Academic Website Data Clustering to Improve Search Results

Anitha Ganesha
CSCI-5502
anitha.ganesha@colorado.edu

Ramya Nair
CSCI-5502
ramya.nair@colorado.edu

ABSTRACT

Academic websites have specific type of data different from the general search engines. This data can be classified into different categories like People, Publications, Courses, etc. Classifying this data into categories and ranking them in increasing order of importance can give improved search results.

This project would be dealing with mining different academic website data, generating classifications dynamically and ranking these classifications so that on a keyword search, the higher rank classification results would be given higher priority. This would provide an improved user experience.

This classification will also help to understand the correlation that exists between different universities within a specific department and within different departments in the same universities.

1. INTRODUCTION

Web mining is one of the major fields in Data mining and Web content mining deals with extracting information from web pages for various purposes. One of the most important such purpose is to cater to search queries.
Currently a large number of generic search engines are widely available which provides satisfactory results to the user provided keywords. In this project, we would specifically look into academic web pages including different departments of various university websites and apply clustering techniques for automatic classification. The generated classifications would stream line the categories within academic websites. Voting algorithms could be used to rank these categories. The rank obtained would be different when applied within academics as compared to that obtained from generic web search. The keyword search can then be mapped to the most highly ranked category. We hope to enhance the user search experience by improving search results applying this technique.
Such automatic classifications could further be applied to varied departments or sub-pages of these websites and results could be correlated between similar departments of various universities or varied departments of same universities

2. RELATED WORK

University of Colorado website http://www.colorado.edu/ has a search engine which has a separate category for people search. It works well, queries quickly and efficiently through its people database which includes Faculty, Staff and even Students. Though it is a fixed category, this search is widely used and is one of the more popular search topics due to its accurate results. This project would be expanding these kinds of search results in dynamic, automated categories.
A number of automated classification techniques have been researched upon in the past. One of the older papers *Web Search Using Automatic Classification* [1], explains some techniques to build automatic classification based on pre-specified categories to improve web search results. Figure [1] briefly shows the basic search architecture. The web crawler obtains the data

from the web, and converts them into pages. A classifier is used to categorize the pages and this data is then stored in the Database. Depending on the keywords used for search, respective pages and categories are provided to the user through the search engine. The automated classifier mentioned in this paper uses some training data to train the classifier. Currently various new techniques are available to dynamically cluster web pages into categories. This project would work on similar search architecture but clusters would be created irrespective of the availability of training data.

One of the more recent papers on *Automatic Structured Web Databases Classification* [2], elaborates of an interesting similarity calculation technique in deep web data using Bipartite graph matching and Hierarchical clustering. Semantic similarity calculation algorithms are used to structure large generic web databases. Our project would be looking for automated structures specifically on academic/university web data using similar techniques.

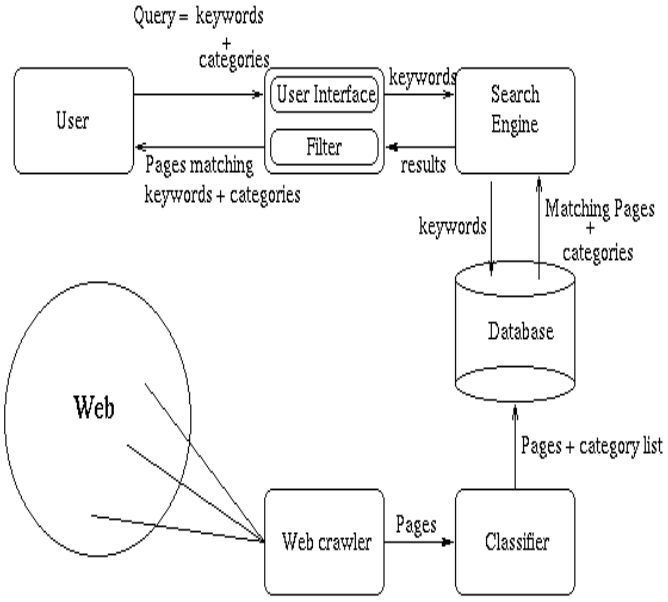


Figure 1. Search architecture [1]

3. PROPOSED WORK

The following steps need to be executed as a part of extracting patterns out of the academic website.

First, a web crawler is run on the websites to extract data in the form of text pages that contains the *url* of the web page and its content. Few of the websites that will be used for initial data set construction include:

http://www-cs.stanford.edu/
http://www.colorado.edu/cs/
http://english.colorado.edu/

Next, with the help of Apache Lucene, a free/open source information retrieval software library, full text indexing will be done.

Following text indexing, dynamic clustering technique will be applied to the data to group the semantically related documents and words into clusters.

Following which, various voting techniques will also be applied to rank the pages based on tokens.

Finally, based on input keywords, the most relevant webpages based on ranking are displayed as output.

4. WORK IN PROGRESS

Data Collection:

The web data from the CU CS department website and the CU english department website was collected using a web crawler called crawler4j. Crawler4j is open source crawler software that provides a simple interface for crawling the web. The advantages of using crawler4j are that it provides a way to setup multi threaded web crawling that fastens the procedure.It also provides options for filtering out pages or contents that are not needed such as .css, .js, .nedia files etc and only allows pages within http://www.colorado.edu/cs/ to be crawled.

The data collected for the CU CS website is about 10 GB and English department is about 18MB

where the contents of each page is stored in different text documents with unique id. The hyperlinks and the actual text contents are separately stored in a different document for every web page.

Web Indexing:

Web indexing refers to indexing the contents of the website. The tool used for this operation is Apache Lucene. It is open source software that contains full text information- retrieval library written in java language. When indexed the document is broken down into a number of terms. The terms are then stored in an index file where each term is associated with the documents that contains it. In order to generate the terms, an analyzer is used.

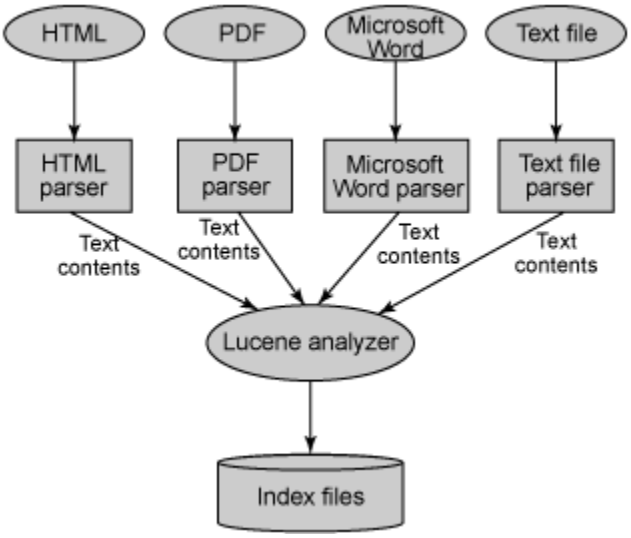
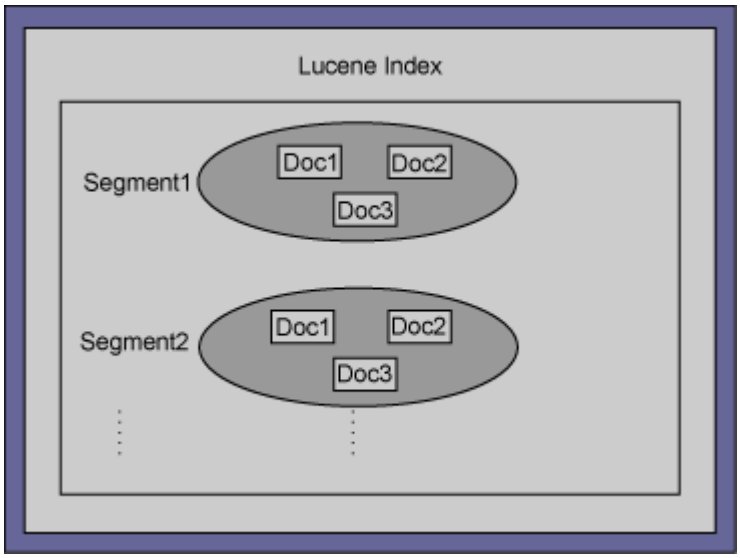


Figure above shows the architecture of lucene. It contains different parser for different types of files. The parser is responsible to extract the text content out of the file. This is passed to a lucene analyzer software which extracts tokens and other related information such as token frequency from the text content. This information is then stored in the index file of lucene.

Analysis of Lucene Index File:



Apache Lucene uses inverted index as its index structure. In inverted indexing, the terms take a center stage where each term points to a list of documents that contain it. The inverted index can be used to find which documents contain certain terms.

The Lucene index file contains segments, wherein it contains a set of documents. The number of segments is determined by the number of documents to be indexed and the maximum number of documents that one segment can contain.

Converting to feature vector:

As described in the previous section, the output of lucene indexing is set of term vectors. But we want to classify the webpages, (also known as features) not the terms. In order to achieve that, we used Apache Mahout to convert these term vectors to feature vectors that could be consumed by any clustering algorithm to form clusters. We used lucene.vector in Mahout and obtained the feature vector in the form out.vec.

Clustering:

Now that feature vectors are ready we worked on various clustering algorithms to evaluate. Apache

Mahout also has many clustering techniques including kmeans, mean –shift, fuzzy kmeans and so on. We started with the basic kmeans clustering for CU CS department and CU English department data.
We gave k ranging from 5 to 50 and iterations ranging from 5 to 50. Various distance measures were selected like, CosineDistanceMeasure, Euclidian distance, Manhattan distance etc. On providing number of clusters, the Mahout algorithm creates its own k initial input and runs the clustering until convergence or until max iteration count is reached.
It created two sets of output files. One set, consists of the cluster results that are obtained after iteration. These are in the form of folder containing the clusters named as 'clusters-N', where N is the iteration number. The final result is stored in a folder post-fixed with the term 'final' and has the iteration number in the folder name. The other set called as clusteredPoints consists of the sequence file, which has the cluster id as the key and a weighted vector variable where weight indicates the probability that the vector belongs to that cluster. The higher the weight the closer it is to the cluster. The distance measure provided as input is used to evaluate the weight.

Manual Evaluation:

The output of the Mahout clustering algorithm is in raw format. For initial understanding of the output we used Mahout clusterdump feature and converted these raw files to .csv format. The csv file consists in tabular form, each cluster as rows. The first column indicates how many objects or document vector are in that cluster.

Results till now:

We saw that both these websites are creating not more than 3 clusters and converged in only 3-4 iterations. This was very suspicious and hence we went back and looked at output at each stage.

Only then we found that there has been some error in the web crawling results, as many of the data files were duplicated. This kind of issue is called crawler trap which needs to be further investigated

Work Remaining:

The entire clustering setup using Lucene and Mahout is completed and any set of data can now be run easily from indexing to clustering.
Our next step is to first fix the issues in web crawling and rerun the rest of the steps. We would continue to evaluate the cluster results until a satisfactory level of clusters are obtained.
After this we intend to look into ways to view these clusters to visually analyze the results.
We would also be running different clustering techniques and comparing the results for same input.
Then if time permits we would rank the these clusters to improve search results

5. EVALUATION

The quality of the results generated needs to be assessed. This includes evaluating the clustering quality, determining the number of clusters generated and the clustering tendency to detect random patterns if any in the structure of the data.
We could expect many similar patterns in academic websites that fall under same educational department. We can evaluate this by comparing the data obtained from CS department of CU boulder website and Stanford university website.
Further, we will also evaluate the patterns that arise when the CU boulder CS website is compared to CU boulder English website. The patterns that will be generated in this case will be different from the patterns generated in the first case.

6. MILESTONES

- Data set collection (2/25/2013 - 3/6/2013): Web crawler such as crawler4j will be used to extract the data sets.
- Indexing (3/7/2013 - 3/15/2013): Apache Lucene tool will be run on the data extracted to generate indexed contents.
- Applying clustering techniques (3/16/2013 - 3/31/2013): Different Clustering techniques will be analyzed and implemented to identify interesting patterns.
- Testing and Evaluation (4/1/2013 - 4/20/2013): Evaluating the quality of the clustering patterns obtained by testing the search results.

7. REFERENCES

[1] Chandra Chekuri; Michael H. Gold ; Prabhakar Raghavan; Eli Upfal , "Web Search Using Automatic Classification,"

[2] XiaoJun Cui; ZhongSheng Ren; HongYu Xiao; Le Xu; , "Automatic structured Web databases classification," Intelligent Computing and Intelligent Systems (ICIS), 2010 IEEE International Conference on , vol.3, no., pp.305-309, 29-31 Oct. 2010

[3] "Latent Semantic Indexing." Wikipedia. Wikimedia Foundation, 27 Jan. 2013. Web. 25 Feb. 2013. http://en.wikipedia.org/wiki/Latent_semantic_indexing.

[4] Han, Jiawei, Micheline Kamber, and Jian Pei. *Data Mining: Concepts and Techniques.*Waltham, MA: Morgan Kaufmann, 2012. Print.