Application of Expectation Maximization Clustering Algorithm in Text Document Summarization: A Query Dependent Approach

Ashwini Ranade[1], Mrunal S. Bewoor[2], Dr. Suhas H. Patil[3]
[*]*Research Student, M.Tech Computer, BVU College of Engineering, Pune, MH, India*
[*#]*Asst. Professor, Bharti Vidyapeeth University College of Engineering, Pune, India*
[1]ashwinimjoshi@rediffmail.com

Abstract – **Document summarization has been receiving much attention these years. Representing the burgeoning data concisely and succinctly is an ever-important task. This carries with itself not only the problem of immensity of scale but also the lack of any uniform structure or arrangement of text. Also, the number of document formats and document structures over the Internet vary enormously and summarization of texts is hence an increasingly challenging task for researchers. Thus there is the need of summarization techniques which will create the summary of the document which includes all the keywords given by the user.**

The Expectation Maximization (EM) algorithm has been applied to the problem of parameter estimation. In the EM framework, the parameters of the component densities are unknown, as are the mixing parameters, and these are estimated from the patterns. The EM procedure begins with an initial estimate of the parameter vector and iteratively rescores the patterns against the mixture density produced by the parameter vector. The rescored patterns are then used to update the parameter estimates. In a clustering context, the scores of the patterns can be viewed as hints at the class of the pattern. Those patterns, placed (by their scores) in a particular component, would therefore be viewed as belonging to the same cluster. [4, 7, 8]

The present research work focuses on analytical study of document clustering and summarization techniques. Currently the most research is focused on Query-Independent summarization. The main aim of this research work is to combine both the approaches of document clustering and query dependent summarization. This mainly includes creation of document graph based on the term frequency, then applying Expectation Maximization clustering algorithm on that document graph, In expectation stage we will get the initial clusters and in Maximization stage the final clusters of the document will be obtained. Section 1 of this paper contains Introduction; section 2 contains related work, where as Section 3 give the description of each module of the system.

Keywords – **Text Summarization, Clustering Techniques, Expectation Maximization clustering algorithm, Clustering algorithms, Summarization algorithms, Summary.**

I. INTRODUCTION

Large amount of data is added to the web constantly and huge amount of data is present to fulfil the user requirement. Users always need to search for the required information by using particular keywords. As the number of documents available on users' desktops and the data available on the Internet are increasing rapidly, there is the need to provide high-quality summaries for the user so that the user can quickly locate the desired information. Summarization is the process of condensing a source text into a shorter version preserving its information content.

1.1 Text Summarization

There are many different descriptions of what summarization is (or should be) and there is no much disagreement in them. For example, text summarization may be described as "to reduce (long) textual information to its most essential points", "to condense information down to critical bit", or "to distill the most important information from a source or sources to produce an abridged version for a particular user (or users) and task (or tasks)" [**1]** A more mechanical way or computational way to look at text summarization is to see it as a **text transformation process**. For example, Sparck-Jones (1999) modelled text summarization as a three-stage text transformation activity that includes interpretation, transformation and generation. Interpretation refers to source text interpretation that analyses source text and transforms it into appropriate text representation. Transformation refers source representation mapped into

summary text representation. This involves key content representation (as key words, key concepts, significant words and significant sentence), concept organization, synthesis of an appropriate summary output and summary text representation. Generation refers to the generation of summary text from summary representation. Such a model presents a more general concept of text summarization. Viewing text summarization as text transformation activities may or may not rely on text understanding.

There is not much debate on what a summary is. The exact form and content of a summary vary greatly depending on its source, its nature, purpose, intended reader, and many more. In terms of structural composition, it may vary from a list of keywords to a list of independent single sentences, or a fully planned and generated coherent summary text. For example, abstracts for scientific articles give a good example of summaries as they have a very clear structure.

The process of producing a summary from a source text consists of the following steps:

1. The interpretation of the text;
2. The extraction of the relevant information which ideally includes the "topics" of the source;
3. The condensation of the extracted information and construction of a summary representation;
4. The presentation of the summary to the reader in natural language.

A compelling application of document summarization is the snippets generated by Web search engines for each query result, which assist users in further exploring individual results. The Information Retrieval (IR) community has largely viewed text documents as linear sequences of words for the purpose of summarization. Although this model has proven quite successful in efficiently answering keyword queries, it is clearly not optimal since it ignores the inherent structure in documents. Furthermore, most summarization techniques are query-independent and follow one of the following two extreme approaches: Either they simply extract relevant passages viewing the document as an unstructured set of passages, or they employ Natural Language Processing techniques. The former approach ignores the structural information of documents while the latter is too expensive for large datasets (e.g., the Web) and sensitive to the writing style of the documents.

In this paper, a method to add structure in form of a graph, to text documents in order to allow effective query specific summarization is discussed. [2]. That is a document is viewed as a set of interconnected text fragments and main focus is on keyword queries. Because of its power and ease of use, keyword search is the most popular information discovery method on documents. This technique has the following key steps:

First, at the pre-processing stage, a structure is added to every document, which can then be viewed as a labelled, weighted graph, called the document graph.[2] Then, at query time, given a set of keywords, a keyword proximity search is performed on the document graphs to discover how the keywords are associated in the document graphs. For each document its summary is the minimum spanning tree on the corresponding document graph that contains all the keywords (or equivalent based on a thesaurus). So data from the minimum spanning tree nodes is collected and presented as a summary of the document.

Document clustering methods usually represent documents as a term document matrix and perform clustering on it with the help of different clustering algorithms. Although these clustering methods can group the documents satisfactorily, it is still hard for people to capture the meanings of the documents since there is no satisfactory interpretation for each document cluster. A single cluster can be represented as a node in the graph. The clusters which are related to each other can be connected with a weighted graph. The weights will be assigned based on the relativity among the clusters using the keywords from the query. The optimal spanning tree of the resulting graph can give the summary of the document.

1.2 Clustering

Clustering is useful in several exploratory pattern-analysis, grouping, decision-making, and machine-learning situations, including data mining, document retrieval, image segmentation, and pattern classification. However, in many such problems, there is little prior information (e.g., statistical models) available about the data, and the decision-maker must make as few assumptions about the data as possible. It is under these restrictions that clustering methodology is particularly appropriate for the exploration of interrelationships among the data points to make an assessment of their structure.

An example of clustering is depicted in Figure 1. The input patterns are shown in Figure 1(a), and the desired clusters are shown in Figure 1(b). Here, points belonging to the same cluster are given the same label. The variety of techniques for representing data, measuring proximity (similarity) between data elements, and grouping data elements has produced a rich and often confusing assortment of clustering methods.

The term “clustering” is used in several research communities to describe methods for grouping of unlabeled data. These communities have different terminologies and assumptions for the components of the clustering process and the context in which clustering is used.

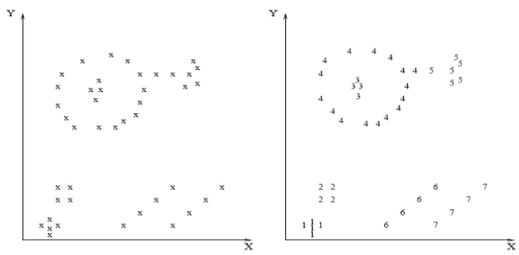


Fig 1(a) Fig 1(b)

Figure 1 Data Clustering

1.2.1 Clustering Techniques

Clustering is a method of unsupervised learning and a common technique for statistical data analysis used in many fields, including machine learning, data mining, pattern recognition, image analysis and bioinformatics.[3] Partition algorithms typically determine all clusters at once, but can also be used as divisive algorithms in the hierarchical clustering. This report includes the Mixture Resolving approach.

The mixture resolving approach to cluster analysis has been addressed in a number of ways. The underlying assumption is that the patterns to be clustered are drawn from one of several distributions, and the goal is to identify the parameters of each and (perhaps) their number. More recently, the Expectation Maximization (EM) algorithm (a general-purpose maximum likelihood algorithm [4] for missing-data problems) has been applied to the problem of parameter estimation. In the EM framework, the parameters of the component densities are unknown, as are the mixing parameters, and these are estimated from the patterns. The EM procedure begins with an initial estimate of the parameter vector and iteratively rescores the patterns against the mixture density produced by the parameter vector. The rescored patterns are then used to update the parameter estimates. In a clustering context, the scores of the patterns can be viewed as hints at the class of the pattern. Those patterns, placed (by their scores) in a particular component, would therefore be viewed as belonging to the same cluster. [4,7,8]

Expectation Maximization is an iterative method which alternates between performing an Expectation step (E Step) which computes the expectation of the log-likelihood evaluated using the current estimate for the latent variables, and a maximization (M) step, which computes parameters maximizing the expected log-likelihood found on the *E* step. [18]

Functions Used:

ClusterArguments()

Expectation()

For Expectation Maximization algorithm takes initial clusters from the user with the threshold to match the nodes with the initial clusters selected. The remaining mismatching clusters are then send to the new extra clusters.

Maximization()

In this function we are refining the output of the expectation step. The nodes which are in the last cluster are again reclustered based on the similarity between them.

II. SYSTEM ARCHITECTURE

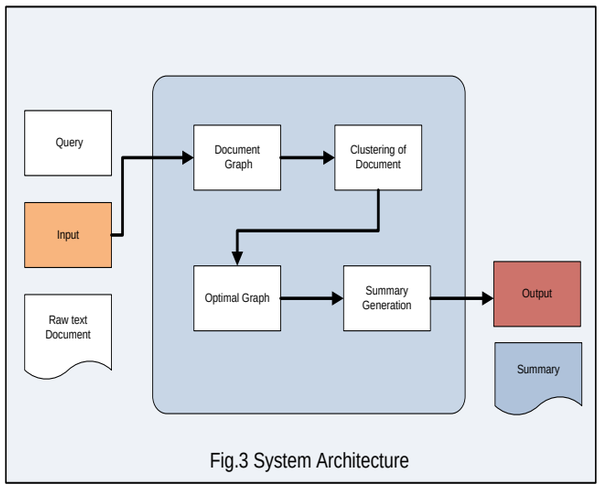


Fig.3 System Architecture

I. SYSTEM DESCRIPTION

2.1 Constraints

The main aim of this Paper is to combine the both approaches of document clustering and query dependent summarization. The main constraints considered in this work can be outlined as:

- The proposed work will be mainly focused on summarization of text files (i.e. .txt).
- The proposed work will be limited to clustering of text files of Standard files related to the topic popular amongst researchers will be used.
- Standard performance evaluation metrics will be used to validate performance.

2.2 Module Description

Module 1: Clustering of the Documents

Objective:

Automatically group semantically related paragraphs in the documents into *clusters.*

Clustering Algorithms:

- Mixture Resolving Clustering algorithm.

Module 2: Creating the document graph.

Each cluster becomes a node in the document graph.
The *document graph G (V,E)* of a document *d* is defined as follows:

- *d* is split to a set of non-overlapping clusters *t(v),* each corresponding to a node *v*□*V*.
- An edge *e(u,v)*□□*E* is added between nodes *u, v*□*V* if there is an association between *t(u)* and *t(v)* in *d*.

Hence, we can view *G* as an equivalent representation of *d,* where the associations between text fragments of *d* are depicted.

A weighted edge is added to the document graph between two nodes if they either correspond to adjacent cluster node or if they are semantically related, and the weight of an edge denotes the degree of the relationship. Here two clusters are considered to be related if they share common words (not stop words) and the degree of relationship is calculated by "***Semantic parsing***". Also notice that the edge weights are **query-independent**, so they can be pre-computed.

Let *D*={d1,d2,,…,dn} be a set of documents d1,d2,,…,dn. Also let size (di) be the length of di in number of words. Term frequency tf (d,w) of term (word) w in document d is the number of occurrences of w in d. Inverse document frequency idf (w) is the inverse of the number of documents containing term w in them.

A keyword query Q is a set of keywords Q={w1,…,wm}. A key component is the document graph G(V,E) of a document d.

Notice that *Q* is only used in *assigning weights to the nodes* of *G and not for assigning weights to the edges,* which is a desirable property since the rest of *G* can be computed before queries arrive.

Module 3: Adding weighted edges to the document graph

(Note: Adding weighted edge is query independent)
The following input parameters are required at the pre computation stage to create the document graph:

1. ***Threshold for edge weights***. Only edges with weight not below *threshold* will be created in the document graph. (A threshold is user configurable value that controls the formation of edges)

2. ***Minimum text fragment size***. This is used when a fragment is too long, which would lead to large nodes (text fragments) and hence large summaries. Users typically desire concise and short summaries.

Adding weighted edge is the next step after generating document graph. Here for **each pair of nodes *u,v* we compute the association degree between them, that is, the score (weight) *EScore(e)* of the edge *e(u,v)*.** If Score *(e)* ≥ *threshold*, then *e* is added to *E*. The score of edge *e (u,v)* where nodes *u*, *v* have text fragments *t(u)*, *t(v)* respectively is:

$$\text{EScore} = \frac{\sum \left(\left(tf(t(u),w) + tf(t(v),w) \right) . idf(w) \right)}{\text{size(t(u))} + \text{size(t(u))}} \quad (1)$$

Where *tf(d,w)* is the number of occurrences of *w* in *d,*
idf(w) is the inverse of the number of documents containing *size(d)* is the size of the document (in words).That is, for every word *w* appearing in both text fragments we add a quantity equal to the $tf \in idf$ score of *w.* Notice that stop words are ignored.

Module 4: Adding weight to nodes in document graph

When a query *Q* arrives, the nodes in *V* are assigned query-dependent weights according to their relevance to *Q*. In particular, we assign to each node *v* corresponding to a text fragment *t(v)* node score *NScore(v)* defined by the Okapi formula as given below (Equation 2).

$$\sum_{t \in Q,d} \ln \frac{N - df + 0.5}{df + 0.5} \frac{(k1 + 1)tf}{\left(k1(1-b) + b\frac{dl}{avdl}\right) + tf} \frac{(k3+1)\,qtf}{k3 + qtf}$$

tf is the term's frequency in document,
qtf is the term's frequency in query,
N is the total number of documents in the collection,
df is the number of documents that contain the term,
dl is the document length (in words),
avdl is the average document length and
k1 (between .0–2.0), b (usually 0.75), and k3 (between 0–1000) are constants.

Module 5: Generating Closure Graph and Finding Minimal Spanning Tree.

Closure graph contains minimal clusters. Minimal clusters are the clusters which shows non zero weight with the in out query. The minimal clusters are the clusters which appear in the result.

Module 6: Result

After getting the minimal clusters, the result can be displayed in two ways:
Top 1 Result Summary
Multi-Result Summary

In top 1 result summary, the minimal cluster having highest weight with the input query is returned, and in multi-result summary all the minimal clusters are returned as result.

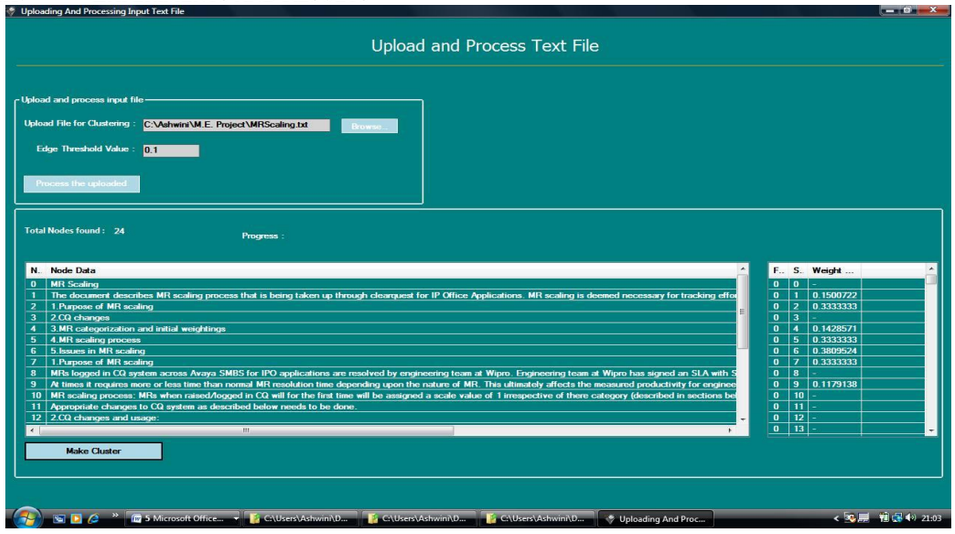
Before displaying the result as a cluster, the cluster is split into its nodes and the weight of every node with the input query is calculated. The nodes are displayed in decreasing order of

the weight with the input query. Means the node having highest weight is displayed at the top and lowest at the bottom.

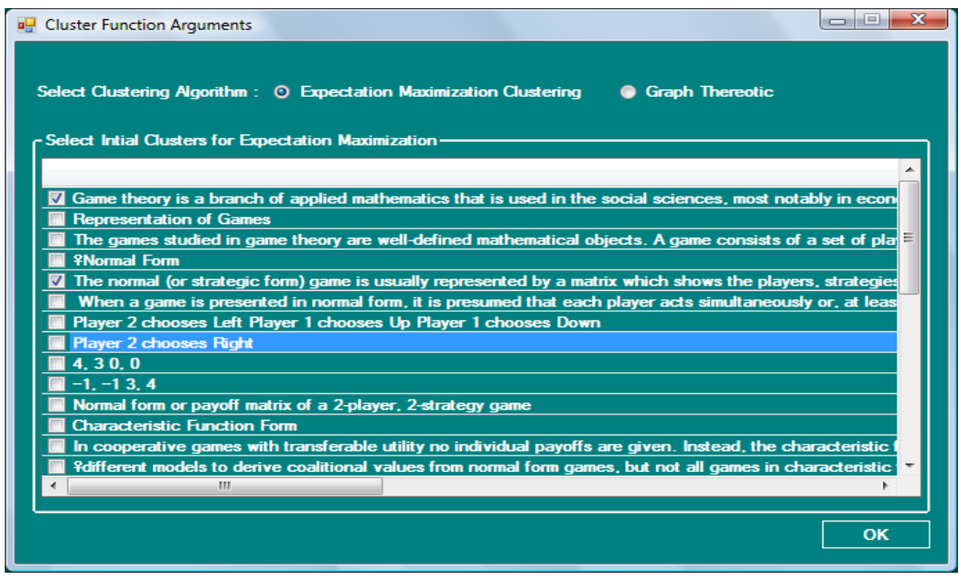
III. EXPERIMENTS AND RESULTS

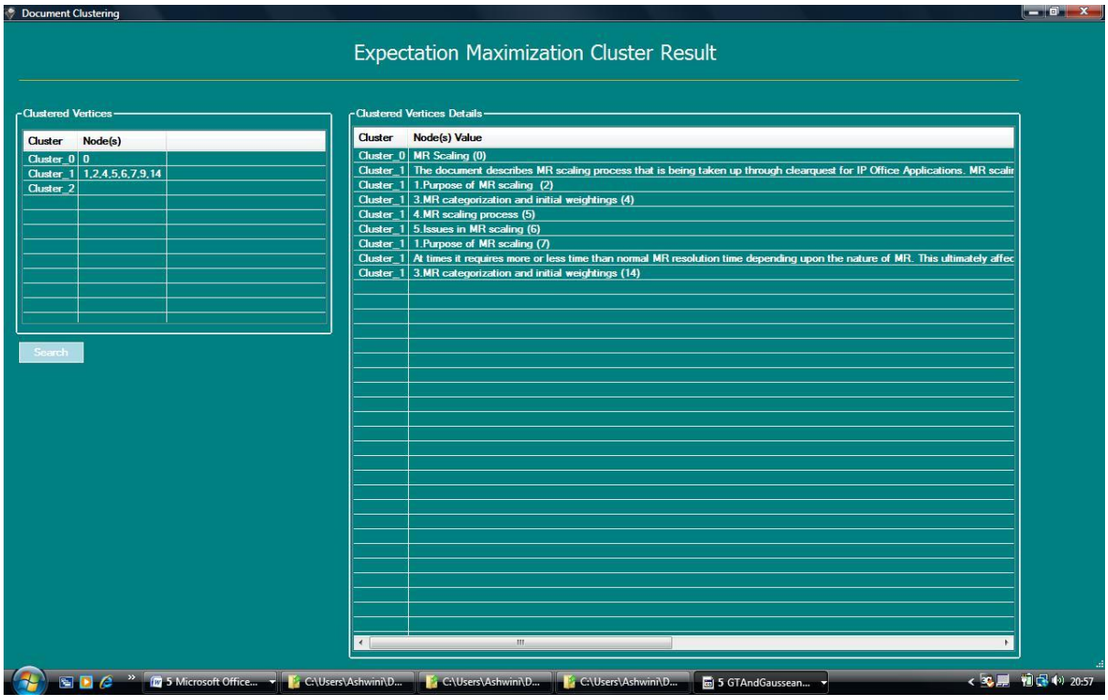
The basic user interface consists of at least three windows. First window is needed to input the text file. For this user has to give input as text file only.

Initially user needs to give text file as input. Then he will click on 'process the input file'. Every newline contains will form a single node and description of the node data will be displayed. Along with this the weight between each node to every other node is calculated. The weight of a node with itself is null. This process is demonstrated by following examples.



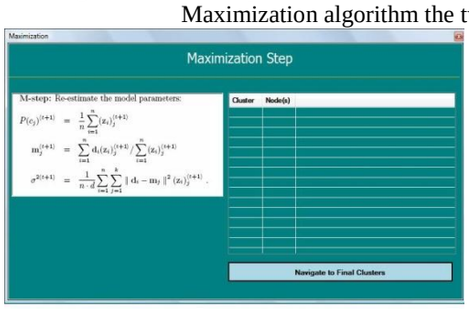
For Expectation Maximization clustering algorithm user has to select the initial clusters, which is as shown below

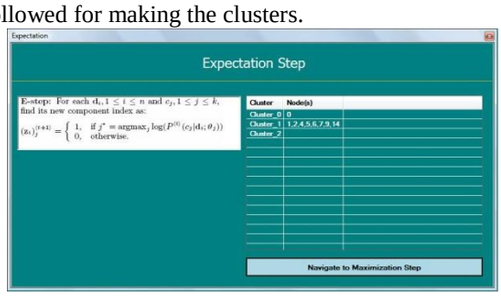




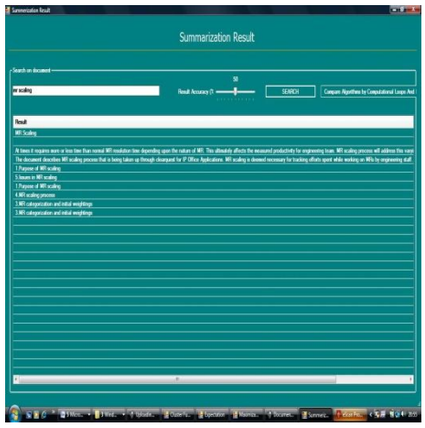
For Expectation Maximization clustering algorithm user has to select the initial clusters, which is as shown below

The clustering will be done as per the thresholds set by the user; It is shown in the following output screens. In Expectation Maximization algorithm the two steps are followed for making the clusters.





Now the summary we have got. To measure the performance of the algorithm we are plotting the graph of the results on the basis of Number of computational loops

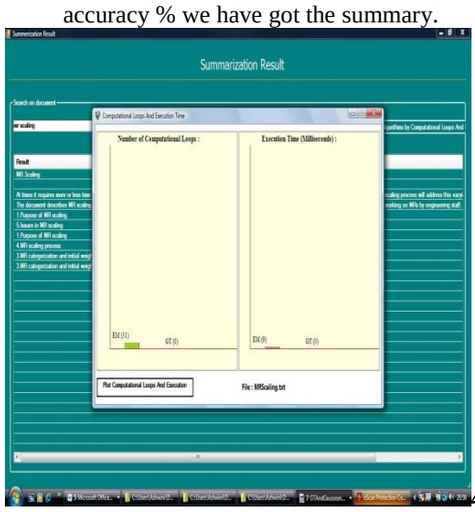


After clustering end user will fire the query and give % to correlate the cluster with fired query. The clusters containing the part of query called as minimal cluster will be displayed.

and time required for execution. For this purpose we have taken some sample special files and the result of the algorithms is plotted.

IV. CONCLUSION

After uploading the text file MRScalin.txt total 109 nodes are formed. After selecting the 4 initial clusters we have got total 5 clusters in Expectation stae and 4 clusters in maximization step. After givin the search string and the accuracy % we have got the summary.



4.0

Thus we can use the Expectation maximization Clustering algorithm for Text document summarization. Here the Query dependent approach for document summarization is proposed.

V. REFERENCES

[1] Endres- Niggemeyer, Mani and MaybUry, Sparck-Jones, "Human Style WWW summarization", ACM 2008
[2] Ramakrishna Varadarajan, Vangelis Hristidis,"A System for Query-Specific
Document Summarization", ACM Nov - 2008
[3] A.K. Jain, M.N. Murty, and P.J. Flynn "Data Clustering: A Review". ACM Sept 199
[4] Jackie CK Cheung,"Comparing Abstractive and Extractive Summarization of Evaluative Text: Controversiality and Content Selection"
[5]The complete Reference of .NET - by Matthew,Tata MacGraw Hill Publication Edition 2003
[6] The complete Reference of .NET - by Matthew,Tata MacGraw Hill Publication Edition 2003
[7] Jie Tang, Limin Yao, and Dewei Chen 'Multi-topic based Query-oriented Summarization', SIAM
[8] Regina Barzilay and Michael Elhadad 'Using Lexical Chains for Text Summarization'
[9]Paul S. Bradley, Usama M. Fayyad, Cory A Raina, ' Scaling EM (Expectation-Maximization)clustering to large Databases', Technical Report MSR-TR-February 1999.
[10] Frank Dellaert, 'The expectation Maximization Algorithm', Technical Report no. GIT-GVU-02-20, February 2002.
[11] http://en.wikipedia.org/wiki

AUTHOR

Ashwini Ranade born in 1980,received her Bachelor of Engineering from Amravati University and perusing her Master of Technology from Bharati Vidyapeeth University, Pune.

She has total 9+ years of teaching experience in renowned Engineering and management colleges. She has published her 2 research papers and working in Data mining area.

Mrunal S. Bewoor is working as Associate Professor in Bharati Vidyapeeth University, Pune. She has total 15+years of experience and her area of research is Business Data Processing.

Dr Suhas H. Patil is Professor and Head of the department (Computer science and Engineering), College of Engineering, Bharati Vidyapeeth University, Pune. His area of specialization is Operating Systems and Distributed Systems.