2007

Clustering Blog Information

Mayank Prakash Jaiswal
San Jose State University

Clustering Blog Information

CS-298 Report

By

Mayank Prakash Jaiswal
Advisor: Professor H. Chris Tseng **Committee Members: Dr.T.Y.Lin**
Dr.Suneuy Kim
Computer Science Department
San Jose State University

Table of Contents:

List of Figures:

List of Tables:

Abstract

Blogs form an important source of information in today's internet world. There are blogs on different topics such as technical, health, electronic gadgets, shopping, etc. However, most of the blog websites have the blogs arranged in chronological order rather than its contents. Such arrangement of blogs makes it difficult for the user searching information about a particular topic from the blog. To resolve this problem, we propose an approach to cluster the blogs based on its content. We studied several clustering algorithms available. The objective of this report is to understand various steps involved in clustering blog information and working of clustering algorithms which best fits in text clustering and improving clustering results by reducing its drawbacks. The report demonstrates a comparison between the K-Means, Vector Space Model (VSM), Latent Semantic Indexing (LSI), and Fuzzy C-Means (FCM) clustering algorithms discussed through the paper and selects the optimum algorithm for blog clustering. The paper proposes modification of selected optimum algorithm to get required blog clustering. A comparison table of the selected algorithm and modified algorithm is included at the end of the paper and the results showed the proposed modified algorithm has performed overall better compared to other clustering algorithms.

Clustering Blog Information

1. Introduction

The definition of Blog provided by Wikipedia is,

> "Blogs are user generated websites where entries are made in journal style and displayed in reverse chronological order."

Blogs form a huge source of information on the internet today. The popularity of blogs is increasing. However, vast source of blog information is organized by blog's chronological order, and not content. This makes the search on a specific topic difficult. The difficulty can be easily resolved by clustering similar content blogs.

2. Blog Clustering:

Clustering of blog consists of three steps, with the output of each step being the input to next step. The first step in clustering is, obtaining the data, which is to be clustered, second is converting the obtained data into the format such that data can be used by clustering algorithms, third and last step, implementing clustering algorithms. Each step within itself would have several sub-steps. The brief overview of flow of data from the blogs to the clusters is shown in the figure 1.

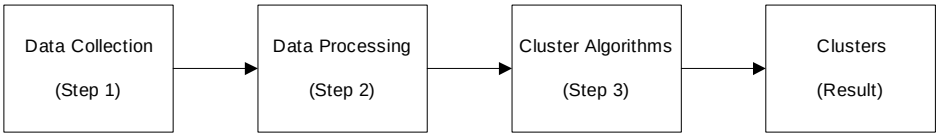


Figure 1 Block Diagram of Blog Clustering

The three steps in Blog Clustering are:

1. Data Collection
2. Data Processing
3. Implementation of Clustering Algorithm

The following section gives a brief description of sub-steps involved within each step. At the end of the description, results are provided to illustrate the working principle of each step.

a. Data Collection

Clustering is performed on a set of data. The data can be online or offline record. In this project, we are performing blog clustering. Blog clustering is the process of clustering the blogs. Blog clustering is of two types:

a. Clustering of blogs with respect to the responses (comments) on a particular topic
b. Clustering of blogs with respect to the topic
c. Clustering blog information

The first method is applicable to websites that have blogs in response to the post (title). In this type of clustering blogs with same response are clustered together. Thus, a cluster contains blogs that have same response to the tile. The advantage of such clustering is that the user can get analytical view of the people's response to a particular topic. Example: Title is "San Jose State University". Blog clustering based on people's response would give a portrait of San Jose State University.

The second method is applicable to websites that have open discussion blogs. Open discussion blogs are blogs entered irrespective to the other blogs. Example a blog may be about weather conditions in a particular state and the remaining blogs talk about their interest in weather, interest in food, interest in places and so on. Surfing through such website is difficult since the user has to go through each blog even though it is not of his/her interest. The solution is to have blogs clustered according to the topic. As a result, the user saves time and goes to the required cluster.

Several bloggers include additional information in their blogs on the same topic. Thus, under a topic each blogger might have added new information or repeated the information added by previous bloggers. On such blog websites, clustering blog information is more useful for users. Example, I was exploring the blog website, where a blogger started new discussion on food. Under this topic, few bloggers included recipes, some included calories content on the food, some included special regional food styles etc. Having clustered blog information in such topic is very useful to the user. The user wants particular information; he/she can scan through the first document of cluster and get the idea what the cluster is about. If cluster contains the required information user need not scan through all the blogs included in the topic. Similarly, user can skip reading blogs in the cluster that does not include required information. Thus, clustering blog information becomes very useful in extracting information within blogs.

The online blogs have hidden HTML tags. These tags carry no information for clustering. Similarly, few blogs have images related to the blogs. These images do carry information but in our project, we concentrate only on the text content of the blogs. In addition to HTML tags and images, few blogs have control buttons also. These controls also carry very little information. Hence, the first step in data collection is filtering of unwanted data and retrieving only the required data. This filtering process eliminates HTML tags, images and controls and retrieves only the textual content of the blogs. Each response is stored in a separate file. The result of this filtering process is used as data for the succeeding steps.

b. Data Processing

The result from Step a, is in text format only. Even though there is only text, the stored result has some data that has less or no importance in clustering process. Example punctuations are integral part of data but they carry almost zero information for clustering.

i. Stop words

Words such as “the”, “on”, “is”, “of” etc are redundantly present in the document. However, these words do not convey the true meaning of the document. Such words are defined stop words. Hence, in order to achieve fast and efficient clustering, removal of non-informative data is mandatory. Consider the documents:

Q: iron diamond truck
D1: Shipment of diamond damaged in volcano
D2: Delivery of iron arrived in iron truck
D3: Shipment of diamond arrived in a truck

After removing the stop words the resultant documents are:

Q: iron diamond truck
D1: Shipment diamond damaged volcano
D2: Delivery iron arrived iron truck
D3: Shipment diamond arrived truck

ii. Stemming

Many words are derived from other words, which can be termed as root words. Example ‘fruits’ and ‘fruit’ have root word “fruit”, ‘worked’, ‘working’, and ‘work’ have root word “work”, ‘fisher’, ‘fishing’, and ‘fish’ have root word “fish”. These words are derivative of same word and hence we can replace these derivates with the root word and then perform clustering. Initially it seems to be loss of data but when considered the documents, there is no loss of information. Several documents contain the derivative word, but not the root word. Such documents are not considered as similar, but after reducing the derivative word to the root word, the documents are similar. Thus, stemming plays an important role in clustering.

The pure textual data retrieved from step one is converted to data that has no punctuation, no stop words and all the derived words are replaced by their root words.

Thus, from section 2.b.i we have documents with no stop words. These documents are then passed through the stemmer. The output of the stemmer is a dictionary, which consists of all the words in the document after the stemming is implemented on each document. For the above documents, the output of stemming is shown below:

Q: iron diamond truck
D1: Shipment diamond damag volcano
D2: Deliveri iron arriv iron truck
D3: Shipment diamond arriv truck

Because of the above processing, we get the terms with significant meaning in the documents. However, we have no information of how important the term is in the particular document. This information is conveyed by assigning weights to the terms in the document.

iii. Weighing Schemes

Various weight assigning methods are available. The following section gives a brief description of several weighing schemes:

a) Binary Value: This is the simplest scheme. In this the weight $w_{ij} = 1$ if the keyword occurs in the document and $w_{ij} = 0$ if the keyword does not occur in the document. This scheme has less significance since it gives information only about the occurrence of the keyword and not about the degree of relevance of the keyword to the document.

b) Term Frequency: In this scheme the weight for the keyword is assigned as:

$$w_{ij} = tf_{ij}$$

where tf_{ij} is the term frequency for the i^{th} keyword in j^{th} document. Term frequency is the number of times a given keyword occurs in the document and is given as:

$$t_{fi} = t_i / N_t \qquad 0 \leq t_{fi} \leq 1$$

where t_i = occurrence of term in the document

N_t = total number of terms in the document

And

Thus, Term frequency does have some information about the relevance of keyword but it still has the shortcoming. Term frequency determines the relevance of occurrence of a keyword in a particular document but does not give information about the relevance of the keyword in the collection of documents.

The following table shows the tf calculation for the document and query considered in above sections:

Q: iron diamond truck
D1: Shipment diamond damag volcano
D2: Deliveri iron arriv iron truck
D3: Shipment diamond arriv truck

Terms	Query	tfq	D1	tf1	D2	tf2	D3	tf3
	3.00			4.00		5.00		4.00
arriv	0.00	0.00	0.00	0.00	1.00	0.20	1.00	0.25
damag	0.00	0.00	1.00	0.25	0.00	0.00	0.00	0.25
deliveri	0.00	0.00	0.00	0.00	1.00	0.20	0.00	0.25
diamond	1.00	0.33	1.00	0.25	0.00	0.00	1.00	0.25
iron	1.00	0.33	0.00	0.00	2.00	0.40	0.00	0.25
Shipment	0.00	0.00	1.00	0.25	0.00	0.00	1.00	0.25
truck	1.00	0.33	0.00	0.00	1.00	0.20	1.00	0.25

volcano	0.00	0.00	1.00	0.25	0.00	0.00	0.00	0.25

Figure 2 : TF calculation

c) Inverse Document Frequency (IDF): IDF is defined as the logarithmic ratio of the total number of documents in the collection to the number of documents in which the keyword occurs. IDF indicates that as the number of documents in which the keyword occur increases, the significance (relevance) of the keyword in the collection decreases. Hence the name "Inverse Document Frequency".

$$Idf = \log (N/d_i) \qquad 0 \leq Idf_i \leq \log N$$

where $\log (N/d_i)$ is the inverse document frequency.

N = total number of documents in the collection

d_i = number of documents in which the term occurs.

Terms	Query	tfq	D1	tf1	D2	tf2	D3	tf3	D/di	IDF
	3.00			4.00		5.00		4.00	3.00	
arriv	0.00	0.00	0.00	0.00	1.00	0.20	1.00	0.25	1.50	0.18
damag	0.00	0.00	1.00	0.25	0.00	0.00	0.00	0.25	3.00	0.48
deliveri	0.00	0.00	0.00	0.00	1.00	0.20	0.00	0.25	3.00	0.48
diamond	1.00	0.33	1.00	0.25	0.00	0.00	1.00	0.25	1.50	0.18
iron	1.00	0.33	0.00	0.00	2.00	0.40	0.00	0.25	1.50	0.18
Shipment	0.00	0.00	1.00	0.25	0.00	0.00	1.00	0.25	1.50	0.18
truck	1.00	0.33	0.00	0.00	1.00	0.20	1.00	0.25	1.50	0.18
volcano	0.00	0.00	1.00	0.25	0.00	0.00	0.00	0.25	3.00	0.48

Figure 3: IDF Calculation

d) TF-IDF (Term Frequency Inverse Document Frequency): This scheme overcomes the shortcoming of TF and IDF. It gives information about the relevance of the term in a particular document as well as in collection of documents.

The TFIDF weight is defined as:

$$w_{ij} = tf_{ij} \cdot \log (N/ d_i) \qquad 0 \leq TFIDF_i \leq \log N$$

Hence if the keyword occurs in all the documents then $d_i = N$ and IDF = 0, indicating the keyword has no significance.

Terms	Query	tfq	D1	tf1	D2	tf2	D3	tf3	D/di	IDF	Q	D1	D2	D3
	3.00			4.00		5.00		4.00	3.00		Weight = TF*IDF			
arriv	0.00	0.00	0.00	0.00	1.00	0.20	1.00	0.25	1.50	0.18	0.00	0.00	0.04	0.04
damag	0.00	0.00	1.00	0.25	0.00	0.00	0.00	0.25	3.00	0.48	0.00	0.12	0.00	0.12
deliveri	0.00	0.00	0.00	0.00	1.00	0.20	0.00	0.25	3.00	0.48	0.00	0.00	0.10	0.12
diamond	1.00	0.33	1.00	0.25	0.00	0.00	1.00	0.25	1.50	0.18	0.06	0.04	0.00	0.04
iron	1.00	0.33	0.00	0.00	2.00	0.40	0.00	0.25	1.50	0.18	0.06	0.00	0.07	0.04
Shipment	0.00	0.00	1.00	0.25	0.00	0.00	1.00	0.25	1.50	0.18	0.00	0.04	0.00	0.04
truck	1.00	0.33	0.00	0.00	1.00	0.20	1.00	0.25	1.50	0.18	0.06	0.00	0.04	0.04
volcano	0.00	0.00	1.00	0.25	0.00	0.00	0.00	0.25	3.00	0.48	0.00	0.12	0.00	0.12

Figure 4: TFIDF Calculation

Thus, data processing consists of applying parsing (removal of punctuation), removing stop words, stemming and assigning weights to the terms.

The result of data processing consists of files that are list of documents and forms a dictionary of the words in the documents.

The next section discusses various clustering algorithms. The section gives a detailed explanation about the working of the algorithm and shows the implementation of each algorithm on group of seven documents.
All algorithms use TFIDF as the weight-assigning scheme. The seven documents used for explaining the clustering algorithms in the subsequent sections and the corresponding TFIDF values are shown below:

Documents:

D1: Large Singular Value computations
D2: Software Library for the Space Singular Value Decomposition
D3: Introduction to Modern Information Retrieval
D4: Using Linear Algebra for Intelligent Information Retrieval
D5: Matrix Computations
D6: Singular Value Analysis of Cryptograms
D7: Automatic Information Organization

Query Vector: Singular Value

The term document matrix with terms in the row and documents in the column with each cell indicating the TFIDF value is shown in figure 2.

Terms	D1	D2	D3	D4	D5	D6	D7
algebra	0.0000	0.0000	0.0000	1.4788	0.0000	0.0000	0.0000
analysi	0.0000	0.0000	0.0000	0.0000	0.0000	1.8200	0.0000
automat	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	1.8972
comput	0.6962	0.0000	0.0000	0.0000	0.9783	0.0000	0.0000
cryptogram	0.0000	0.0000	0.0000	0.0000	0.0000	1.8200	0.0000
deomposit	0.0000	1.3415	0.0000	0.0000	0.0000	0.0000	0.0000
inform	0.0000	0.0000	0.3298	0.2804	0.0000	0.0000	0.3597
intellig	0.0000	0.0000	0.0000	1.4788	0.0000	0.0000	0.0000
introoduct	0.0000	0.0000	1.7397	0.0000	0.0000	0.0000	0.0000
larg	1.6796	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
librari	0.0000	1.3415	0.0000	0.0000	0.0000	0.0000	0.0000
linear	0.0000	0.0000	0.0000	1.4788	0.0000	0.0000	0.0000
matrix	0.0000	0.0000	0.0000	0.0000	2.3605	0.0000	0.0000
modern	0.0000	0.0000	1.7397	0.0000	0.0000	0.0000	0.0000
organ	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	1.8972
retriev	0.0000	0.0000	0.7210	0.6129	0.0000	0.0000	0.0000
scale	1.6796	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000

singular	0.3184	0.2543	0.0000	0.0000	0.0000	0.3451	0.0000
softwar	0.0000	1.3415	0.0000	0.0000	0.0000	0.0000	0.0000
spars	0.0000	1.3415	0.0000	0.0000	0.0000	0.0000	0.0000
valu	0.3184	0.2543	0.0000	0.0000	0.0000	0.3451	0.0000

Figure 5.Term Document Matrix Indicating TFIDF value

3. Clustering Algorithms

There are many clustering algorithms available today. In this section, we discuss and implement clustering algorithms.

I. K-means Algorithm

K means algorithm partitions the given set of data into k different sets, where a unique centroid (mean) characterizes each set. The elements in the set are close to the centroid of the set to which they belong and have large variance to the other sets.

K-means algorithm consists of the following steps:

1. Take the data space, which needs to be clustered
2. Pre-determine the number of clusters as k
3. Initialize k means for the data, viz, m1,m2, m3, ……, mk
4. Determine the distance of each data point from the mean using Euclidean distance
5. Group the data points having minimum distance to the mean, to the corresponding mean.
6. Calculate the new mean of the each group formed in Step 5
7. Repeat Steps 4 – 6 until the new mean formed is same as the previous mean.

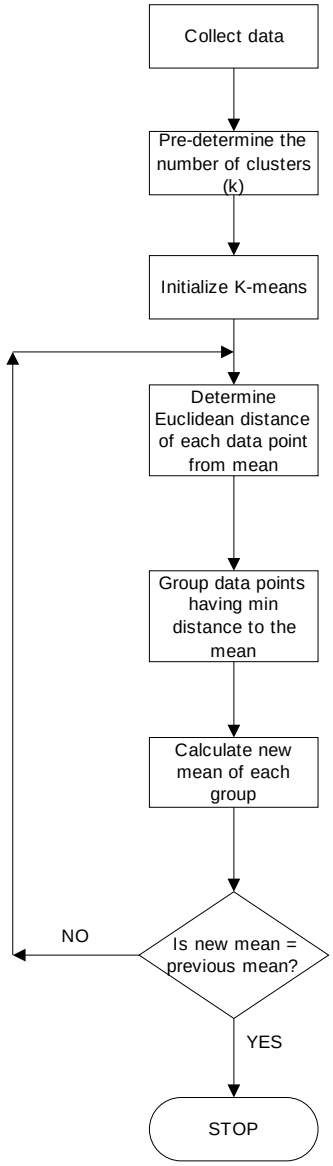


Figure 6. K means Algorithm

a. Generalization of K-means algorithm:

Consider the following 2x4 matrix where the row indicates the number of attributes to determine the objects. Thus, there are four objects and each object is determined by two attributes.

Step 1: Data Space to be clustered:

A	B	C	D
0	2.0	0.1	1.0
0.5	6.0	0.8	0.05

Table 1.K means Data

Step 2: Number of clusters = 2. For k-means calculation demo, we consider value of k as the two. The range of k is greater than two and equal to the number of documents. The value of k cannot be greater than the total number of documents.

Step 3: Initialize two means for the data:
m1= {0, 0.5} and m2= {2.0, 0.6}

Step 4: Determine the distance of each data point from the mean using Euclidean distance.
Euclidean Distance:
The definition of Euclidean distance is:
The distance between points x and y in Euclidean space is:

$$d=|x-y| = (\Sigma |xi - yi|^2)^{1/2} \text{ for } i= 1 \text{ to } n;$$

Distance of Point C from two means:

m1= {0, 0.5} is $d_{c,m1} = ((0.1-0)^2 + (0.8-0.5)^2)^{1/2}$
m2= {2.0, 0.6} is $d_{c,m2} = ((0.1-2)^2 + (0.8-0.6)^2)^{1/2}$

The distance table is:

A	B	C	D
0	2.0	0.1	1.09
2	0	1.09	1.14

Table 2. Distance Table

Step 5: Group data points to the mean with minimum distance to the mean. Thus, A, C and D form one cluster and B forms other.

A	B	C	D
1	0	1	1
0	1	0	0

Table 3 . Intermediate Data Cluster using K-mean

1: Belongs to the cluster
0: Does not belong to the cluster

Step 6: Calculate new mean of each cluster. New mean for cluster one would be:

$m_{11} = (A + C + D) / 3$
$= \{(+0.1+1.0)3, (0.5 + 0.8 + 0.05)/ 3\}$
$m_{11}= (0.366, 0.45)$

Similarly $m_{21} = (2, 0.6)$

Step 7: Repeat step 4, calculating distance, of each data point from the new mean calculated in step 6. Similarly repeat step 5, step 6, and calculate new mean m_{12} and m_{22}. Check whether $m_{12} = m_{11}$ and $m_{22} = m_{21}$. If equal stop the iteration and clusters formed in step 6 are the final clusters. If not equal repeat steps, 4-6 until the new mean calculated is equal to the previous mean.

For the above example the new distance table for new mean (m_{11}, m_{21}) is:

A	B	C	D
0.37	1.67	0.44	0.74
2	0	1.9	1.14

Table 4 . New Distance Calculation

Grouping objects with the lowest distance from the mean.

A	B	C	D
1	0	1	1
0	1	0	0

Table 5. Final Data cluster using K means

Calculating the new mean from the above cluster, we get

m_{12} = (0.366, 0.45) and m_{22}= (2, 0.6)

Since $m_{11}=m_{21}$ and $m_{21}=m_{22}$, stop the iteration and above cluster is the final cluster.

b. Implementation of K-means algorithm:

Consider the documents and the corresponding TFIDF values given in Figure 5.
And Number of cluster (k) = 2
The result is:

D1	D2	D3	D4	D5	D6	D7
1	1	2	2	1	1	1

Figure 7. Cluster Index

The cluster index matrix displays result in the ascending order of documents, (D1, D2,…, D7) and the value of each cell indicates the cluster to which the corresponding document belongs. Thus from the above matrix the first cell is for document #1 and its value is 1 i.e. Document D1 belongs to the cluster #1.

Cluster 1	Cluster 2
D1	D3
D2	D4
D5	
D6	
D7	

Figure 8. Cluster using K-means

One of the important advantages of K-means algorithm is that it gives the output-clustered data very fast. However, the cluster data depends on how well defined initial means is. Usually first k columns are the means in the first iteration. As a result if the columns are interchanged the cluster result changes. In addition, the result is not optimized.
For every iteration, k-means algorithm scans through the entire database, which makes the algorithm cumbersome for huge database.

II. VSM Method

In vector space model, each document is defined as a multidimensional vector of keywords in Euclidean space whose axis correspond to the keyword. For a document, the keywords are extracted from the document and weight associated with each keyword determines the importance of the keyword in the document and the collection of the document. Thus, a document is represented as,

$$d_i = \{wi1, w_{i2}, w_{i3}, \ldots\ldots, w_{ij}\}$$

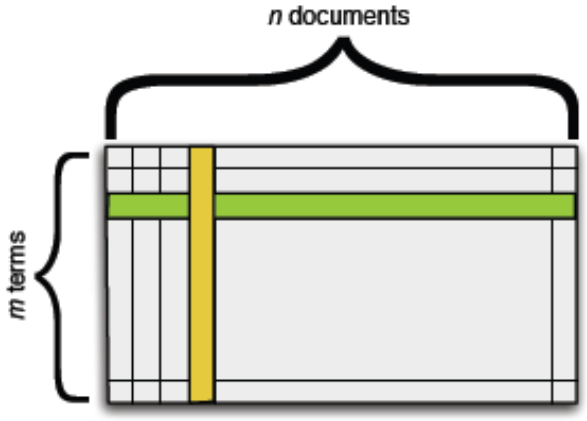
where wij = weight of term i in document j.

Similarly, query is represented as the vector consisting of the weight defined to the keywords.

The coordinate of document d in the direction of the keyword is determined by the weights associated with the keyword. Assigning weight to the keyword is the process of calculating the degree of relevance of the keyword to the document. Weight w_{ij} indicates the weight of the term i in the document j. Different weight assigning schemes are discussed in section 2.b.*iii*.

a. Vector Space Model:

In vector space model, the collection of n documents and m keywords is represented by n x m matrix and each cell indicates the weight associated with the keyword.



The column vector represents the document vectors, which includes all the documents in the collection. The row vectors modeled as the keyword vector, which defines the document vector. The query vector is added as a document vector whose keyword vector is the keywords. Thus, the model treats the query vector like the document vector.

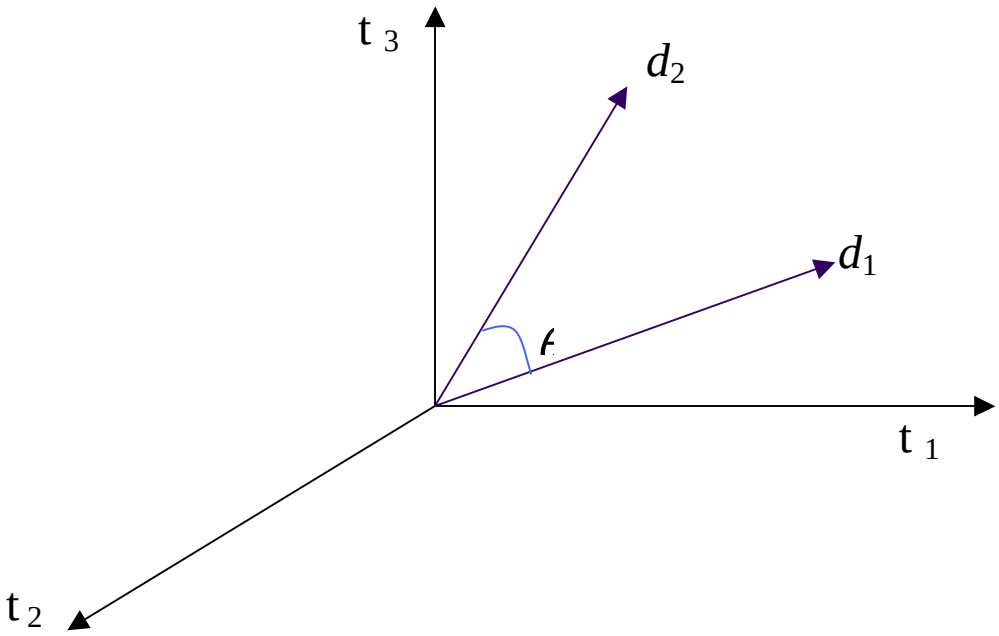


Figure 10. Vector Representation of Documents

Similarly, the query is represented as the vector in the vector space model. The angle θ indicates the similarity between the document and the query. This angle gives the measure of how close a document is to the query. More close the document to the query is, more similar the document to the query. Index of the document depends on the closeness with the query vector. Several algorithms are available to determine the closeness (similarity). One such algorithm is Cosine Similarity measurement Algorithm. The following sections give a brief description of the cosine similarity and then elaborate the working of VSM using cosine similarity to determine the closeness of the document to the query.

b. Vector Space Model Using Cosine Similarity Measurement:

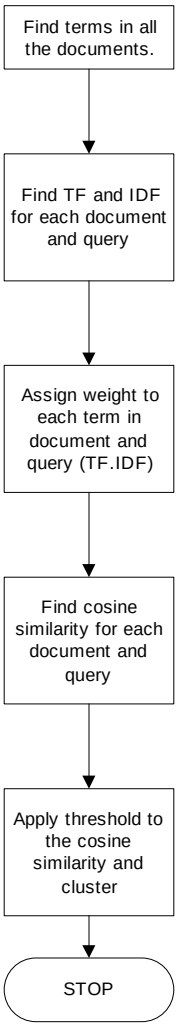


Figure 5. VSM Flowchart

The angle between vector representation of the document vector and the query vector determines the similarity between the document and the query. Thus:

- If angle = 0; document and query similar
- If angle = 90; no similarity;
- D2 more similar to query than D1 if the angle between D2 and query is smaller than the angel between D1 and query

One of the techniques to measure the similarity between the document and the query is cosine measure.

Cosine Measure:

$$\text{Cosine } \theta = \text{Sim } (Q, D) = (\sum w_{Q,j}\, w_{i,j}) / (\sqrt{\sum} w^2_{Q,j}\, \sqrt{\sum} w^2_{i,j})$$

Where

$\sqrt{\sum} w^2_{Q,j}$ = Vector Length of query vector

$\sqrt{\sum} w^2_{i,j}$ = Vector Length of Document vector

c. Generalization of Vector Space Model

Q: iron diamond truck
D1: Shipment of diamond damaged in volcano
D2: Delivery of iron arrived in iron truck
D3: Shipment of diamond arrived in a truck

Consider the following table indicating the calculation for Term Frequency, Inverse Document Frequency and weight to the terms in the document and query.

	Count										Weight = TF * IDF			
Terms	Querry	tfq	D1	tf1	D2	tf2	D3	tf3	D/di	IDF	Q	D1	D2	D3
	3.00			6.00		7.00		7.00	3.00					
a	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.14	3.00	0.48	0.00	0.00	0.00	0.07
arrived	0.00	0.00	0.00	0.00	1.00	0.14	1.00	0.14	1.50	0.18	0.00	0.00	0.03	0.03
damaged	0.00	0.00	1.00	0.17	0.00	0.00	0.00	0.00	3.00	0.48	0.00	0.08	0.00	0.00
delivery	0.00	0.00	0.00	0.00	1.00	7.00	0.00	0.00	3.00	0.48	0.00	0.00	3.34	0.00
diamond	1.00	0.33	1.00	0.17	0.00	0.00	1.00	0.14	1.50	0.18	0.06	0.03	0.00	0.03
in	0.00	0.00	1.00	0.17	1.00	0.14	1.00	0.14	1.00	0.00	0.00	0.00	0.00	0.00
iron	1.00	0.33	0.00	0.00	1.00	0.14	0.00	0.00	3.00	0.48	0.16	0.00	0.07	0.00
of	0.00	0.00	1.00	0.17	1.00	0.14	1.00	0.14	1.00	0.00	0.00	0.00	0.00	0.00
Shipment	0.00	0.00	1.00	0.17	0.00	0.00	1.00	0.14	1.50	0.18	0.00	0.03	0.00	0.03
truck	1.00	0.33	0.00	0.00	1.00	0.14	1.00	0.14	1.50	0.18	0.06	0.00	0.03	0.03
volcano	0.00	0.00	1.00	0.17	0.00	0.00	0.00	0.00	3.00	0.48	0.00	0.08	0.00	0.00

Figure 6. TFIDF Calculation in VSM

Calculate Similarity Value for each document. Cosine similarity is used.

$$\text{Cosine } \theta = \text{Sim}(Q, D) = (\sum w_{Q,j}\, w_{i,j}) / (\sqrt{\sum} w^2_{Q,j} \sqrt{\sum} w^2_{i,j})$$

Where
$\sqrt{\sum} w^2_{Q,j}$ = Vector Length of query vector

$\sqrt{\sum} w^2_{i,j}$ = Vector Length of Document vector

The following table depicts the calculation steps of the cosine similarity of the documents:

Vector Length									
Q	D1	D2	D3	wq,j*wd1,j	wq,j*wd2,j	wq,j*wd3,j	cosine Similarity		
							cos D1	cos D2	cos D3
0.18	0.12	3.34	0.08	0.00000	0.00000	0.00000	0.08010	0.02055	0.19431
				0.00000	0.00000	0.00000			
				0.00000	0.00000	0.00000			
				0.00000	0.00000	0.00000			
				0.00172	0.00000	0.00148			
				0.00000	0.00000	0.00000			
				0.00000	0.01084	0.00000			
				0.00000	0.00000	0.00000			
				0.00000	0.00000	0.00000			
				0.00000	0.00148	0.00148			
				0.00000	0.00000	0.00000			
				0.00172	0.01232	0.00295			

Figure 7. Cosine Similarity Calculation

Clusters are formed by giving a threshold for the cosine similarity. Example all documents with cosine similarity of 0.5 and greater are clustered together whereas documents with cosine similarity from 0 to 0.5 are in second cluster.

Thus from the above calculations:

Cluster 1: D3
Cluster 2: D1, D2

d. Implementation of VSM:

The VSM implementation for the documents and query using the TFIDF values mentioned in Figure 5, we get the following cosine similarity matrix:

D1	D2	D3	D4	D5	D6	D7
0.2254	0.1329	0.0000	0.0000	0.0000	0.1863	0.0000

Figure 8. Cosine Similarity Matrix

Thus, the cosine similarity matrix gives the similarity of the documents to the query. The output has only the similarity measurement and we do not see any cluster formation. Thus, in order to form a cluster we need to apply threshold to the above similarity matrix. In this example, we consider documents with non-zero similarity in one cluster and documents with zero similarity measure in other.
Thus,

Cluster 1	Cluster 2
D1: Large **Singular Value** computations	D3: Introduction to Modern Information Retrieval
D2: Software Library for the Space **Singular Value** Decomposition	D4: Using Linear Algebra for Intelligent Information Retrieval
D6: **Singular Value** Analysis of Cryptograms	D5: Matrix Computations
	D7: Automatic Information Organization

Figure 9. Cluster using VSM

e. Disadvantages of VSM:

I. VSM becomes cumbersome for huge dataset. Huge database result in multidimensional space and similarity calculation becomes difficult.
II. It fails to show similarity for documents with same content but different vocabulary.

III. Latent Semantic Indexing

As discussed in section 3.II.e, VSM fails to cluster documents that have different terms but are talking about the same concept. This forms a vital disadvantage for VSM. In search, the query entered by the user is usually different and contains different words from user to user. In such situation, the clustering engine should be able to cluster data based on additional parameters other than mere presence or absence of terms. Latent Semantic Indexing provides a method to provide such clustering. It introduces additional parameter called as concept.

According to LSI, if a term is related to a concept and the document is related to the same concept then the terms and documents are related to each other even if the terms are not present in the document.

Thus, LSI overcomes the deficiencies of VSM model by implementing "concepts" in the clustering process.

LSI tries to overcome the problems of lexical matching by using statistically derived conceptual indices instead of individual words for retrieval. LSI assumes that there is some underlying or latent structure in word usage that is partially obscured by variability in word choice. A truncated Singular Value Decomposition (SVD) is used to estimate the structure in word usage across documents. Retrieval is then performed using a database of singular values and vectors obtained from the 27 truncated SVD. Performance data shows that these statistically derived vectors are more robust indicators of meaning than individual terms. [Berry et al., 95]

Singular Value Decomposition:

To extract the conceptual relation between terms and documents from term-document matrix Singular Value Decomposition is used. It decomposes the matrix into product of three terms.

The middle matrix is diagonal matrix and the other two matrices are orthogonal matrix. The theorems related to Singular Value Decomposition that defines how a matrix is decomposed into three different matrices. The theorem also states the properties of the three resulting matrices.

Theorem I:

Each real-valued mxn matrix A with rank r can be decomposed into the form
$A = U x \Delta x V_T$ with an
mxr matrix U with orthonormal column vectors, an
rxr diagonal matrix Δ , and
an nxr matrix V with orthonormal column vectors.
This decomposition is called singular value decomposition and is unique when the elements of Δ or sorted. [Feilong, 2007]

Theorem II:

In the singular value decomposition $A = U x \Delta x V^T$ of matrix A the matrices U, Δ , and V can be derived.
U: The columns of U are the Eigenvectors of $A_x A_T$.
Term-Topic matrix.
Δ: The positive roots of the Eigen values of A_T xA
Singular values matrix.
V^T: The columns of V are the Eigenvectors of A_T x A.
Topic-Document matrix. [Feilong, 2007]

Thus, we can write,

[term document matrix] = [term-topic matrix] $_{mxr}$ * [singular value matrix] $_{rxr}$ * [topic document matrix] $_{rxn}$

Thus, matrix A is decomposed into three matrices:

a. Implementation of Latent Semantic Indexing:

Consider the documents and query from Figure 5.
The SVD of the matrix A gives three matrices U, S, V as:

U=

-0.34	0.00	0.28	0.00	0.00	-0.35	0.00	-0.41	0.00	-0.40	0.00	-0.41	0.00	0.00	0.05	-0.17	-0.40	-0.08	0.00	0.00	-0.08
0.00	-0.24	0.00	0.11	0.63	0.00	0.10	0.11	0.00	-0.09	-0.18	0.11	0.56	0.00	-0.01	0.04	-0.09	-0.17	-0.18	-0.18	-0.17
-0.42	0.00	-0.56	0.00	0.00	-0.03	0.00	0.00	-0.41	0.00	0.00	0.00	0.00	-0.41	-0.39	-0.17	0.00	0.00	0.00	0.00	0.00
0.00	-0.17	0.00	0.39	-0.15	0.00	0.04	0.39	-0.01	-0.42	0.00	0.39	-0.34	-0.01	-0.04	0.16	-0.42	-0.08	0.00	0.00	-0.08
0.00	-0.24	0.00	0.11	0.63	0.00	0.10	-0.11	0.00	0.13	0.18	-0.11	-0.60	0.00	0.01	-0.05	0.13	-0.01	0.18	0.18	-0.01
0.00	-0.41	0.00	-0.24	-0.13	0.00	0.03	-0.07	0.00	0.07	-0.45	-0.07	-0.36	0.00	0.01	-0.03	0.07	0.00	-0.45	-0.45	0.00
-0.20	0.00	-0.01	0.00	0.00	0.04	0.00	-0.04	0.43	0.03	0.00	-0.04	0.00	0.43	-0.74	0.16	0.03	0.01	0.00	0.00	0.01
-0.34	0.00	0.28	0.00	0.00	-0.35	0.00	0.73	0.00	0.18	0.00	-0.27	0.00	0.00	0.04	-0.11	0.18	0.03	0.00	0.00	0.03
-0.31	0.00	0.22	0.00	0.00	0.56	0.00	0.03	0.50	-0.02	0.00	0.03	0.00	-0.50	0.06	-0.20	-0.02	0.00	0.00	0.00	0.00

0.00	-0.20	0.00	0.34	-0.07	0.00	-0.53	-0.08	0.00	0.60	0.00	-0.08	0.05	0.00	0.01	-0.03	-0.40	-0.07	0.00	0.00	-0.07
0.00	-0.41	0.00	-0.24	-0.13	0.00	0.03	0.02	0.00	-0.01	0.82	0.02	0.11	0.00	0.00	0.01	-0.01	-0.06	-0.18	-0.18	-0.06
-0.34	0.00	0.28	0.00	0.00	-0.35	0.00	-0.27	0.00	0.18	0.00	0.73	0.00	0.00	0.04	-0.11	0.18	0.03	0.00	0.00	0.03
0.00	-0.21	0.00	0.59	-0.28	0.00	0.62	-0.16	0.00	0.17	0.00	-0.16	0.14	0.00	0.02	-0.07	0.17	0.03	0.00	0.00	0.03
-0.31	0.00	0.22	0.00	0.00	0.56	0.00	0.03	-0.50	-0.02	0.00	0.03	0.00	0.50	0.06	-0.20	-0.02	0.00	0.00	0.00	0.00
-0.42	0.00	-0.56	0.00	0.00	-0.03	0.00	0.00	0.33	0.00	0.00	0.00	0.00	0.33	0.53	0.14	0.00	0.00	0.00	0.00	0.00
-0.27	0.00	0.21	0.00	0.00	0.09	0.00	-0.10	-0.21	0.07	0.00	-0.10	0.00	-0.21	0.04	0.87	0.07	0.01	0.00	0.00	0.01
0.00	-0.20	0.00	0.34	-0.07	0.00	-0.53	-0.08	0.00	-0.40	0.00	-0.08	0.05	0.00	0.01	-0.03	0.60	-0.07	0.00	0.00	-0.07
0.00	-0.16	0.00	0.04	0.08	0.00	-0.08	0.00	0.00	-0.09	0.00	0.00	0.09	0.00	0.00	0.00	-0.09	0.97	0.00	0.00	-0.03
0.00	-0.41	0.00	-0.24	-0.13	0.00	0.03	0.02	0.00	-0.01	-0.18	0.02	0.11	0.00	0.00	0.01	-0.01	-0.06	0.82	-0.18	-0.06
0.00	-0.41	0.00	-0.24	-0.13	0.00	0.03	0.02	0.00	-0.01	-0.18	0.02	0.11	0.00	0.00	0.01	-0.01	-0.06	-0.18	0.82	-0.06
0.00	-0.16	0.00	0.04	0.08	0.00	-0.08	0.00	0.00	-0.09	0.00	0.00	0.09	0.00	0.00	0.00	-0.09	-0.03	0.00	0.00	0.97

Table 6. U Matrix

S =

2.74319	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
0.00000	2.73295	0.00000	0.00000	0.00000	0.00000	0.00000
0.00000	0.00000	2.68555	0.00000	0.00000	0.00000	0.00000
0.00000	0.00000	0.00000	2.66185	0.00000	0.00000	0.00000
0.00000	0.00000	0.00000	0.00000	2.60483	0.00000	0.00000
0.00000	0.00000	0.00000	0.00000	0.00000	2.50742	0.00000
0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	2.38969
0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000

Table 7. S Matrix

V^T=

0.00	0.00	-0.48	-0.63	0.00	0.00	-0.61
-0.33	-0.84	0.00	0.00	-0.24	-0.36	0.00
0.00	0.00	0.34	0.51	0.00	0.00	-0.79
0.55	-0.48	0.00	0.00	0.67	0.16	0.00

-0.12	-0.26	0.00	0.00	-0.31	0.91	0.00
0.00	0.00	0.81	-0.59	0.00	0.00	-0.04
-0.76	0.06	0.00	0.00	0.63	0.14	0.00

Table 8. V Matrix

The first two highest singular values are considered. (k=2)

The corresponding matrices are:

U_k =

-0.34	0.00
0.00	-0.24
-0.42	0.00
0.00	-0.17
0.00	-0.24
0.00	-0.41
-0.20	0.00
-0.34	0.00
-0.31	0.00
0.00	-0.20
0.00	-0.41
-0.34	0.00
0.00	-0.21
-0.31	0.00
-0.42	0.00
-0.27	0.00
0.00	-0.20
0.00	-0.16
0.00	-0.41
0.00	-0.41
0.00	-0.16

S_k=

2.74	0.00
0.00	2.73

V^T_k=

0.00	0.00	-0.48	-0.63	0.00	0.00	-0.61
-0.33	-0.84	0.00	0.00	-0.24	-0.36	0.00

$A_k = U_k * S_k * V^T_k$

A_k

0.00	0.00	0.45	0.58	0.00	0.00	0.57
0.22	0.55	0.00	0.00	0.16	0.24	0.00
0.00	0.00	0.56	0.73	0.00	0.00	0.71
0.16	0.39	0.00	0.00	0.11	0.17	0.00
0.22	0.55	0.00	0.00	0.16	0.24	0.00
0.37	0.94	0.00	0.00	0.27	0.41	0.00
0.00	0.00	0.27	0.35	0.00	0.00	0.34
0.00	0.00	0.45	0.58	0.00	0.00	0.57
0.00	0.00	0.41	0.53	0.00	0.00	0.51
0.19	0.47	0.00	0.00	0.14	0.20	0.00
0.37	0.94	0.00	0.00	0.27	0.41	0.00
0.00	0.00	0.45	0.58	0.00	0.00	0.57
0.19	0.48	0.00	0.00	0.14	0.21	0.00
0.00	0.00	0.41	0.53	0.00	0.00	0.51
0.00	0.00	0.56	0.73	0.00	0.00	0.71
0.00	0.00	0.35	0.46	0.00	0.00	0.45
0.19	0.47	0.00	0.00	0.14	0.20	0.00
0.15	0.37	0.00	0.00	0.11	0.16	0.00
0.37	0.94	0.00	0.00	0.27	0.41	0.00
0.37	0.94	0.00	0.00	0.27	0.41	0.00
0.15	0.37	0.00	0.00	0.11	0.16	0.00

Query Vector q

q^T =

0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.22	0.00	0.00	1.22

Table 9. Query Vector

Query Document Similarity Vector:

0.36	0.91	0.00	0.00	0.26	0.39	0.00

Table 10. Query Document Similarity Matrix

The query document similarity matrix has the documents serially, i.e. D1, D2,…..,D7. Clusters are formed with the documents arranged in the descending score order. Thus, for the above example document D2 with score of 0.91 is the closest match to the query and then document D5, D1 and D6.

In this example, we consider Documents with non-zero value are in one cluster and documents with zero value are in one cluster.
The non-zero value is an infinitesimal small value of the order of -16 and hence for this example we consider it as zero. For cases where the value is non-zero, we apply a threshold to form the cluster and documents in each cluster are arranged in the descending order of the score.

Thus,

Cluster 1	Cluster 2
D2	D3
D6	D4
D1	D7
D5	

Figure 10 : Clusters from LSI

IV. Fuzzy-C Means Algorithm

All the blogs are clustered into groups on some similarity measure. Due to the vagueness in similarity between the blogs, it is inefficient to categorize a blog to a particular cluster. In order to group the blog in such a way that it belongs to more than one cluster we use Fuzzy clustering technique.

Fuzziness is defined by a membership function that gives the membership degree or the belongingness of the data to a particular cluster. The value of the membership degree ranges from [0, 1]. Fuzzy C Means (FCM) is a simple iterative Fuzzy based algorithm that returns the centers (centroid) of the clusters. It terminates on satisfying an objective function given by

$$J_m = \sum_{i=1}^{N} \sum_{j=1}^{C} u_{ij}^{m} \left\| x_i - c_j \right\|^2$$

Where m ϵ [1, ∞] - Fuzzy Coefficient
uij - membership degree of xi
w.r.t to cj; Range [0, 1]
cj - centroid(vector) of cluster j;
C - Number of Clusters
N - Number of data vectors
xi - Data vector

FCM starts with a random initial weight matrix or membership matrix U such that

- $u_{ij} \in [0,1] \quad \forall i, j$
- $\sum_{j=1}^{C} u_{ik} = 1 \quad \forall i$
- $0 < \sum_{i=1}^{N} u_{ij} < N \quad \forall N$

and, a fixed number of clusters. Number of columns and rows of the matrix U depends on the data and number of clusters. (rows – clusters & columns - membership degrees of the document). Clusters center vectors are updated with each iteration. The membership values are calculated w.r.t to the new centers. Belongingness of the data to the cluster is calculated using Euclidian distance between the center and data point.

Fuzzy C Means algorithm is composed of

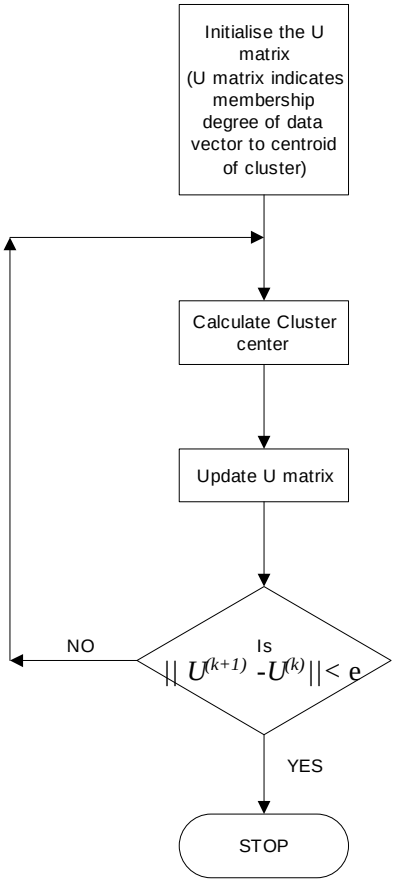


Figure 11. Fuzzy C-means Flowchart

1. Initialize the U matrix
   U0 = [uij]
   Where 0 – the iteration step
2. Calculate the cluster center vector

$$c_j = \frac{\sum_{i=1}^{N} u_{ij}^{m} \cdot x_i}{\sum_{i=1}^{N} u_{ij}^{m}}$$

   Where uij - membership degree of xi
   w.r.t to cluster cj
   cj - centroid(vector) of cluster j;
   N - Number of data vectors
   xi - Data vector
   m - Fuzzy Coefficient
3. Update the U^k matrix such that

$$u_{ij} = \frac{1}{\sum_{k=1}^{c}\left(\frac{\|x_i - c_j\|}{\|x_i - c_k\|}\right)^{\frac{2}{m-1}}}$$

Where u_{ij} - membership degree of x_i
w.r.t to cluster c_j
c_j - centroid(vector) of cluster j;
x_i - Data vector
m - Fuzzy Coefficient
C - Number of Clusters
K – the iteration step

4. If $|| U^{(k+1)} - U^{(k)}|| < \varepsilon$ then stop, else go to step 2

a. Generalization of Fuzzy C Means Algorithm for Blog clustering

Each blog (b) is represented as N- TFIDF value vector of relevant terms. The input $(X^T)^T$ to the FCM is a data set containing n- blogs.

$$b_i = \begin{bmatrix} t_{11} \\ t_{21} \\ M \\ t_{ji} \\ M \\ t_{N1} \end{bmatrix}$$

$$X^T = \begin{array}{c} \begin{matrix} b1 & b2 & & bn \end{matrix} \\ \begin{bmatrix} t_{11} & t_{12} & K & t_{1n} \\ t_{21} & t_{22} & \Lambda & t_{2n} \\ M & M & t_{ij} & M \\ t_{N1} & t_{N2} & K & t_{Nn} \end{bmatrix} \end{array}$$

where b_i - blog
N - Number of terms
t_{ij} - TFIDF value for the term w.r.t blog b_i
n - Number of Blogs

Let $C = \{C_1, C_2 \ldots C_p\}$ be set of cluster centers where C_i - Cluster center of Cluster i
P - total number of Clusters
n - total number of blogs

Step 1: Initialize the U_0 matrix

$$U_0 = \begin{bmatrix} u_{11} & u_{12} & \Lambda & u_{1p} \\ u_{21} & u_{22} & \Lambda & u_{2p} \\ M & M & u_{ij} & u_{ip} \\ u_{n1} & u_{n2} & u_{nj} & u_{np} \end{bmatrix}$$

Step 2: Find the cluster centers.

$$C_1 = \frac{u_{11}^m \begin{bmatrix} t_{11} \\ t_{21} \\ M \\ t_{N1} \end{bmatrix} + u_{21}^m \begin{bmatrix} t_{12} \\ t_{22} \\ M \\ t_{N2} \end{bmatrix} + \Lambda + u_{n1}^m \begin{bmatrix} t_{1p} \\ t_{2p} \\ M \\ t_{Np} \end{bmatrix}}{u_{11}^m + u_{21}^m + \Lambda + u_{n1}^m}$$

$$= \begin{bmatrix} c_{11} \\ c_{21} \\ M \\ c_{N1} \end{bmatrix}$$

$$C_2 = \frac{u_{12}^m \begin{bmatrix} t_{11} \\ t_{21} \\ M \\ t_{N1} \end{bmatrix} + u_{22}^m \begin{bmatrix} t_{12} \\ t_{22} \\ M \\ t_{N2} \end{bmatrix} + \Lambda + u_{n2}^m \begin{bmatrix} t_{1p} \\ t_{2p} \\ M \\ t_{Np} \end{bmatrix}}{u_{12}^m + u_{22}^m + \Lambda + u_{n2}^m}$$

$$= \begin{bmatrix} c_{12} \\ c_{22} \\ M \\ c_{2p} \end{bmatrix}$$

.
.
.

$$C_p = \frac{u_{1p}^m \begin{bmatrix} t_{11} \\ t_{21} \\ M \\ t_{N1} \end{bmatrix} + u_{2p}^m \begin{bmatrix} t_{12} \\ t_{22} \\ M \\ t_{N2} \end{bmatrix} + \Lambda + u_{np}^m \begin{bmatrix} t_{1p} \\ t_{2p} \\ M \\ t_{Np} \end{bmatrix}}{u_{1p}^m + u_{2p}^m + \Lambda + u_{np}^m}$$

$$= \begin{bmatrix} c_{1p} \\ c_{2p} \\ M \\ c_{Np} \end{bmatrix}$$

Step 3: Find the new U^1 matrix

$$u_{11}^1 = \frac{1}{\left(\frac{\left\| \begin{bmatrix} t_{11} \\ t_{21} \\ M \\ t_{N1} \end{bmatrix} - \begin{bmatrix} c_{11} \\ c_{21} \\ M \\ c_{N1} \end{bmatrix} \right\|}{\left\| \begin{bmatrix} t_{11} \\ t_{21} \\ M \\ t_{N1} \end{bmatrix} - \begin{bmatrix} c_{11} \\ c_{21} \\ M \\ c_{N1} \end{bmatrix} \right\|} \right)^{\frac{2}{m-1}} + \Lambda + \left(\frac{\left\| \begin{bmatrix} t_{11} \\ t_{21} \\ M \\ t_{N1} \end{bmatrix} - \begin{bmatrix} c_{11} \\ c_{21} \\ M \\ c_{N1} \end{bmatrix} \right\|}{\left\| \begin{bmatrix} t_{11} \\ t_{21} \\ M \\ t_{N1} \end{bmatrix} - \begin{bmatrix} c_{1p} \\ c_{2p} \\ M \\ c_{Np} \end{bmatrix} \right\|} \right)^{\frac{2}{m-1}}}$$

$$u_{12}^1 =$$

$$\frac{1}{\left(\frac{\left\| \begin{bmatrix} t_{11} \\ t_{21} \\ M \\ t_{N1} \end{bmatrix} - \begin{bmatrix} c_{11} \\ c_{21} \\ M \\ c_{N1} \end{bmatrix} \right\|}{\left\| \begin{bmatrix} t_{11} \\ t_{21} \\ M \\ t_{N1} \end{bmatrix} - \begin{bmatrix} c_{11} \\ c_{21} \\ M \\ c_{N1} \end{bmatrix} \right\|} \right)^{\frac{2}{m-1}} + \Lambda + \left(\frac{\left\| \begin{bmatrix} t_{11} \\ t_{21} \\ M \\ t_{N1} \end{bmatrix} - \begin{bmatrix} c_{11} \\ c_{21} \\ M \\ c_{N1} \end{bmatrix} \right\|}{\left\| \begin{bmatrix} t_{11} \\ t_{21} \\ M \\ t_{N1} \end{bmatrix} - \begin{bmatrix} c_{1p} \\ c_{2p} \\ M \\ c_{Np} \end{bmatrix} \right\|} \right)^{\frac{2}{m-1}}}$$

$$\vdots$$

$$u_{ij}^1 =$$

$$\frac{1}{\left(\frac{\left\|\begin{bmatrix} t_{ij} \\ t_{21} \\ \mathrm{M} \\ t_{N1} \end{bmatrix} - \begin{bmatrix} c_{11} \\ c_{21} \\ \mathrm{M} \\ c_{N1} \end{bmatrix}\right\|}{\left\|\begin{bmatrix} t_{11} \\ t_{21} \\ \mathrm{M} \\ t_{N1} \end{bmatrix} - \begin{bmatrix} c_{11} \\ c_{21} \\ \mathrm{M} \\ c_{N1} \end{bmatrix}\right\|}\right)^{\frac{2}{m-1}} + \Lambda + \left(\frac{\left\|\begin{bmatrix} t_{11} \\ t_{21} \\ \mathrm{M} \\ t_{N1} \end{bmatrix} - \begin{bmatrix} c_{11} \\ c_{21} \\ \mathrm{M} \\ c_{N1} \end{bmatrix}\right\|}{\left\|\begin{bmatrix} t_{11} \\ t_{21} \\ \mathrm{M} \\ t_{N1} \end{bmatrix} - \begin{bmatrix} c_{1p} \\ c_{2p} \\ \mathrm{M} \\ c_{Np} \end{bmatrix}\right\|}\right)^{\frac{2}{m-1}}}$$

$$\therefore U^1 = \begin{bmatrix} u_{11}^1 & u_{12}^1 & \Lambda & u_{1p}^1 \\ u_{21}^1 & u_{22}^1 & \mathrm{M} & u_{2p}^1 \\ \mathrm{M} & \mathrm{M} & u_{ij}^1 & \mathrm{M} \\ u_{n1}^1 & u_{n2}^1 & \Lambda & u_{np}^1 \end{bmatrix}$$

Step 4: check the condition $< \varepsilon$

i.e. $\left\|U^1 - U^0\right\| < \varepsilon$

Where ε – termination factor

if yes: STOP

else Go to Step 2.

b. Implementation of FCM Algorithm

FCM algorithm is implemented on the documents and the corresponding TFIDF values given in Figure 5 we get the results as:

The following table indicates the center points for two clusters:

Center1	Center2
0.1199	0.3109
0.2463	0.2628
0.2334	0.2534
0.3384	0.1634
0.2463	0.2628
0.252	0.1282
0.091	0.1802
0.1199	0.3109
0.1269	0.3863
0.2641	0.1641
0.252	0.1282

0.1199	0.3109
0.4126	0.2517
0.1269	0.3863
0.2334	0.2534
0.1023	0.289
0.4039	0.1427
0.1711	0.1012
0.252	0.1282
0.252	0.1282
0.1711	0.1012

Figure 12. Center Points for Two Clusters

Objective function is used to determine the stopping criteria. The following table indicates the objective function for each iteration. As seen from the results, the value of objective function decreases with iteration. As the difference between the previous and present objective function value decreases to the order of 1e-3 to 1e-5 the algorithm stops.

28.1055	27.4795	27.4501	27.4335	27.4222	27.4134	27.4062	27.3998	27.394	27.3887

Figure 13. Objective function

Membership function obtained:

Cluster	D1	D2	D3	D4	D5	D6	D7
Cluster 1	0.6765	0.6124	0.3224	0.342	0.6001	0.5013	0.4988
Cluster 2	0.3235	0.3876	0.6776	0.658	0.3999	0.4987	0.5012

Figure 14 : Membership function

Thus, from the above membership function value we say that document D1 is closer to cluster 1 than to cluster 2. Since, the membership function (1) > membership function (2) for document 1. Thus larger the value of membership function closer the document to the cluster is. Hence, by membership function we can say to which cluster a document belongs. In case of binary value, we say the documents belong to the cluster or not, whereas by looking at the membership function we say how close the document to the cluster is, in other words the degree of membership of the document to the cluster.

Thus, the clusters from FCM are:

Cluster 1	Cluster 2
D1	D 3
D 2	D 4
D 5	D 7
D 6	

Figure 15. Clusters from FCM

4. Comparison of Clustering Algorithm

In previous sections, we presented four different clustering techniques. Each clustering technique is implemented on the same set of seven documents with the corresponding

TFIDF value. However, the results obtained from each algorithm are different. In this section, we discuss pros and cons of each method and propose the optimum method.

The clustering algorithm VSM discussed in section 3.2 is actually a method to transform the documents and query into a method. Additional cosine similarity method is involved on this transformation to obtain the similarity between the documents. Thus, we can say that we do not get clusters from VSM but we get the similarity measure of the documents to the query. In order to find the clusters we impose an additional threshold to determine which documents are in which cluster.

Similarly, LSI model discussed in section 3.3 is the indexing method. The LSI method gives the score of the documents. Score gives the measurement of how close the document is to the query. Thus, higher the score closer the document to the query is. The output of LSI also does not give cluster and we need to impose additional threshold to find the cluster.

Both the methods discussed above have the disadvantage that the clustering depends on the threshold value. As the threshold, value changes the clustering changes. And maintaining the same threshold value would not give accurate results with the changing database. For example, a threshold value would give accurate results for a smaller database however; the same threshold value does not give accurate result when more documents are added to the database.

On the other hand, K-means and FCM algorithm there is no need of additional threshold to get the cluster. The output of the algorithm is the cluster. This section depicts the comparison of the clusters obtained using K-means and FCM.

Clustering Algorithm	No. of Clusters								
	#2		#3			#4			
	Cluster 1	Cluster 2	Cluster 1	Cluster 2	Cluster 3	Cluster 1	Cluster 2	Cluster 3	Cluster 4
FCM	D1,D2,D5,D6	D3,D4,D7	D3,D4,D7	D1,D5	D2,D6	D1,D2,D6	D3,D4	D5	D7
K-means	D1,D2,D5,D6,D7	D3,D4	D2,D5,D6,D7	D1	D3,D4	D6	D1,D5,D7	D3,D4	D2

Figure 16. Comparison of FCM and K-means for different clusters

D1: Large Singular Value computations
D2: Software Library for the Space Singular Value Decomposition
D3: Introduction to Modern Information Retrieval
D4: Using Linear Algebra for Intelligent Information Retrieval
D5: Matrix Computations
D6: Singular Value Analysis of Cryptograms
D7: Automatic Information Organization

k-means clustering algorithm takes first k columns as mean. Thus when the number of clusters is 2 it takes first two columns of TFIDF as the mean. Hence, in first cluster document D7 is present. As can be seen from above documents there is no correlation

between document D7 and D1, D2, D5 and D6. For the same number of clusters we get accurate results from FCM. D1, D2, D5 and D6 are correlated.

Thus from above results it is evident that for K-means the mean changes with the number of clusters and the documents in a cluster are not correlated to each other, whereas for FCM the documents in the cluster are strongly correlated.

5. FCM Shortcomings:

Documents in the FCM clusters are strongly correlated; however FCM clusters are sensitive to the initialization of membership matrix and center. Sensitivity of algorithm to initialization results in different cluster with single execution. The following table depicts the change in clusters with every execution. The input data is same as in the example considered above:

Run #	Clusters	Document	Objective Function
1	1 2	D1,D2,D5,D6 D3,D4,D7	20.3186
2	1 2	D1,D2,D5,D7 D3,D4,D6,D7	20.3186
3	1 2	D1,D5 D2,D3,D4,D6,D7	20.3186
4	1 2	D1,D2,D3,D4,D6,D7 D5	20.3186
5	1 2	D1,D2,D5,D6,D7 D3,D4	20.3186

Table 11: FCM Clusters Comparison for each run (TFIDF) Data

The FCM clusters for the butterfly data are shown in the table below:

Run #	Clusters	Document	Objective Function
1	1 2	D8,D9,D10,D11,D12,D13,D14,D15 D1,D2,D3,D4,D5,D6,D7,D8	26.32
2	1 2	D8,D9,D10,D11,D12,D13,D14,D15 D1,D2,D3,D4,D5,D6,D7,D8	26.32
3	1 2	D8,D9,D10,D11,D12,D13,D14,D15 D1,D2,D3,D4,D5,D6,D7,D8	26.32
4	1 2	D1,D2,D3,D4,D5,D6,D7,D8 D8,D9,D10,D11,D12,D13,D14,D15	26.32
5	1 2	D8,D9,D10,D11,D12,D13,D14,D15 D1,D2,D3,D4,D5,D6,D7,D8	26.32

Table 12: FCM Clusters Comparison for each run (Butterfly data)

Referring table 11, we see that the documents in the cluster change, i.e documents forming a cluster are different with every run. As long as same documents form a cluster, which cluster they form (1 or 2) does not matter. As observed in table 12, Documents 1,2,3,4,5,6,7,8 always form one cluster , either cluster #1 or cluster #2. This is not observed in the Table 11. Documents forming cluster change with ever run. Thus, we can see that the cluster in Table 11, is sensitive to the initialization of membership matrix.

However, the clusters in Table 12 are independent of the initialization of the membership matrix and at every run we get the same result.

FCM gives the cluster depending on the cluster size given by the user. As the cluster size changes the documents belonging to the cluster changes. Thus, clustering of document depends on the optimum number of clusters.

6. FCM Data and Cluster Analysis

Tables 11 and 12 included in section 5 are examples of small data analysis. Large data analysis was performed in understanding the changing behavior of the FCM clusters. Genetic Algorithm (GA) was performed on the FCM to reduce the sensitivity of the clusters on the random initialization of the membership function. GA succeeded in obtaining the minimum objective function and the number of clusters; however the documents forming clusters changed.

Cluster merging was implemented to reduce the effects of random initialization of membership function on clusters. Cluster merging assisted in obtaining the optimum number of clusters however, the effect of initialization on the clusters sustained.

Several data sets, consisting of large and small number of documents was implemented to narrow down the factor, which is affected by the initialization of the membership function.

Based on the results summarized in table 11 and 12, the following is concluded:

" As long as the input data to the FCM is correct, the random initialization of membership function has null effect on the clusters for membership value 2"

The conclusion resulted in evolution of a modified keyword weight assigning scheme.

7. Modified TFIDF

The parameters that are added in modified TFIDF are explained briefly in the subsequent sections:

a. Global Frequency
   The minimum number of term occurrence in the data collection. A term occurs in one document but does not occur in other documents should be least important. The minimum global frequency set is 2. The term should at the least occur in two documents.

b. Local Frequency
   The minimum number of term occurrence in the document. The term appears only once in the document, such term is irrelevant. Such term could be an effect of typo error and should not be considered. In order to consider a term for clustering, it should occur at the least twice in the document.

Part A:

$$Modified\ TFIDF\ (t3) = \left[\frac{(maxterm\ per\ document)}{[(term\ occurence\ in\ that\ document) + (term\ document\ length) + (maxterm\ occurence)]}\right] + \left[\frac{(maxterms\ in\ a\ document)}{(total\ number\ of\ terms)}\right]$$

Where

Max term per document – Total number of terms in a document
Term occurrence in that document – tells how many times the term occurs in the document
Term document Length – max occurrence of active terms in a document in which the term occurs.
Max term occurrence – maximum of all the term occurrence in the document
Total number of terms – total number of terms in a document

Part B:

$$Modified\ TFIDF = \left[\frac{((t3)*(total\ documents\ in\ which\ term\ occurs))}{(sum\ of\ tf\ along\ row)}\right] * (global\ frequency)$$

If the document has only one term then the above formula changes to:

$$Modified\ TFIDF = (global\ frequency) * \left[\frac{(sum\ of\ tf\ along\ row)}{(t3 * total\ documents\ in\ which\ term\ occurs)}\right]$$

Modified TFIDF performs cumulation of terms. The weight is assigned such that the value indicates total number of terms in the document in which term occurs and the number of documents in which the term occurs. The modified TFIDF weight of the term determines the contribution of the term in the document and is independent of the terms occurring in other documents. Modified TIIDF assigns weigh to the term such that the terms (say t1) occurring in one document should not affect the weight of terms (say t2) occurring in other documents which do not include the terms (t1).

Terms	D1	D2	D3	D4	D5	D6	D7
algebra	0	0	0	0	0	0	0
analysi	0	0	0	0	0	0	0
automat	0	0	0	0	0	0	0
comput	0.362	0	0	0	0.577	0	0
cryptogram	0	0	0	0	0	0	0
deomposit	0	0	0	0	0	0	0
inform	0	0	0.453	0.431	0	0	0.749
intellig	0	0	0	0	0	0	0
introoduct	0	0	0	0	0	0	0
larg	0	0	0	0	0	0	0
librari	0	0	0	0	0	0	0
linear	0	0	0	0	0	0	0
matrix	0	0	0	0	0	0	0
modern	0	0	0	0	0	0	0
organ	0	0	0	0	0	0	0
retriev	0	0	0.363	0.35	0	0	0
scale	0	0	0	0	0	0	0
singular	0.465	0.422	0	0	0	0.466	0
softwar	0	0	0	0	0	0	0
spars	0	0	0	0	0	0	0
valu	0.465	0.422	0	0	0	0.466	0

Table 13: Modified TFIDF

8. Cluster Size

In the preceding sections we mention dependence of FCM clusters on the cluster size. As the cluster size changes the documents in the cluster change. This problem could be solved using cluster merging. In cluster merging, FCM starts with large number of clusters and stops when the cluster could no longer be merged. However, considering the application, clustering blog information, the size of the cluster should meet user's requirement. User should have the option, if he/she wants a general overview of what types of documents are present under the topic or he is looking for specific information within the blogs. The approach to meet this requirement is, provide two clusters for the dataset and have a certain depth within each cluster. The peculiarity of clustered information increases with the depth within each cluster. Thus the user has both general overview and specific information at its display and has the choice as per his/her requirement.

9. Comparison of Classical TFIDF and Modified TFIDF

In classical TFIDF weight of a term in a document affects the weight of other terms in other documents. Classical TFIDF gives the closeness of the documents however; it does not include information about how different the documents are from each other.

Run #	Clusters	Document	Objective Function
1	1 2	D3,D4,D7 D1,D2,D5,D6	0.5176
2	1 2	D3,D4,D7 D1,D2,D5,D6	0.5176
3	1 2	D1,D2,D5,D6 D3,D4,D7	0.5176
4	1 2	D3,D4,D7 D1,D2,D5,D6	0.5176
5	1 2	D3,D4,D7 D1,D2,D5,D6	0.5176

Table 14: FCM clusters with modified TFIDF

10. Analysis of FCM Parameters

As discussed in section 3.4 FCM is based on minimizing the objective to function. Objective function includes the product of membership matrix and the distance between document and the center of the cluster. FCM starts with initialization of membership matrix, calculates the center using the membership matrix. Membership function is updated using the calculated center. It calculates the distance between the documents and the center. Using this distance and membership function objective function is calculated. The process continues until the stopping criterion is fixed.

Referring to section 3.4 the formula for membership matrix is:

$$u_{ij} = \frac{1}{\sum_{k=1}^{c} \left(\frac{\|x_i - c_j\|}{\|x_i - c_k\|} \right)^{\frac{2}{m-1}}}$$

Where m is the membership value.

We varied the membership value and noted the corresponding objective function and the clusters for the TFIDF data shown in table 15. Table 16- 24 shows variation in objective function and clusters for the given cluster size and membership value.

	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15	D16	D17	D18	D19
breast	0.4185	0.4185	0.4185	0.4185	0.3895	0	0.4185	0	0	0.3895	0	0	0	0	0	0	0	0	0
cancer	0.4185	0.4185	0.4185	0.4185	0.3895	0	0.4185	0	0	0.3895	0	0	0	0	0	0	0	0	0
download	0	0	0	0	0	0	0	0	0	0	0	0.2183	0	0.2555	0	0	0	0	0
incom	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.2354	0.2354	0	0
inform	0	0	0	0	0	0	0	0	0	0	0.2325	0	0	0	0	0	0	0	0.2384
java	0	0	0	0	0	0	0	0	0	0	0.3398	0.3563	0.7666	0.3057	0	0	0	0	0
page	0	0	0	0	0	0	0	0	0	0	0	0.2343	0	0.2365	0	0	0	0	0
prayer	0	0	0	0	0	0.2846	0	0.2846	0.4878	0	0	0	0	0	0	0	0	0	0
product	0	0	0	0	0	0	0	0	0	0	0.2325	0	0	0	0	0	0	0	0.2384
strength	0	0	0	0	0	0.2354	0	0.2354	0	0	0	0	0	0	0	0	0	0	0
tax	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.4571	0.3125	0.3125	0.4571	0.3821
treatment	0	0	0	0.249	0	0	0.2233	0	0	0	0	0	0	0	0	0	0	0	0
tutori	0	0	0	0	0	0	0	0	0	0	0.2373	0.2336	0	0	0	0	0	0	0
women	0.273	0.273	0.273	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 15: TFIDF Membership analysis

For Cluster =2

C1	6,8,9,11,12,13,14,15,16,17,18,19	6,8,9,11,12,13,14,15,16,17,18,19	6,8,9,11,12,13,14,15,16,17,18,19	6,8,9,11,12,13,14,15,16,17,18,19	1,2,3,4,5,7,10
C2	1,2,3,4,5,7,10	1,2,3,4,5,7,10	1,2,3,4,5,7,10	1,2,3,4,5,7,10	6,8,9,11,12,13,14,15,16,17,18,19
	m=0.5	m=1	m=1.5	m=2	m=2.5
	obj	obj	obj	obj	obj
	0.111543375862551	0.073922220377798	0.131038077831631	0.062767914904736	0.345157540661493
	0.109483389496695	0.034142898682597	0.111779545729879	0.029677689529154	0.292413985115773
	0.101095503218478	0.004461319571317	0.084737579807990	0.027643057924450	0.288720353577483
	0.075590168028555	0.000145919449408	0.075726629889793	0.027582618345660	0.278210352777879
	0.030729533027130	0.000000088676252	0.075595513939674	0.027579499412983	0.239082998790607
	0.003388518666656	0.000000000000066	0.075595055852157	0.215776297378664	0.185199979668308
	0.000035014236461			0.215775653221851	0.174721812664289
	0.000000003525256				0.174489701099793
	0.000000000000000				0.174482024118356

Table 16: Objetive Function for 2 Clusters (a)

1,2,3,4,5,7,10	1,2,3,4,5,7,10	1,2,3,4,5,7,10,11,12,13,14,15,16,17,18,19	1,2,3,4,5,7,10	15,18,19
6,8,9,11,12,13,14,15,16,17,18,19	6,8,9,11,12,13,14,15,16,17,18,19	6,8,9	6,8,9,11,12,13,14,15,16,17,18,19	1,2,3,4,5,6,7,8,9,10,11,12,13,14,16,17
m=3	m=15	m=35	m=50	m=80
obj	obj	obj	obj	obj
5.056285005869E-02	5.893871093880E-04	1.915620E-10	6.323908E-02	7.745420E-10

4.441649295919E-02	6.907610897600E-05	4.980000E-13	1.000000E-15	0.000000E+00
4.409205598443E-02	6.484069645200E-05		1.000000E-15	
4.309851742465E-02				
4.062908389914E-02				
3.690361775790E-02				
3.439319462776E-02				
3.350186712042E-02				
3.315704516906E-02				
3.300539949191E-02				
3.2943570344528000 0000E-02				
3.2920552421851000 0000E-02				
3.2912501324352000 0000E-02				

Table 17: Objective function for 2Clusters (b)

6,8,9	1,2,3	6,8,9,11,12,13,14 ,15,16,17,18,19
1,2,3,4,5,7,10,11, 12,13,14,15,16,1 7,18,19	4,5,7,10,6,8,9,1 1,12,13,14,15,1 6,17,18,19	1,2,3,4,5,7,10
m=100	m=300	m=500
obj	obj	obj
5.519800E-16	3.406300E-64	7.684400E-145
9.108600E-32		8.425700E-152

Table 18: Objective Function for 2Clusters (c)

For Clusters 3:

6,8,9,11,12,13,14,15,16 ,17,18,19	6,8,9,11,12,13,14	6,8,9,15,16,17,1 8,19	11,12,13,14	15,16,17,18,19
1,2,3,4,5,7,10	1,2,3,4,5,7,10	11,12,13,14	6,8,9,15,16,17,1 8,19	6,8,9,11,12,13,14
	15,16,17,18,19	1,2,3,4,5,7,10	1,2,3,4,5,7,10	1,2,3,4,5,7,10
m=0.5	m=1	m=1.5	m=2	m=2.5
obj	obj	obj	obj	obj
1.0328710140777E-01	6.43927232647E-02	5.4365488E-02	2.8657535E-02	5.4217661E-02
8.1581703365138E-02	6.41657191305E-02	7.3199950E-03	1.7119929E-02	7.9787785E-03
3.8369421121289E-02	6.32784408254E-02	6.8153000E-11	5.6253674E-04	3.2727172E-06
5.5757277745300E-03	5.98557957099E-02	0.0000000E+00	3.2941802E-04	0.0000000E+00
9.6677320698000E-05	4.81284987900E-02		3.2940807E-04	
2.6703485000000E-08	2.23648400647E-02			
2.0000000000000E-15	2.63590248077E-03			

Table 19:Objective Function for 3Clusters (a)

1,2,3,4,5,7,10	11,12,13,14,15,16,17,18,19	1,2,3,4,5,7,10	6,8,9,15,16,17,18,19	6,8,9,11,12,13,14,15,16,17,18,19
6,8,9,11,12,13,14	6,8,9	6,8,9,11,12,13,14	11,12,13,14	4,5,7,10
15,16,17,18,19	1,2,3,4,5,7,10	,15,16,17,18,19	1,2,3,4,5,7,10	1,2,3
m=3	m=15	m=35	m=50	m=80
obj	obj	obj	obj	obj
1.7809212E-02	1.8276845E-04	1.4503636E-08	7.38250E-28	8.64380E-15
1.2013049E-03	5.2783425E-06	0.0000000E+00	1.43690E-38	2.57090E-04
5.7118012E-08	6.6847180E-09			
0.0000000E+00				

Table 20:Objective Function for 3Clusters (b)

4,5,7,10	1,2,3,5,10	15,16,17,18
1,2,3	6,8,9,11,12,13,14,15,16,17,18,19	11,19
6,8,9,11,12,13,14,15,16,17,18,19	4,7	1,2,3,4,5,7,10,12,13,14,6,8,9,
m=100	m=300	m=500
obj	obj	obj
6.62260E-37	9.949E-38	2.1269E-113
	1.253E-09	8.4257E-152

Table 21: Objective Function for 3Clusters (c)

For Clusters=4

C1	6,8,9,11,12,13,14,15,16,17,18,19	1,2,3,4,5,7,10	6,8,9	6,8,9	11,12,13,14
C2	1,2,3,4,5,7,10	15,16,17,18,19	11,12,13,14	11,12,13,14	15,16,17,18,19
C3		6,8,9,11,12,13,14	1,2,3,4,5,7,10	1,2,3,4,5,7,10	1,2,3,4,5,7,10
C4		nil	15,16,17,18,19	15,16,17,18,19	6,8,9
	m=0.5	m=1	m=1.5	m=2	m=2.5
	obj	obj	obj	obj	obj
	8.864151E-02	3.84083081615769000000E-01	6.1582721E-02	3.7927818E-02	4.1156862E-02
	8.233415E-02	3.87014916970164000000E-01	4.8380607E-02	1.7631867E-03	1.0849010E-02
	6.235639E-02	3.88674807524683000000E-01	3.4667451E-03	7.1275400E-10	7.7191718E-05
	2.619430E-02	3.89336186949300000000E-01	1.1400000E-12	0.0000000E+00	6.9400000E-13
	3.074809E-03	3.89548009408815000000E-01	0.0000000E+00		0.0000000E+00
	3.565939E-05	3.89539623571849000000E-01			
	4.515178E-09				
	0.000000E+00				

Table 22: Objective Function for 4Clusters(a)

11,12,13,14	6,8,9,11,12,14	11,12,14	4,7	11,15,16,17,18,19
1,2,3,4,5,7,10	1,2,3,4,5,7,10	1,2,3,4,5,7,10	5,6,8,9,10,15,16,17,18,19	1,2,3,4,5,7,10
6,8,9	13	6,8,9,15,16,17,18,19	1,2,3	9
15,16,17,18,19	15,16,17,18,19	13	11,12,13,14	6,8,12,13,14
m=3	m=15	m=35	m=50	m=80
obj	obj	obj	obj	obj
3.4133116E-02	4.1137841E-04	0.0000000E+00	2.1642636E-07	9.9890E-22
5.3498146E-03	1.1991750E-06	1.3005000E-15	4.0332000E-16	1.0264E-25
4.5084378E-06	1.8799900E-10			
7.0000000E-15				

Table 23: Objective Function for 4Clusters (b)

6,8,9,11,12,13,14	6,8,9,15,16,17,18,19	11,15,16,17,18,19
15,16,17,18	11,12,13,14	6,8,12,13,14
1,2,3,4,5,7,10	1,2,3,5,7,10	1,2,3,4,5,7,10
19	4	9
m=100	m=300	m=500
obj	obj	obj
4.4192E-11	5.4050E-79	7.86510E-09
3.8792E-31	1.3533E-91	

Table 24: Objective Function for 4Clusters (c)

From above tables we observe that m=0.5 fails for Clusters 3 and 4, m=1 fails for cluster#4, m=15 fails for cluster4, m=35 fails for cluster#2, m=50 fails for cluster#4, m=80 fails for cluster#2 and 4, m=100 fails for cluster#2 and 4, m=300 fails for cluster 4 and m=500 fails for cluster4.

The clusters and objective function values are observed approximately same for m=1.5, 2 and 2.5. However, under several iterations, m=1.5 and m=2.5 fail. Hence we choose m=2.

The graph of data and cluster center is shown:

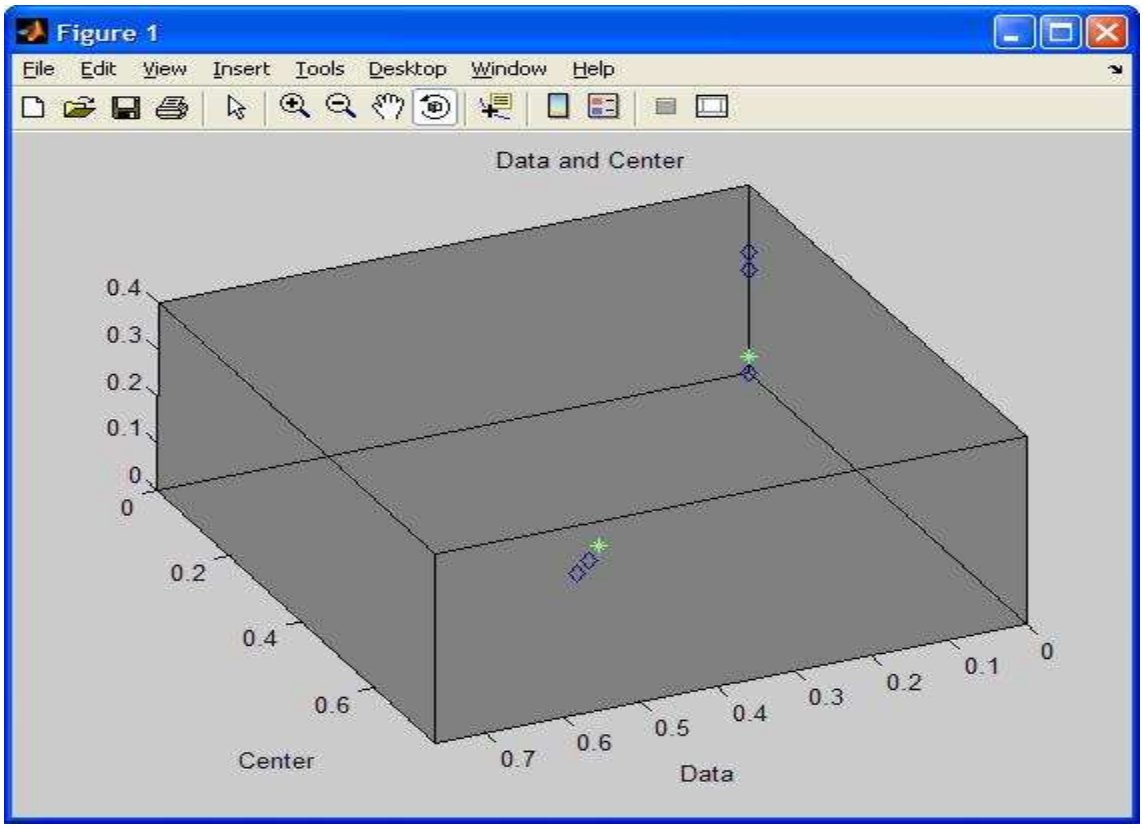


Figure 17 : Data Center Plot

The graph below shows the cluster and sub-cluster objective function:

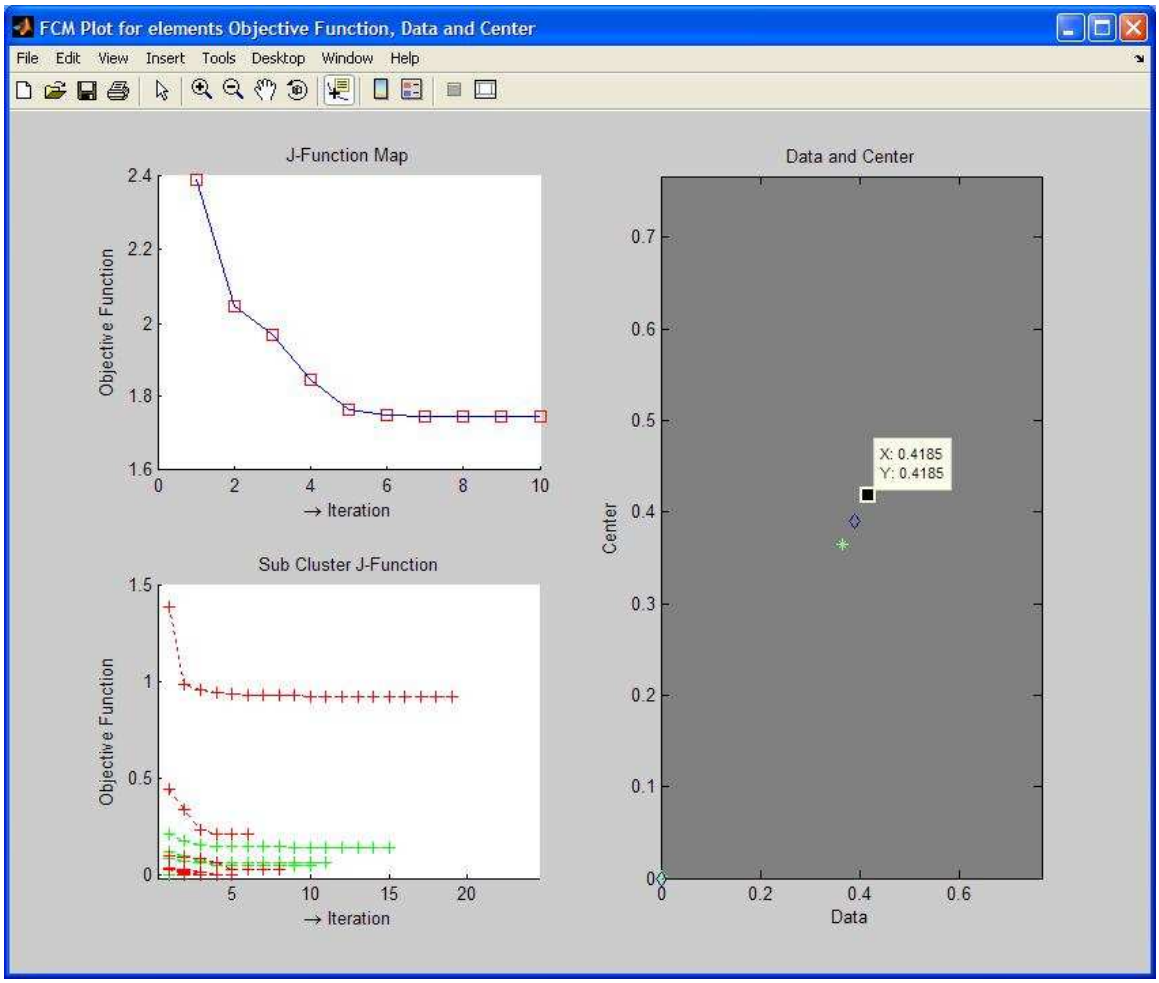


Figure 18: Cluster and Sub-Cluster Objective Function

The following figure shows the behavior of objective function for FCM

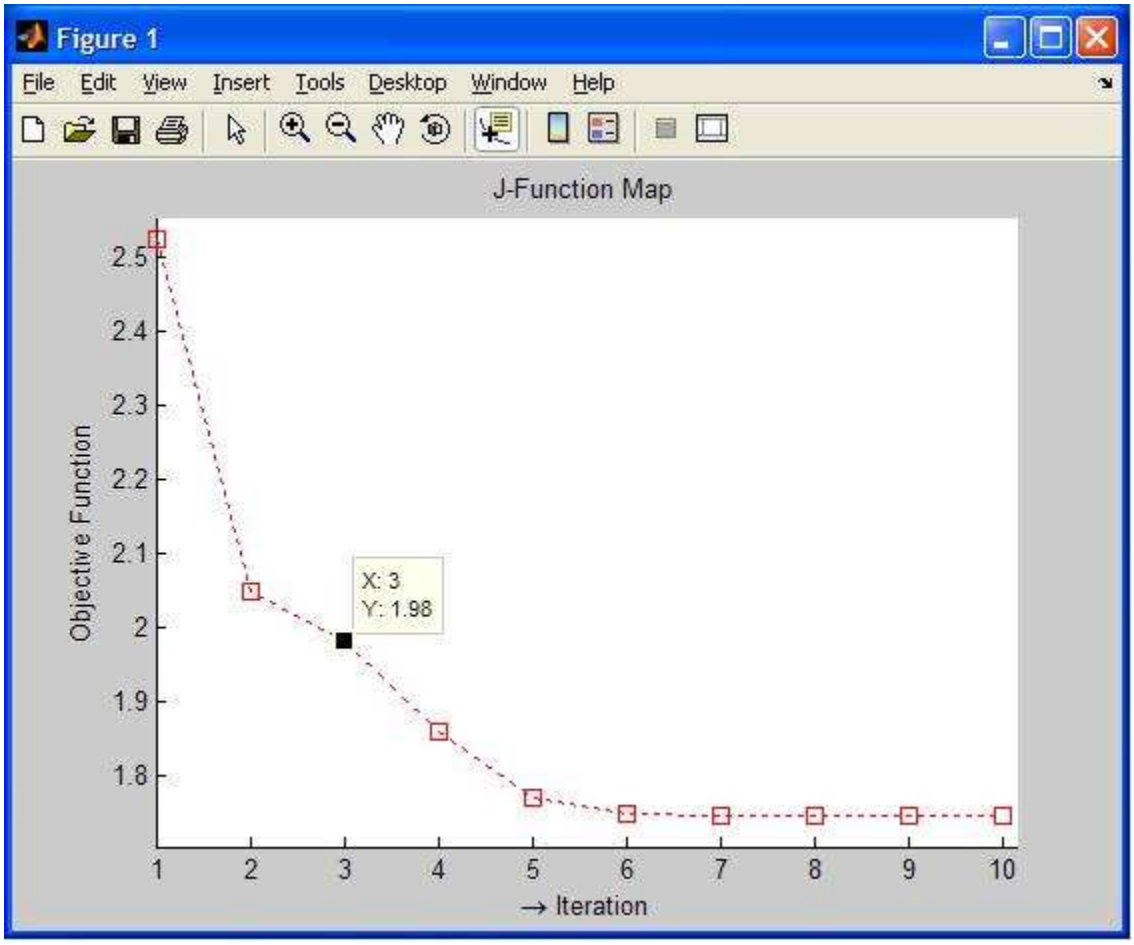


Figure 19: FCM Cluster Objective Function

FCM Sub-Cluster Objective Function

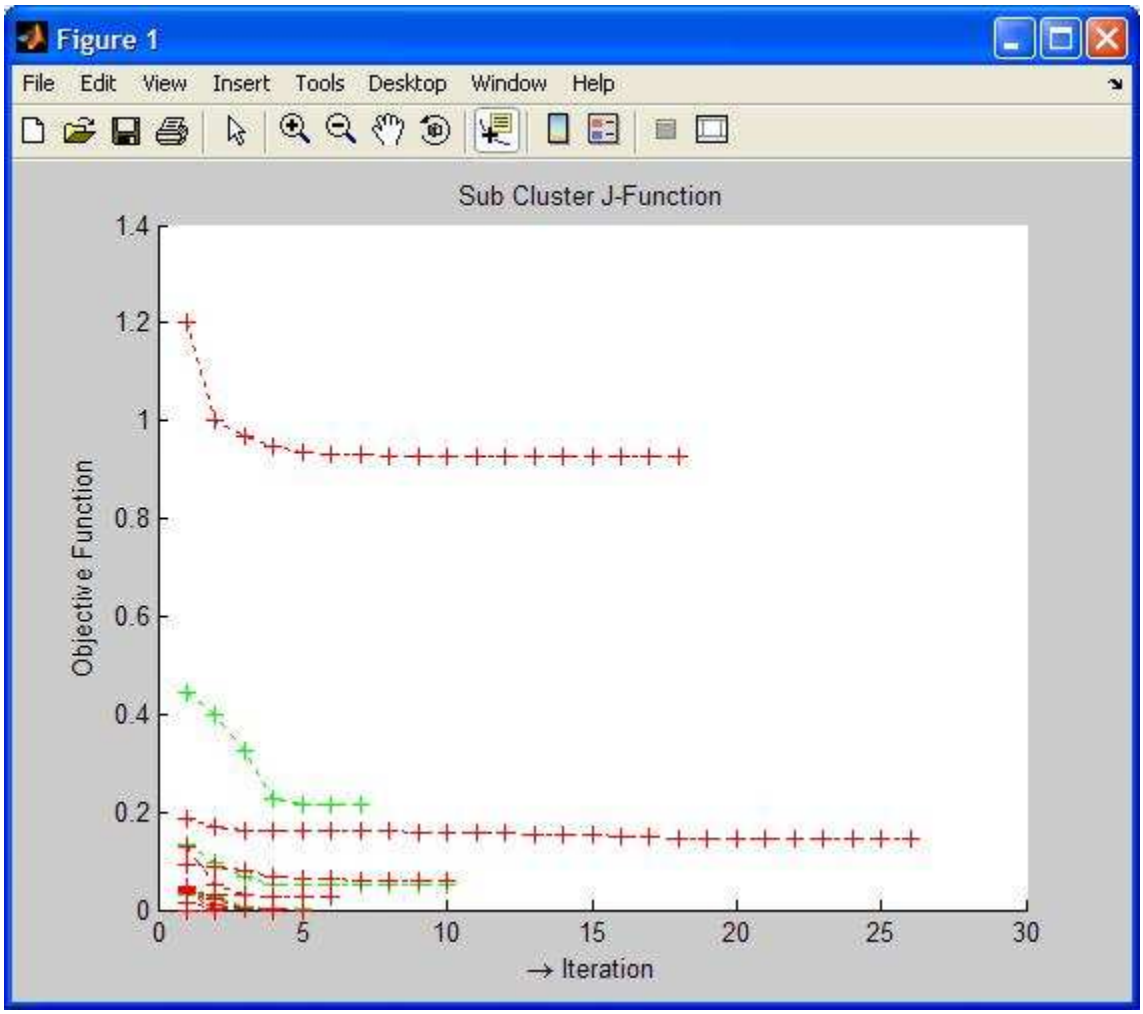


Figure 20: FCM Sub-Cluster Objective Function

11. Software Environment

The data collection step mentioned in above section is implemented using Java. A java program is written to collect the data from the website.

The remaining steps of data processing and clustering algorithms are implemented using Matlab 7.1

The application window is shown as:

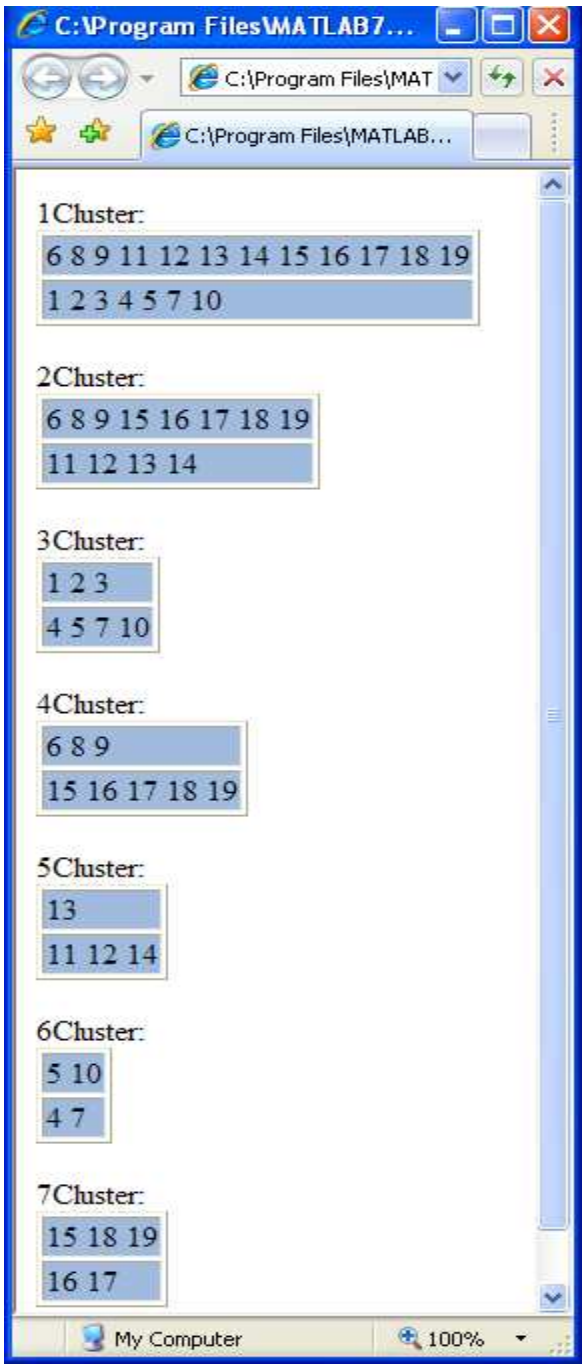


Figure 21 Clustering Blog Information Application Window

The above figure shows the Clustering Blog Information Application window. It shows the cluster and corresponding clusters with the document number. As can be seen, the more the depth of each cluster more information is retrieved.

The documents can be viewed by clicking the “blue” colored area. The area expands and user can read the documents within each cluster. User can read documents in one cluster

or more clusters at a given time. The application window with documents is shown below:

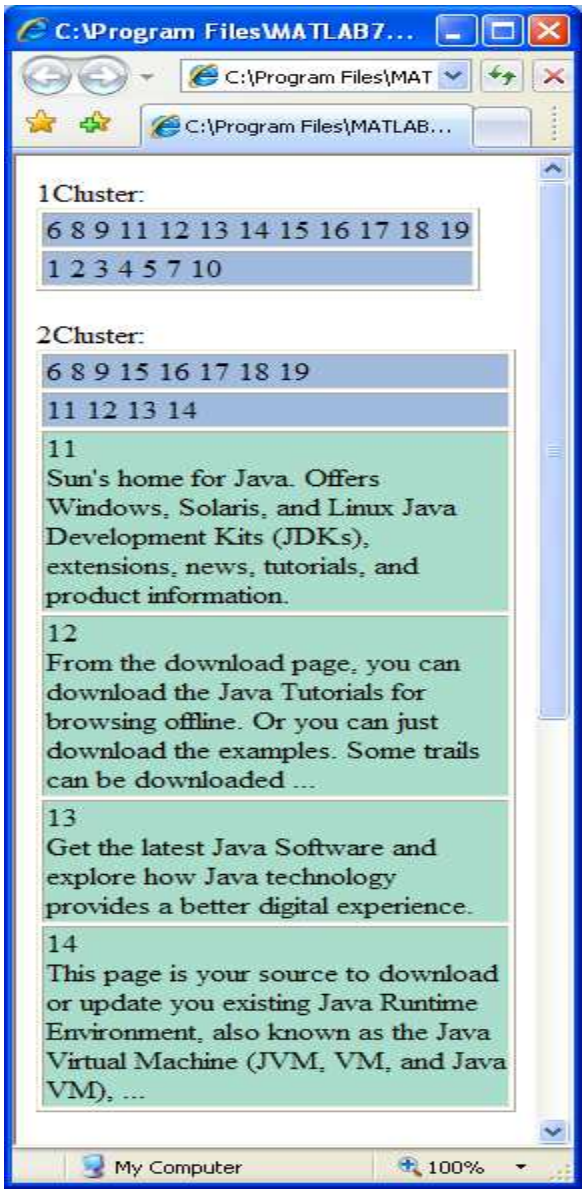


Figure 22 Application Window with Documents

The matlab code displays the information of the first two steps involved in Clustering Blog Information. It shows total number of documents, number of terms, number of stop words. The following figure is the snapshot of matlab results

Figure 23: Matlab Results

12. Conclusion

In our project, Clustering Blog Information, we divided clustering process in three steps; viz; Data Collection, Data Processing and Clustering Algorithm. Data collection is an elementary process in clustering blog information used to obtain data to be clustered. Data can be obtained online or offline. We discussed blogs can be clustered with respect to the response to a particular topic, with respect to the topic or with respect to the blog

information. For this project we consider blogs data as the source information and ignore any images, additional control buttons within the blogs. Data collection is used to eliminate the images or any factor that is included in blogs other than the text. Online blogs consists of HTML tags which carry no information for clustering. Eliminating HTML tags from the blogs forms an important step in the data collection process.

Data processing follows the data collection process of clustering blog information. Data retrieved from the blog website using data collection, consists on many repetitive and less important information. Example, punctuation marks, pronouns, etc carry very little almost null information. Hence such the data should be filtered to get rid of such repetitive and less important information. Data processing performs the function and converts the blog data in format that could be used by clustering algorithms. Data processing uses different weight assigning schemes to assign weight to terms in the blogs. The weight assigned terms are passed as an input data to the clustering algorithms. The paper gives a brief description of few weight assigning algorithms.

Several clustering algorithms are available. We follow the approach of understanding, implementing and comparing, the working of four clustering algorithm and selected the optimum clustering algorithm depending on the output of the clustering algorithms. The clustering algorithms discussed in the paper are K-means, VSM (Vector Space Model) cosine similarity measurement based clustering, LSI (Latent Semantic Indexing) and FCM(Fuzzy C-Means Clustering).

VSM represents documents and query as vectors. The angle between the document and query defines the similarity between them. In the experiments performed cosine similarity measurement is used to determine the angle between document and query. Thus output of VSM is vectorial representation of documents and query; incorporation of cosine similarity measurement with VSM gives the similarity measure of document to the query. To retrieve clusters from the similarity measurement an additional threshold is required. Also VSM with cosine similarity measurement fails to cluster documents with different vocabulary but same contents.

LSI overcomes the disadvantages of VSM by introducing clustering based on concepts rather than terms within the documents. However, output of LSI is a score that indicates the similarity of documents to the query based on concept. In order to retrieve the clusters from LSI an additional threshold needs to be implemented on the output score.
VSM cosine similarity measure and LSI have a common disadvantage. The similarity measure in VSM and the score in LSI depend on the query. Hence with change in query, clusters change and eliminate documents that are not related to the query even though they are related to the cluster.

Unlike VSM cosine similarity measure and LSI, output of k-means and FCM is clusters. However, for k-means, clustering depends on the mean and the mean changes with the number of clusters. Dependence of clustering on means, results in clusters with documents that are not correlated to each other. Contrary, the documents in the FCM

clusters are correlated to each other. Thus, amongst the discussed clustering algorithms, we select FCM as the clustering algorithm for blog clustering.

FCM has its own disadvantages though. The clusters for a given set of data are not fixed in FCM. The clusters formed change with the cluster number due to un-proportionate weight assignment of the term. Several experiments were performed on the same data set with different number of clusters and iteration gave different cluster. Proportionate weight assignment of term, which is achieved by using modified TFIDF, overcomes the above mentioned disadvantage of FCM. A brief description of modified TFIDF and its comparison with classical TFIDF is presented in the paper. Modified TFIDF proportionately assigns weight to the term, such that each term knows its contribution in the document. Modified TFIDF gives consistent clusters with the same objective function. Document clustering using FCM depends on the clustering size. As the cluster size changes the documents in the cluster changes. Considering the application, clustering blog information, the size of the cluster should be as per user's requirement. We meet the requirement by dividing the dataset into two clusters and provide depth within each cluster. The user can go to a particular depth depending on the granularity of information required by the user. Thus with modified TFIDF and sub-clustering the clustering blog information algorithm gives correct result and the user can view the clusters depending on the granularity of information required by the user.

References

1. Agarwal.G., Goswami.A., and Jin. R. (2004) *Fast and Exact Out of Core K-means Clustering.* Proceedings of the Fourth IEEE International Conference on Data Mining (ICDM'04)

2. Berry. M.W., Drmac. Z., znd Jessup.R.E. *Matrices, Vector Spaces and Information Retrieval.* SIAM Rev. 41, pp. 335-362

3. Carven, M. *Introduction to Information Retrieval,* Retrieved on March 6, 2007 from University of Wisconsin, Department of Computer Science Website: http://www.cs.wisc.edu/~shavlik/cs540/introToIR.ppt

4. Church, K.W and Gale, W.A. *Inverse Document Frequency (IDF): A Measure of Deviations from Poisson.* AT&T Bell Laboratories

5. Djouani.K & Nefti.S (2003) *Extended Fuzzy Clustering Algorithm Based on Inclusion Concept* The 12th IEEE International Conference on Fuzzy Systems, 2003. Fuzz'03 (869-874) vol.2

6. El-Sharkawi, M.A. *Fuzzy System and Control,* Retrieved on March 10, 2007 from University of Washington, Department of Electrical Engineering, Computational Intelligence Applications Laboratory (CIA) lab website: http://cialab.ee.washington.edu/index_files/tutorial/fuzzy.pdf

7. Fan.W, Gordon.M.D, and Pathak.P *A generic ranking function discovery framework by genetic programming for information retrieval*. Retrieved on September 7, 2007 from http://filebox.vt.edu/users/wfan/paper/ARRANGER/ip&m2003.pdf

8. Fielong. X. *Latent Semantic Indexing,* Retrieved on March 20, 2007 from computational Linguistics and Phonetics Department, University of Saarlandes website, http://www.coli.uni-saarland.de/~schulte/Teaching/Klassifikation-04/feilong.pdf

9. Gen.M, Tsujimura.Y, and Zhao.L *Genetic Algorithm for Fuzzy Clustering.* Dept. of Ind. & Syst. Eng., Ashikaga Inst. of Technol., Japan; Proceedings of IEEE International Conference on Evolutionary Computation, 1996. 20-22 May 1996 716 – 719

10. Jea.Y.T *Basic Concepts of Data Mining, Clustering and Genetic Algorithms* Department of Computer Science and Engineering SUNY at Buffalo

11. Jing.L., Ng. M., and Huang.M.Z (2006) *Text Clustering: Algorithms, Semantics and Systems.* The University of Hong Kong and Hong Kong Baptist University.

12. Joachims, T (2004). *Representing and Accessing Digital Information: Information Retrieval: Indexing.* Retrieved March 10, 2007, from Cornell University Department of Computer Science website:
http://www.cs.cornell.edu/courses/cs630/2004fa/lectures/tclust_6up.pdf

13. Karayiannis.N.B, & Randolph-Gips.M.M (2002) *Non-Euclidean c-means clustering algorithms.* Intelligent Data Analysis 7 *(2003)* (405-425)

14. Keller. M.J, Krishnapuram.R., Kuncheva.I.L., Bezdek.C.J, & Pal.R.N (1999) *Will the Real Iris Data Please Stand Up?* IEEE Transactions on Fuzzy Systems (368-369) Vol.7

15. Lazarinis, F. *Porter.Java, IR Linguistic Utilities.* Retrieved on March 7, 2007 from
http://www.dcs.gla.ac.uk/idom/ir_resources/linguistic_utils/porter.java

16. Liu,J. and Xie,W. *A Genetics Based Approach to Fuzzy Clustering.*(1995) Proceedings of 1995 IEEE International Conference on International Joint Conference of the Fourth IEEE International Conference on Fuzzy Systems and The Second International Fuzzy Engineering Symposium., 20-24 March 1995 2233 - 2240 vol.4

17. Losee.M.R, (1998). *Comparing Boolean and Probabilistic Information Retrieval Systems Across Queries and Disciplines.* J. of the American Society for Information Science

18. Mendes, M.E.S. and Sacks, M.L. *Knowledge Based Content Navigation in e-Learning Applications.* Retrieved on February 4, 2007 from University College London, Department of EE website
http://www.ee.ucl.ac.uk/lcs/papers2002/LCS115.pdf

19. Mishne.G. & Rijke.D.M (2005) *Vector Space Model.* Informatics Institute University of Amsterdam

20. Oroumchian,F. and Garamalek, F.M. *An Evaluation of Retrieval Performance Using Farsi Text.* Retrieved on February 20, 2007 from University of Tehran, Department of Electrical and Computer Engineering website:
http://www.ut.ac.ir/fa/farsi-writting-and-reading/articles/3.pdf

21. Osinski, S. (2003). *An Algorithm for clustering of web search results.* Poznan University of Technology, Poland.

22. Porter, M.F. (1980). *An Algorithm for suffix stripping.* Program 14 (3 1980), 130—137 Retrieved on March 1, 2007 from: http://www.tartarus.org/~martin/PorterStemmer/def.txt

23. Sanderson, M. *IR Linguistic Utilities,* Retrieved on March 7, 2007 from http://www.dcs.gla.ac.uk/idom/ir_resources/linguistic_utils/stop_words

24. Sm. Kannan; V. Jayabalan; K. Jeevanantha (2003) *Genetic algorithm for minimizing assembly variation in selective assembly* International Journal of Production Research, 41:14, 3301 - 3313

25. Vazirgiannis.M., Halkidi.M, & Batistakis.Y. (2002) *Cluster Validity Methods: Part 1* SIGMOD Record, Vol.31, No. 2, June 2002

26. Zheng,Z and Erbach,G. *Specialised Search in Linguistics and Languages.* Retrieved on February 15, 2007 from Saarland University, Computational Linguistics Department website: http://answerbus.coli.uni-saarland.de/zheng/CIC2002.pdf

27. *A Tutorial on Clustering Algorithms, Fuzzy C-Means Clsutering,* Retrieved on March 10, 2007, from http://www.elet.polimi.it/upload/matteucc/Clustering/tutorial_html/cmeans.html

28. Matlab Help files. Reference to Mathworks.com

29. Pedrycz.W. *Knowledge Based Clustering- From Data to Information Granules*

30. Zeimpekis.D, and Gallopoulos.E (2005)*Design of Matlab Toolbox for term-document Matrix generation* Retrieved from University of Patras, Computer Engineering and Informatics Department