IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661, p- ISSN: 2278-8727Volume 12, Issue 5 (Jul. - Aug. 2013), PP 24-29
www.iosrjournals.org

Enhanced Clustering Algorithm for Processing Online Data

Ankita Choubey[1], Dr. Sadhna K. Mishra[2],
Department of Computer Science Engineering M.Tech.(SE) LNCT Bhopal(M.P.)
Department of Computer Science Engineering LNCT Bhopal (M.P.)

Abstract: *Clustering of data has been of intense need for any organization in the past and various researchers in the field of data mining have been continuously working to find efficient and accurate tools and algorithms for the same. From the proliferation of internet and network applications is pressing the same need more deeply and it is becoming more and more necessary for providing such algorithms by the researchers of data mining. Researchers have been working continuously and finding incremental clustering mechanism to be best suitable for online data. Clustering of very large document databases is useful for both searching and browsing. The Periodic updating of clusters is required due to the dynamic nature of databases. For this purpose incremental clustering is a profitable approach.*

Incremental clustering algorithm clusters data in dynamic form. It requires initial clusters to be decided in advance i.e. they must be pre exist and fixed.

This work proposed a dynamic and novice approach of incremental clustering algorithm for creating efficient clusters and rearrangement of the clusters on the basis of characteristics of the data. Also an approach of retrieval and searching of some specific data from the fixed clusters using a new frequency check method will be compared with the proposed work.

Keywords: *Incremental Clustering, Characteristics, Parameters, Clustering, Hybrid, Hierarchical.*

I. Introduction

We focus here on clustering as an unsupervised learning process, which attempts to represent the partitions of data of unknown class origin without feedback or information beyond what is inherently gained from the data. Its purpose is to find underlying groups, or clusters, which ideally share similar features. Most commonly, the purpose of unsupervised clustering is to autonomously learn how to best discretize a space with the intent of classifying unseen data samples into one of several clusters with the assumption that commonly classed samples share a common features. As class labels are unnecessary for training or adjusting parameters, learning here implies a presentation of a set of samples that need not be segmented into training, test, and validation subsets. In this paper, we introduce a new approach to incremental or online clustering that puts strict restraints on centroid estimation and modification in an attempt to handle the stability/plasticity dilemma that plagues all data-driven systems. As will be detailed in the following section, our approach augments traditional competitive learning clustering algorithms. [1]

One drawback of the partitional clustering is the difficulty in determining the optimal number of clusters (*k*). Incremental clustering is an efficient method and runs in linear time to the size of input data set. In most related studies, the dissimilarity between two clusters is defined as the distance between their centroid or the distance between two closest data points. Hierarchical clustering algorithms create a hierarchical decomposition of data set based on some criterion. The decomposition can be described as a dendrogram that represents a series of nested partitions. A partition is obtained by cutting the dendrogram at some desired level.

Hierarchical algorithms do not suffer from the problem of choosing a pre-specified number for the output clusters [3].

Hierarchical clustering algorithms can differ in their operation. Agglomerative clustering methods start with each object in a distinct cluster and successively merge them to larger clusters until a stopping criterion is satisfied. Alternatively, divisive algorithms begin with all objects in a single cluster and perform splitting until a stopping criterion is met. Both agglomerative and divisive hierarchical algorithms are static in the sense they never undo what was done previously, which means that objects which are committed to a cluster in the early stages, cannot move to another cluster. In other words, once a cluster is split or two clusters are merged, the split objects will never come together in one cluster or the merged objects will be never in the same cluster, no matter whether the splitting or the merging is the correct action or not. But in practice, some splitting or merging actions may not be correct and there is a need to rearrange the partition. This problem is a cause for inaccuracy in clustering, especially for poorly separated data sets [2].

II. Existing System

A. Leader Clustering Algorithm Leader clustering algorithm [2] is a single pass algorithm and very efficient in terms of computational requirements. When leader algorithm is applied on the data set, it chooses one data object and assumes it as a first leader. Then, it chooses all the data objects one by one and compares it with the leader. The comparison is based on the Euclidean distance between leader and the current data object.

Whether to put the current data object within the cluster lead by that particular leader or make that data object as a new leader is decided based on the threshold value. This process will continue until the entire data set has been processed.

Following is the leader algorithm:

Algorithm: Leader Clustering

```
Input: Data set, Threshold value (Th).
Output: Number of Clusters (k).
Initialize a leader, add it to the leader list and set leader counter L =1
Do for all patterns i =2 to N
{
Calculate the distance between pattern i and all leaders
Find the nearest leader
If (distance between pattern i and nearest leader<Threshold) then
{
        Assign pattern i to the nearest leader Label the cluster number
        Add pattern i to member list of this cluster
        Increment member count of this cluster
}
else
{
        Add it to the leader list
        Increment leader counter L = L + 1
}
}
```

For a given threshold value (Th), leaders method [2] works as follows. It maintains a set of leaders L, which is incrementally built. For each pattern x in dataset (D) if there is a leader l Є L such that distance between x and l is less than Th, then x is assigned to the cluster represented by l.

Note that even if there are many such leaders, only one (the first encountered one) is chosen. Because of this the cluster represented by a leader is of semi-spherical shape. If there is no such leader then x itself becomes a new leader and is added to L. Along with each leader, a count indicating number of patterns that are grouped under the leader is also stored.

Leader clustering algorithm scans the input data set only once and gives the result very quickly. The advantage with leaders clustering algorithm is that there is no need to Document Similarity Calculation Document Clustering Clusters Document analysis Frequency matrix Representation Vector Representation Vector Normalization knows the number of clusters in advance. Leaders clustering algorithm finds appropriate number of clusters based on the threshold value (Th). The leader algorithm requires only O(n) time to get the clusters from the data set with n number of data objects. The drawbacks with this algorithm are: (1) Threshold value must be very appropriate, (2) output depends on the order in which data is presented, and (3) arbitrary shape of the cluster may not be possible to determine. Leader's algorithm produces only the circular clusters. Also, the result of leader clustering algorithm is highly affected by the presence of noise.

III. Clustering Based On Similarity

Wan [6] reviews existing similarity measures, including the measures in the vector space model, the information theoretic measure, the measures derived from popular retrieval functions and OM-based measures, and also proposes a novel measure based on the earth mover's distance to evaluate document similarity by allowing many-to-many matching between subtopics.

When designing a clustering algorithm, similarity measure choice is sensitive to the specific representation, which determines whether the algorithm can accurately reflect the structure of various components in high dimensional data [10]. However, there are many methods to measure pair-wise document similarity and no selection criteria so that too often it is an arbitrary choice to choose similarity measure although it is very important to a clustering algorithm.

Single-Pass Clustering

It is a very simple clustering method. The method makes the first object as the centroid for the first cluster, and then calculates the similarity between the next object and each existing cluster centroid using some similarity coefficient. If the highest value of the similarity is greater than the threshold value appointed beforehand, the new object is added to the corresponding cluster and the centroid is updated; otherwise, the object is put into a new cluster.

3.5.2 K-Nearest Neighbor Clustering

KNN is an approach to classifying objects based on closest training data in pattern recognition. According to this algo, an object is classified by a majority vote of its neighbors, with the object being assigned to the class most common amongst its K nearest neighbors [11].

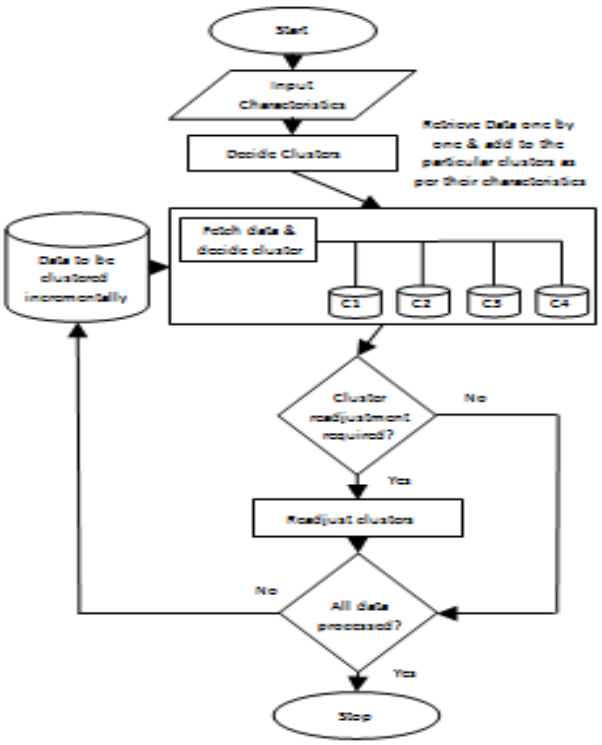
IV. Proposed System

Incremental Clustering requires initial clusters to be decided in advance i.e. they must pre exist for processing. If the initial clusters are to be fixed, then there are several ways it can be achieved. The algorithm being proposed is a dynamic and novice algorithm for clustering and rearrangement of the initial clusters dynamically. In this work I am offering readjustment and efficient retrieval of clusters dynamically on the basis of characteristics in the system

The Method shall be allowing the users to rearrange the data into the existing clusters:

- Initially a few characteristics of data are chosen to decide the initial clusters.
- Data will be incrementally clustered into the user defined clusters along with the weight values indicating the percentage by which data matches with the particular cluster
- User can add any more clusters in the system by adding additional characteristics.
- Now data will be clustered in increased number of clusters incrementally.
- Already clustered data will be filtered and moved to the relevant clusters. This process will be done using a threshold weight value, which is matched with the stored weight values in specific clusters.
- Now the clustered which arranges all required data properly can be used for searching and retrieving of any specific data.

The algorithm can be viewed as per the following flow chart:



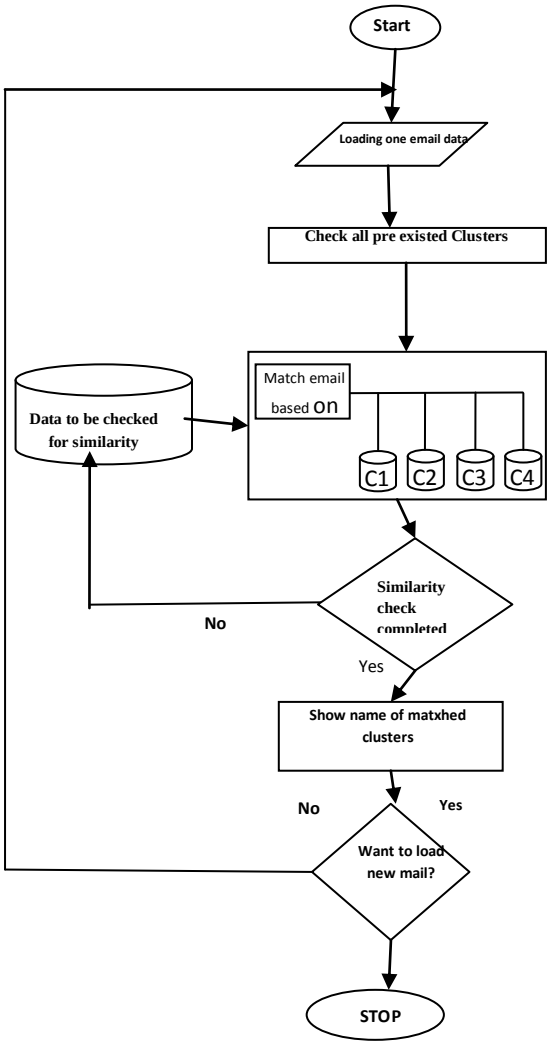
Method of Frequency Check Approach:

- A new frequency checking process is introduced and examined to compare the efficiency of this work.
- In frequency check all the email data is initially clustered within four groups on the basis of their similar characteristics.

- In this approach any specific data can be retrieved and the result will show its belonging with particular pre existed cluster.
- By this approach we can find out the similarity measurement of any data with other data or clusters.
- Both the approaches will be compared for time complexity taken by email data for retrieval and arrangement and we will calculate the efficiency of our work.

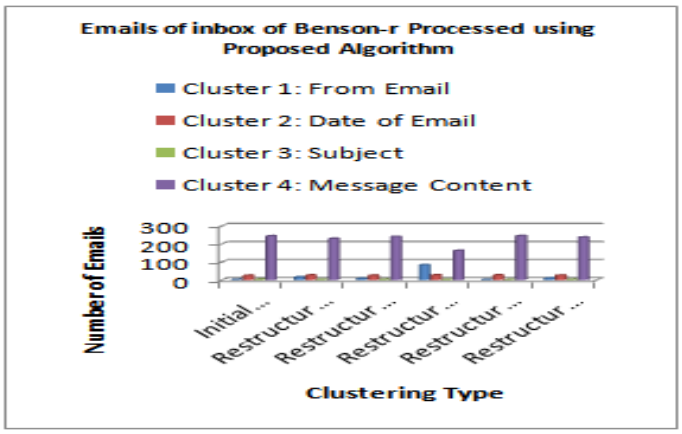
V. Results

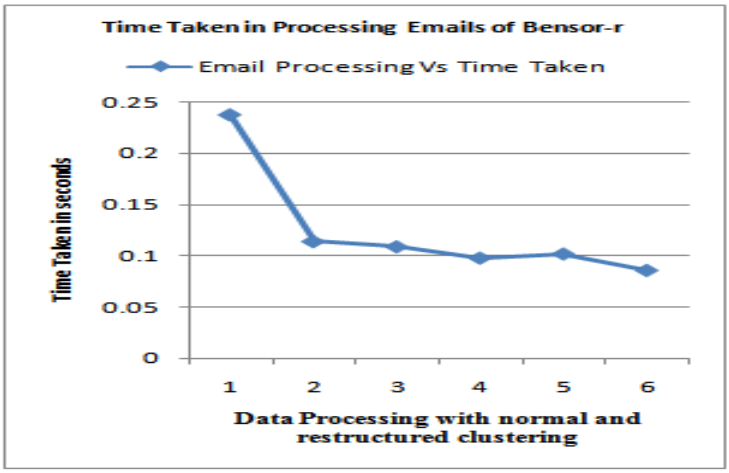
Incremental clustering algorithm clusters data in dynamic form. But assuming initial clusters for clustering process is not justifiable in a ll applications. Therefore, the proposed algorithm as shown above, very well deals with the problem. In the proposed algorithm the initial clusters have been decided on the basis of the characteristics provided by the user of the data.



Flow Chart for the Frequency based similarity check

Graphs & Inference For Incremental Algorithm:

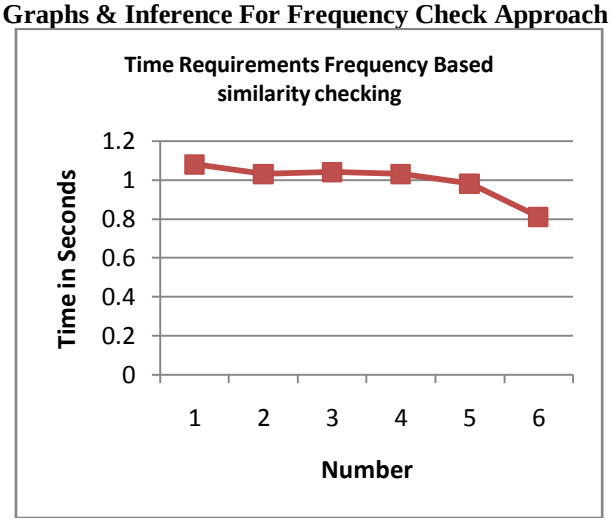




VI. Inference

The graph and the data in table show the time taken in processing the emails initially and by using restructuring of the clusters and it is found that the time required in restructuring is very less in comarison with initial clustering and that also goes down as more and more restructured clustering is performed. This depicts the high performance of the proposed algorithm.

Graphs & Inference For Frequency Check Approach:



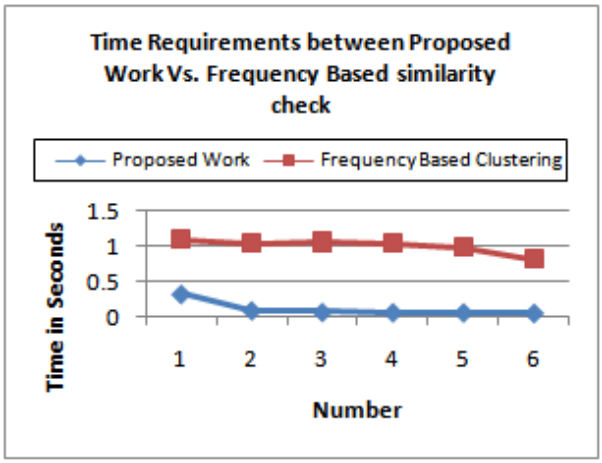


Figure 18: Graph Showing Comparison between Proposed Work and Frequency based similarity measure

From the chart, the comparison of time requirements of proposed work and frequency based similarity measure has been shown. From the data and the graph it is clear that the time taken in frequency based similarity measure is too much in comparison with proposed work due to collection of frequency data. The pattern shows that the time requirements is in decreasing but for all runs the graph is much higher than the proposed work.

Cluster Type	Average Time Taken in Processing Emails Using Proposed Work (Seconds)	Time Taken in Processing Emails of Using Frequency based similarity measure (Seconds)
First Run	**0.32**	**1.08**
Second Run	**0.087**	**1.032**
Third Run	**0.064667**	**1.04**
Fourth Run	**0.064667**	**1.031**
Fifth Run	**0.058333**	**0.982**
Sixth Run	**0.058333**	**0.812**

Table 8: Data of time taken in processing for proposed work and Frequency based similarity measure method

From the above discussion it is clear that the problem taken into consideration in this work has been addressed fully and will produce the results according to the requirements.
In future, the proposed work shall be implemented using C# language to show the working and solving the problem by using proposed algorithm.

References

[1] Yongli Liu1, Qianqian Guo2, Lishen Yang1, Yingying Li1 "Research on Incremental Clustering",This work was supported in part by national social science fund (No. 11CYY019) and the natural scientific research project of education department of Henan Province (No. 2011A520015). 978-1-4577-1415-3/ © 2012 IEEE
[2] Steven Young, Itamar Arel, Thomas P. Karnowski, Derek Rose, "A Fast and Stable Incremental Clustering Algorithm", April 12-14, 2010, PP 204-209
[3] M. Srinivas and C. Krishna Mohan, "Efficient Clustering Approach using Incremental and Hierarchical Clustering Methods", July 18-23, 2010, PP. 1-7, 978-1-4244-8126-2/10 2010 IEEE
[4] A. K. Jain, M. N. Murty, and P. J. Flynn, "Data clustering: A review", ACM Computing Survey (CSUR), 31(3):264-323, 1999.
[5] P.Vijaya, M. N. Murthy and D.K. Subramanian. Leaders-Sub leaders: "An efficient hierarchical clustering algorithm for large data sets". Pattern Recognition Letters 25
[6] M. Perkowitz and 0. Etzioni, "Adaptive web sites: An ai challenge". 'In Proceedings of IJCAI-97. Volume 1, 1997
[7] M. Perkowitz and 0. Etzioni. "Towards adaptive web sites: Conceptual framework and case study". Artificial Intelligence, 118245 - 275, 2000
[8] T. Toolan and N. Kusmerick. "Mining web logs for personalized site maps". In Proceedings of the Intentional Conference on Web Information System Engineering, volume I, 2002
[9] ChengRuLin and Ming-Syan Chen "A Robust and Efficient Clustering Algorithm based on Cohesion Self- Merging". In Proceedings of the Intentional Conference on SIGKDD '02 Edmonton, Alberta, Canada 2002 ACM
[10] Yue Xu "Hybrid Clustering with Application to Web Mining". In Proceedings of the IEEE Intentional Conference 2005
[l1] Barges and M. Levene. "Data mining of user navigation patterns". In Proceedings of the Web Usage Analysis and User Profiling, volume I, pages 31-36, 1999.