Document Clustering

Pankaj Jajoo

In partial fulfillment of the
requirements for the degree of

Master of Technology

Indian Institute of Technology
Kharagpur

2008

Under the Guidance of
Prof. Sudeshna Sarkar
Professor
Department of Computer Science & Engineering
Indian Institute of Technology Kharagpur
WB, India 721302

Abstract

Clustering is an automatic learning technique aimed at grouping a set of objects into subsets or clusters. The goal is to create clusters that are coherent internally, but substantially different from each other. In plain words, objects in the same cluster should be as similar as possible, whereas objects in one cluster should be as dissimilar as possible from objects in the other clusters.

Automatic document clustering has played an important role in many fields like information retrieval, data mining, etc. The aim of this thesis is to improve the efficiency and accuracy of document clustering. We discuss two clustering algorithms and the fields where these perform better than the known standard clustering algorithms.

The first approach is an improvement of the graph partitioning techniques used for document clustering. In this we preprocess the graph using a heuristic and then apply the standard graph partitioning algorithms. This improves the quality of clusters to a great extent.

The second approach is a completely different approach in which the words are clustered first and then the word cluster is used to cluster the documents. This reduces the noise in data and thus improves the quality of the clusters.

In both these approaches there are parameters which can be changed according to the dataset inorder to improve the quality and efficiency.

Table of Contents

1. Introduction
    1.1. Motivation
    1.2. Thesis statement
    1.3. Thesis outline
2. Background
    2.1. Clustering
    2.2. Clustering Applications
    2.3. Document Clustering
    2.4. Challenges in Document Clustering
3. Literature Study
    3.1. Clustering methods
        3.1.1. K-Means
        3.1.2. Expectation Maximization
        3.1.3. Hierarchical Clustering
    3.2. Preprocessing Techniques
        3.2.1. Term filtering
        3.2.2. Stemming
        3.2.3. Graph preprocessing
    3.3. Evaluation
        3.3.1. User Surveys
        3.3.2. Shanon’s Entropy
        3.3.3. F-Measure
        3.3.4. Overall Similarity
4. Triplet based graph partitioning
    4.1. Introduction
    4.2. Motivation
    4.3. Algorithm

4.3.1. Get Similarity

4.3.2. Graph Creation and Clustering

5. Feature Based clustering

5.1. Introduction

5.2. Motivation

5.3. Algorithm

5.3.1. Feature Extraction

5.3.2. Feature Clustering

5.3.3. Document Clustering

5.3.4. Multilevel Graph Partitioning Algorithm

5.3.5. Multilevel Graph Partitioning Tools

5.3.5.1. Metis

5.3.5.2. Graclus

6. Results

6.1. Datasets

6.1.1. 20 Newsgroups

6.1.2. Reuters – 21578

6.1.3. Keepmedia dataset

6.2. Results for Triplet based Graph Partitioning

6.2.1. 20 newsgroups dataset

6.2.2. Reuters – 21578 dataset

6.2.3. Keepmedia dataset

6.3. Results for Feature based clustering

6.3.1. 20 newsgroups dataset

6.3.2. Reuters – 21578 dataset

6.3.3. Keepmedia dataset

6.4. Results for K-Means clustering

6.4.1. 20 newsgroups dataset

6.4.2. Reuters – 21578 dataset

6.4.3. Keepmedia dataset

6.5 Result Analysis

7. Conclusion
8. References

Acknowledgement

With great pleasure and deep sense of gratitude, I express my indebtedness to Prof. Sudeshna Sarkar for her invaluable guidance and constant encouragement at each and every step of my project work. She exposed us to the intricacies of relevant topics through paper counseling and discussions and always showed great interest in providing timely support and suitable suggestions.

I would also like to express my gratitude to all my friends in the Department of Computer Science and my hostel for their constant support and encouragement. Words are not enough to express my gratitude towards my parents to whom I owe every success and achievements of my life. Their constant support and encouragement under all odds has brought me where I stand today.

Date: May 6th 2008

Pankaj Jajoo
03CS3024
Department of CSE
IIT Kharagpur

Chapter1

Introduction

Clustering is a division of data into groups of similar objects. Each group, called cluster, consists of objects that are similar between themselves and dissimilar to objects of other groups. In other words, the goal of a good document clustering scheme is to minimize intra-cluster distances between documents, while maximizing inter-cluster distances (using an appropriate distance measure between documents). A distance measure (or, dually, similarity measure) thus lies at the heart of document clustering.

Clustering is the most common form of unsupervised learning and this is the major difference between clustering and classification. No super-vision means that there is no human expert who has assigned documents to classes. In clustering, it is the distribution and makeup of the data that will determine cluster membership.

Clustering is sometimes erroneously referred to as automatic classification; however, this is inaccurate, since the clusters found are not known prior to processing whereas in case of classification the classes are pre-defined. In clustering, it is the distribution and the nature of data that will determine cluster membership, in opposition to the classification where the classifier learns the association between objects and classes from a so called training set, i.e. a set of data correctly labeled by hand, and then replicates the learnt behavior on unlabeled data.

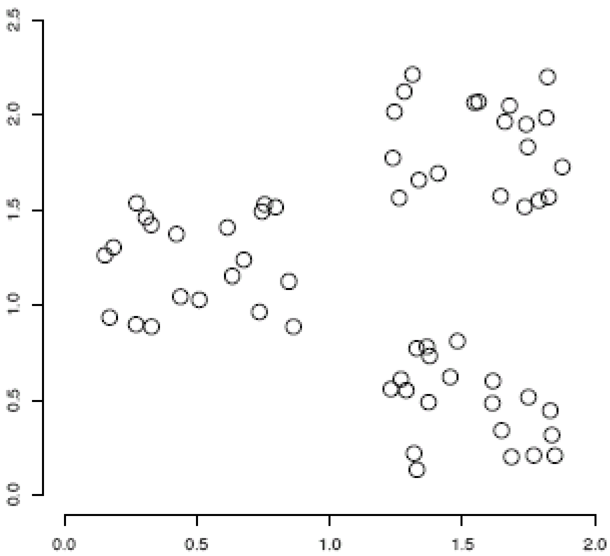


Figure1. An example of a data set with a clear cluster structure

1.1 Motivation

We were developing an application for recommendations of news articles to the readers of a news portal. The following challenges gave us the motivation to use clustering of the news articles:

1. The number of available articles was large.
2. A large number of articles were added each day.
3. Articles corresponding to same news were added from different sources.
4. The recommendations had to be generated and updated in real time.

By clustering the articles we could reduce our domain of search for recommendations as most of the users had interest in the news corresponding to a few number of clusters. This improved our time efficiency to a great extent. Also we could identify the articles of same news from different sources.

The main motivation of this work has been to investigate possibilities for the improvement of the effectiveness of document clustering by finding out the main reasons of ineffectiveness of the already built algorithms and get their solutions.

Initially we applied the K-Means and Agglomerative Hierarchical clustering methods on the data and found that the results were not very satisfactory and the main reason for this was the noise in the graph, created for the data. Thus we tried for pre-processing of the graph to remove the extra edges. We applied a heuristic for removing the inter cluster edges and then applied the standard graph clustering methods to get much better results.

We also tried a completely different approach by first clustering the words of the documents by using a standard clustering approach and thus reducing the noise and then using this word cluster to cluster the documents. We found that this also gave better results than the classical K-Means and Agglomerative Hierarchical clustering methods.

1.2 Thesis Statement

The statement of this thesis is that the quality of document clustering can be improved by reducing the noise in the data by pre-processing the structure of data representation and also by applying some new clustering techniques.

We first study the effectiveness of pre-processing of data representation and then applying the classical clustering methods. We then detect the effectiveness of a new clustering algorithm in which the noise is reduced by first clustering the features of the data and then clustering the data on the basis of their feature's clusters.

1.3 Thesis Outline

This thesis is organized into 7 chapters. A brief outline of the concepts of remaining chapters follows:

Chapter 2 This gives the background of the clustering technique, its applications and the challenges faced in this problem.

Chapter 3 This contains brief overview of the methods that have been applied till date for solving the problem of document clustering and also the evaluation measures that are applied to compare two methods.

Chapter 4 This contains the details of the Triplet based graph partitioning algorithm including the motivation behind the algorithm.

Chapter 5 This contains the details of the Feature based clustering approach.

Chapter 6 This contains the results of the implementation of the above two algorithms and the standard k-Means algorithm applied on three datasets. It also contains the analysis of the results.

Chapter 7 This contains the final conclusion of the thesis and the future work that can be done using the results of this thesis.

Chapter2

Background

2.1 Clustering

A general definition of clustering stated by Brian Everitt et al. [6]

Given a number of objects or individuals, each of which is described by a set of numerical measures, devise a classification scheme for grouping the objects into a number of classes such that objects within classes are similar in some respect and unlike those from other classes. The number of classes and the characteristics of each class are to be determined.

The clustering problem can be formalized as an optimization problem, i.e. the minimization or maximization of a function subject to a set of constraints. The goal of clustering can be defined as follows:

Given

I. a dataset X = {x1, x2, …. , xn}

II. the desired number of clusters k

III. a function f that evaluates the quality of clustering

we want to compute a mapping

$$\gamma : \{1, 2, ..., n\} \longrightarrow \{1, 2, ..., k\}$$

that minimizes the function f subject to some constraints.

The function f that evaluates the clustering quality are often defined in terms of similarity between objects and it is also called distortion function or divergence. The similarity measure is the key input to a clustering algorithm.

2.2 Clustering Applications

Clustering is the most common form of unsupervised learning and is a major tool in a number of applications in many fields of business and science. Hereby, we summarize the basic directions in which clustering is used.

- **Finding Similar Documents** This feature is often used when the user has spotted one "good" document in a search result and wants more-like-this. The interesting property here is that clustering is able to discover documents that are conceptually alike in contrast to search-based approaches that are only able to discover whether the documents share many of the same words.
- **Organizing Large Document Collections** Document retrieval focuses on finding documents relevant to a particular query, but it fails to solve the problem of making sense of a large number of uncategorized documents. The challenge here is to organize these documents in a taxonomy identical to the one humans would create given enough time and use it as a browsing interface to the original collection of documents.
- **Duplicate Content Detection** In many applications there is a need to find duplicates or near-duplicates in a large number of documents. Clustering is employed for plagiarism detection, grouping of related news stories and to reorder search results rankings (to assure higher diversity among the topmost documents). Note that in such applications the description of clusters is rarely needed.
- **Recommendation System** In this application a user is recommended articles based on the articles the user has already read. Clustering of the articles makes it possible in real time and improves the quality a lot.
- **Search Optimization** Clustering helps a lot in improving the quality and efficiency of search engines as the user query can be first compared to the clusters instead of comparing it directly to the documents and the search results can also be arranged easily.

2.3 Document Clustering

The goal of a document clustering scheme is to minimize intra-cluster distances between documents, while maximizing inter-cluster distances (using an appropriate distance measure between documents). A distance measure (or, dually, similarity measure) thus lies at the heart of document clustering. The large variety of documents makes it almost impossible to create a general algorithm which can work best in case of all kinds of datasets.

2.4 Challenges in Document Clustering

Document clustering is being studied from many decades but still it is far from a trivial and solved problem. The challenges are:

1. Selecting appropriate features of the documents that should be used for clustering.
2. Selecting an appropriate similarity measure between documents.
3. Selecting an appropriate clustering method utilising the above similarity measure.
4. Implementing the clustering algorithm in an efficient way that makes it feasible in terms of required memory and CPU resources.
5. Finding ways of assessing the quality of the performed clustering.

Furthermore, with medium to large document collections (10,000+ documents), the number of term-document relations is fairly high (millions+), and the computational complexity of the algorithm applied is thus a central factor in whether it is feasible for real-life applications. If a dense matrix is constructed to represent term-document relations, this matrix could easily become too large to keep in memory - e.g. 100, 000 documents × 100, 000 terms = 10^{10} entries ~ 40 GB using 32-bit floating point values. If the vector model is applied, the dimensionality of the resulting vector space will likewise be quite high (10,000+). This means that simple operations, like finding the Euclidean distance between two documents in the vector space, become time consuming tasks.

Chapter3

Literature Study

It is important to emphasize that getting from a collection of documents to a clustering of the collection, is not merely a single operation, but is more a process in multiple stages. These stages include more traditional information retrieval operations such as crawling, indexing, weighting, filtering etc. Some of these other processes are central to the quality and performance of most clustering algorithms, and it is thus necessary to consider these stages together with a given clustering algorithm to harness its true potential. We will give a brief overview of the clustering process, before we begin our literature study and analysis.

We have divided the offline clustering process into the four stages outlined below:

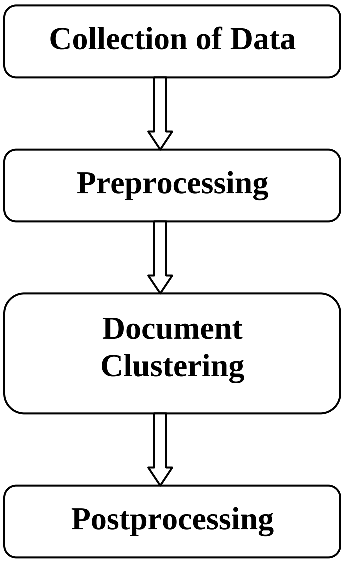


Figure2. The Stages of the Process of Clustering

Collection of Data includes the processes like crawling, indexing, filtering etc. which are used to collect the documents that needs to be clustered, index them to store and retrieve in a better way, and filter them to remove the extra data , for example, stopwords.

Preprocessing is done to represent the data in a form that can be used for clustering. There are many ways of representing the documents like, Vector-Model, graphical model, etc. Many measures are also used for weighing the documents and their similarities.

Document Clustering is the main focus of this thesis and will be discussed in detail.

Postprocessing includes the major applications in which the document clustering is used, for example, the recommendation application which uses the results of clustering for recommending news articles to the users.

3.1 Clustering Methods

3.1.1 K-Means

K-means is the most important flat clustering algorithm. The objective function of K-means is to minimize the average squared distance of objects from their cluster centers, where a cluster center is defined as the mean or centroid μ of the objects in a cluster C:

$$\vec{\mu}(\mathrm{C}) = \frac{1}{|C|}\sum_{\vec{x}\in C}\vec{x}$$

The ideal cluster in K-means is a sphere with the centroid as its center of gravity. Ideally, the clusters should not overlap. A measure of how well the centroids represent the members of their clusters is the Residual Sum of Squares (RSS), the squared distance of each vector from its centroid summed over all vectors

$$\mathrm{RSS}_\mathrm{i} = \sum_{\vec{x}\in C_i} \| \vec{x} - \vec{\mu}(C_i) \|^2$$

$$\mathrm{RSS} = \sum_{i=1}^{K} RSS_i$$

K-means can start with selecting as initial clusters centers K randomly chosen objects, namely the seeds. It then moves the cluster centers around in space in order to minimize RSS. This is done iteratively by repeating two steps until a stopping criterion is met

1. reassigning objects to the cluster with closest centroid
2. recomputing each centroid based on the current members of its cluster.

We can use one of the following termination conditions as stopping criterion

- A fixed number of iterations I has been completed.
- Centroids μ_i do not change between iterations.
- Terminate when RSS falls below a pre-estabilished threshold.

Algorithm for K-Means

```
procedure KMEANS(X,K)
    {s1, s2, · · · , sk}   SelectRandomSeeds(K,X)
    for i ←1,K do
        μ(Ci) ← si
    end for
    repeat
        $\min_{k\sim x_n - -\sim\mu(C_k)k}$ $C_k = C_k [ \{\sim x_n\}$
        for all $C_k$ do
            $\mu(C_k) = 1$
        end for
    until stopping criterion is met
end procedure
```

3.1.2 Expectation Maximization

The EM algorithm fall within a subcategory of the flat clustering algorithms, called Model-based clustering. The model-based clustering assumes that data were generated by a model and then tries to recover the original model from the data. This model then defines clusters and the cluster membership of data.

The EM algorithm is a generalization of K-Means algorithm in which the set of K centroids as the model that generate the data. It alternates between an *expectation step*, corresponding to reassignment, and a *maximization step*, corresponding to recomputation of the parameters of the model.

3.1.3 Hierarchical Clustering

Hierarchical clustering approaches attempt to create a hierarchical decomposition of the given document collection thus achieving a hierarchical structure. Hierarchical methods are usually classified into Agglomerative and Divisive methods depending on how the hierarchy is constructed.

Agglomerative methods start with an initial clustering of the term space, where all documents are considered representing a separate cluster. The closest clusters using a given inter-cluster similarity measure are then merged continuously until only 1 cluster or a predefined number of clusters remain.

Simple Agglomerative Clustering Algorithm:

1. Compute the similarity between all pairs of clusters i.e. calculate a similarity matrix whose ijth entry gives the similarity between the ith and jth clusters.
2. Merge the most similar (closest) two clusters.
3. Update the similarity matrix to reflect the pairwise similarity between the new cluster and the original clusters.
4. Repeat steps 2 and 3 until only a single cluster remains.

Divisive clustering algorithms start with a single cluster containing all documents. It then continuously divides clusters until all documents are contained in their own cluster or a predefined number of clusters are found.

Agglomerative algorithms are usually classified according to the inter-cluster similarity measure they use. The most popular of these are single-link, complete-link and group average. In the *single link* method, the distance between clusters is the minimum distance between any pair of elements drawn from these clusters (one from each), in the *complete link* it is the maximum distance and in the *average link* it is correspondingly an average distance

3.2 Preprocessing Techniques

Most of the clustering methods depend on various preprocessing techniques to achieve optimal quality and performance. We discuss here some of the common preprocessing methods.

3.2.1 Term Filtering

The removal of stopwords is the most common term filtering technique used. There are standard stopword lists available but in most of the applications these are modified depending on the quality of the dataset. Some other term filtering methods are:

- Removal of terms with low document frequencies. This is done to improve the speed and memory consumption of the application.
- Numbers do not play much importance in the similarities of the documents except dates and postal codes. Thus these can also be removed.

3.2.2 Stemming

Stemming is the process of reducing words to their stem or root form. For example ‘cook’, ’cooking’, ‘cooked’ are all forms of the same word used in different constraint but for measuring similarity these should be considered same.

3.2.3 Graph preprocessing

The algorithms using the graphs of documents or features require preprocessing of the graph inorder to improve the quality and time efficiency. Some simple graph preprocessing techniques include removal of edges having weight lower than threshold, removal of nodes which are not connected to any other nodes, etc.

3.3 Evaluation

One of the most important issues in clusters analysis is the evaluation of the clustering results. Evaluating clustering results is the analysis of the output to understand how well it reproduces the original structure of the data. However, the evaluation of clustering results is the most difficult task within the whole clustering workflow.

The ways of evaluation are divided in two parts:

1. Internal quality measure
2. External Quality measure

In internal quality measures, the overall similarity measure is used based on the pair wise similarity of documents and there is no external knowledge to be used.

For external quality measure some external knowledge for the data is required

Following are some of the measures used for evaluation

3.3.1 User Surveys

User surveys are a very common external measure of evaluating clustering algorithms and often the only one possible. Unfortunately, a number of elements speak against this method of evaluation.

- It is difficult to find a significantly large and representative group of evaluators. When the users are familiar with the subject (like computer science students or fellow scientists), their judgment is often biased. On the other hand, people not familiar with clustering and used to regular search engines have difficulty adjusting to a different type of search interface.
- People are rarely consistent in what they perceive as "good" clusters or clustering.
- User surveys usually take a long time and effort in both preparation of the experiment and its practical realization.

3.3.2 Shanon's Entropy

Entropy is often used to express the disorder of objects within the cluster with information-theoretic terms. For each cluster, the category distribution of data is calculated first i.e p_{ij} is the probability that a member of cluster j belongs to category i. Then the entropy of each cluster j is calculated as

$$E_j = -\sum_i p_{ij} \log(p_{ij})$$

The total entropy is calculated as the sum of the entropies of each cluster weighted by the size of each cluster:

$$E_{en} = \sum_{j=1}^{m} \frac{n_j * E_j}{n}$$

Where m is the total number of clusters, n_j is the size of j^{th} cluster and n is the total number of documents

3.3.3 F-measure

This is an aggregation of precision and recall, here adopted to clustering evaluation purposes. *Precision* is the ratio of the number of relevant documents to the total number of documents retrieved for a query. *Recall* is the ratio of the number of relevant documents retrieved for a query to the total number of relevant documents in the entire collection. In terms of evaluating clustering, the f-measure of each single cluster c_i is:

$$F(c_i) = \max_{j=1....m} \frac{2P_jR_j}{P_j + R_j}$$

where

$$P_j = \frac{|c_i \cap k_j|}{|k_j|}, \quad R_j = \frac{|c_i \cap k_j|}{|c_j|}$$

The final F-measure for the entire set is given as :

$$F = \sum_{i=1}^{m} F(i) \frac{|c_i|}{N}$$

where N is the total number of documents.

Higher value of F-measure indicates better clustering.

3.3.4 Overall Similarity

This is an internal quality measure when no external information is available. In this case the cohesiveness of clusters can be used as a measure of cluster similarity. One method for computing the cluster cohesiveness is to use the weighted similarity of the internal cluster similarity.

Chapter4

Triplet based graph partitioning

4.1 Introduction

Many graph partitioning algorithms have been used for document clustering. The major features of any such algorithm are:

1. **Document Representation**

   The most common way of representing the documents is as a set of keywords, where the keywords can be simple words or word phrases obtained using part-of-speech tagging, named entity recognition, etc. In some cases the documents are also represented as vectors of features, where the features can be the names of entities, places, etc.

2. **Similarity measure**

   This is the most important part of such algorithms which mainly differentiates them. There are many standard similarity measures but in most of the applications it is modified depending upon the input data and output required.

   Some of the common similarity measures are:

   a. Cosine vector similarity
   b. Euclidian distance

3. **Graph preprocessing**

   The graph initially created is a clique with an edge between every pair of nodes, where a node corresponds to a document and en edge-weight is the similarity between the two documents. There are two major problems with such a clique:

   1. It is a very dense graph and thus the time efficiency of the partitioning algorithm reduces to a high extent.
   2. It contains a lot of noise added by the extra, non-required edges.

   A lot of heuristics are applied inorder to sparse the graph and reduce the noise. One of the most common and simple example of such a heuristic is to remove all the edges having weight less than a threshold value. Many complex heuristics are also applied.

4. **Graph partitioning**

   Graph partitioning is an NP-complete problem and thus has no perfect solution but there are many heuristic algorithms available which partition the graph into required number of clusters in polynomial time.

5. **Postprocessing**

   Many applications require some post-processing of the obtained clusters, for example, in case of the recommendation engine there is no need of clusters with single or very few number of documents and thus such clusters are merged applying other heuristics.

4.2 Motivation

The available graph partitioning algorithms are very efficient but when applied for document clustering they do not produce good results. The major reason for this is that the graph created is not very optimum. The two major problems in the graph are:

1. There are many extra edges in the graph having low edge weight. Individually they do not create much noise but together they cause a lot of noise and reduce the quality of partitioning.
2. Some of the documents are very general and thus are very similar to documents of various categories. They create a lot of inter-cluster edges, which should not be present for an optimum graph.

The first problem can be solved by applying a threshold for the edge-weight and removing the edges having weight less than the threshold. It is the second problem which gave us the motivation to think for a heuristic which can be used for removing the edges created by the general graphs while maintaining the intra-cluster edges.

4.3 Algorithm

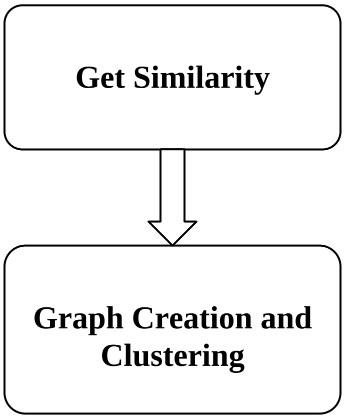


Figure 3. Triplet based graph partitioning algorithm

4.3.1 Get Similarity

Similarity between every pair of documents is calculated.

For any two documents X and Y, where X and Y are the sets of unique words in the documents except the stopwords, the similarity is defined as :

$$\mathrm{Sim(X,Y)} = \frac{|X \cap Y|}{\min.(|X|,|Y|)}$$

4.3.2 Graph Creation and Clustering

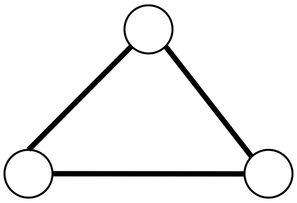
This is the most important part of the algorithm. In this, a graph is created with every document as a node and the edges are drawn using following algorithm:

1. Draw an edge between two documents, i and j, if there exists a third document, k , such that Sim(i,k) >= *Threshold* and Sim(j,k) >= *Threshold*, where *Threshold* is a value between 0 and 1.
2. Take Sim(i,j) as the edge weight.

3. Now, in this graph, keep an edge between two nodes, i and j, if there exists a third node k, such that edge-weight(i,k) >= *Threshold* and edge-weight(k,j) >= *Threshold* and edge-weight(i,j) > 0.
4. Cluster the above graph, with each edge considered equal weighted, using a standard graph clustering algorithm.
5. If some documents remain then reapply the algorithm with a lower value of *Threshold* and each created cluster as a node.
6. Merge the clusters if less number of clusters are required.

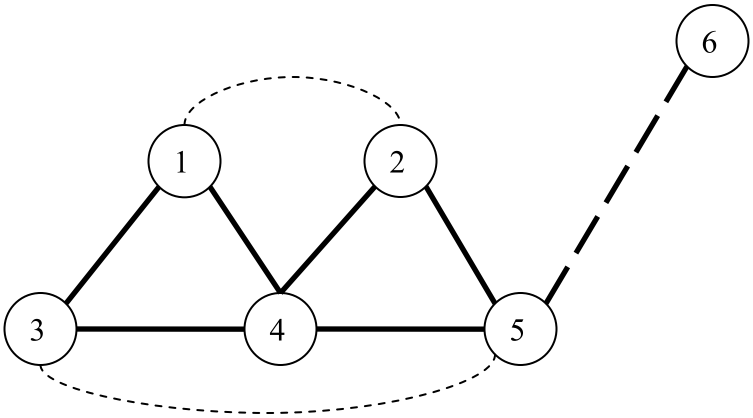
The heuristic used is that here the triplets or the triangles with all the three edges having weight great than the threshold are considered as the basis of clusters. After the steps 1-3 the graph formed will have two types of edges:

1. The edges which are part of a triangular clique, of which all the three edges have weight greater than threshold.



2. The edge is between two nodes which are part of two triangular cliques, having a common node and edge weights greater than the threshold. The edge weight of this node may not be greater than threshold but this is important because it increases the density of the cluster.

   For example, in the graph shown in figure below, the edges 1-2 and 3-5 will be part of the remaining edges even if their edge-weights are less than the threshold because they join two triangles having common node and high edge-weights and the edge 5-6 will not be a part of remaining edges even if its edge-weight is greater than the threshold.


1
2
3
4
5
6

Chapter5

Feature Based Clustering

5.1 Introduction

Two types of clustering have been studied - clustering the documents on the basis of the distributions of words that co-occur in the documents, and clustering the words using the distributions of the documents in which they occur. In this algorithm I have used a *double-clustering* approach in which I first cluster the words and then use this word-cluster to cluster the documents. The clustering of words reduces the feature space and thus reduces the noise and increases the time efficiency.

In general, this algorithm can be used for clustering of objects based on their features. A recently introduced principle, termed the *information bottleneck method* is based on the following simple idea. Given the empirical joint distribution of two variables, one variable is compressed so that the mutual information about the other is preserved as much as possible. Here the two variables are the object and the features. First, the features are clustered to preserve the information of objects and then these clusters are used to reduce the noise in the object graph.

The main advantage of this procedure lies in a significant reduction of the inevitable noise of the original co-occurrence matrix, due to its very high dimension. The reduced matrix, based on the word-clusters, is denser and more robust, providing a better reflection of the inherent structure of the document corpus.

5.2 Motivation

Clustering is being studied since a long time, and many state-of-art algorithms have been applied till date but still the results are not very satisfactory and we are looking for some better algorithms. This gave us the motivation to think out of box and try something simple but different. While calculating the similarity between the documents we first tried to use synonyms of the words also as same words but in case of documents like news articles, its not the synonymy but the co-occurrence of words which plays importance in the similarity. For example the words like ‘Bill Gates’ and ‘Microsoft’ are not synonyms but the news articles containing these words will belong to same clusters and this can be found out from the frequency of co-occurrence of these words. This gave us the motivation to first cluster the words based on their co-occurrence in the given dataset and then use this cluster to cluster the documents.

5.3 Algorithm

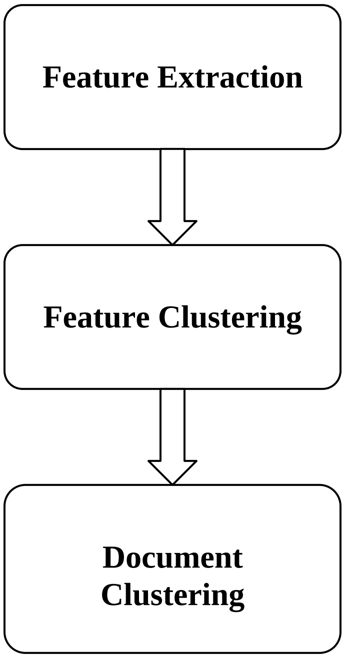


Figure4. Feature based Clustering Approach used for Clustering of Documents

5.3.1 Feature Extraction

This is used for extraction of features (important words and phrases in this case) from the documents. We have used Named-Entity tagger and frequency of unigrams and bigrams to extract the important words from the documents.

5.3.2 Feature Clustering

This is the most important phase in which the extracted features are clustered based on their co-occurrence. For this we tried many algorithms and found Multi-level graph clustering algorithms to be best for large data set as it reduces the time taken to a large extent.

5.3.3 Document Clustering

This is the final phase in which documents are clustered using the feature clusters. For this we have used a simple approach in which a document is assigned to the cluster of words of which it has the maximum words.

5.3.4 Multi-level Graph Partitioning Algorithm

The multilevel algorithms are graph clustering algorithms which take a graph as input in which an edge defines the similarity between two nodes it is connecting. Based on these similarities it clusters the nodes.

The overview of a multilevel algorithm is this:

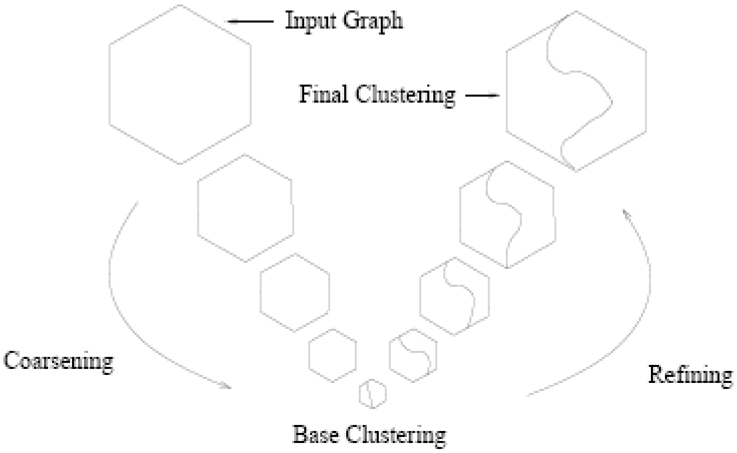


Figure5. Multi-level graph partitioning algorithm

The three phases are:

1. *Coarsening*

   In the coarsening phase the graph is repeatedly transformed into smaller and smaller graphs by combining set of nodes to form supernodes. When combining a set of nodes into a single supernode, the edge weights out of the supernode are taken to be the sum of the edge weights out of the original nodes.

2. *Base Clustering*

   The graph is coarsened until it becomes small enough to be clustered easily and effectively. At this point base clustering is performed by directly clustering the coarsened graph. The algorithms used for base clustering are the usual graph clustering algorithms.

3. *Refining*

   In the refinement phase the clustered initial graph is gained by separating the nodes which were combined in the coarsening phase. Given graph G_i , the graph G_{i-1} is obtained which is the graph used in level i-1 of the coarsening phase.

   The clustering in G_i induces a clustering in G_{i-1} as follows:

   if a supernode in G_i is in cluster c, then all nodes in G_{i-1} formed from that supernode are in cluster c. This yields an initial clustering for the graph G_{i-1}, which is then improved using a refinement algorithm.

5.3.5 Multi-level Graph Partitioning tools

The two most efficient tools using multi-level graph partitioning algorithm are:

1. *Metis*
2. *Graclus*

5.3.5.1 Metis

The algorithms used in Metis for different phases are:

1. Coarsening

   The coarsening approach used in Metis is as follows:

   Given a graph, start with all nodes unmarked. Visit each vertex in a random order. For each vertex x, if x is not marked, merge x with the unmarked vertex y that corresponds to the highest edge weight among all edges between x and unmarked vertices. Then mark x and y. If all neighbors of x have been marked, mark x and do not merge it with any vertex. Once all vertices are marked, the coarsening for this level is complete.

2. Base Clustering

   Metis uses region growing algorithm in which random nodes are selected and regions are grown around these nodes in a breadth-first fashion to form clusters. This method is run several times by selecting different random nodes and the best cluster is selected.

3. Refining

   Metis uses weighted kernel k-means algorithm as refinement algorithm. The initialization for the weighted kernel k-means is taken to be the clustering induced by the previous level. This algorithm converges quickly because the initial clustering is good at each level.

5.3.5.2 Graclus

The algorithms used in Graclus for different phases are:

1. Coarsening

   Given a vertex x, instead of merging using the criterion of heavy edges, it looks for the unmarked vertex y that maximizes

   $$\frac{e(x,y)}{w(x)}+\frac{e(x,y)}{w(y)}$$

   Where $e(x,y)$ is the edge weight and $w(x)$ & $w(y)$ are weights of nodes.

2. Base Clustering

   Graclus uses spectral clustering algorithm for base clustering.

3. Refining

   Graclus also uses weighted kernel k-means algorithm as refinement algorithm.

Both Metis and Graclus are efficient and quick algorithms and are very effective for large data. It depends on the kind of data that which of these work better.

Chapter6

Results

We have produced for three datasets using three algorithms. The first two algorithms are the algorithms proposed by us to be better and the third one is the standard K-Means algorithm. The results for the K-Means algorithm have been generated to compare them with the proposed algorithms.

6.1 Datasets

We have used three datasets

1. 20 newsgroups
2. Reuters
3. Keepmedia

6.1.1 20 newsgroups

This is a very standard and popular dataset used for evaluation of many text applications, data mining methods, machine learning methods, etc.

Its details are as follows:

- Number of unique documents = 18,828
- Number of categories = 20
- Number of unique words after removing the stopwords = 71,830

6.1.2 Reuters -21578

This is the most common dataset used for evaluation of document categorization and clustering.

Its details are as follows:

- Number of unique documents = 19715
- Number of categories = 5
- Number of unique words after removing the stopwords = 39,096

It contains many sub-categories also but for this experiment I am using only the broad categories.

6.1.3 Keepmedia dataset

This is a set of news articles provided by a company.

Its details are as follows:

- Number of unique documents = 62,239
- Number of categories = 69
- Number of unique words after removing the stopwords = 3,36,656

6.2 Results for Triplet based graph partitioning

In this case, two types of results have been generated based on two parameters for each dataset. The two parameters are:

1. Similarity threshold: This is the threshold given in the algorithm described in section 4.3.2.
2. Number of clusters

The entropy has been calculated for two result sets:

1. One is for only those documents which are contained in the graph generated after the preprocessing.
2. Second is for the complete dataset in which the remaining documents are added simply to the clusters they are closest to.

6.2.1 20 newsgroups dataset

Similarity threshold = 0.5

Number of extracted documents = 12,542

Number of clusters	Entropy for only the extracted documents	Entropy after adding the remaining documents
10	2.76	2.89
50	2.67	2.78
100	2.64	2.73
200	2.53	2.67
500	2.4	2.47
1000	2.21	2.34

Similarity threshold = 0.6

Number of extracted documents = 5976

Number of clusters	Entropy for only the extracted documents	Entropy after adding the remaining documents
10	2.65	2.8
50	2.46	2.73
100	2.28	2.68
200	2.06	2.57
500	1.82	1.93
1000	1.44	1.52

Similarity threshold = 0.7

Number of extracted documents = 3655

Number of clusters	Entropy for only the extracted documents	Entropy after adding the remaining documents
10	2.67	2.92
50	2.49	2.65
100	2.21	2.41
200	2.00	2.11
500	1.53	1.6
1000	1.15	1.21

Similarity threshold = 0.8

Number of extracted documents = 2668

Number of clusters	Entropy for only the extracted documents	Entropy after adding the remaining documents
10	2.64	2.97
50	2.41	2.73
100	2.17	2.23
200	1.98	2.12
500	1.5	1.73
1000	0.89	1.2

6.2.2 Reuters – 21578 dataset

Similarity threshold = 0.5

Number of extracted documents = 10,112

Number of clusters	Entropy for only the extracted documents	Entropy after adding the remaining documents
10	0.63	0.75
50	0.67	0.72
100	0.54	0.69
200	0.51	0.73
500	0.46	0.53
1000	0.36	0.47

Similarity threshold = 0.6

Number of extracted documents = 6124

Number of clusters	Entropy for only the extracted documents	Entropy after adding the remaining documents
10	0.58	0.98
50	0.53	0.87
100	0.46	0.67
200	0.44	0.61
500	0.37	0.43
1000	0.24	0.39

Similarity threshold = 0.7

Number of extracted documents = 2934

Number of clusters	Entropy for only the extracted documents	Entropy after adding the remaining documents
10	0.62	0.96
50	0.57	0.83
100	0.49	0.59
200	0.48	0.56
500	0.36	0.47
1000	0.29	0.36

6.2.3 Keepmedia dataset

Similarity threshold = 0.5

Number of extracted documents = 23,456

Number of clusters	Entropy for only the extracted documents	Entropy after adding the remaining documents
10	2.71	2.98
50	2.45	2.89
100	2.21	2.73
200	2.20	2.62
500	2.16	2.43
1000	2.02	2.15

Similarity threshold = 0.6

Number of extracted documents = 12,687

Number of clusters	Entropy for only the extracted documents	Entropy after adding the remaining documents
10	2.69	2.91
50	2.38	2.73
100	2.15	2.56
200	2.13	2.59
500	2.06	2.41
1000	1.98	2.34

Similarity threshold = 0.7

Number of extracted documents = 2934

Number of clusters	Entropy for only the extracted documents	Entropy after adding the remaining documents
10	2.31	2.97
50	2.19	2.86
100	2.04	2.54
200	2.0	2.4
500	1.57	2.32

6.4 Results for Feature based clustering

In this case also we have calculated the entropy of the output generated by feature based approach. We have used two tools for Multi-level graph partitioning algorithm for clustering of words:

1. Graclus
2. Metis

The results have been generated for each dataset for different number of clusters by using the two tools separately.

6.3.1 20 newsgroups dataset

Number of Clusters	Entropy with use of Graclus	Entropy with use of Metis
100	2.48	2.42
500	1.79	1.70
1000	1.30	1.32

6.3.1 Reuters - 21578 dataset

Number of Clusters	Entropy with use of Graclus	Entropy with use of Metis
100	0.72	0.65
500	0.68	0.59
1000	0.91	0.47

6.3.1 Keepmedia dataset

Number of Clusters	Entropy with use of Graclus	Entropy with use of Metis
100	1.92	1.64
500	1.92	1.55
1000	2.41	1.42

6.4 Results for K-Means clustering

In this the entropy is calculated for the clusters generated by using the standard K-Means algorithm for different number of clusters for each dataset.

6.4.1 20 newsgroups dataset

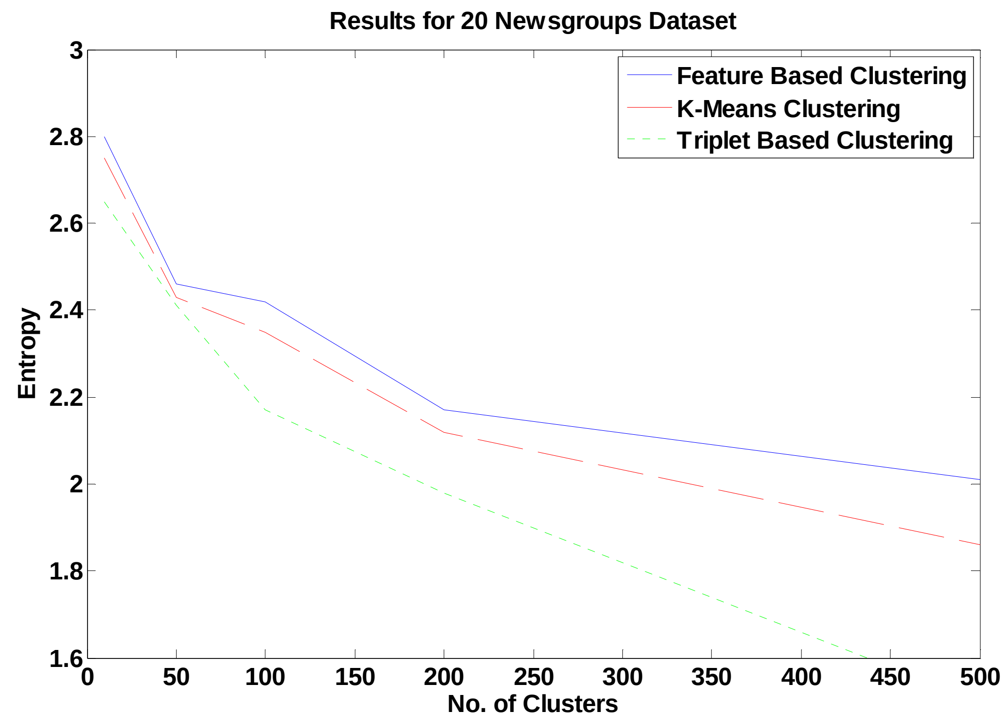
Number of clusters	Entropy
10	2.75
100	2.35
500	1.86
1000	No result

6.4.2 Reuters – 21578 dataset

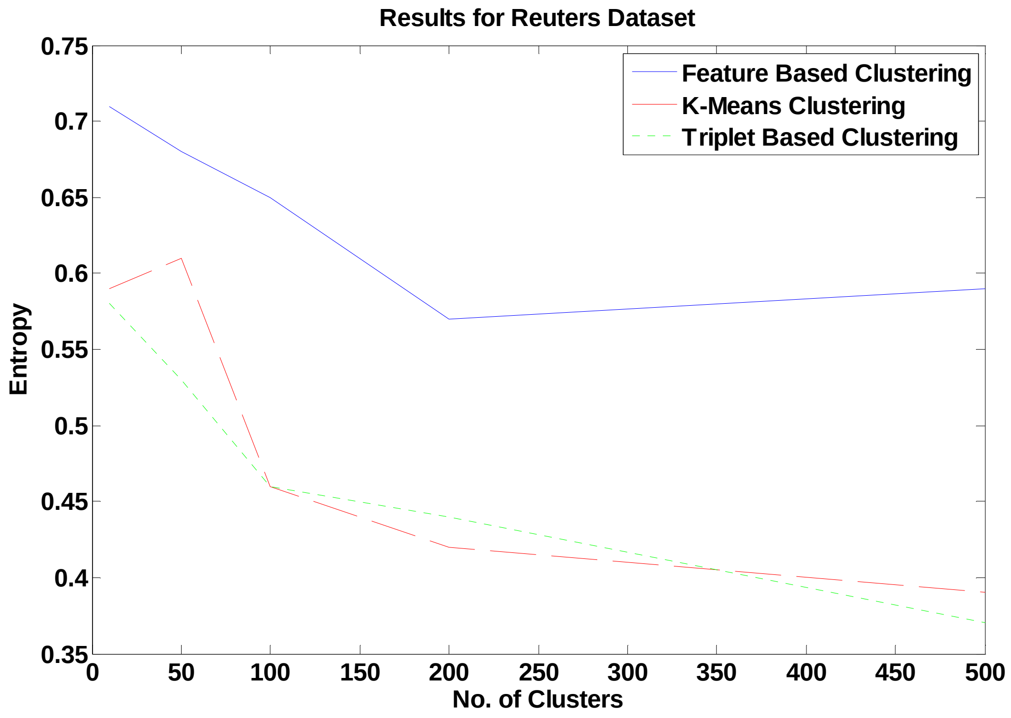
Number of clusters	Entropy
10	0.59
100	0.46
500	0.39
1000	0.35

6.4.3 Keepmedia dataset

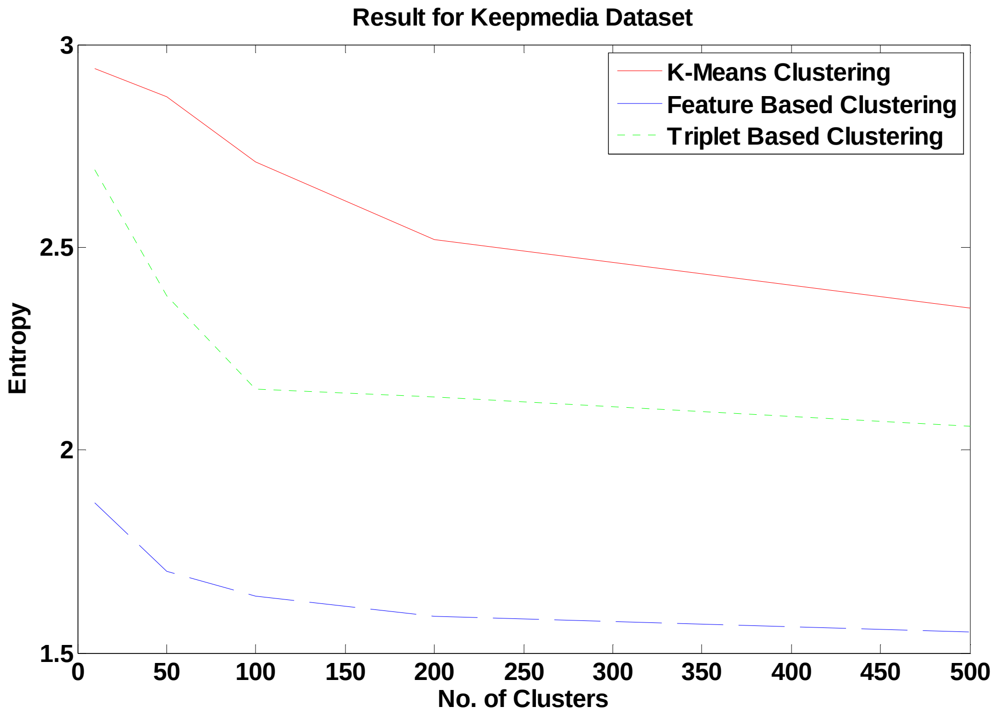
Number of clusters	Entropy
10	2.94
100	2.71
500	2.35
1000	No result



Results for 20 Newsgroups Dataset



Results for Reuters Dataset



6.5 Result Analysis

A set of documents used for evaluation has following features:

1. Number of documents per category
2. Evenness in number of documents in each category
3. Size of each document i.e. the number of words in each document
4. Similarity of documents of same category compared to similarity of documents of different categories.
5. Number of unique words in all the documents.

The quality of the results of the clustering algorithms depends very much on the features of the set of documents on which it is applied. For example, some algorithms may give good results in case of large documents as compared to small documents.

In the datasets used in our project, the reuters dataset has a large number of documents per category (around 4000) as compared to the other two (around 1000). The number of documents in each category is uneven in the keepmedia dataset but it is almost even in the other two. The documents are of very large in the keepmedia dataset, medium size in the reuters dataset and smaller in the 20 newsgroup dataset. Documents of same category in reuters dataset are very much similar as compared to documents of different categories. In 20 newsgroups dataset many categories are related and so the documents of different categories are also similar. The keepmedia dataset has a very large number of unique words.

For the 20 newsgroups dataset, as can be seen from the results, the Triplet based approach works better than the other two. Following are the logical reasons that can be stated for this observation:

- As the categories in this dataset are related, we require a stronger basis for each cluster. The triplet based approach gives a very strong basis i.e. a triangle with high edge weights, as compared to the K-Means algorithm in which a single node forms the basis of a cluster.
- The related categories also give many common words associated with different categories. This reduces the quality of the feature based clustering in which the word clusters are created first.
- The document size being less also gives an advantage to the triplet based method because this improves the similarity measure.

The feature based clustering works much better in case of keepmedia dataset. The reasons for this can be:

- The keepmedia dataset has large number of unique words, which are the features in this case, and this gives the feature based approach an advantage over the other approaches. In other algorithms the similarity measure looses its importance because even if the number of common words is large the similarity is low because of large number of total words.
- The documents of keepmedia dataset are large news articles and thus the names of people, places, organizations, etc. play an important role in them and this gives the consideration of co-occurrence of words a huge importance. For example, ‘Tendulkar’ and ‘cricket’ are two different words which co-occur many numbers of times in news articles. Now, if an

article contains only ‘Tendulkar’ then the feature based approach will still put it in the cluster of articles related to cricket or sports but this will not be the case with other algorithms.

Also, as can be seen from the graph plots, the dependency of the quality of result on the number of clusters is very much dependent upon the quality of the dataset in case of Feature based and K-means clustering but the graph is almost same for the Triplet based partitioning with only change in the absolute value of entropy. This may be because the graph first created, in case of Triplet based clustering, is dependent on the threshold value and then the number of clusters only change the way the graph is partitioned, which will be same for all the datasets.

Chapter7

Conclusion and Future Work

We started with an application for recommendations, in which clustering was required just to improve the time efficiency without decreasing the quality of the application, but while doing the research we realized that clustering has got a large application and its use is increasing with the increase in the use and applications of web. Also, we found that its not just the number of uses but the ways in which clustering is used in various applications is changing and this motivated us to think of some new algorithm for clustering.

7.1 Conclusion

In this thesis we investigated many existing algorithms and proposed two new ones. We conclude that it is hardly possible to get a general algorithm, which can work the best in clustering all types of datasets. Thus we tried to implement two algorithms which can work well in two different types of datasets.

The algorithm described in chapter 4, the Triplet based graph partitioning, suits the set of documents in which the required classes are related to each other and we require a strong basis for each cluster. Thus, this algorithm can be very effective in applications like a search engine for a particular field.

The next algorithm proposed in chapter 5 is the feature based clustering which is suited for set containing documents from very different fields and where co-occurrences of words plays an important role in deciding the cluster. The applications like recommendation of news articles in case of newsportals can be effective for this algorithm.

Finally we would conclude that though many algorithms have been proposed for clustering but it is still an open problem and looking at the rate at which the web is growing, for any application using web documents, clustering will become an essential part of the application.

7.2 Future work

The algorithms proposed in this thesis are at their very rudimentary stage and there are many possible improvements that can be implemented. Some of the possible implementations are:

1. The adding of remaining documents in case of triplet based clustering can be improved using many existing algorithms. Since the clusters are already defined, we can use the categorization algorithms for allotting clusters to the remaining documents.
2. The triplets in case of triplet based clustering can be allotted weights based on the weights of the edges belonging to them and then the other edges can be allotted weights depending on which triplets they are associated with.
3. For feature based clustering, the similarity measure used at present is very simple, and this can be changed according to the application to improve the results.

Chapter 8

References

[1] Noam Slonim and Naftali Tishby.
"The Power of Word Clusters for Text Classification"
School of Computer Science and Engineering and The Interdisciplinary Center for Neural Computation The Hebrew University, Jerusalem 91904, Israel

[2] Noam Slonim and Naftali Tishby.
"Document Clustering using Word Clusters via the Information Bottleneck method"
School of Computer Science and Engineering and The Interdisciplinary Center for Neural Computation The Hebrew University, Jerusalem 91904, Israel

[3] Inderjit S. Dhillon, *Member, IEEE*, Yuqiang Guan and Brian Kulis
"Weighted Graph Cuts without Eigenvectors: A Multilevel Approach"

[4] Michael Steinbach, George Karypis, and Vipin Kumar.
"A Comparison of Document Clustering Techniques"
Department of Computer Science and Engineering, University of Minnesota

[5] Anton V. Leouski and W. Bruce Croft
"An Evaluation of techniques for clustering search results"
Computer Science Department, University of Massachusetts at Amherst

[6] Brian S. Everitt, Sabine Landau, and Morven Leese. *Cluster Analysis*. Oxford University Press, fourth edition, 2001.

[7] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *An Introduction to Information Retrieval.* Cambridge University Press, 2008

[8] Kirk Schloegel, George Karypis, and Vipin Kumar. *Parallel static and dynamic multi-constraint graph partitioning*

[9] Tao Liu, Shengping Liu, Zheng Chen, and Wei-Ying Ma. *AN Evaluation on Feature Selection for Text Clustering.* Proceedings of the Twentieth International Conference on Machine Learning (ICML-2003), Washington DC, 2003.