A step towards Interactive Document Clustering

Incorporating dynamic analysis of a dynamic database

[1]Sushila Aghav, [2]Anukriti Sinha, [3]Sarika Shitole, [4]Sandhya Muthe

[1]Assistant Professor, [2,3,4]Student
Department of Computer Engineering,
MIT College of Engineering, Pune, India

***Abstract*—Document clustering has been implemented in innovative ways but has till date refrained from making better use of data and information which can be extracted from the World Wide Web. Most research is dependent on static databases like the well investigated Reuters-21578 news corpus for news articles categorization analysis. This paper presents the use of a dynamic news database which is obtained by using a web crawler and this database is updated daily. On the fly clustering is performed and categories are given as input by the user. To reduce the volume of text being analyzed and avoid misleading results preprocessing techniques has been used. To understand the context of the articles and categorize them further TF-IDF has been performed in this experiment.**

***Index Terms*— Document clustering, Dynamic database, Preprocessing, TF-IDF.**

I. INTRODUCTION

Over the years Document Clustering has been a hot topic but although it is a well investigated domain, the problem is far from being completely solved and also from being considered inconsequential. Various challenges prevail in the field of document clustering amongst which feature selection and size of databases are the most commonly faced issues.

Despite the challenges experienced in the process of Document Clustering, it is a much needed technique which is applied in various applications. Few of these applications are listed below.

1. Finding Similar Documents:- This method is used when the user has spotted a document they like in a search result and want to obtain more documents like that one. The interesting fact here is that clustering is able to discover documents that are conceptually alike in comparison to approaches which are search-based which are only able to discover whether the documents share many of the same words.
2. Organization of Large Document Collections**:-** Information and Document retrieval focuses on finding documents relevant to a particular query, but it fails to solve the problem of finding context or making sense of large numbers of uncategorized documents. It is needed to organize the documents in a taxonomy identical to the one humans would create given enough time and use it as a browsing interface to the original collection of documents.
3. Duplicate Content Detection:- In many applications there is a need to find redundant data or almost similar data in a large number of documents. Clustering is employed for, grouping of related news stories, plagiarism detection and to reorder search results rankings.
4. Recommendation System:- Recommender systems are used by various sites today. In this system a user is recommended products based on what the user has bought or suggested reading based on the articles the user has already read. Clustering of the articles makes it possible in real time and improves the quality a lot.
5. Search Optimization:- Unsupervised learning techniques like clustering can be really helpful in improving the efficiency and quality of search engines as the user query can be first compared to the clusters instead of comparing it directly to the documents and the search results can also be arranged easily.

II. THE PROCESS OF DOCUMENT CLUSTERING

The process of Document clustering is divided into four phases.

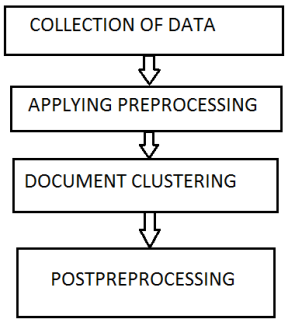

Fig. 1. The four stages of a typical offline document clustering procedure.

Collection of Data refers to generation of the database on which clustering should be applied. For collection of data and storing it in a database, processes like crawling, indexing, filtering and other techniques are used. These techniques collect the documents, index them to store and retrieve the documents in a better way, as well as filter them to remove the noisy irrelevant data like stopwords.
Preprocessing is done to represent the data in a clean form that can be used for efficient and faster clustering.
Document Clustering:- It refers to clustering of documents into categories To begin with let us first understand the aim of document clustering .The aim of a document clustering scheme is to minimize intra-cluster distances between documents, while maximizing inter-cluster distances (using an appropriate distance measure between documents).
Postprocessing is applied in major applications where the document clustering is used, for example, the recommendation system which uses the results of clustering for recommending products or even news articles to the users.

III. GETTING THE DATABASE

While working on a project to create an online news recommendation system, we were faced with the challenge of incorporating the vast amount of news articles present on different websites. Acquiring this information would help us to provide the users with the best articles available on a particular topic on the Internet at that time.

Understanding Document Model

We first need to be able to visualize the web document structure. The web document can be seen as a tuple which consists of following columns-i)Incoming link text ii)Title iii)Page content (contains subparts) iv) A unique document id. A web search can really be understood as a simple query which is given as below:-
SELECT * FROM docs where docs.txt LIKE 'userquery' AND docs.title LIKE 'userquery' AND … ORDER BY 'relevance'
Modern search engines use various features and clues for ranking web documents. These features include page title, meta tags, bold texts, text position on a page and word frequency to name a few. The concept of inverted indexes is commonly used for mapping between the content on the webpage and its location in a database file or a set of named documents. Even inverted indexes cannot handle the millions of documents on a single machine. This calls for parallelizing queries by either segmenting documents or segmentation by search term.

Extraction of news articles using RSS protocol and ROME parser

RSS is an acronym for Really Simple Syndication which is a protocol for extracting news summaries in XML format over the World Wide Web. ROME is a freely available open source parser used to handle RSS feeds. In a more sophisticated explanation, ROME is Apache 2.0 licensed Java based framework which includes parsers and generators for syndication feeds.

IV. PROCESSING OF DATA

We have crawled and stored the data present over the internet. This data is useless unless it is further analyzed. Now the process of information retrieval and data processing needs to be performed to get meaningful results.
Preprocessing is a means of ensuring that optimal performance and quality is attained. It greatly makes processing and analysis of documents easier.The two preprocessing techniques used in our project are:-

1. Stopword Removal or Term Filtering

The words or terms which are used several times in a document and are of no importance for the processing of the document are called stopwords. The most common stopwords are the articles, ‘a’, ’an’ and ’the’. The process of removing these words is known as stopword removal. Standard stopword lists are available which can be modified as the application demands. Apart from removing the redundant words, another term filtering trick is to remove those terms which have low frequencies. This leads to improved speed and memory consumption of the application. Numbers do not play much importance in the similarities of the documents except where dates and postal codes are needed. Thus numbers can also be removed during term filtering.

2. Stemming

The process of reducing words to their stem or root form is known as stemming. For example ‘eat’, ’eating’, ‘ate’ are all forms of the same word used in a different constraint but in terms of measuring similarity these should be considered same.

V. IMPLEMENTING TF-IDF

TF-IDF stands for term frequency-inverse document frequency. This technique is divided into two phases- term frequency and inverse document frequency. Term Frequency is used to predict the importance of a word in specific document .The word which frequently appears in a document are regarded to have a high score. For example, if we have a query like ‘red car’, then term frequency will search for all documents that have a high score for the terms ‘red’ and ‘car’ as well as ignoring documents which do not have a considerable score for theses two terms. Now, some terms or words which appear more frequency in many document like the articles, ’a’, ’an’ and ’the’. These redundant words should have a low score because of their diminished importance in analyzing the documents but Term Frequency can wrongly emphasize such redundant words over meaningful words like ‘red’ or ‘car’. Thus it can become difficult to differentiate relevant and non-relevant documents based on the keywords decided by term frequency alone. This is where IDF or inverse document frequency comes into the picture. IDF runs over the entire document set and diminishes the weight of terms that occur very frequently in the document set as well as increases the weight of terms that occur rarely. Basically, tf-idf is the multiplication of term frequency(tf) and inverse document frequency(idf).Mathematically it is represented as,

$$tf\text{-}idf = tf(t,d) * idf(t,D) \quad (1)$$

where,
tf(t,d)=Raw frequency of term in a document.
Idf(t,D)=It is a frequency/importance of term across all document.

VI. CLUSTERING

The clustering algorithm used in our project is K-mans clustering algorithm.
Let X = {x1,x2,x3,……..,xn} be the set of data points and V = {v1,v2,…….,vc} be the set of centers.
1) Randomly select 'c' cluster centers.
2) Calculate the distance between each data point and cluster centers.
3) Assign the data point to the cluster center whose distance from the cluster center is minimum of all the cluster centers.
4) Recalculate the new cluster center using:

$$v_i = \frac{x_1+x_2+x_3+x_4+x_5\ldots\ldots\ldots\ldots x_n}{c_i} \quad (2)$$

where, 'ci' represents the number of data points in ith cluster.
5) Recalculate the distance between each data point and new obtained cluster centers.
6) If no data point was reassigned then stop, otherwise repeat from step 3.

VII. RESULT

This project uses Apache based framework support in the form of Hadoop ecosystem, mainly HDFS (Hadoop Distributed File System) and Apache Mahout.
The flow of the project is shown below.

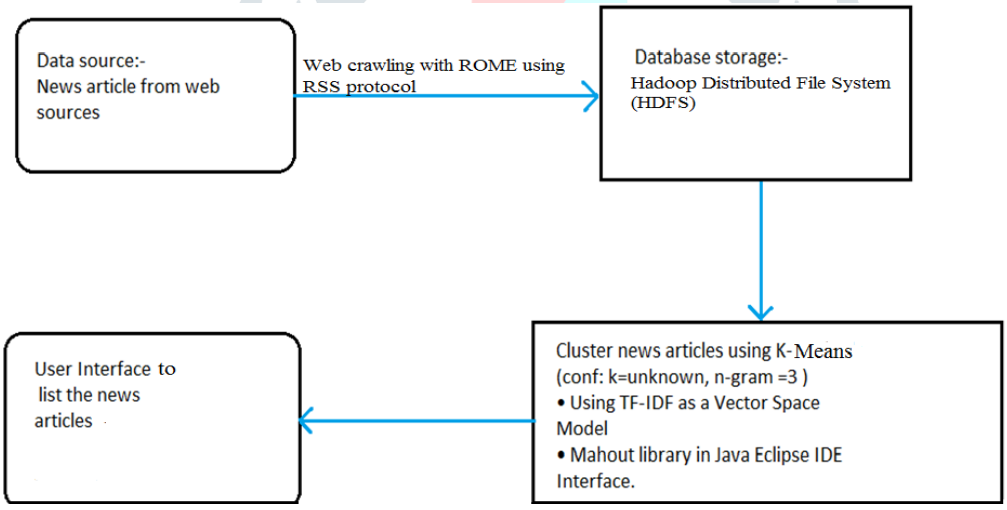

Fig. 2. The flow/block diagram of the project.

As discussed earlier, the database is extracted from the internet using RSS and ROME parser. This database needs to be stored in a place that can handle large amounts of data and for this purpose we have implemented the use of Hadoop's internal storage and file sytem called HDFS(Hadoop Distributed File System) . HDFS is a client-server based architecture which consists of two types of nodes, Datanodes and Namenodes. A single Namenode is first created to store the metadata and the locations of the various data which is to be stored. Datanodes contain the actual data being stored, which gets divided into chunks of blocks to manage the large amounts of stored data. A secondary namenode exists just in case if the original namenode breaks down.

Preprocessing in the form of stopword removal and stemming has been performed for converting raw data into understandable format. It also reduces the volume of data being analyzed and increases efficiency of the analysis.

TF-IDF has been used in the project to understand the context of the article by calculating the most occurring as well as the considering the rarest words present in a document.
Considering the aspect of using K-means clustering algorithm, it is a basic fact that k-means only deals with numbers and not text or words as input. Thus using TF-IDF as a vector space model helps convert words to a number format which can be given as input to K-means algorithm. Using TF-IDF results in creation of key-value pairs called sequence files. These sequence files contain

indexes for the terms starting from 0 and the corresponding term frequency of that particular indexed word. These sequence files are passed using vector format to k-means clustering algorithm.
We have implemented this project on netbeans IDE using Mahout libraries for k-means and TF-IDF. The code has been deployed on Tomcat server.

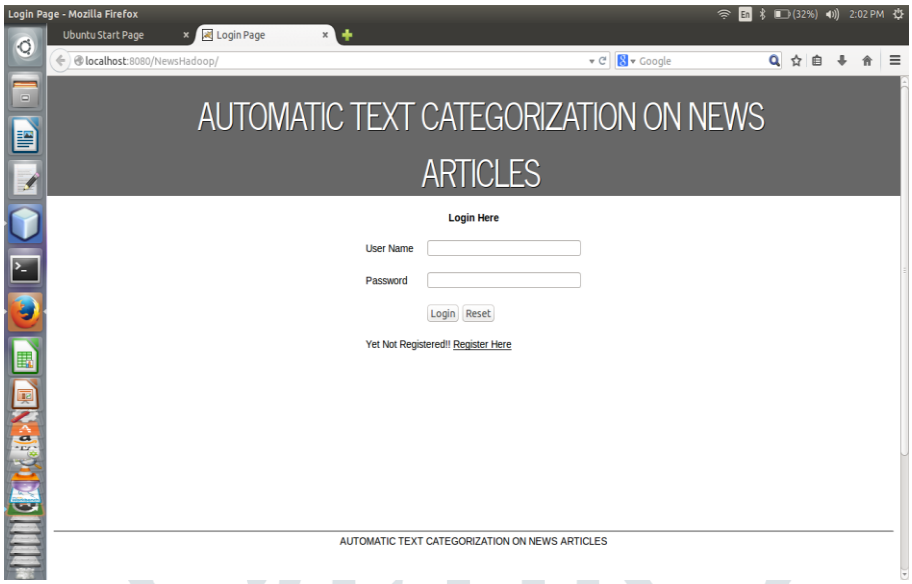

Fig. 3. Snapshot of login page.

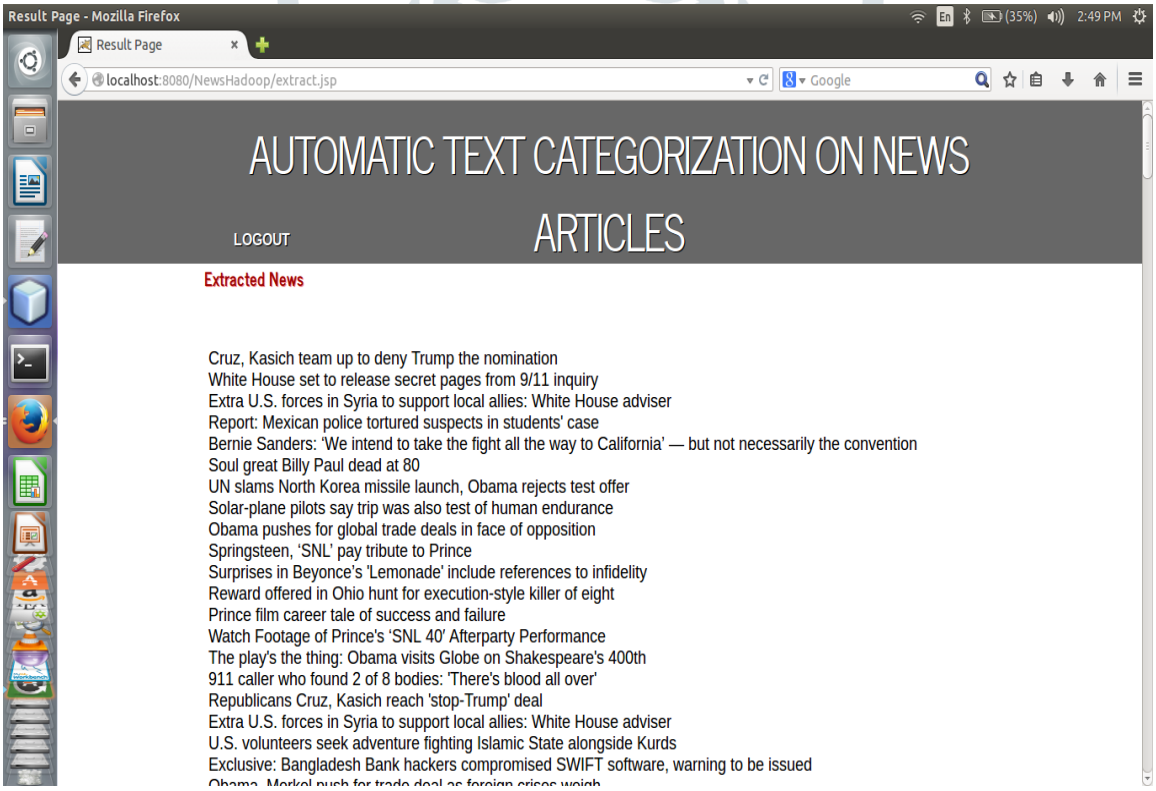

Fig. 4. Snapshot of page showing the extracted news.

```
204wt: 1.0 distance: 16.768553581114013  vec: Document 0 = [168:5.879,
221:5.879, 335:4.780, 958:5.879, 1123:5.879, 1131:5.474, 1176:5.186,
1267:5.474, 1293:5.879]
246wt: 1.0 distance: 16.5197130328023  vec: Document 1 = [261:5.474,
578:5.186, 794:5.879, 809:5.879, 833:5.879, 955:4.270, 1153:5.879,
1174:5.879, 1354:5.186]
204wt: 1.0 distance: 10.722890881276777  vec: Document 10 = [329:4.780,
464:5.879, 956:4.780, 1247:5.879]
219wt: 1.0 distance: 15.260906660197069  vec: Document 100 = [254:5.474,
263:5.186, 298:5.474, 348:5.879, 753:5.879, 830:5.474, 848:5.879,
1299:3.800]
109wt: 1.0 distance: 19.703869403579272  vec: Document 101 = [70:5.879,
527:5.879, 599:5.474, 728:5.186, 790:5.879, 800:5.474, 837:5.474,
920:5.474, 1102:4.626, 1197:5.879, 1234:5.879, 1296:4.087, 1373:5.879]
211wt: 1.0 distance: 14.9646785895068  vec: Document 102 = [64:4.270,
233:4.963, 428:5.879, 505:4.780, 924:5.879, 1012:5.474, 1018:5.879,
1082:5.186]
198wt: 1.0 distance: 16.099000658443046  vec: Document 103 = [93:4.963,
261:5.474, 283:4.963, 334:4.174, 530:5.186, 700:5.879, 905:5.879,
919:5.879, 1049:5.879]
248wt: 1.0 distance: 16.825784363993815  vec: Document 104 = [34:4.493,
188:5.879, 397:5.879, 610:5.474, 849:5.879, 1098:5.879, 1127:5.879,
1171:5.186, 1270:5.879]
107wt: 1.0 distance: 11.500536245911348  vec: Document 105 = [283:4.963,
556:4.493, 733:5.186, 978:5.186, 1060:5.879]
191wt: 1.0 distance: 15.142052073432632  vec: Document 106 = [165:5.474,
223:4.626, 266:5.186, 574:5.879, 805:4.963, 874:5.879, 1090:5.879,
1289:4.963]
142wt: 1.0 distance: 16.637704019759727  vec: Document 107 = [169:5.879,
265:5.879, 645:5.879, 861:5.879, 1069:5.879, 1164:5.879, 1260:5.879,
1300:5.879]
```

Fig. 5. Snapshot of page showing part of the sequence file generated and the final clustering.

The Figure 5 shows the final clustering process. The combined weights of documents are calculated and sequence files are created using TF-IDF and k-means algorithm. Then the documents having the same weights are combined into single clusters.

ACKNOWLEDGEMENT

The authors would like to appreciate the guidance and encouragement provided by the head of department, teachers and students of Department of Computer Engineering, MIT College of Engineering.

REFERENCES

[1] Document Clustering, Pankaj Jajoo, IITR.
[2] Noam Slonim and Naftali Tishby. *"The Power of Word Clusters for Text Classification*"
School of Computer Science and Engineering and The Interdisciplinary Center for Neural Computation The Hebrew University, Jerusalem 91904, Israel
[3] Mahout in Action, Sean Owen, Robin Anil, Ted Dunning and Ellen Friedman Manning Publications, 2012 edition
[4] Michael Steinbach, George Karypis, and Vipin Kumar. *"A Comparison of Document Clustering Techniques*"
Department of Computer Science and Engineering, University of Minnesota