ISSN XXXX XXXX © 2018 IJESC

Research Article Volume 8 Issue No.7

Design and Development of Optimum HAC and Comprision with Flat Clustering in Searching Web Documents

Priti Rawat[1], Jay Kumar[2]
M.Tech Student[1], Assistant professor[2]
Department of CSE
RGEC, Meerut, India

Abstract:
Users of Web search engines are often forced to sift through the long ordered list of document "snippets" returned by the engines. The IR community has explored document clustering as an alternative method of organizing retrieval results, but clustering has yet to be deployed on the major search engines. Document clustering is a subset of the larger data clustering, which carries concepts from the fields of information retrieval (IR), natural language processing, and machine learning, among others. Document clustering will hereafter be simply referred to as clustering. Document clustering has been investigated in different areas of text mining and information retrieval. Document clustering has been studied deeply because of its wide application in areas such as Web Mining, Search Engine and Information Retrieval. Document clustering is the automatic organization of documents into clusters or groups, so that, documents within a cluster have high similarity in comparison to one another, but are very dissimilar to documents in other clusters . In other words, the grouping is based on the principle of maximizing intra cluster similarity and minimizing inter-cluster similarity. Clustering can also speed up search. For grouping documents in a cluster their similarity will be calculated, various similarity measure algorithms are available. A similarity measure is a function which computes the degree of similarity between a pair of vectors or documents – since queries and documents are both vectors, a similarity measure can represent the similarity between two documents, two queries, or one document and one query. With similarity measure between query and documents – it is possible to rank the retrieved documents in the presumed importance. The proposed methodology clusters the web documents in a single cluster, for this pre processing of web documents into textual form is required, then flat cluster of documents are generated .The web documents are grouped into single cluster by measuring the similarity between the documents using cosine similarity measure. Then the clusters are merged according to agglomerative approach from bottom to top.

1. INTRODUCTION

Today web is the main resource for the text documents. The amount of textual data available to us is consistently increasing; approximately 80% of the information of an organization is stored in unstructured textual form in the form of reports, email, views and news. Information intensive business processes demand that we transcend from simple document retrieval to "knowledge" discovery. The need of automatically extraction of useful knowledge from the huge amount of textual data in order to assist the human analysis is fully apparent. Market Trends based on the content of the online news articles, sentiments, and events is an emerging topic for research in data mining and text mining community. We have a set of training records D = {X_1, . . . , X_N}, such that each record is labeled with a class value drawn from a set of k different discrete values indexed by {1 . . . k}. The training data is used in order to construct a classification model, which relates the features in the underlying record to one of the class labels. So extracting information from many web resources and proper categorization and knowledge discovery is an important area for research. Clustering means grouping of documents which are similar to each other into one group. The main uses of clustering of documents are –

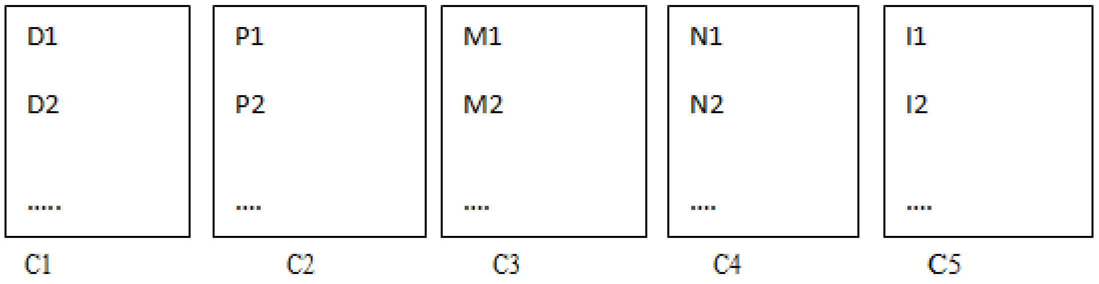
1. If a collection is well clustered, we can search only the cluster that will contain relevant documents.
2. Searching a smaller collection improved effectiveness and efficiency.

Proper clustering of web documents is required by search engine for efficient searching of documents according to the terms. The clustering technique limited the search of the query to a specific set of documents and so the time of the searching to find the relevant document could be saved. This technique can be utilized by search engines to provide relevant results to the user according to query. This paper discusses the three main issues of clustering –

1. To decide the nature of for comparison with other documents.
2. Selection of algorithm is selected for sending the document to a particular cluster.
3. Increasing the vocabulary dictionary word files, which are used for deciding to which
cluster the document belongs?

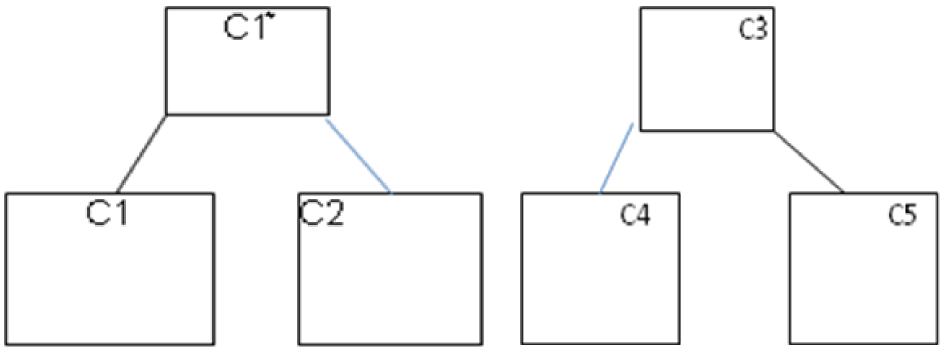
The proposed research work suggests creation of cluster keyword file, which contain keywords (or terms) related to the documents in the cluster. Each cluster maintains its own cluster keyword file , which is used to decide the appropriate cluster for a web document. The work has been divided into two sections: html document processing and automatic generation of clusters. In first section web documents which are in the form of html documents are parsed by removing tags from the html document and converting into a text file, then from this text file stop words, cue words and most frequently used words such as the, is, are, they etc are removed. The automatic generation of cluster is done with the help of maintaining cluster keyword file that contains the keywords presented in the documents of a cluster, the cluster keyword file maintains the keywords of the documents appearing in the cluster , for deciding the appropriate cluster for a new document the similarity is measured between the keywords of the document and the terms of the cluster keyword file of each of the cluster and if the similarity measure is equal to or greater than a decided threshold value , the new web document is assigned to that particular cluster.

Flat Cluster Generation



Here we are having five clusters. Cluster C1 contains different documents like D1, D2, D3 upto Dk. In the same way cluster C2 contains different documents like P 1 , P2, P3 upto pk. Cluster C3 contains documents M 1 , M2, M3 upto mk. Cluster C4 contains documents N1, N2, N3 upto nk and Cluster C5 contains documents I1, I2, I3 upto ik. This representation is at the ground level.

First Level Cluster Generation by Single Linkage



Here the generation of new cluster is done, the documents of cluster C1 is having similarity with the documents of cluster C2 so these clusters will be merged and will form a new cluster C1' at first level. In the same way the documents of cluster C4 is having similarity with the documents of cluster C5 so they will be merged and will form a new cluster C3'.

2. PROPOSED METHODOLOGY

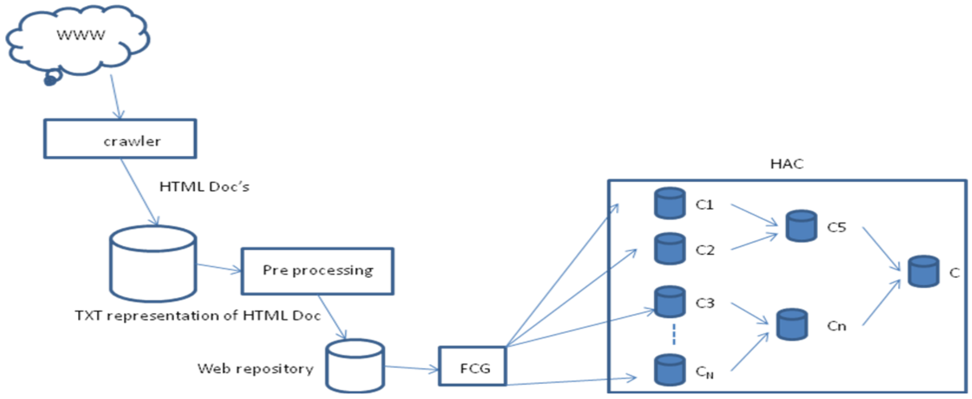


Figure. 2.1 Architecture of HAC

The proposed work is divided into two modules: html document processing and automatic generation of clusters for web documents.

2.1.1 Html document processing: Html preprocessing of web documents consists of steps that take as input a Html document and output a text file. These steps typically consist of:

Filtering: The process of removing special characters and punctuation that are not thought to hold any discriminative power under the vector model. This is more critical in the case of formatted documents, such as web pages, where formatting tags can either be discarded.

Stopword removal: A stopword is defined as a term which is not thought to convey any meaning as a dimension in the vector space (*i.e.* without context). A typical method to remove stopwords is to compare each term with a compilation of known stopwords.

Pruning: Removes words that appear with very low frequency throughout the corpus. The underlying assumption is that these words, even if they had any discriminating power, would form too small clusters to be useful. A pre-specified threshold is typically used, *e.g.* a small fraction of the number of words in the corpus. Sometimes words which occur too frequently (*e.g.* in 40% or more of the documents) are also removed.

General Architecture for Html document processing

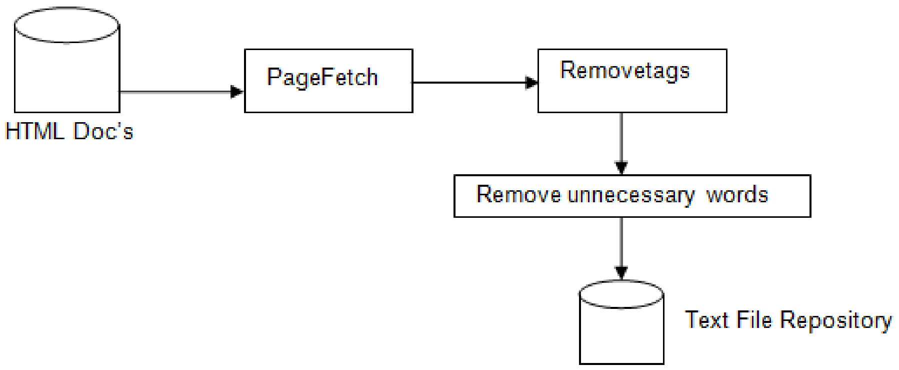


Figure 2.1.1 Html document processing

Modules Description

PageFetch

Fetch page one by one from the HTML Doc's repository and send them to the process "Remove Tags".

Remove Tags

Remove tags from the html file such as <html>, <h1>, <img>, <script>, <style> etc. which have no meaning with the contents of the web document.

Remove unnecessary words

Remove stop and cue words which are frequently used and has no necessary meaning in the document such as this, that, is, are, am, yesterday, Sunday, Monday, these, those etc.

Algorithm for Html document processing

```
HtmlDocs[]
while(no more files in HtmlDocs)
{      file1 ← RemoveTags(file)
textfile ← RemoveStopWords(file1)
save(file1)
}
```

2.1.2 Domain Specific keyword based automatic web document clustering

The automatic generation of cluster is the process of maintaining a cluster keyword file, which contains the keywords(terms) related to the documents for a given cluster, along with terms with one more field is maintained in the cluster keyword file that accounts the docid's of the documents in which the term is available. The cluster is shown in the figure below-

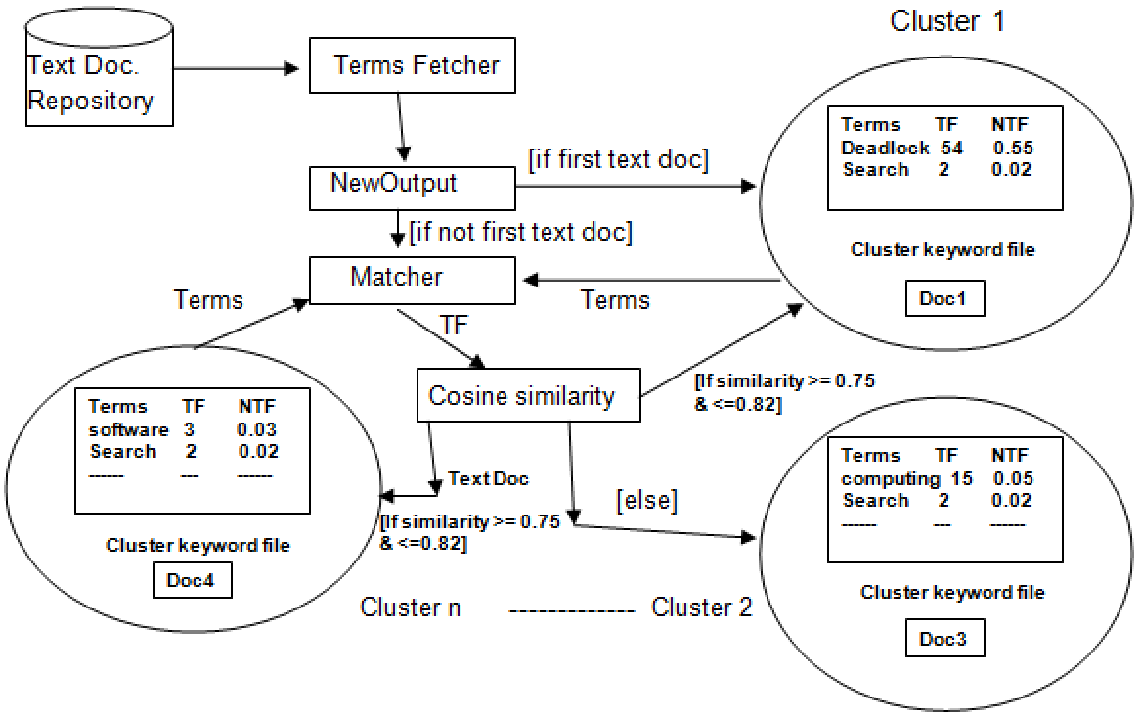


Figure 2.1.2 Domain Specific keyword based automatic web document clustering

Modules description

Term Fetcher

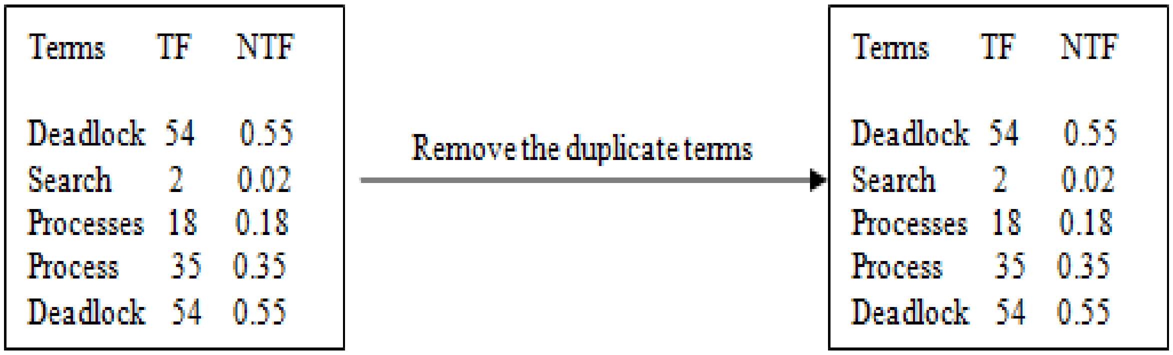
Extract the keywords (terms) from the text document repository. The algorithm is shown in the fig. below-

```
TermFetcher(textfile)
{
file_tokens[]← converttotokens(textfile)
while(no more tokens in file_tokens[])
{
if(file_tokens[i] = numeric value)
{
Remove file_tokens[i] from file_tokens[]
}
NewOutput(filetokens[])
}
```

Figure. 2.1.3 Algorithm for term fetcher

NewOutput

Remove the duplicate terms from the cluster keyword file and also checks the existence of clusters when first document comes and if the no new cluster exists then create a new cluster and assign the document to the new cluster. The figure below shows the removal of duplicate term “Deadlock” that is appearing twice in the cluster keyword file and generates a new cluster keyword file that contains each term only once in the file.



Terms	TF	NTF
Deadlock	54	0.55
Search	2	0.02
Processes	18	0.18
Process	35	0.35
Deadlock	54	0.55

Terms	TF	NTF
Deadlock	54	0.55
Search	2	0.02
Processes	18	0.18
Process	35	0.35
Deadlock	54	0.55

Figure. 2.1.4 Removal of duplicate terms from cluster keyword file

The algorithm is shown in the fig. below-

```
NewOutput(file_tokens[])
{
str[][]← file_tokens[]
str[][]← RemoveDuplicateterms(str[][])
str[][] ← calculateTF(str[][])
if(cluster exsists)
{
while(has more clusters)
{
cluster_st[] ← extracttermsofCKF(cluster[i])
matcher(str[][],cluster_st[])     }
}
else
{
createnewcluster()
createclusterkeywordfile()
}
```

Figure. 2.1.5 Algorithm for NewOutput

Matcher

Extracts the terms of the cluster keyword file one by one and calculate the term frequency and normalized term frequency for the terms of the cluster keyword file existing in the incoming document. The algorithm is shown below:

```
Matcher(str[][], cluster_st[])
{    while(cluster_st[] has more elements)
{      if(cluster_st[i] = str[i][])
writetoexcel(str[i][i]) // Keyword, Term Frequency
}   calculate NTF and write to excel file
cosinesimilarity() }
```

Figure. 2.1.6 Algorithm for Matcher

cosine Similarity measure: It will measures the similarity between the cluster keyword file and the new coming document by taking the summation of multiplication of NTF values of both the files and if the value comes lies between the threshold value 0.75 and 0.82, the document assigned to that particular cluster. The algorithm is shown below:

```
Cosinesimilarity()
{
threshold ← sim(clusterkeywordfile, text_file)
if(threshold >= 0.75 and <=0.82)
cluster[i] ← textfile
}
else
{    newcluster ← createnewcluster()
newcluster ← textfile }
```

Figure. 2.1.7 Algorithm for Cosine Similarity

3. EXPERIMENT AND RESULT ANALYSIS

3.1 Calculation of threshold value for similarity measure between two documents

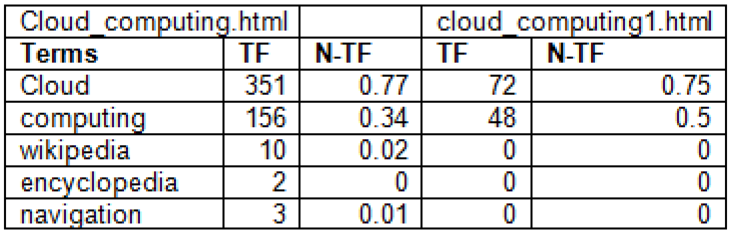
For deciding the threshold value of the cosine similarity between two similar documents is done by taking ten documents of which some of the documents are similar to each other by content wise. The documents taken are as follows- cloud_computing.html, deadlock_in_operatingsystem.html, deadlock.html, cloud_ computing1. html, process.html, microprocessor.html, image processing. html, computerorganization.html, kernel.html and microarchitecture.html. Out of these ten documents two sets of documents are similar to each other by content wise, are as follows -

S1= {cloud_computing.html, cloud_computing1.html} and

S2 = {deadlock_in_opeatingsystem.html, deadlock.html}.

The tables below show the calculation of cosine similarity between the documents of the sets S1 and S2.

Table 3.1.1 TF and NTF values of 2 documents



Cloud_computing.html			cloud_computing1.html	
Terms	TF	N-TF	TF	N-TF
Cloud	351	0.77	72	0.75
computing	156	0.34	48	0.5
wikipedia	10	0.02	0	0
encyclopedia	2	0	0	0
navigation	3	0.01	0	0

Table 3.1.1 shows the term frequency (TF) and normalized term frequency (N-TF) of the terms of the document cloud_computing.html and term frequency (TF) and normalized term frequency (N-TF) of the document cloud_computing1.html with respect to the terms of the cloud_computing.html. Term frequency (TF) of a specific term is calculated as the total number of occurrences of a specific term in a given document. The normalized term frequency or normalized Euclidean length (N-TF) is calculated as –

N-TF (cloud) for cloud_computing.html = 351 / sqrt ($351^2+156^2+2^2+3^2+6^2+$………for all terms)

= 0.77

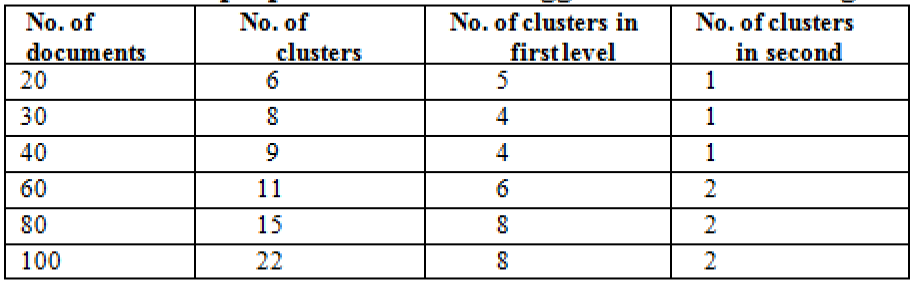
N-TF (cloud) for cloud_computing1.html = 72 / sqrt ($72^2+48^2+1^2$………for all terms)

= 0.75

Now, the cosine similarity between two documents is calculated as-

Simarity_measure (cloud_computing.html, cloud_computing1.html) = 0.77*0.75 + 0.34*0.5 + ………for all the terms = **0.86** ……….. value 1.

5.1a Cluster Structure for proposed hierarchical agglomerative clustering model :



No. of documents	No. of clusters	No. of clusters in first level	No. of clusters in second
20	6	5	1
30	8	4	1
40	9	4	1
60	11	6	2
80	15	8	2
100	22	8	2

4. CONCLUSION AND SCOPE OF FUTURE WORK

This research work can be enhanced further by making the model distributed so, that the work is divided among the different machines and the efficiency increases, by applying agglomartive concept so, that the clusters having more inter cluster similarity combine to form the one cluster by this the number of clusters may reduced, as cluster keyword file maintain another file that contains the summary of all the docments in the cluster, this may help the viewer to identify the cluster of his or her choice.

5. REFERENCES

[1]. Nicholas O. Andrews and Edward A. Fox, "Recent Developments in Document Clustering", thesis, Deptt. Of CS, Virginia Tech, October 16, 2007.

[2]. Christopher D. Manning, Prabhakar Raghavan, Hinrich Schütze, "An Introduction to Information Retrieval", Online book, Cambridge University Press Cambridge, England, 2009.

[3]. Pankaj Jajoo, "Synopsis for project on document clustering", thesis(M.tech), CSE, IIT Khargpur.

[4]. Jayanthi Manicassamy, "Rank Based Clustering For Document Retrieval From Biomedical Databases", International Journal on Computer Science and Engineering Vol.1(2), 2009, 111-115.

[5]. P. Ponmuthuramalingam, "Effective Term Based Text Clustering Algorithms", (IJCSE) International Journal on Computer Science and Engineering Vol. 02, No. 05, 2010, 1665-1673.

[6]. Jain and R. Dubes. "Algorithms for Clustering Data." Prentice Hall, 1988

[7]. Sanjiv K. Bhatia. "Adaptive KMeans Clustering" American Association for Artificial Intelligence, 2004.

[8]. Chris Staff: Bookmark Category Web Page Classification Using Four Indexing and Clustering Approaches. AH 2008:345-348

[9]. Fabrizio Silvestri, Raffaele Perego and Salvatore Orlando. "Assigning Document Identifiers to Enhance Compressibility of Web Search Engines Indexes" In the proceedings of SAC, 2004.