INTERNATIONAL JOURNAL OF COMPUTER ENGINEERING & TECHNOLOGY (IJCET)

ISSN 0976 – 6367(Print)
ISSN 0976 – 6375(Online)
Volume 6, Issue 5, May (2015), pp. 08-12
© IAEME: www.iaeme.com/IJCET.asp
Journal Impact Factor (2015): 8.9958 (Calculated by GISI)
www.jifactor.com

IJCET

© I A E M E

AUTOMATIC DOCUMENT CLUSTERING

Mona Pardeshi[1], Neha Puranik[2], Aishwarya Tiwari[3], Prof. P.Y.Pawar[4]

[1,2,3,4]Department of Information Technology, Sinhgad Academy of Engineering, Kondhwa, Pune, Maharashtra, India

ABSTRACT

Automatic document clustering has played an important role in the field of information retrieval. The aim of the developed this system is to store documents in clusters and to improve its retrieval efficiently. Clustering is a technique aimed at grouping a set of objects into clusters. Document clustering is the task of combining a set of documents into clusters so that similar type of documents will be store in one cluster. We applied non overlapping method to store document into cluster. In this project, we write an algorithm which will calculate similarity of document's keywords and according to its similarity points it will either put into existing cluster or new cluster is created and stored into that cluster. To find keywords from document various techniques are used like tokenization, stop word removal, stemmer, TF*IDF calculation.

Keywords: Document Clustering, Stemmer, Stop words removal, TF*IDF, Tokenization

1. NEED OF THE SYSTEM

We developed software for stand-alone system. Today each person has its own computer or laptop. In general they used to store their all types text, multimedia documents in it. We dealing with all types of text documents which include file types having extension pdf, doc, docx, ppt, pptx, txt,xls,xlsx. its drawback of human memory that it can forget things so its possible that today you store text file on pc and later after few weeks or month or years forgot its path. Now to get that file either you have to remember file path or you have to go through all drives and need check in each folder. This is very time consuming task to solve this problem we developed our system. You can store file into system by specifying folder or by specifying particular file as input to the system. User can also specify its keywords explicitly. Once you make entry of file in application you can search that file using our software by just entering file keywords, extension or by name. These three levels of search provide user flexibility in search and also save time of user.

2. WORK FLOW OF SYSTEM

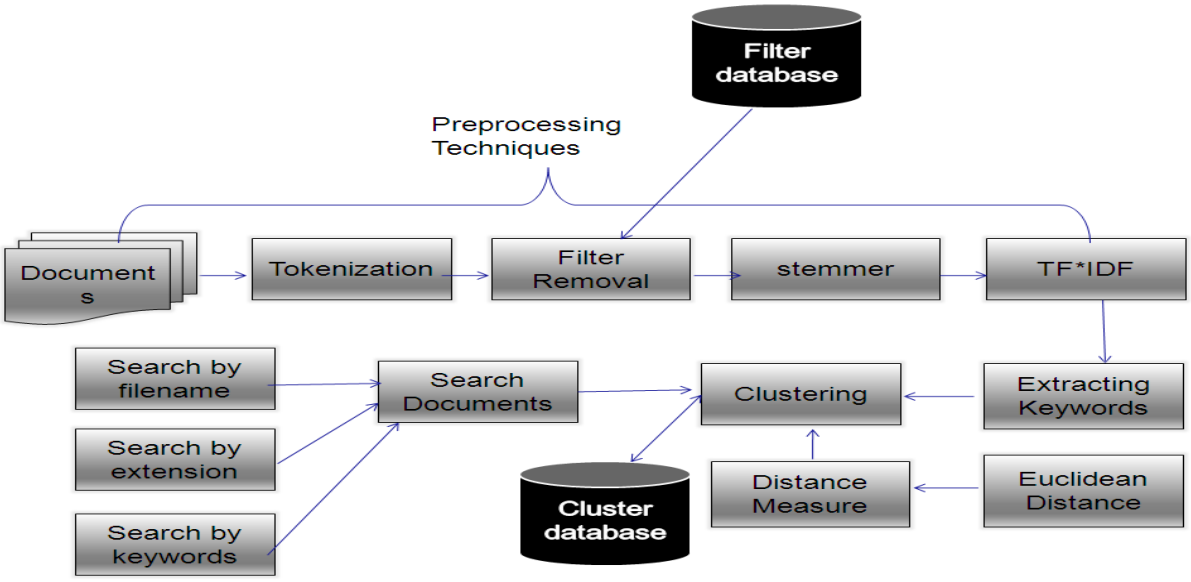

Fig: System Architecture and Work Flow of system.

System workflow is mainly divide into three parts they are:- first is Pre-processing phase , second is Clustering phase and third is document searching phase.

INPUT DOCUMENTS

All types of documents having extensions like txt, doc ,docx, pdf, xls, xlsx, ppt, pptx can be given as input to the system.

2.1 Document pre-processing phase

To make clustering process more efficient, before the documents are given to the main system all the input documents are pre-processed. The pre-processing phase brings accuracy and efficiency in our developed system for clustering process. The pre-processing phase is divided into following stages:-

2.1.1 Tokenization

Tokenization means breaking up the streams of text into small tokens like separating words, phrases, symbols and punctuation marks. All types of documents like txt, doc ,docx, pdf, xls, xlxs, ppt, pptx can be given as input to the system. How actually the Tokenization process takes place that means how to break streams to small words? Java provides us with a class named as StringTokenizer it can break stream of text to small unit of string. Or one can also use split function.

Tokens can be separated or recognized by whitespace characters like space or blank space or line break or punctuation characters. The resulting list of tokens may contain or may not contain punctuation characters and whitespace like blank space. Most of the programming languages also make use of English language, where words are delimited by whitespace, where tokenization process can be easily applied. However, tokenization process is more difficult for languages such as Chinese and Japanese which have no word boundaries like whitespaces. [3]

After tokenization process is completed the output of this step is given as input to next step Filter in pre-processing phase.

2.1.2 Filter (stop word removal)

While clustering the documents, Stop words are those words which occur more frequently but which does not have meaning. Words like the, is a, of, then, this, these, etc which have less importance are removed in this step. Removal of these words reduces the number of words in document. These words are also called as stop words or high frequency words. For this system, we have created a database, which have all the common stop words. The stop words are removed by comparing the input documents tokens and database of stop words.
Some most common term filtering methods are:

- Removing the terms which have low document frequencies. This technique is used to improve the memory and speed consumption for execution of application.
- Removing numbers which do not play much importance in documents except dates and postal codes. So these numbers are removed.[2]
  After Filtering process is completed the output of this step is given as input to next step Stemming in pre-processing phase.

2.1.3 Stemming

Stemming is the step which brings the root word for the particular word. Stemming process is used to reduce derived word to their root or base form. [3]
Example:-
1) Clustering can be shorten to its root word called Cluster.
2) The word Clusters can be shortens to its root word called Cluster.
In Stemming process, tokens are eliminated by reduce frequency of the same word which are occurring in its different form/inflected form. Stemmer algorithm finds root word by eliminating postfix of the words for obtaining keywords. Postfix like ing, ed, tion, tive , etc are removed.

2.1.4 TF*IDF

TF stands for term frequency and IDF stands for Inverse document frequency. Product of TF and IDF weight is a statistical measure which is used to determine how important a word is to a document in a collection or corpus. TF*IDF weighting scheme are used by search engines as a tool for ranking and scoring a document's relevance for a given a user query. [1] To get the product of TF and IDF first we have calculate TF and IDF individually. Formula for calculating TF and IDF is described below:

2.1.4.1 Term Frequency (TF)

The Term Frequency computes how frequently a particular term(word) occurs in a particular document. As every document will be having different length, it is possible that a term would appear much more times in long documents than shorter ones. Thus, the term frequency is often divided by the document length (i.e the total number of terms in the document) as a way of normalization:

Calculation of TF:

TF (t) = (Number of times term t appears in a document) / (Total number of terms in the document). [1]

2.1.4.2 Inverse Document Frequency (IDF)

The Inverse Document Frequency measures how important a term is in the collection of documents or corpus. Inverse document frequency actually measures in how many document a particular term (word) occurs.

Calculation of IDF:

IDF (t) = log_e (Total number of documents / Number of documents with term t in it). [1]

2.2 Clustering Phase

Clustering Algorithm is used to store the document in particular cluster. if document similar with any cluster then it will stored in existing cluster else new cluster is created.

Algorithm Steps:-

1) check current file is first file in the system <if true then>
1.1) Create new cluster.
1.2) Add current file to newly created cluster.
1.3) Calculate centroid.

2) else (if not first file then)
2.1) Check if file match is not to any cluster <if true then>
2.1.1) Create new cluster
2.1.2) Add current file to cluster
2.1.3) Calculate Centroid.

2.2) Else Check if file is match to only one cluster <if true then>
2.2.1) Update TF*IDF of current cluster in which document is going to store.
2.2.3) Add unmatched keywords and its TF*IDF of document to cluster.
2.2.3) Add current file to cluster.
2.2.4) Calculate Centroid.

2.3) Else Check if File match to more than one cluster <if true then>
2.3.1) Find cluster to which document is very much similar than any other cluster.
2.3.2) Calculate distance measure using Euclidean Distance formula.
2.3.3) Assign document to that cluster having minimum ED value.
2.3.4) If two cluster have same ED value with document then assign document to first cluster in a queue.
2.3.5) Update TF*IDF of current cluster in which document is going to store
2.3.6) Add Unmatched keywords and its TF*IDFs of document to cluster.
2.3.7) Calculate Centroid

2.3 Searching Phase

We allow user to search for document by name of file or by extension of file or by entering keywords. This make user's task easy and also simplify the time and complex computation required for searching. If user knows/ remember only name of the file he/she can search by name, if user knows /remember extension of file he/she can search by file extension, if user knows/ remember only content of the file he/she can search by entering keywords of file.

3 FUTURE SCOPE

Currently we developed system to retrieve text document in a faster way and in an organized manner.

In future we can also build a system that can stored process and search the multimedia objects. Also we can work to improve the performance of the algorithm.

4 CONCLUSION

Thus we learnt different pre-processing techniques and implement them by simplifying it. And based on concept of clustering we implement an algorithm that can store document in cluster. By using above all concept we devised a system for easy retrieval of document. In this paper we mention all steps for implement document clustering technique in simple manner. Thus we successfully developed system by simplifying data retrieval techniques which is useful for any person who wants to store its preserved file from loss due to forgetting location or name.

5. ACKNOWLEDGEMENT

Our group would like to thank our project guide Prof. P.Y.Pawar for her support and guidance while developing our project. we to thankful to our entire IT department staff for giving us guidance and also thanks to all friends for their support.

REFERENCES

Journal Papers

1. "Interpreting TF-IDF term weights as making relevance decisions". ACM Transactions on Information Systems, 26 (3). 2008. by H. Wu and R. Luk and K. Wong and K. Kwok.
2. Neeti Arora and Dr.Mahesh Motwani, "A Distance Based Clustering Algorithm" International journal of Computer Engineering & Technology (IJCET), Volume 5, Issue 5, 2014, pp. 109 - 119, ISSN Print: 0976 – 6367, ISSN Online: 0976 – 6375.
3. Syed Abdul Sattar, Mohamed Mubarak.T, Vidya Pv and Appa Rao, "Corona Based Energy Efficient Clustering In WSN" International journal of Computer Engineering & Technology (IJCET), Volume 4, Issue 3, 2013, pp. 233 - 242, ISSN Print: 0976 – 6367, ISSN Online: 0976 – 6375.
4. Ms. Meghana. N.Ingole, Mrs.M.S.Bewoor, Mr.S.H.Patil,, "Context Sensitive Text Summarization Using Hierarchical Clustering Algorithm" International journal of Computer Engineering & Technology (IJCET), Volume 3, Issue 1, 2012, pp. 322 - 329, ISSN Print: 0976 – 6367, ISSN Online: 0976 – 6375.

Books

5. Document Clustering by Pankaj Jajoo, Master of Technology, Indian Institute of Technology, Kharagpur in 2008. Under the Guidance of Prof. Sudeshna Sarkar Professor, Department of Computer Science & Engineering Indian Institute of Technology Kharagpur WB, India-721302
6. "Topic-Based Clustering of News Articles", http://pdf.aminer.org/000/006/766/ topic_ based_clustering_of_news_articles.pdf