**Average linkage clustering with Multi Viewpoint Based similarity Measure**

A Thesis-Equivalent Project

Presented to the

Department of Computer Science

and the

Faculty of the Graduate College

University of Nebraska

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

University of Nebraska at Omaha

by

Divya Teja Ravoori

December 2014

Supervisory Committee:

Dr. Zhengxin Chen, Chair

Dr. Sanjukta Bhowmick

Dr. Gerardus de Vreede

Average Linkage Clustering with Multi View point based Similarity measure

Divya Teja Ravoori

University of Nebraska at Omaha, 2014

Advisors: Dr. Zhengxin Chen, Dr. Sanjukta Bhowmick, Dr. Gerardus de Vreede

**Abstract:**

Clustering is the process of finding intrinsic structures among large quantities of unordered data and organizing them into meaningful subgroups which can be used for further study and analysis. In order to group the data, cluster algorithms assume some relationship among data. The similarity among data can be defined either explicitly or implicitly. In this project, I compare two clustering algorithms MVSC-IR and K-means which uses different similarity measures. The first one uses multi-view point based similarity (MVS) measure which calculates similarity based on the documents in all other clusters, while the second one uses single view point which is centroid of its own cluster. Then, I embed MVS, which forms better clusters than the single viewpoint, with the average linkage cluster criterion function that follows hierarchical clustering. Embedding the average linkage algorithm with multi view point similarity measure helps to remove dependency on random initialization of clusters which is the main drawback of K-means and MVSC-IR algorithms. The embedded approach shows improved results when compared with other two algorithms in terms of various measures specifically, F-score, Accuracy and NMI.

**Dedication and Acknowledgement:**

I deem it my great pleasure to have done a project “Average Linkage Clustering with Multi View Point based similarity measure” related to Document Clustering.

I would like to extend my sincere thanks to my supervisor *Dr. Zhengxin Chen* for his guidance, inputs and support that helped me make progress with my project in a timely fashion. I also want to thank my committee members *Dr. Sanjukta Bhowmick, Dr. Gerardus de Vreede* for their support and their valuable suggestions that led to the completion of my project.

I would also like to thank the professors of the Department of Computer Science at University of Nebraska at Omaha who facilitated numerous courses. The hands-on experience I gained during my coursework helped me understand the criticalities related to the project easily and fuelled my interest in doing research in document clustering techniques and similarity measures for the completion of my project.

Most importantly, I would like to acknowledge my sincere love to my family and friends who have supported and motivated me all through my life.

Table of Contents

[1. Introduction: 5](#_Toc403339366)

[2. Literature Survey: 8](#_Toc403339367)

[3. Clustering Techniques: 13](#_Toc403339368)

[3.1 MVSC- IR Clustering Technique: 13](#_Toc403339369)

[3.2 K-Means clustering Technique: 15](#_Toc403339370)

[3.3 MVS with Average Linkage Clustering Technique: 16](#_Toc403339371)

[4 Project Description: 20](#_Toc403339372)

[4.1 Program Flow: 20](#_Toc403339373)

[4.2 Implementation Details: 23](#_Toc403339374)

[4.3 Technical Challenges: 27](#_Toc403339375)

[5 Experimentation Details: 29](#_Toc403339376)

[5.1 Documents Collection: 29](#_Toc403339377)

[5.2 Experiment Setup: 36](#_Toc403339378)

[5.3 Evaluation Measures: 36](#_Toc403339379)

[5.4 Screenshots: 38](#_Toc403339380)

[5.5 Results: 42](#_Toc403339381)

[6 Conclusions & Future Work: 46](#_Toc403339382)

# Introduction:

Clustering is the process of finding intrinsic structures in data and organizing them into meaningful subsets called clusters, such that objects within a cluster are more similar than the objects in other clusters. Clustering has been widely used in many applications such as pattern recognition, market research, image processing, and information retrieval. In business, clustering can help marketers discover distinct groups in their customer bases and characterize customer groups based on purchasing patterns. In information retrieval, clustering can be used to create more relevant set of search results when compared to normal search engines. It is the main task of exploratory data mining. As a data mining function, it is a standalone tool to gain insight into the distribution of data, to observe the characteristics of each cluster, and to focus on a particular set of clusters for further analysis.

There have been many clustering algorithms published every year and are developed using totally different approaches and techniques. These cluster algorithms are sensitive to various parameters like initialization of clusters, number of clusters expected, threshold value, similarity measure used. Basically, there is an implicit assumption that the true intrinsic structure of data could be correctly described by the similarity formula defined and embedded in the clustering criterion function [1].

The similarity measure used to form clusters can be defined in many ways. Euclidean distance is one of the most popular similarity measures that uses sum of the squared error. Cosine similarity is used instead of Euclidean distance for sparse and high dimensional data like text documents, which is in fact more suitable. Similarity of an object w.r.to a cluster can be considered from two different viewpoints. One is the single viewpoint which is centroid of the cluster. The most popular clustering algorithm k-means follow single view point based similarity. In this, Cosine similarity is used directly to calculate the distance between document and cluster centroid. The other one is the Multi-viewpoint based similarity, uses different viewpoints, which are objects assumed to not be in the same cluster with the two objects being measured. When we look a pair of points from different viewpoints which are objects outside the cluster we can have more accurate assessment of how close or distant a pair of points are. MVSC-IR algorithm uses multi view point based similarity.

These two clustering algorithms follow Partitional clustering approach. The sensitiveness to various parameters like initialization, number of clusters expected and threshold value depends on the approach used to follow the cluster process. Partitional clustering attempts to decompose the dataset into set of ‘k’ disjoint clusters and then the criterion function is used to improve the initial clusters formed by minimizing the local dissimilarity of clusters and similarity among other clusters. The algorithms which follow the partitional clustering approach have few basic drawbacks like sensitiveness to initialization, number of clusters expected in advance. These type of clustering algorithms work well for spherical shaped clusters in small and medium sized datasets but they are not suitable to find cluster with complex structures in large datasets.

Hierarchical clustering approach overcomes some of the drawbacks of partitional clustering approach like sensitiveness to initialization; able to form clusters irrespective of their shapes. The work in this project is motivated by the investigations from above and similar research findings. It appears to me that the similarity measure used along with clustering approach plays a vital role in the successful formation of clusters for the given dataset. My first objective in this project is to compare two clustering algorithms MVSC-IR and k-means that use multi-view similarity and single view point similarity measures to observe their performance experimentally. From my observation on comparison results of these two different similarity measures, I embedded the multi-view similarity measure which forms better clusters than the single view point based similarity measure with the average-linkage clustering algorithm. The Average-linkage clustering algorithm follows the Hierarchical approach which can form clusters irrespective of their shape and removes the dependency on cluster initialization process. I also compared and evaluated the embedded average linkage algorithm with the two other clustering algorithms in terms of F-score, NMI and Accuracy. In addition to this, I also observed how the considered content of document affects the cluster process. I used two parts of documents one is the entire text in document and the other is portion of the document which are meta tags in the input file. Performance of clusters may be improved in terms of accuracy when the whole text is taken into consideration than using part of the document.

What follows is a Literature review, description of algorithms, details of project implementation, and datasets used and experimental results, conclusion, future work and references.

# Literature Survey:

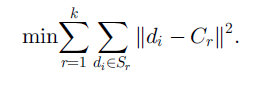
The following is the list of fundamental notations that will be used to signify documents and its associated concepts.

1. n represents the number of documents in the input set
2. m represents the number of terms in the document
3. c represents the number of classes
4. k represents the number of clusters
5. d represents the document vector
6. r represents a cluster
7. S represents the set of all documents
8. Sr represents the set of documents in cluster r
9. D represents composite1 vector of all documents
10. Dr represents the composite vector of all the documents in cluster r
11. C represents the centroid vector of all the documents
12. nr represents the number of documents in cluster r
13. Cr represents the centroid vector for a cluster r

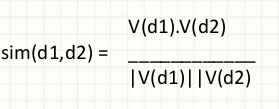
Composite vector means sum of all document vectors. The number of classes’ ‘c’ should be equal to the number of clusters ‘k’ in the ideal case. Each document considered for clustering is represented by an m-dimensional vector d, where m represents the number of terms the document has. Document vectors are the result of some weighing schemas like term frequency and inverse document frequency where term frequency represents the number of times the term occurs in a document and inverse document frequency is the logarithmic value of total number of input documents considered to the number of documents which contains the term.

Document clustering is one of the data mining techniques. It is the process of grouping objects into categories in a way that there is maximization of intra-cluster object similarity and inter-cluster object dissimilarity. Here, an ‘object’ means a document and ‘term’ refers to a word in the document. The concept of clustering itself represents that there is need for some form of measurement to determine the cluster similarity or dissimilarity. There are some clustering techniques that do not have any form of measurement to define the similarity like model based [2] and non negative matrix factorization [3]. In this project, I primarily focus on methods that use specific measures.

In literature, Euclidean is one of the most popular similarity measures [1]. It is used in traditional k-means for which the objective function is to minimize the distance between the objects and their respective cluster centroids.



However, for sparse and high dimensional objects like text documents, cosine similarity is popularly used instead of Euclidean distance. It is also a popular similarity score in information retrieval [4]. Cosine similarity is the measure of the cosine of angle between two documents.



The difference between the k-means with cosine similarity and Euclidean distance is the former represents vector directions while the latter represents vector magnitude.

Besides cosine similarity, another measure called extended jaccard coefficient is also used to represent the similarity between two documents and is used in a graph based clustering technique called CLUTO [5]. It measures similarity between two documents as ratio of intersection to the union of terms. i.e., for text documents it measures the sum weight of the terms that are common in the two documents divided by the sum weight of all the terms in the two documents without repetition. The formal definition is



Extended jaccard coefficient takes both magnitude and direction of vectors into consideration when compared to Euclidean distance and cosine similarity. Another method for clustering called correlation clustering [6] uses Pearson correlation measure. It divides a set of objects into best possible clusters without specifying the number of clusters in advance. In their paper [3] Xu, Liu and Gong defined two metrics for correlation clustering namely, minimizing disagreements and maximizing agreements between clusters.

In [7] Steh et al. compared four similarity measures Euclidean, Cosine similarity, Extended jaccard coefficient and Pearson correlation. From their results the cosine similarity and the extended coefficient are better than the other two measures especially for web documents. In [1] Multi-viewpoint based similarity measure is defined to measure similarity between text documents. It is quite opposite to traditional cosine similarity measure as this measures similarity between two documents with respect to all documents in other clusters. It also uses cosine similarity internally but considers different viewpoints instead of single view point which is the centroid of the cluster. The definition for measuring MVS is defined as follows:



From the experimental results in [1], multi-view similarity is better than cosine similarity and extended jaccard coefficient in most of the cases. The other factor that affects the clustering process is the approach it follows. There are many ways to define the cluster process. Popular algorithms like k-means and MVSC-IR follows the partitioning clustering. It forms k arbitrarily clusters initially and then improves the clusters from them trying to maximize the intra cluster similarity and inter cluster dissimilarity. Though these algorithms result in good clustering processes, there are a few basic drawbacks like sensitiveness to initialization, and feeding number of clusters expected in advance. Also, these methods can find only spherical-shaped clusters and encounter difficulty in discovering clusters of arbitrary shapes [8].

Unlike Partitional based methods, Density based clusters deals irrespective of the cluster shapes. The idea of this is to continue growing the given cluster as long as the number of objects in the neighborhood exceeds some threshold [8]. It is ambiguous to partition the points that share a border with more than one cluster using density based approach and also it cannot cluster the datasets with large differences in densities. Constraint based clustering performs clusters based on the constraints setup by user or constraints related to application. These constraints are the properties of the desired clustering results. This approach is biased based on the conditions and are different for different applications.

Hierarchical clustering solutions [13] are of great attention for a number of application domains. It can be a top down or a bottom up approach. The top down approach, called divisive hierarchical clustering, considers the complete dataset as a single cluster initially and divides into sub groups until the termination condition holds. In the other way, the bottom up approach, called agglomerative hierarchical clustering, considers every object or document as a single cluster and merges the clusters until the termination condition holds [9]. This type of algorithms covers the basic drawbacks of partitional clustering techniques such as sensitiveness to initialization, able to handle clusters even with arbitrary shapes.

I used ideas from different algorithms and came up with an approach that suits the clustering process. I.e., the average linkage clustering technique embedded with multi view point based similarity measure which covers the drawbacks of other algorithms like sensitiveness to initialization, shape of the clusters, defined similarity measure.

# 

# Clustering Techniques:

## MVSC- IR Clustering Technique:

MVSC-IR clustering technique uses Multi View point based Similarity (MVS) measure to cluster the documents. In general, all distance measures for document clustering is based on the distance of the document as measured from the origin. In MVS, we measure the distance of a document from every document that is not there in the cluster to which the document under consideration belongs. The average of all these distances is used to come to a conclusion about the correctness of clustering of the document.

The MVS measure hence provides much more in-depth and minutely detailed information of the validity of a document in a cluster as opposed to just the distance of the document from the origin as used in the cosine similarity. This is the principle advantage of using the MVS method.

Hence, initially it begin with a random number of documents as cluster seeds and then it assigns all the remaining documents to one or the other of these clusters depending on cosine similarity measure. A document gets assigned to the cluster whose seed document has the maximum cosine similarity.

Iteratively, it went through every document and calculated the difference in similarity caused if a document was moved from its current cluster to a new cluster. If the difference lead to better clustering then the movement is carried out, otherwise it would move to the next document. I did not want to bias the results by following the same traversal patterns every run and hence for every run the order in which the documents will be processed is randomly generated.

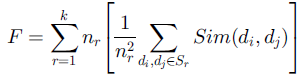
Local convergence of this algorithm depends on the initial cluster seed document choice. Convergence is achieved when a complete run takes place without any movement taking place from one cluster to another cluster. I built in a constraint on the number of times this algorithm runs in case convergence is not achieved. This prevents an infinite loop condition.

The two steps of this algorithm are respectively named as the initialization and refinement step. The measure that is used in the algorithm is represented by IR and given by the formula below:



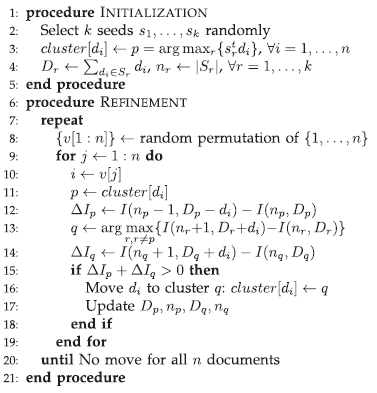
In the above equation α represents the regulating factor, where α can take any values between 0 and 1 (both inclusive). It is representative of the size of a cluster and is used to remove the excessive dependency that these algorithms have on the cluster size. The above formula is represented in the algorithm as the function I(x, y).

In the above formula the norm squared term represents the intra-cluster similarity and the dot product represents inter-cluster similarity. The measure IR is called the cluster size-weighted sum of average pair-wise similarities of documents in the same cluster. It is derived from the representative formula:



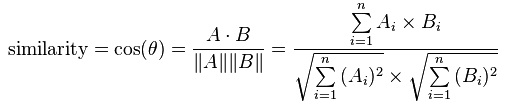
Where Sim(di, dj) indicates the MVS of the document pair in cluster Sr.

The algorithm for initialization of the clusters and the refinement of clusters is presented below as obtained from [1]:



## K-Means clustering Technique:

K-means is also a Partitional based clustering technique which uses cosine similarity measure to calculate the similarity between document and its centroid. It uses a single view point similarity measure which is centroid of the cluster. Cosine similarity is calculated as follows:



It takes k value as input parameter and partitions a set of n objects into k clusters. Cluster similarity is measured in regard to the mean value in cluster which can be viewed as centroid [4]. The algorithm is implemented as follows:

1. First, randomly selects k objects and makes them as initial centroids.
2. For each point find the nearest centroid and assign the point to the cluster associated with the nearest centroid.
3. Update the centroid of each cluster based on the objects in the cluster.
4. Repeat steps 2 & 3 till no object switches the clusters.

## MVS with Average Linkage Clustering Technique:

In a hierarchical algorithm no assumptions are made about the documents, therefore every document is assigned to its own cluster rather than starting with a random cluster initialization process. Then similarity measures between these clusters are used to find the cluster-pair closest to each other. They are merged and the process of finding the cluster-pair with lowest distance is repeated. This process of search and merge continues until all the documents have been grouped in the same cluster. I chose at what depth to stop the algorithm. The clusters available at that depth are the resultant clusters in that case.

In this case, the MVS measure of similarity is a much more robust method than the origin-distance based similarity. I took this one step further by using this in hierarchical agglomeration of the documents thus removing the dependency on the randomness of the initial cluster formation in other algorithms discussed above.

I believe this will yield more accurate clusters.

The similarity between two clusters in this algorithm is calculated using the formula:



Here DKL indicates the distance between the two clusters K and L. d(xi, xj) indicates the MVS measure between the documents xi and xj. The formula DKL looks at the average MVS of the document pair one-each from each cluster. The following illustrative steps show how the hierarchical agglomeration algorithm functions in theory:

Let us assume we have 5 documents indicated by d1, d2, … , d5.

**Step 1:** Initializing every document to its own cluster.

C1= {d1}, C2= {d2}, C3= {d3}, C4= {d4} and C5= {d5}

**Step 2:** Calculating DKL for all cluster pairs and finding the two with minimum distance (maximum DKL score). Let us assume they are clusters C2 and C5. We combine them to form a new cluster C2’ = {d2, d5}

**Step 3:** We now have the clusters C1= {d1}, C2’= {d2, d5}, C3= {d3}, C4= {d4}.

**Step 4:** Again the DKL parameter is calculated and in this round we find C3 and C4 are the closest. So the clusters are: C1= {d1}, C2’= {d2, d5}, C3’= {d3, d4}.

**Step 5:** Again DKL parameter is calculated and in this round we find C1 and C2’ to be the closest. So the clusters are: C1’= {d1, d2, d5}, C3’= {d3, d4}

**Step 6:** In the last step we find that C1’ and C3’ are the closest and we combine them to form:

C1’= {d1, d2, d5, d3, d4}.

Now in this chain we can terminate the algorithm at any step. If we want all documents to be in 2 clusters we stop after Step 5. If we want the documents in 3 clusters then we stop after Step 4.

So in this algorithm we decide the terminating condition which is the difference of total number of clusters and the number of ideal clusters. The algorithm can be represented in pseudo-code as follows:

**Procedure** INITIALIZATION

**for** i 🡨 1 to n

Clusters [i] 🡨 di

**end-for**

**end procedure**

**Procedure** HIERARCHICAL\_STEP

**While** (count (clusters[]) > desired number of clusters)

D[K][L] 🡨 calculate DKL score for all cluster pairs

Find the minimum of D[K][L]

The corresponding row and column of D[K][L] indicate the two clusters

which have minimum distance

The documents in clusters[col] are added to clusters[row] and clusters[col] is removed

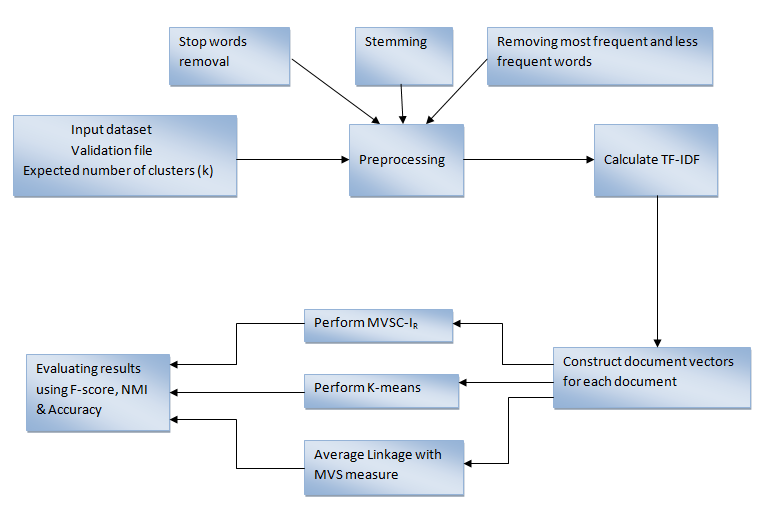
**end-while**

**end-procedure**

1. Project Description:

## Program Flow:

The block diagram of the project is as shown in the figure below followed by the description of the program flow.



The code requires input as an index file that will contain the name of all the files that will be considered as input for clustering. Then the number of ideal clusters as per the ground-truth assumed about the input documents. Also required is a validation file that contains the document groupings in each cluster as per the ground-truth or ideal conditions.

The index file is parsed to obtain all the file names and then assigned indices corresponding to each of the document names according to their order in index file.

We need to then perform several pre-processing steps that will clean up the input, removing elements that skew the output falsely like stop word removal, stemming, and removing most frequent and less frequent words.

The examples of this are words like ‘is’, ‘are’, ‘the’, etc, which are so common in the English language that they do not add any additional meaning to the documents and which may in fact mislead the clustering process by including the frequent terms that are not informative. These kinds of words are removed to reduce the processing load. This is called stop-words removal.

Secondly words like ‘fish’, ‘fishing’, ‘smooth’, ‘smoothly’, ‘smoothest’ are all derivatives of the word ‘fish’ and ‘smooth’ respectively. Hence for clustering purposes the meaning of a document is contributed by the root words only and the derivatives do not need to be considered separately. This is called stemming. We use the Porter Stemmer algorithm for this purpose [11].

Once these two steps are complete we also remove the words that occur in less than 2 documents and also in more than 99.5% of the documents. This is done as analysis has shown that too rarely occurring words and too frequently occurring words do not contribute positively towards the clustering process.

After these steps are performed we have only the list of unique words across all documents which will be used for the clustering process. The documents given as input should be represented in a way that can be mathematically operated upon. This purpose is served by representing the documents as document vectors. In each of the vectors each element is the TF-IDF (Term Frequency-Inverse Document Frequency) of the words occurring in that document.

Term Frequency is defined as the ratio of the number of times a word occurs in the document to the total number of words in the document. Inverse Document Frequency is defined as the ratio of the total number of documents in the input to the number of documents in which that particular word occurs.

The first algorithm run is MVSC-Ir. The initialization step is performed using the document matrices. After this the refinement step is performed with either one of two exit criterion. If the algorithm reaches convergence then the algorithm terminates. Otherwise there is a repeat limit assigned which when exceeded the algorithm will terminate. The calculation of variables required for performance measures related to MVSC-Ir are then calculated.

This is followed by the performance of the K-Means algorithm. Once again the algorithm is terminated after the number of runs as suggested by the repeat limit is exceeded or when it reaches the convergence. The variables related to the calculation of performance measures related to K-Means are then calculated.

The matrix of MVS similarity measures between every pair of documents in the input set is constructed. The hierarchical agglomeration algorithm is then executed. This algorithm terminates when we have the desired number of clusters as specified by the ideal number of clusters input to the system. The variables related to the calculation of performance measures for this algorithm are calculated.

The performance measures for all the algorithms are now calculated. The performance measures used in this project are F-Score, NMI and Accuracy. A description of these measures is presented in section 5.3.

There are event based calls to display the clusters formed by all three algorithms. There are also event based calls to display in a bar chart the comparison of the performance measures for each of the three algorithms. A bar chart is generated for each measure comparing the measure across all the three algorithms.

## Implementation Details:

**User Interface**

The user interface is made using JAVA Swing. A main re-sizable window is provided to the user to select the index file, to provide the ideal number of cluster expected and the validation file. There is a messages area which displays useful messages during the course of execution. There are five buttons provided to achieve the following functions:

1. Process: Invokes the parsing of input files, pre-processing, running of the three algorithms and calculation of the performance measures for each.
2. F-Score: Invokes rendering of a bar chart comparing the F-Score measure across all three algorithms.
3. NMI: Invokes rendering of a bar chart comparing the NMI measure for all the three algorithms.
4. Accuracy: Invokes rendering of a bar chart comparing the accuracy measures of all the three algorithms.
5. Clusters: Invokes the rendering of a window showing the clusters formed by all three algorithms. The documents are represented in the form of index numbers assigned while parsing the index file.

**Process functionality**

When the process button is clicked the actionPerformed() method for the process button is invoked. In this method all the required variables are initialized.

The parseClassInformation() method is then called where we parse the validations file and store in appropriate data structures the ideal cluster groupings and also the number of ideal clusters.

The process() method is called next. In this method first the index file is parsed to read all the input file names, and then we read every single file and store all the words in the documents in list data structures. After that we perform stop-word removal from this list. We then perform stemming of the words using Porter Stemmer algorithm.

Only the unique words are filtered from the list. Then we calculate the number of documents in which each of the words occurs. After that the words which occur too frequently and the words which occur rarely are removed from the list.

Vectors representing each of the documents are constructed using the TF-IDF measures. The data structure docNormVectors contains the document vectors representing each document. Henceforth we will be using these vectors for processing only and not refer back to the text based input files.

showContentParsing() is the method which is invoked next and this is where the actual implementation of the three algorithms is done.

**MVSC-Ir**

formInitialClusters() generates the random numbers which will be used as indices for the documents acting as the seeds to form the initial cluster. The actual initial clustering is done using the cosine similarity measure. The cluster variables, composite vectors for each clusters, composite vector for the entire input set, centroid for each cluster and centroid for the whole input set are calculated.

performRefinementStep() then performs the refinement algorithm steps using the initial clusters formed above. We repeat the process until convergence is reached or the maximum number iterations set by repeat\_limit are reached. At the end of this step finalClusters will have the clusters formed by this algorithm.

The populatenijir() method is called next to calculate the variables nij and fij required to calculate the performance measures for this algorithm.

**K-Means**

doKMeans() is the wrapper method that is called to perform the K-Means algorithm.

initializeClusterCentroidKMeans(),initializeClusterCounts(), initializeClusterMatrices() are used to initialize the data structures that will be used. assignInitialClusterCentroids() is used to randomly decide the documents that will act as the initial cluster centroids.

In every run, every document vector is taken and we determine the cluster whose centroid it is closest to using findClosestClusterCentroidIndex(). In this run, the document is assigned to this cluster.

The clusters formed in this run are compared with those formed in the previous run. If any documents change clusters between runs then the cluster centroids are recalculated and the process is repeated. If there are no documents that change clusters between the runs then the algorithm terminates and the resultant clusters are found in finalClustersKmeans structure. Otherwise the algorithm repeats until the limit set by repeat\_limit is reached. It then terminates and the resultant clusters are stored in the finalClusterKmeans structure.

populatenijkmeans() is then called to calculate the nij and fij variables required for calculation of the performance measures corresponding to the K-Means algorithm.

**Hierarchical Agglomeration Algorithm**

doHAC() is the wrapper method which is called to perform this algorithm.

initHACProcess() is the method where every document is assigned to its own cluster. The clustering information in each step is maintained in the structure HACMap.

findMinDKL() is then repeated until we have the desired number of clusters. In findMinDKL() the closest two clusters are found out using the DKL measure. The cluster with the higher index is merged into the cluster with the lower index. The cluster with the higher index is then removed from HACMap. The cluster arrangement changes after every iteration. This mandates recalculation of the MVS scores for the involved documents. Here by involved documents I mean those documents which are part of the newly formed cluster by merging two clusters which were closest to each other. This reduces processing load as only those rows related to the documents in the new cluster need to be changed and not the whole similarity matrix. This process is carried out in the method adjustHACSimilarityMatrix().

The resulting clusters are then stored in the finalClustersMVSCHAC structure.

populatenijmvschac() is then invoked to calculate nij and fij variables required to calculate the performance measures for the Hierarchical Agglomeration algorithm.

**Performance Measures**

calculateFScoreIr(), calculateNMIIr(), calculateAccuracyIr() are the methods invoked to calculate the performance measures F-Score, NMI and accuracy for the MVSC-IR algorithm. The corresponding functions are called for the remaining algorithms as well.

## Technical Challenges:

The main difficulty faced in implementing this project was the handling of the large number of input datasets in code and efficient parsing of the input files. Memory management was the biggest concern. Several times during initial development I ran out of heap space due to large memory requirements. The use of BufferedReader sped up the reading and parsing of the input files a lot faster.

Although the most efficient data structures have been used, the sheer volume of calculations make the application a memory intensive one. Hence a technical difficulty I faced was that it cannot be run effectively on standard desktop machines. So I switched to loki environment to resolve this issue.

# Experimentation Details:

## Documents Collection:

The Reuters-21578 [10] dataset is used as the input for this application. It is a research standard data set that Reuters provides and is now maintained by NIST. It is composed of several news articles or reports sent via newswire (telegram, telephone and other modes) by reporters to Reuters in 1987.

The Reuters set contains the data within sgm files from where we parse the information into formatted html files.

Each Reuters-21578 sgm files contain several datasets of the form:

<REUTERS TOPICS="YES" LEWISSPLIT="TRAIN" CGISPLIT="TRAINING-SET" OLDID="5544" NEWID="1">

<DATE>26-FEB-1987 15:01:01.79</DATE>

<TOPICS><D>cocoa</D></TOPICS>

<PLACES><D>el-salvador</D><D>usa</D><D>uruguay</D></PLACES>

<PEOPLE></PEOPLE>

<ORGS></ORGS>

<EXCHANGES></EXCHANGES>

<COMPANIES></COMPANIES>

<UNKNOWN>

&#5;&#5;&#5;C T&#22;&#22;&#1;f0704&#31;reuteu f BC-BAHIA-COCOA-REVIEW 02-26 0105</UNKNOWN><TEXT>&#2;<TITLE>BAHIA COCOA REVIEW</TITLE>

<DATELINE> SALVADOR, Feb 26 - </DATELINE><BODY>Showers continued throughout the week in

the Bahia cocoa zone, alleviating the drought since early January and improving prospects for the coming temporao, although normal humidity levels have not been restored, Comissaria Smith said in its weekly review. The dry period means the temporao will be late this year. Arrivals for the week ended February 22 were 155,221 bags of 60 kilos making a cumulative total for the season of 5.93 mln against 5.81 at the same stage last year. Again it seems that cocoa delivered earlier on consignment was included in the arrivals figures. Comissaria Smith said there is still some doubt as to how much old crop cocoa is still available as harvesting has practically come to an end. With total Bahia crop estimates

around 6.4 mln bags and sales standing at almost 6.2 mln there are a few hundred thousand bags still in the hands of farmers, middlemen, exporters and processors.

There are doubts as to how much of this cocoa would be fit for export as shippers are now experiencing dificulties in obtaining +Bahia superior+ certificates. In view of the lower quality over recent weeks farmers have sold a good part of their cocoa held on consignment. Comissaria Smith said spot bean prices rose to 340 to 350 cruzados per arroba of 15 kilos. Bean shippers were reluctant to offer nearby shipment and only limited sales were booked for March shipment at 1,750 to 1,780 dlrs per tonne to ports to be named. New crop sales were also light and all to open ports with June/July going at 1,850 and 1,880 dlrs and at 35 and 45 dlrs under New York july, Aug/Sept at 1,870, 1,875 and 1,880 dlrs per tonne FOB. Routine sales of butter were made. March/April sold at

4,340, 4,345 and 4,350 dlrs. April/May butter went at 2.27 times New York May, June/July

at 4,400 and 4,415 dlrs, Aug/Sept at 4,351 to 4,450 dlrs and at 2.27 and 2.28 times New York Sept and Oct/Dec at 4,480 dlrs and 2.27 times New York Dec, Comissaria Smith said. Destinations were the U.S., Covertible currency areas, Uruguay and open ports. Cake sales were registered at 785 to 995 dlrs for

March/April, 785 dlrs for May, 753 dlrs for Aug and 0.39 times New York Dec for Oct/Dec.

Buyers were the U.S., Argentina, Uruguay and convertible currency areas. Liquor sales were limited with March/April selling at 2,325 and 2,380 dlrs, June/July at 2,375 dlrs and at 1.25 times New

York July, Aug/Sept at 2,400 dlrs and at 1.25 times New York Sept and Oct/Dec at 1.25 times New York Dec, Comissaria Smith said. Total Bahia sales are currently estimated at 6.13 mln bags

against the 1986/87 crop and 1.06 mln bags against the 1987/88 crop. Final figures for the period to February 28 are expected to be published by the Brazilian Cocoa Trade Commission after

carnival which ends midday on February 27.

Reuter

&#3;</BODY></TEXT>

</REUTERS>

So we copy the text enclosed within the <BODY></BODY> tags as the content for the <body><h3></h3></body> portion of the corresponding html file. The categorizing labels according to reuters dataset is provided under <TOPICS><TOPICS>, <PLACES></PLACES>, <PEOPLE></PEOPLE>, <ORGS></ORGS>, <EXCHANGES></EXCHANGES>, <COMPANIES></COMPANIES> and <UNKNOWN></UNKNOWN> are used to form the <meta> tags in the corresponding html files. The text within <TITLE></TITLE> in the sgm files is used to form the <title></title> block in the corresponding html files.

For the purposes of clustering all the text in <title>, <meta> and <body> are considered. The groupings of all the html files are decided by the algorithm based on document vectors constructed from all valid words in each document and the resultant TF-IDF measures.

**Index.html file format**

*<html>*

*<head>*

*<title>index for 2000 files</title>*

*</head>*

*<body>*

*<h3></h3>*

*<table>*

*<tr>*

*<td><a href=reut2-000.sgm-index.html>reut2-000.sgm-index.html</a> </td>*

*</tr>*

*<tr>*

*<td><a href=reut2-000.sgm-0.html>reut2-000.sgm-0.html</a></td>*

*</tr>*

*…………………*

*………………..*

*</table>*

*</body>*

*</html>*

**Data file format**

*<html>*

*<head>*

*<title>bahia cocoa review</title>*

*<meta name="keywords" content="cocoa">*

*<meta name="keywords" content="el-salvador">*

*<meta name="keywords" content="usa">*

*<meta name="keywords" content="uruguay">*

*</head>*

*<body>*

*<h3>showers continued throughout the week in the bahia cocoa zone, alleviating the drought since early january and improving prospects for the coming temporao, although normal humidity levels have not been restored, comissaria smith said in its weekly review. the dry period means the temporao will be late this year. arrivals for the week ended february 22 were 155,221 bags of 60 kilos making a cumulative total for the season of 5.93 mln against 5.81 at the same stage last year. again it seems that cocoa delivered earlier on consignment was included in the arrivals figures. comissaria smith said there is still some doubt as to how much old crop cocoa is still available as harvesting has practically come to an end. with total bahia crop estimates around 6.4 mln bags and sales standing at almost 6.2 mln there are a few hundred thousand bags still in the hands of farmers, middlemen, exporters and processors. there are doubts as to how much of this cocoa would be fit for export as shippers are now experiencing dificulties in obtaining +bahia superior+ certificates. in view of the lower quality over recent weeks farmers have sold a good part of their cocoa held on consignment. comissaria smith said spot bean prices rose to 340 to 350 cruzados per arroba of 15 kilos. bean shippers were reluctant to offer nearby shipment and only limited sales were booked for march shipment at 1,750 to 1,780 dlrs per tonne to ports to be named. new crop sales were also light and all to open ports with june/july going at 1,850 and 1,880 dlrs and at 35 and 45 dlrs under new york july, aug/sept at 1,870, 1,875 and 1,880 dlrs per tonne fob. routine sales of butter were made. march/april sold at 4,340, 4,345 and 4,350 dlrs. april/may butter went at 2.27 times new york may, june/july at 4,400 and 4,415 dlrs, aug/sept at 4,351 to 4,450 dlrs and at 2.27 and 2.28 times new york sept and oct/dec at 4,480 dlrs and 2.27 times new york dec, comissaria smith said. destinations were the u.s., covertible currency areas, uruguay and open ports. cake sales were registered at 785 to 995 dlrs for march/april, 785 dlrs for may, 753 dlrs for aug and 0.39 times new york dec for oct/dec. buyers were the u.s., argentina, uruguay and convertible currency areas. liquor sales were limited with march/april selling at 2,325 and 2,380 dlrs, june/july at 2,375 dlrs and at 1.25 times new york july, aug/sept at 2,400 dlrs and at 1.25 times new york sept and oct/dec at 1.25 times new york dec, comissaria smith said. total bahia sales are currently estimated at 6.13 mln bags against the 1986/87 crop and 1.06 mln bags against the 1987/88 crop. final figures for the period to february 28 are expected to be published by the brazilian cocoa trade commission after carnival which ends midday on february 27. reuter &#3;empty file</h3>*

*</body>*

*</html>*

In the validations file each row represents a cluster. The file indices in that cluster form the content of the row. The indices are comma separated values.

Among various categories available in the Reuters dataset, I used Zinc, Propane, Livestock, Housing, Barley, Trade, Ship, Money-fx, Interest, Earn, Crude, Acq, Cocoa to create the different sizes of datasets. The details of datasets used to conduct experiment are shown in the figure below:

|  |  |  |  |
| --- | --- | --- | --- |
| Source | N | c | m |
| Reuters | 2000 | 13 | 4430 |
| Reuters | 4000 | 6 | 7394 |
| Reuters | 6000 | 7 | 8487 |
| Reuters | 8000 | 7 | 9008 |
| Reuters | 9100 | 13 | 9409 |

n: # of documents; c: # of classes; m: number of words

## Experiment Setup:

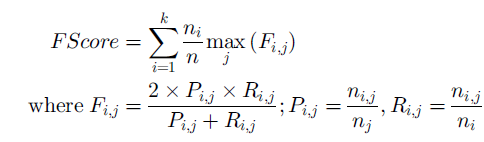
The ideal number of cluster input that is taken is used as the initial number of clusters for MVSC-IR and K-Means. It also serves as the stopping criteria for the hierarchical agglomeration algorithm.

The evaluation of the effectiveness of an algorithm in clustering the datasets is evaluated using the measures F-Score, NMI and accuracy which are discussed in next section. The average of 10 runs is presented as a single value for F-Score, NMI and Accuracy for all the algorithms as IR and k-means are initialization dependent. The application is run on the Loki system.

## Evaluation Measures:

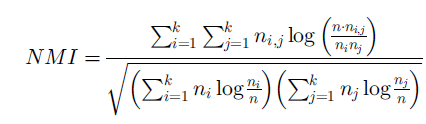
The performance of the three algorithms is compared against the given validation file. The cluster groupings in the validation file are taken as the ground-truth.

I consider three measures to evaluate the performance of the algorithms. They are F-score, NMI, and Accuracy. F-Score is the equally weighted combination of Recall (R) and Precision (P) measures used widely in document clustering. Its formula is given as:

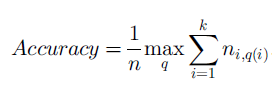


Where ni represents the number of documents in the ideal cluster i. nj represents the number of documents assigned to cluster j. nij indicates the number of documents shared between ideal cluster i and resultant cluster j. It gives us the idea of how capable the cluster process is of repeating same results. It gives us the idea that there are some documents in the set which do not belong but it does not give us any way of determining those documents. This disadvantage of F-Score is partly offset by the use of NMI.

Normalized Mutual Information (NMI) measures the information the true class partition and cluster assignment share. It is calculated using the following formula:



It measures how much knowing about the clusters helps us to know about true classes. Accuracy is a measure that allows us to gauge the purity of clustering. This means that it tells us how well the documents that are supposed to be in the same cluster as per the ground truth are actually in the same cluster in the resultant. The Hungarian Algorithm [12] is used to calculate the appropriate value of q(i) in the following formula used to derive accuracy:



All of the measures yield a value between and including 0 and 1. The greater the value of a measure, the better the algorithm.

## Screenshots:

The following (Fig 1.) is the opening screen when we execute the project. It needs three inputs to run the project which are index file that has links to all the input files, expected number of clusters and validation file to evaluate the performance of the algorithms.

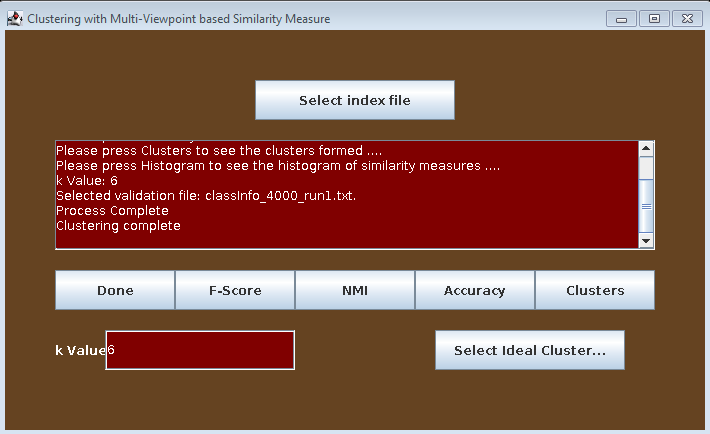


Fig.1 Screenshot of home screen

After passing all of the three inputs we need to hit the process button which preprocess the data, runs cluster algorithms and calculate scores for all the algorithms. Once all these steps are completed the processed information from each file is displayed as shown in Fig.2 below.

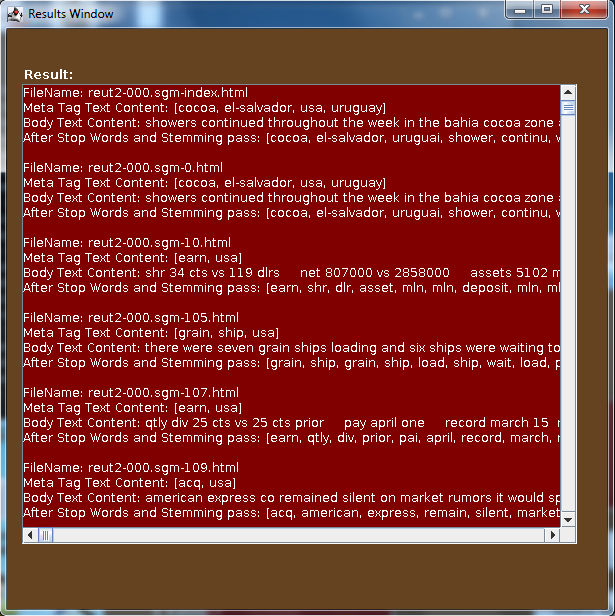


Fig.2 Screenshot of document content after parsing

When we hit the cluster button, it shows the clusters formed according to the index number for all the three algorithms as shown in the Fig.3.

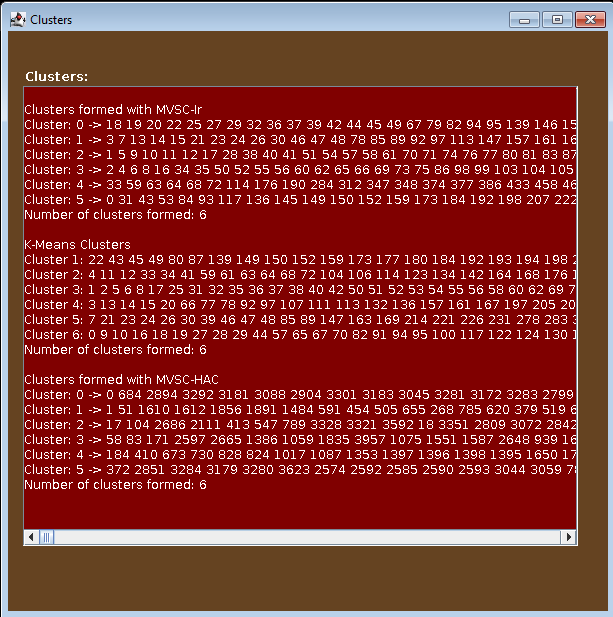


Fig.3 Screenshot of clusters formed

The graphs for F-score, NMI and Accuracy are displayed for all the three algorithms when we hit appropriate buttons. The screenshots of graphs for F-score, NMI, and Accuracy are shown in Fig 4, Fig.5 and Fig.6.

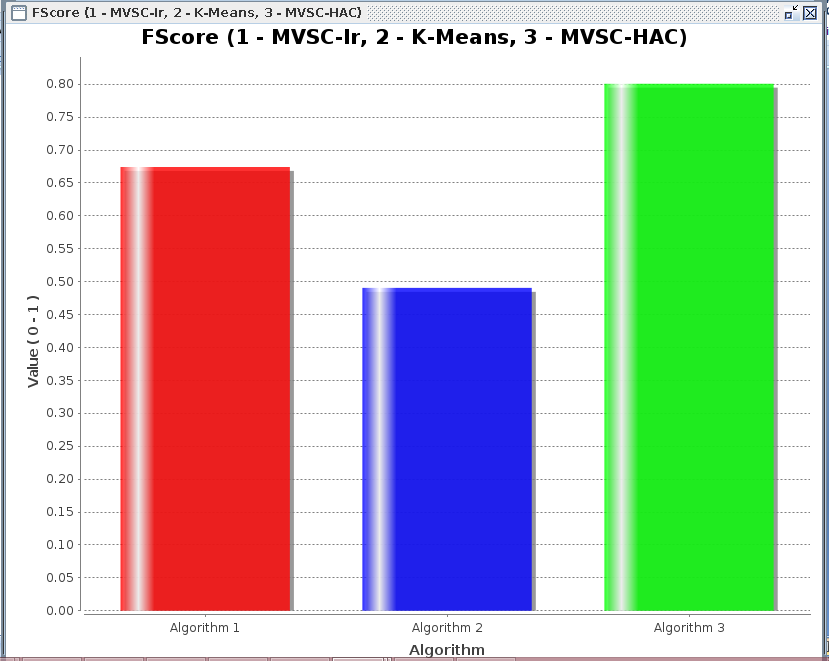


Fig.4 Screenshot of F-score

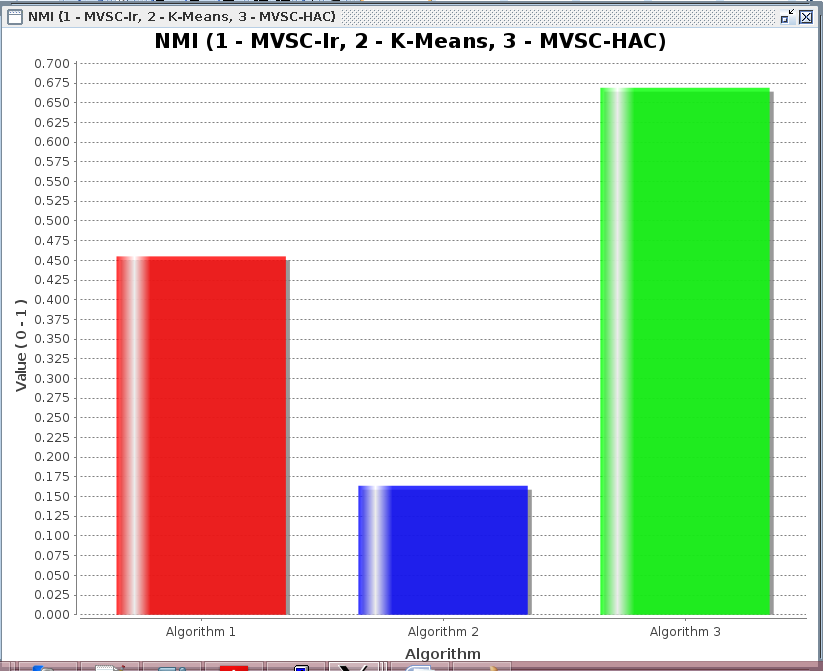


Fig.5 Screenshot of NMI

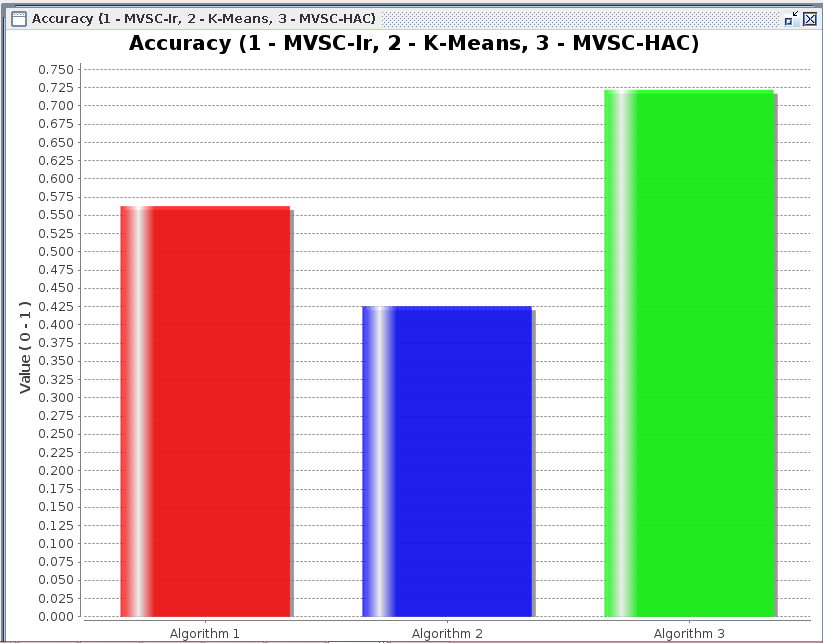


Fig.6 Screenshot for Accuracy

## Results:

I observed the results by using the complete input file and also using only a portion of file which is under meta-tags. The results are more accurate when I consider the complete file but it is taking more time when compared to meta-tags.

Cluster results based on F-score, NMI and accuracy are reported in table 1, 2 & 3 when whole document is taken into consideration. From the values, we can observe that the MVSC-HAC algorithm performs consistently well across all datasets.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Number of Datasets (n)** | **Ideal Number of Clusters (k)** | **F-Score**  **MVSC-IR** | **F-Score**  **K-Means** | **F-Score**  **MVS-Average Linkage** |
| 2000 | 13 | 0.64 | 0.47 | 0.87 |
| 4000 | 6 | 0.67 | 0.49 | 0.80 |
| 6000 | 7 | 0.74 | 0.52 | 0.87 |
| 8000 | 7 | 0.78 | 0.47 | 0.79 |
| 9100 | 13 | 0.65 | 0.49 | 0.72 |

Table 1. Clustering results in F-score

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Number of Datasets (n)** | **Ideal Number of Clusters (k)** | **NMI**  **MVSC- IR** | **NMI**  **K-Means** | **NMI**  **MVS-Average Linkage** |
| 2000 | 13 | 0.58 | 0.35 | 0.86 |
| 4000 | 6 | 0.46 | 0.17 | 0.67 |
| 6000 | 7 | 0.50 | 0.25 | 0.73 |
| 8000 | 7 | 0.58 | 0.26 | 0.69 |
| 9100 | 13 | 0.50 | 0.29 | 0.61 |

Table 2: Clustering results in NMI

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Number of Datasets (n)** | **Ideal Number of Clusters (k)** | **Accuracy**  **MVSC- IR** | **Accuracy**  **K-Means** | **Accuracy**  **MVS-Average Linkage** |
| 2000 | 13 | 0.60 | 0.43 | 0.82 |
| 4000 | 6 | 0.56 | 0.46 | 0.72 |
| 6000 | 7 | 0.71 | 0.47 | 0.83 |
| 8000 | 7 | 0.76 | 0.47 | 0.77 |
| 9100 | 13 | 0.60 | 0.42 | 0.68 |

Table3: Clustering results in Accuracy

Cluster results based on F-score, NMI and accuracy are reported in table 4, 5 & 6 when only part of document is considered. Here we can observe that MVS-Average linkage clustering showed better results than K-means and MVS-IR irrespective of the size of the document considered.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Number of Datasets (n)** | **Ideal Number of Clusters (k)** | **F-Score**  **MVSC-IR** | **F-Score**  **K-Means** | **F-Score**  **MVS-Average Linkage** |
| 2000 | 13 | 0.63 | 0.30 | 0.64 |
| 4000 | 6 | 0.65 | 0.47 | 0.74 |
| 6000 | 7 | 0.63 | 0.51 | 0.72 |
| 8000 | 7 | 0.51 | 0.46 | 0.65 |
| 9100 | 13 | 0.62 | 0.49 | 0.64 |

Table 4. Clustering results in F-score

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Number of Datasets (n)** | **Ideal Number of Clusters (k)** | **NMI**  **MVSC- IR** | **NMI**  **K-Means** | **NMI**  **MVS-Average Linkage** |
| 2000 | 13 | 0.55 | 0.16 | 0.58 |
| 4000 | 6 | 0.45 | 0.15 | 0.63 |
| 6000 | 7 | 0.47 | 0.21 | 0.48 |
| 8000 | 7 | 0.44 | 0.25 | 0.45 |
| 9100 | 13 | 0.42 | 0.27 | 0.44 |

Table 5: Clustering results in NMI

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Number of Datasets (n)** | **Ideal Number of Clusters (k)** | **Accuracy**  **MVSC- IR** | **Accuracy**  **K-Means** | **Accuracy**  **MVS-Average Linkage** |
| 2000 | 13 | 0.58 | 0.42 | 0.58 |
| 4000 | 6 | 0.54 | 0.44 | 0.68 |
| 6000 | 7 | 0.57 | 0.44 | 0.68 |
| 8000 | 7 | 0.58 | 0.43 | 0.61 |
| 9100 | 13 | 0.59 | 0.40 | 0.60 |

Table 6: Clustering results in Accuracy

The time taken to complete the process using only meta tags in file and also when using complete file among all datasets are shown in table 7.

|  |  |  |
| --- | --- | --- |
| Datasets(n) | Meta tag information | Whole text |
| 2000 | 0hr 6mins | 0hr 17minutes |
| 4000 | 0hr 56mins | 1hr 30minutes |
| 6000 | 3hrs 06mins | 5hrs 18minutes |
| 8000 | 5hrs 54mins | 8hrs 56minutes |
| 9100 | 8hrs 54mins | 12hrs 36minutes |

Table 7: time taken to complete the process

# Limitations of Project:

The time taken to finish the clustering process for MVS-Average linkage algorithm is more as it follows hierarchical agglomerative approach when compared to MVSC-IR and K-means which follow partitional clustering approach. It’s time complexity which is O(n2).

# Conclusions & Future Work:

In this project, I compared two clustering techniques namely, MVSC-IR and K-means which use different similarity measures. MVSC-IR uses similarity based on multiple viewpoints which are objects outside the clusters with the two documents being measured while the K-means uses single view point which is centroid. Both of them follow Partitional clustering approach. From experimental results, Multi view point similarity is potentially more suitable for text documents than the single view point measure. However, in Partitional clustering approach there are few basic drawbacks like sensitiveness to initialization and number of clusters expected in advance. With the sensitiveness to initialization, the results of the algorithm are unpredictable and may run for infinite loops during refinement of clusters. In order to random initialization, I embedded MVS measure with average linkage algorithm which follows Hierarchical approach. It forms potentially improved clusters than the other two algorithms. In addition to this, I also observed the performance of clustering techniques using part of the document to the whole text. The results are more accurate when whole document is taken into consideration.

The key contribution of this project is to embed MVS measure in average linkage algorithm. Future methods could make use of the same principle but define termination condition in terms of similarity score instead of having it on ‘K’ value which is the number of clusters expected. Also, they can try to embed MVS with other types of clustering approaches. I considered average of 10 runs to evaluate the results of the clustering algorithms. Future comparisons or improvements on this technique may take the results by considering probability instead of taking the average.

References:

1. Duc Thang Nguyen; Lihui Chen; Chee Keong Chan, "Clustering with Multiviewpoint-Based Similarity Measure," *Knowledge and Data Engineering, IEEE Transactions on*, vol.24, no.6, pp.988, 1001, June 2012.
2. A. Banerjee, I. Dhillon, J. Ghosh, and S. Sra, “Clustering on the Unit Hypersphere Using Von Mises-Fisher Distributions,” J. Machine Learning Research, vol. 6, pp. 1345-1382, Sept. 2005.
3. W.Xu, X.Liu and Y.Gong “Document Clustering based on Non Negative Matrix Factorization” Proceedings of the 26th annual international ACM SIGIR conference on Research and development in information retrieval pp. 267-273, 2003.
4. C.D. Manning, P. Raghavan, and H. Schu¨tze, An Introduction to Information Retrieval. Cambridge Univ. Press, 2009.
5. G. Karypis, “CLUTO a Clustering Toolkit,” technical report, Dept. of Computer Science, Univ. of Minnesota, http://glaros.dtc.umn.edu/~gkhome/views/cluto, 2003.
6. X. Wu, V. Kumar, J.R. Quinlan, J. Ghosh, Q. Yang, H. Motoda, G.J. McLachlan, A. Ng, B. Liu, P.S. Yu, Z.-H. Zhou, M. Steinbach, D.J. Hand, and D. Steinberg, “Top 10 Algorithms in Data Mining,” Knowledge Information Systems, vol. 14, no. 1, pp. 1-37, 2007
7. A. Strehl, J. Ghosh, and R. Mooney, “Impact of Similarity Measures on Web-Page Clustering,” Proc. 17th Nat’l Conf. Artificial Intelligence: Workshop of Artificial Intelligence for Web Search (AAAI), pp. 58-64, July 2000
8. Han and Kamber Data Mining: Concepts and Techniques, 2nd ed.
9. Christopher, Prabhakar, Hinrich An Introduction to Information Retrieval, 2009
10. David D. Lewis: ResearchReuters-21578 text categorization test collection Distribution 1.0, AT&T Labs – Research
11. <http://chianti.ucsd.edu/svn/csplugins/trunk/soc/layla/WordCloudPlugin/trunk/WordCloud/src/cytoscape/csplugins/wordcloud/Stemmer.java>
12. <https://github.com/KevinStern/software-and-algorithms/blob/master/src/main/java/blogspot/software_and_algorithms/stern_library/optimization/HungarianAlgorithm.java>
13. G. Karypis, E.H. Han, and V. Kumar. Chameleon: A hierarchical clustering algorithm using dynamic modeling. IEEE Computer, 32(8):68–75, 1999.