Information Retrieval and Web Search

Project Report

Prepared By

Dheeraj Reddy Aguthu

**Topics**

1. Implementations
2. How to run the code
3. Results
4. References

**Problem Statement:**

You are required to implement a search engine for unt.edu.

1. Implement vector space retrieval model for the search
2. Evaluation of the system:
   1. Select a word of your choice
   2. Run the query on the original unt.edu
   3. Run the same query on your system
   4. Compare the result and report any discrepancies
3. **Please note:**You need to crawl the unt.edu to collect webpages in unt.edu and parse them to get terms that may end up in your dictionary.
4. You can use any library to crawl and parse web pages or you can use your own custom-built crawler/parser.

**Implementation:**

In this project, we need to create a search engine where we need to crawl the “unt.edu” website and store the information in the form of index and use them to find the links for the given keyword.

**Step 1**: In the first step we need to import certain libraries from the python which are used for crawling and indexing.

Text

Description automatically generated

**Step 2**: After installing the required libraries, we are going to crawl around the website using the “beautiful soup”. First, we are going to give the URL as a parameter to parse the website. Later we need to remove the stop words from the text and later we extract all the links from the website by using the “find\_all” after that we are parsing through all the nested links in the website and storing them in the index. We are using the word tokenizer to break the text into the terms and store. We are training the dataset which we got from the word\_tokenizer and store them in the data frames.

Graphical user interface, text, application, email

Description automatically generated

**Step 3:** In this step we are going to use the w2v model (word to vector) model to find the term frequency and provide the rank to the given words or terms. This model helps us to identify which links come under priority based on the similarity and frequency, this helps us to find the exact link which we are seeing for. For example, we have given “UNT” as a search word and it shows the results of it what we get when we use the keyword “unt”. This model helps to parse the given terms and provide the correct score.

Graphical user interface, text, application, email

Description automatically generated

**Step 4:** In the next step, we are traversing the index to find the keyword which matches the given word. It traverses the text which is stored in the data frames. We got around 1400 keywords when we parse through the text.

Graphical user interface, text, application, email

Description automatically generated

**Step 5:** In this step, we are going to define a function and pass the parameter, later we are going to search the terms which are saved in the superdf.text(which is the combination of df1 and df2). It will append to the variable, and it iterates through the process.

Text

Description automatically generated

**Step 6:** In the next step, we are going to implement the search engine, where we are going to iterate through the index using the w2v model and find the most similar used words and provide the output for the given word.

Graphical user interface, text, application, email

Description automatically generated

**Comparing Results:** We have provided the search term as “sports”, it has provided the given output

Graphical user interface, text, application, email

Description automatically generated

Similarly, we have searched for, “admissions”.

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, application, table

Description automatically generated

**How To Run the Program:**

* 1. Install certain libraries as mentioned above
  2. Run the Program in jupyter notebook
  3. Run the cells one by one.
  4. Enter the keyword of choice and it provides the results

**References:**

1. <https://www.researchgate.net/publication/33>

5576922\_Data\_Analysis\_by\_Web\_Scraping\_using\_Python

1. chrome-extension://efaidnbmnnnibpcajpcglclefindmkaj/http://ijream.org/papers/IJREAMV07I02SJ003.pdf
2. https://ieeexplore.ieee.org/document/8822022