Eagle Search: A Search Engine for UNT

http://129.120.151.94:8000/eagle_search

Venugopal Gonela
Department of Computer Science, UNT
Mobile: +1 (940)312-8472
Email: venugopalgonela@my.unt.edu

ABSTRACT

This paper is a project report for the course 5200: Information Retrieval and Web Search. The paper discusses the implementation of web based search engine for the University of North Texas (UNT). The search engine described in this paper implements crawler which crawls the UNT domain pages, implements vector space model to index and retrieve the pages for any query given to the engine. The key components involved in the implementation are crawler, text preprocessor & indexing, retrieval & ranking system and web user interface. The paper also includes the experiments, results and evaluation of the performance of search engine.

Keywords

Information retrieval, Web Search, Crawler

1. INTRODUCTION

Data is increasing tremendously over the years in this digital world, where the data formats include a variety of formats such as .pdf, .gif, .mp3 etc. Retrieving information from this huge collection of data accurately and efficiently is a very challenging task in information retrieval system. Along with an increase in the data, the frequency of changes made to the data available on the web is also increasing, which adds more challenges in retrieving the up to date information. A lot of research is going on actively to improve the efficiency of the retrieval process.

This paper implements a web based search engine which allows a user to search for pages available in UNT domain. The components involved are crawler, text processor, indexer, documents retrieving & ranking and web interface to the user. Vector space model representation is used in representing documents and query. The indexer uses tf-idf weighting scheme in indexing the documents.

The primary motivation behind this report is to demonstrate the fundamental concepts of implementing the web based search engine for UNT. This is achieved by designing and implementing a search engine which searches and retrieves documents for a given query. There are around 4000 pages crawled from the UNT domain which are indexed for faster retrieval.

The following report has the following structure. Section 2 discusses the key components involved in the implementation in detail, section 3 talks about the challenges faced while implementing the project, section 4 details the experiments conducted, the results obtained and manual evaluation of performance of search engine, section 5 presents the conclusion of the report and possible future work, section 6 contains acknowledgements and section 7 includes references that are used for writing the report.

2. EAGLE SEARCH: THE WEB SEARCH ENGINE

This section describes the various components used in the implementation of web search engine. The web based search engine implemented is named as ‘Eagle Search’. The components involved are classified as follows:

- Web Crawler
- Text processing and Indexing
- Retrieving and Ranking
- Web user interface

The architecture used in implementing the web search engine is shown in Figure 1 [2].

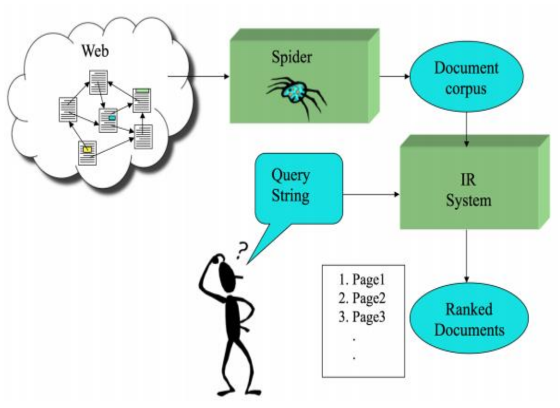


Figure 1. Web Search system

The detailed architecture used in the Information Retrieval system is shown in Figure 2 [2].

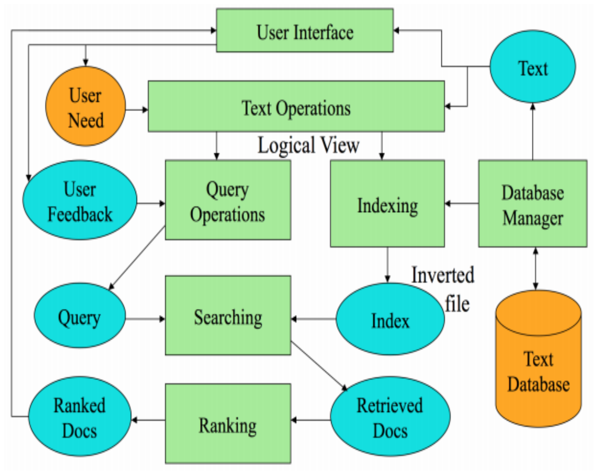


Figure 2. IR System Architecture

The components shown in the architecture are discussed in detail in the following sections. The project represents the development and integration of the components that are shown in figure 2. This paper does not cover the implementation of User feedback component that is shown in the figure 2.

2.1 Web Crawler/ Spider

The web crawler starts with UNT homepage http://www.unt.edu and follow all the links from the homepage recursively to find additional pages. The crawler implements the breadth first search strategy in crawling the pages and uses python's beautifulsoup package, to parse the HTML pages. The URLs parsed from the pages are stored in the SQLite database. The html pages are converted to plain text by removing all the html tags and saves the html content as a file in a directory with name of the file as ID of the URL stored in the database. This way mapping from URL to its page is established. The following techniques are used for URL validation to store it in database while crawling a page.

- Canonicalize URL's (e.g. delete ending "/")
- Handle URLs with relative path. (e.g. ./xyx, /xyx)
- Change URL to lower case
- Handle URLs pointing to within page (e.g. #maincontent)

Of all the links crawled and saved in database, only around 4000 pages are downloaded from UNT domain which are valid and active. There are more than 4000 pages which are valid but I have limited the count to 4000 due to memory space constraints.

2.2 Text processing and indexing

This is one of the core parts of the search engine where the html pages that are downloaded are preprocessed and then indexed for efficient retrieval of documents. Vector space model is implemented to index and retrieve the documents for a given query. The documents and query from user are treated as vectors in vocabulary size dimensional space and document vectors closer to the query vector are more similar to the query and are displayed to the user. The preprocessing of the text includes tokenization, stop word removal and stemming. The text processing follows the steps:

- Converts the bytes in the files into sequence of characters, which are only alphanumeric removing special characters
- Each file downloaded is chosen as the document unit for indexing.

The character sequence from each document is tokenized by splitting into tokens with each token having a semantic meaning. Splitting is done on whitespace by throwing away the punctuation. Extremely common words or stop words are of little value in selecting the documents for matching a user query, so tokens are checked whether it is contained in stop words list and removed if it is a stop word. Stemming is done to make different tokens of same root to appear identical, so that query from user better matches with identical words. All the tokens are applied to stemming using Porter Stemmer [1] which is an effective algorithm to stem a word. This results all the document's text processed to proceed with indexing.

The next step is indexing, the tokens obtained from the preprocessing module forms vocabulary. Each entry for each token in the vocabulary keeps a list of all the documents where it appears together with the corresponding frequency which forms Term Frequency (TF) and also keeps the total number of documents in which the corresponding word appears which forms Document Frequency (IDF). A dictionary (map) is maintained for each entry as a key and values is a posting list containing first element as IDF and second element again a dictionary whose key is document name and value is TF of that entry. The term frequency is normalized by formula

$$TF_{norm} = \frac{tf}{\max\_tf}$$

The IDF is calculated for each document as follows

$$IDF = log_2(N/df)$$

Where N is total number of documents and df is document frequency of each token added to the dictionary.

A typical tf-idf weighting for each token i and each document j is given as follows,

$$w_{ij} = tf_{ij} * idf_i$$

The pictorial representation of inverted index is shown in figure 3 [2]

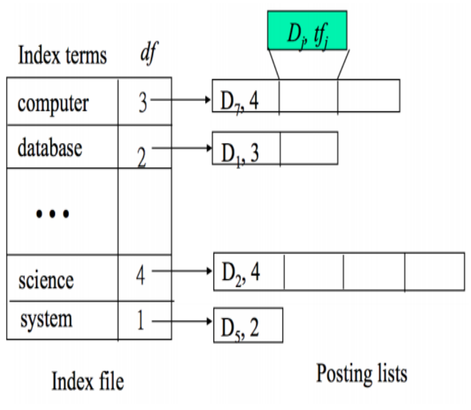


Figure 3. TF- IDF Inverted Index

There are many other weighting schemes that are used in implementing document retrieval model. Few popular are binary independence model, language model which are based on probabilistic theory.

2.3 Retrieving and Ranking

Query obtained from the user from web interface is also treated as a document and then passed through the same preprocessing module used before for all previous documents and also tf-idf weighted. The retrieving of the documents is obtained by comparing the similarity between each document with the query. There are many similarity measure schemes available of which cosine similarity is very frequently used. The cosine similarity of a document and a query is given as follows,

$$\text{Cosine}(d_j, q) = \frac{dj\ .q}{\|d_j\|\|q\|} = \frac{\sum_{i=1}^{t} w_{ij} * w_{iq}}{\|d_j\|\|q\|}$$

where q is the query as document and d is the jth document, w is the tf-idf weight

For each document cosine similarity is calculated with respect to given query and are stored in a dictionary where key is document and value is similarity measure value. The dictionary is then sorted with similarity measure in descending order. The URLs of the documents in the dictionary are then read from database and displayed in the web interface to the user as results. Only top 10 results are shown to the user and a button “Next” is shown to show more results as user clicks. So the user will see the documents with cosine similarity from higher to lower as documents from top to bottom in the interface.

There are other similarity measures available as follows:

Inner product

$$\sum_{i=1}^{t} w_{ij} * w_{iq}$$

Dice Coefficient

$$\frac{2 * \sum_{i=1}^{t} w_{ij} * w_{iq}}{\|d_j\| + \|q\|}$$

Jaccard coefficient

$$\frac{\sum_{i=1}^{t} w_{ij} * w_{iq}}{\|d_j\| + \|q\| - \|d_j * q\|}$$

The cosine similarity is widely used compared with above similarities.

2.4 Web user interface

The final part of search engine implementation is graphical web user interface, which via a web browser enables user to take query to search engine and to display results back for the given query. The results are displayed as list showing top 10 results initially and navigation links like “Previous” and “Next” to fetch more results and view previous results.

The snapshots of user interface are shown in Figure 4 and Figure 5, where the home page of ‘eagle search’ is given in Figure 4 and the results page for the query ‘college of engineering’ is as given in Figure 5.

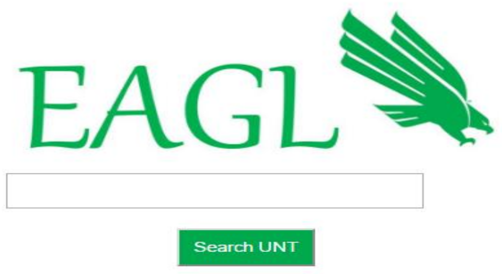
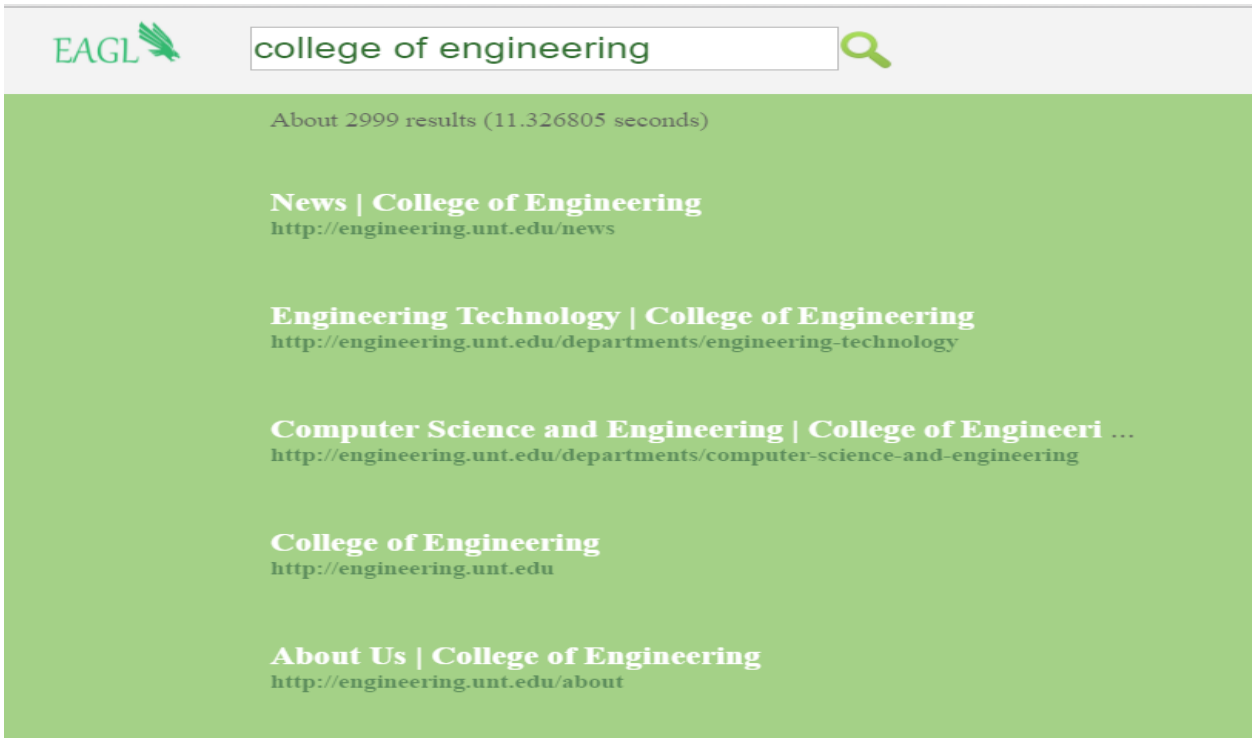


Figure 4. EagleSearch Home page



About 2999 results (11.326805 seconds)

News | College of Engineering
http://engineering.unt.edu/news

Engineering Technology | College of Engineering
http://engineering.unt.edu/departments/engineering-technology

Computer Science and Engineering | College of Engineeri ...
http://engineering.unt.edu/departments/computer-science-and-engineering

College of Engineering
http://engineering.unt.edu

About Us | College of Engineering
http://engineering.unt.edu/about

Figure 5. Results page for query 'college of engineering'

3. CHALLENGES FACED

This section discusses the challenges faced while implementing the search engine.

Most of the challenges faced are during the implementation of crawler for search engine. This is due to the various file formats or extensions available in the web. UNT domain contains various file formats like ppt, pdf, zip, xml, txt, image files, etc. So handling this issue was bit challenging and different URLs pointing to same web page.

a. **Duplicate Pages and URLs –** There are many URLs which are pointing to the same page in various ways. The URLs referring to some parts of the page like #main-content are handled easily, but there are some URLs which are different and points to same page like "http://www.cse.unt.edu/site/index.php" and "http://www.cse.unt.edu/site" which causes duplicate pages in the search engine. This is handled by comparing the text of the documents pointing by two URLs and ignoring the latest one. I took additional code and time to handle this issue.

b. **Blocking various file extensions –** Checking for various file extensions while crawling for URLs in a webpage is very challenging. One need to specify all the extensions available in both upper and lower case to pattern match the file extension of URL to stop it from storing that URL, which makes crawling slow and inefficient. The alternative is checking the URL content of a URL by making a request to it and check for page type. This way I have resolved the issue and have no other file formats in the crawled URLs.

c. **Indexing for every new session of user –** Initially documents are indexed for every new session established by a user with search engine. A session is a semi-permanent interactive information interchange, also known as a dialogue, a conversation or a meeting, between two or more communicating devices, or between a computer and user [3]. Latter indexed dictionary object is serialized once it is built and saved in a file. Serialization is the process of translating data structures or object state into a format that can be stored (for example, in a file or memory buffer, or transmitted across a network connection link) and reconstructed later in the same or another computer environment [4]. As user established a session with search engine by making first request, index object is formed by deserialization of the serialized index file and used it to retrieve results. This has saved lot of time between a user query and displaying results, hence improving the system's efficiency.

d. **Mapping of URL's and Pages downloaded –** The mapping between downloaded pages and its URL has to be established in way that retrieving URL for a given document should be efficient. Initially the mapping is stored in a file which was very inefficient in retrieving the results. A light weight database called SQLite is used to store mappings and other metadata about URL and its page. This way the efficiency of search engine increased a lot.

4. EXPERIMENTS & RESULTS

This section discusses the experiments conducted on search engine and evaluates the results. The search engine is deployed in CSE machines at Computer Science and Engineering department and experiments are conducted on it. The set of five queries are chosen to conduct experiment and manual evaluation of results are recorded. The six queries chosen are:

1. College of engineering
2. College of music
3. Cornelia Caragea
4. Financial aid
5. Blackboard
6. Graduate Student Council

The results of these queries to search engine are displayed in the Table 1. The manual evaluation of each query is measured and displayed in Table 2.

Table 1: Top five results for six queries

Query	Rank	Results
College of engineering	1	http://engineering.unt.edu/news
	2	http://engineering.unt.edu/departments/engineering-technology
	3	http://engineering.unt.edu/departments/computer-science-and-engineering
	4	http://engineering.unt.edu/
	5	http://engineering.unt.edu/about
College of music	1	http://musiced.music.unt.edu/
	2	http://music.unt.edu/musiced
	3	http://calendar.unt.edu/event-calendar/music
	4	http://music.unt.edu/future-students
	5	http://www.music.unt.edu/future-students
Cornelia Caragea	1	http://www.cse.unt.edu/~ccaragea/cv_cornelia.html
	2	http://www.cse.unt.edu/~ccaragea
	3	http://www.cse.unt.edu/~ccaragea/color-picture.html
	4	http://www.cse.unt.edu/~ccaragea/research.html
	5	http://www.cse.unt.edu/~ccaragea/teaching.html
Financial aid	1	http://financialaid.unt.edu/financial-aid-forms
	2	http://financialaid.unt.edu/financial-aid-policies
	3	http://financialaid.unt.edu/how-apply-financial-aid
	4	http://admissions.unt.edu/transfer/how-to-apply-for-scholarships-and-financial-aid
	5	http://financialaid.unt.edu/sources
Blackboard	1	https://bbsupport.unt.edu/content/blackboard-policies
	2	https://bbsupport.unt.edu/
	3	http://learn.unt.edu/
	4	http://clear.unt.edu/blackboard
	5	https://bbsupport.unt.edu/blackboardprivacypolicy
Graduate student council	1	http://vpaa.unt.edu/councils-and-committees
	2	http://research.unt.edu/aout-us/councils-committees
	3	http://studentaffairs.unt.edu/org/university-program-council
	4	http://www.international.unt.edu/content/china-advisory-council
	5	http://gsc.unt.edu/

Table 2: Average Precision for top five results for five queries

Query	Precision
College of engineering	1.0
College of music	0.8
Cornelia Caragea	1.0
Financial aid	1.0
Blackboard	1.0
Graduate student council	0.2
Average Precision	**0.833**

For the experiments where precision is very good, the terms used in the query are equally distributed and weighed in the documents that are retrieved. But for the query "Graduate student council", some terms in the query are present in documents more frequently than documents containing all the terms in the query leading to poor precision of the query. The performance of the search engine recorded average precision of 0.833 for the queries selected.

5. CONCLUSION AND FUTURE WORK

This report discussed all the components involved in implementing the search engine, where the search engine's page searches are limited to UNT domain pages. Results obtained from the experiments show that the implementation has efficacy in retrieving the accurate pages for the user queries.

Possible future work is to implement query expansion to retrieve more relevant documents. And the terms present in the query are not given equal importance or weigh equally while retrieving the documents which reduces the precision a lot for some queries, there can be a possible work in improving in this area in future by using bigram model or other techniques resolving the issue.

6. ACKNOWLEDGEMENTS

The search engine describe in this report was implemented under the guidance of Dr. Cornelia Caragea at the Department of Computer Science and Engineering, University of North Texas.

7. REFERENCES

[1]. The Porter Stemming Algorithm http://tartarus.org/~martin/PorterStemmer/
[2]. Cornelia Caragea. CSCE 5200 Information Retrieval and Web Search Lectures, 2016
[3]. Session https://en.wikipedia.org/wiki/Session
[4]. Serialization https://en.wikipedia.org/wiki/Serialization