8 Result

The results are based on 3000 crawled pages as per the query provided below.

Query1 :- UIC Employment

http://studentemployment.uic.edu
http://www.uic.edu/depts/st_empl
http://uic.edu/about/job-opportunities
https://uic.edu/about/job-opportunities
https://www.uic.edu/about/job-opportunities
https://www.cs.uic.edu/visit-us-at-an-open-house
https://www.cs.uic.edu/our-department
https://jobs.uic.edu/job-board/job-details?jobid=104249
http://www.uic.edu/life-at-uic
http://www.uic.edu/apps/departments-az/search?dispatch=letter&letter=U

Query 2 :- UIC computer science

https://www.cs.uic.edu/our-department
https://www.cs.uic.edu/visit-us-at-an-open-house
https://www.cs.uic.edu
http://www.cs.uic.edu
http://cs.uic.edu
https://jobs.uic.edu/job-board/job-details?jobid=104249
https://www.cs.uic.edu/undergraduate-admissions
http://studentemployment.uic.edu
http://www.uic.edu/depts/st_empl
http://uic.edu/about/job-opportunities

Query 3 :- UIC alumni

http://uic.edu/alumni
http://www.uic.edu/apps/departments-az/search?dispatch=letter&letter=U
http://www.uic.edu/life-at-uic
http://advance.uic.edu/alumni-association
https://admissions.uic.edu/explore-uic
https://www.cs.uic.edu/visit-us-at-an-open-house
http://uic.edu/about/job-opportunities
https://uic.edu/about/job-opportunities
https://www.uic.edu/about/job-opportunities
http://www.uic.edu

Query4 :- UIC EVL

https://www.cs.uic.edu/386k-nsf-grant-visualization-and-collaboration-services-for-global-cyberinfrastructure
http://www.uic.edu/apps/departments-az/search?dispatch=letter&letter=U
http://uic.edu/alumni
http://www.uic.edu/life-at-uic
https://www.cs.uic.edu/evl-to-be-featured-in-chicago-new-media-1973-1992-exhibition
https://www.cs.uic.edu/evls-lance-long-recognized-with-uics-award-of-merit
https://www.cs.uic.edu/visit-us-at-an-open-house
http://accc.uic.edu/service/computer-labs
http://advance.uic.edu/alumni-association
https://admissions.uic.edu/explore-uic

Query5 :- uic library

http://www.uic.edu/life-at-uic
http://www.uic.edu/apps/departments-az/search?dispatch=letter&letter=U
http://uic.edu/alumni
https://admissions.uic.edu/explore-uic
http://advance.uic.edu/alumni-association
https://library.uic.edu/help/article/1955/use-accessibility-services
http://www.uic.edu
https://today.uic.edu/university-library-extended-hours-for-end-of-semester-and-finals-week-2
https://today.uic.edu/contact/social-media-directory
https://news.uic.edu/social-media-directory

If the number of crawled pages is increased the crawler provides better and efficient results depending on the query provided**.**

8 Future Work

I plan to incorporate Query expansion and Relevance Feedback.

Query expansion:- It is the process of reformulating a given query to improve retrieval performance in information retrieval operations, particularly in the context of query understanding.

Relevance Feedback:- The idea behind relevance feedback is to take the results that are initially returned from a given query, to gather user feedback, and to use information about whether or not those results are relevant to perform a new query.

ACKNOWLEDGMENTS

Professor Cornelia Caragea
CS 582

REFERENCES

[1] Assignment 1
[2] Assignment 2 and Assignment 3
[3] Stackoverflow
[4] Paper provided for the course