A Novel K means Clustering Algorithm for Large Datasets Based on Divide and Conquer Technique

Rajesh Ahirwar

Assistant Professor
Department of Computer Science and Engineering
Technocrats Institute of Technology
Bhopal, (M.P.), INDIA

Abstract: In this paper we propose an efficient algorithm that is based on divide and conquers technique for clustering the large datasets. In our research work we have applied divide and conquer technique on partitions of the large datasets and we have used squared Euclidean distance for measuring the similarity between data points. The partitioning of datasets is done according to the number of clusters desired. Finally clusters are obtained from each partition of the dataset and we merge those clusters to get more precise clusters. Our proposed technique uses two phases with seven steps for clustering the large datasets. The advantage of using divide and conquer technique is that the large datasets which require a large amount of physical memory to load into the system can also be clustered using our proposed algorithm as it requires a small amount of physical memory because the clustering is done on parts of the dataset. Finally we have used three performance measures namely Fmeasure, purity and entropy to compare our results from the existing algorithms. The results have shown that our approach is much better than existing algorithms.

Keywords: divide and conquer, kmeans algorithm, novel kmeans, partition approach

I. Introduction

Clustering is a fundamental component of real-world problems in nearly every computational discipline, probably in large part due to the human tendency to use categorization as a tool for understanding data. Clustering is primarily used for two purposes. First, clusters provide compact approximate density representations for multimodal or difficult-to-describe distributions. Second, clustering is used to recover underlying categories in data. The most popular and the simplest partitional algorithm is K-means. The disadvantage of k-means clustering are difficulty in comparing quality of the clusters produced (e.g. for different

Initial partitions or values of K affect outcome), fixed number of clusters can make it difficult to predict what K should be. Different initial partitions can result in different final clusters.

The k-means clustering algorithm consists of two separate phases: the first phase is to describe *k* centroids, single for each cluster. The next phase is to take each point belonging to the given data set and associate it to the nearest centroid. When all the points are included in some clusters, the first phase is completed and an early grouping is done. At this point we need to recalculate the new centroids, as the inclusion of new points may lead to a change in the cluster centroids. Once we find *k* new centroids, a new binding is to be created between the same data points and the nearest new centroid, generating a loop. As a result of this loop, the *k* centroids may change their position in a step by step manner. Eventually, a situation will be reached where the centroids do not move anymore. This signals the convergence of clustering. The k-means algorithm is effective in producing clusters for many practical applications in emerging areas like Bioinformatics. But the computational complexity of the original k means algorithm is very high. Moreover, this algorithm results in different types of clusters depending on the random choice of initial centroids.

Typically, the square-error criterion is used, defined as

$$\sum_{i=1}^{k}\sum_{p\varepsilon C_i} |p - m_i|2$$

where *E* is the sum of the square error for all objects in the data set; ***p*** is the point in space representing a given object; and ***mi*** is the mean of cluster *Ci* (both ***p*** and ***mi*** are multidimensional). In other words, for each object in each cluster, the distance from the object to its cluster center is squared, and the distances are summed. This criterion tries to make the resulting *k* clusters as compact and as separate as possible.

Algorithm: *k*-means.

The *k*-means algorithm for partitioning, where each cluster's center is represented by the mean value of the objects in the cluster. The steps of the original kmeans algorithm are described as follows-

Input:

k: the number of clusters,
D: a data set containing *n* objects.

Output: A set of *k* clusters.

Method:

(1) Arbitrarily choose *k* objects from *D* as the initial cluster centers;
(2) Repeat
(3) (Re)assign each object to the cluster to which the object is the most similar, based on the mean value of the objects in the cluster;
(4) Update the cluster means, i.e., calculate the mean value of the objects for each cluster;
(5) Until no change

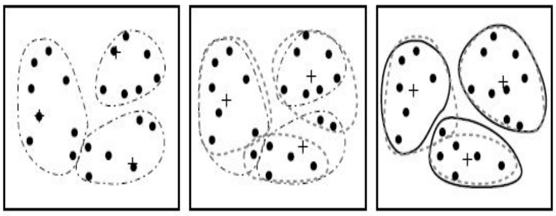

Fig.1. Example of clusters by kmeans

II. Literature Survey

In 2006, David Cheng, Ravi Kannan, Santosh Vempala and Grant Wang presented a divide-and-merge methodology for clustering a set of objects that combines a top down “divide” phase with a bottom-up “merge” phase. For the divide phase, which produces a tree whose leaves are the elements of the set, they suggested an efficient spectral algorithm. The merge phase quickly finds the optimal partition that respects the tree for many natural objective functions, e.g., k-means, min-diameter, min-sum, correlation clustering.

In 2006, Fahim A.M, Salem A.M., Torkey F.A., Ramadan M.A. presented a simple and efficient clustering algorithm based on the *k*-means algorithm, which they called enhanced *k*-means algorithm. This algorithm requires a simple data structure to keep some information in each iteration to be used in the next iteration. Their approach improved the computational speed of the *k*-means algorithm by the magnitude in the total number of distance calculations and the overall time of computation.

In 2007, Nicholas O. Andrews and Edward A. Fox considered the problem of reducing a potentially very large dataset to a subset of representative prototypes. Rather than searching over the entire space of prototypes, they divided the data into balanced clusters using bisecting k-means and spectral cuts, and then find the prototypes for each cluster by affinity propagation.

In 2009, Madjid Khalilian, Farsad Zamani Boroujeni, Norwati Mustapha, and Md. Nasir Sulaiman focused on the fact that most clustering techniques ignore the fact about the different size or levels – where in most cases, clustering is more concern with grouping similar objects or samples together ignoring the fact that even though they are similar, they might be of different levels. For really large data sets, data reduction should be performed prior to applying the data-mining techniques which is usually performing dimension reduction, and the main question is whether some of these prepared and preprocessed data can be discarded without sacrificing the quality of results. Existing clustering techniques would normally merge small clusters with big ones, removing its identity. They proposed a method which uses divide and conquer technique to improve the performance of the K-Means clustering method.

In 2009, Jirong Gu, Jieming Zhou, Xianwei Chen proposed the method in which the number of clusters is predefined and the technique is highly dependent on the initial identification of elements that represent the clusters well. If the numbers of sample data are too large, it may let the cluster members unstable. Another problem is selecting initial seed points because clustering results always depend on initial seed points and partitions. To prevent this problem, Refining initial points algorithm is provided; it can reduce execution time and improve solutions for large data by setting the refinement of initial conditions.

In 2010, Madjid Khalilian, Norwati Mustapha, MD Nasir Suliman, and MD Ali Mamat suggest that having both accuracy and efficiency for high dimensional data sets with enormous number of samples is a challenging arena. For really large and high dimensional data sets, vertical data reduction should be performed prior to applying the clustering techniques which is performing dimension reduction, and the main disadvantage is sacrificing the quality of results. However, because dimensionality reduction methods inevitably cause some loss of information or may damage the interpretability of the results, even distorting the real clusters, extra caution is advised. Existing clustering techniques would normally apply in a large space with high dimension; dividing big space into subspaces horizontally can lead us to high efficiency and accuracy. They proposed a method that uses divide and conquer technique with equivalency and compatible relation concepts to improve the performance of the K-Means clustering method for using in high dimensional datasets.

Ding-yin XIA, Fei WU, Xu-qing Zhang and Yue-ting Zhuang presents two variants of AP for grouping large scale data with a dense similarity matrix. The local approach is partition affinity propagation (PAP) and the global method is landmark affinity propagation (LAP). PAP passes messages in the subsets of data first and then merges them as the number of initial step of iterations; it can effectively reduce the number of iterations of clustering. LAP passes messages between the landmark data points first and then clusters non-landmark data points.

III. Proposed Algorithm

In this section we introduced the proposed idea to find the clusters in large datasets. Since in each iteration kmeans algorithm computes the distance between data points and all centroids, this is computationally very expensive for large datasets so we are using divide and conquer technique to reduce the number of computations which results in less execution time.

Steps of Proposed Algorithm: -

Input: Dataset *D* and number of clusters, *k.*

Output: FCL, A set of k clusters.

Method:

Divide Phase:

- **Step 1**: Partition the dataset $\boldsymbol{D}_{rxd}$ into *k* parts as an average. At each iteration, 1<*k*<*N*/ (4*C*), where *C* is the maximal number of clusters prospected and N is the length of the dataset.
- **Step 2:** Apply Kmeans algorithm on each of the partition of the dataset to get the initial clusters of each partition.

- **Step 3:** Calculate the means of each clusters in all partitions separately.
- **Step 4:** Calculate the average of all means within a partition**.**
- **Step 5:** Repeat step 4 for all partitions.

Merge Phase:

- **Step 6:** Now taking the average means from each partition as the final centroids for the final clustering, calculate the square of Euclidean distance of each data point in the dataset to the above average means.
- **Step 7:** Finally based on the minimum distance criterion, assign each data points to the cluster to which it has minimum distance. So as to minimize the within groups sum of squared errors.

IV. Experimental Results

In the research work, we have evaluated Fmeasure Purity and Entropy for comparing the results between kmeans algorithm and our proposed algorithm. To measure these performance parameters we have used ten data sets namely Iris dataset, wine dataset, Segmentation dataset, Balance dataset, Yeast dataset, Pendigits dataset, Waveform Database Generator Dataset, Optdigits dataset, Shuttle_trn and Shuttle_tst dataset. As per the experimental results, the proposed clustering algorithm clusters the dataset with great accuracy. The main purpose of the Proposed Algorithm is to improve accuracy for the dataset of any size. Our proposed algorithm achieves this goal for datasets where number of data points is less than 50000(N<50000). The Proposed Algorithm clusters data, using squared Euclidean distance as the distance measure to calculate the similarity between two clusters. Clusters are each represented by a cluster center (the "centroid"). Our proposed algorithm is iterative and converges when no data points moves from one cluster to another. It is concluded from the experimental result that Proposed Algorithm is a good clustering algorithm as it gives better clustering with higher accurate results.

Dataset	Fmeasure	
	Kmeans	Proposed
Iris	0.91	0.97
Wine	0.94	0.98
Segmentation	0.71	0.72
Balance	0.72	0.75
Yeast	0.65	0.75
Pendigits	0.70	0.71
Waveform database generator	0.91	0.92
Optdigits	0.68	0.69
Shuttle_tst	0.85	0.92
Shuttle_trn	0.73	0.81

Table1. Description of datasets

DESCRIPTION OF OBSERVATION TABLES AND GRAPHS

In this section, the explanation regarding the observation tables and graphs is revealed. We have made observation on three performance measures Fmeasure, Purity and Entropy and the observed values are tabulated in three tables for Fmeasure, purity and entropy respectively. Table 2 contains the calculated value of Fmeasure on ten datasets with kmeans algorithm and our proposed algorithm. Similarly, Table 3 and Table 4 contain the values of Purity and Entropy on ten datasets for kmeans algorithm and our proposed algorithm.

The corresponding graphs for Fmeasure, Purity and Entropy are given below. Figure 2, Figure 3 and Figure 4 are the graphs for Fmeasure, Purity and Entropy on ten datasets for kmeans algorithm and our proposed algorithm. From the graphs, it is clearly visible that the performance of our proposed algorithm is better than kmeans algorithm for large datasets.

From the above graphs, it is also clearly visible that the performance of our proposed algorithm is much better than the kmeans algorithm for large datasets as well as for small and medium datasets.

S. No	Dataset	No. of records	No. of attributes
1.	Iris	150	4
2.	Wine	178	13
3.	Segmentation	210	19
4.	Balance	625	3
5.	Yeast	1484	8
6.	Pendigits	3498	17
7.	Waveform Database Generator	5000	21
8.	Optdigits	5620	64
9.	Shuttle_tst	14500	9
10.	Shuttle_trn	43500	9

Table2. Observation Table for Fmeasure

Dataset	Purity	
	Kmeans	Proposed
Iris	0.92	0.97
Wine	0.95	0.98
Segmentation	0.70	0.72
Balance	0.83	0.84
Yeast	0.83	0.77
Pendigits	0.70	0.72
Waveform database generator	0.91	0.92
Optdigits	0.69	0.70
Shuttle_tst	0.87	0.94
Shuttle_trn	0.84	0.85

Table3. Observation Table for Purity

Dataset	Entropy	
	Kmeans	Proposed
Iris	0.18	0.08
Wine	0.13	0.02
Segmentation	0.31	0.27
Balance	0.32	0.31
Yeast	0.14	0.16
Pendigits	0.24	0.24
Waveform database generator	0.22	0.21
Optdigits	0.24	0.25
Shuttle_tst	0.12	0.10
Shuttle_trn	0.17	0.08

Table4. Observation Table for Entropy

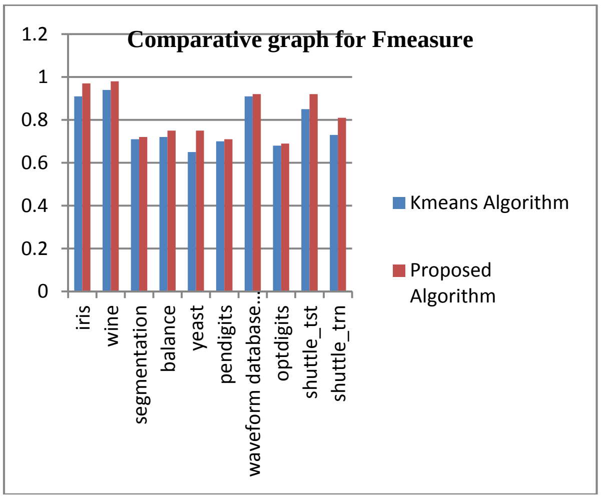

Figure2.Performance comparison based on Fmeasure

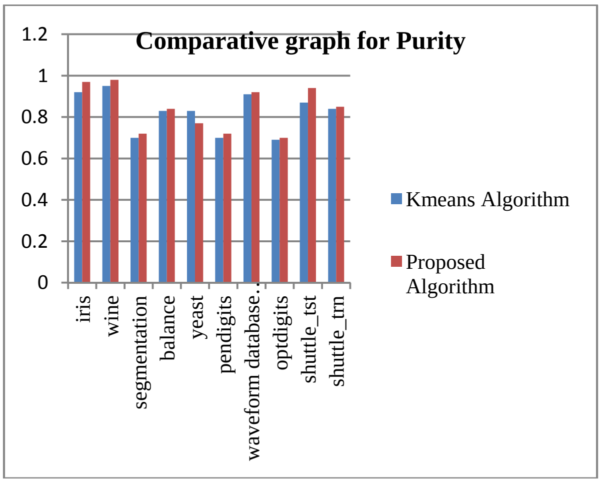

Figure3. Performance comparison based on Purity

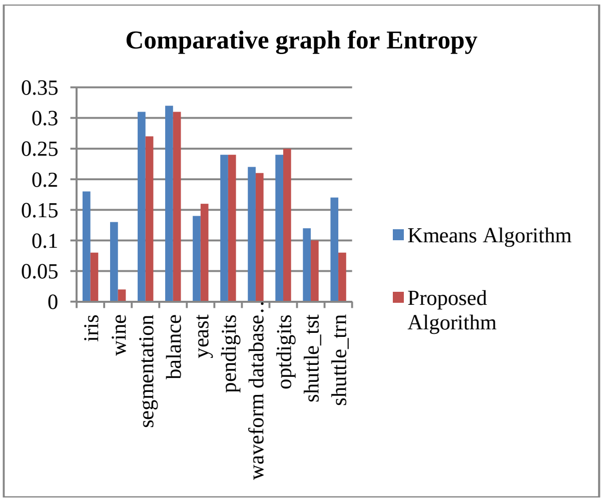

Figure4. Performance comparison based on Entropy

V. Conclusion And Future Work

The proposed research work focuses on the effects of divide and conquers technique on large datasets. We have used divide and conquer technique to solve the problems of the kmeans algorithm. To measure the performance of our proposed method, we have used three performance parameters, Fmeasure, purity and entropy. We have also used ten datasets to implement the concept of our proposed method and to verify the results.

From the experimental results on the different sizes of dataset, it is concluded that the proposed method based on divide and conquer technique correctly identifies the data points and assign the data points to the best cluster so that the intra cluster distance is minimize and inter cluster distance is maximize. The values of Fmeasure, purity and entropy are improved in this research work and that is proved by the experimental results.

References

[1] Ding-yin Xia, Fei Wu, Xu-qing Zhang, Yue-ting Zhuang,(2008).” Local and global approaches of affinity propagation clustering for large scale data”, Journal of Zhejiang University Science A, Xia et al. / J Zhejiang Univ Sci A 2008 9(10):1373-1381.

[2] Jain, A. K., Murthy, M. N., & Flynn, P. J. (1999). “Data clustering: A review”. ACM Computing Surveys, 31(3), 264-323.

[3] Jain & Dubes, (1988) Jain, Anil K., & Dubes, Richard C. 1988. “Algorithms for clustering data”. Prentice Hall.

[4] Clustering, by Rui Xu and Donald C. Wunsch, II Copyright c 2009 Institute of Electrical and Electronics Engineers

[5] Guha, S., Rastogi, R., Shim, K., 1998. Cure: An Efficient Clustering Algorithm for Large Databases. Proc. ACM SIGMOD Int. Conf. on Management of Data. Seattle, WA, p.73-84.

[6] T. Zhang, R. Ramakrishnan, and M. Livny: “BIRCH: An efficient data clustering method for very large databases.” Proc. ACM-SIGMOD int. Conf. Management of Data (SIGMOD 96), Montreal, Canada, June, Page: 103-114, 1996.

[7] K. A. Abdul Nazeer and M. P. Sebastian, “Improving the accuracy and efficiency of the k-means clustering algorithm,” in International Conference on Data Mining and Knowledge Engineering (ICDMKE), Proceedings of the World Congress on Engineering (WCE-2009), Vol 1, July 2009, London, UK

[8] David Cheng, Ravi Kannan, Santosh Vempala, and Grant Wang. A divide-and-merge methodology for clustering. ACM Trans. Database Syst., 31(4):1499{1525, 2006.

[9] Nicholas O. Andrews and Edward A. Fox, “Clustering for Data Reduction: A Divide and Conquer Approach”

[10] Madjid Khalilian, Farsad Zamani Boroujeni, Norwati Mustapha, Md. Nasir Sulaiman "K-Means Divide and Conquer Clustering", DOI 10.1109/ICCAE.2009.59

[11] R. Varshavsky, D. itorn and M. Linial, "cluster algorithm optimizer: A framework for large datasets", ISBRA pp 85- 96, Springer 2007.

[12] W. Tang, H. Xiong, S. Zhang, J. Wu, "Enhancing semi supervised clustering", KDD07 California USA , ACM 2007.

[13] Madjid Khalilian, Norwati Mustapha, MD Nasir Suliman, MD Ali Mamat "A Novel K-Means Based Clustering Algorithm for High Dimensional Data Sets" ,In Proceedings of the International Multiconference of engineers and computer scientists 2010 Vol I, IMECS 2010 March 17-19,2010 Hongkong.

[14] A. M. Fahim, A. M. Salem, F. A. Torkey and M. A. Ramadan, "An Efficient enhanced k-means clustering algorithm," journal of Zhejiang University, 10(7): 16261633, 2006.

[15] Chen Zhang and Shixiong Xia, " K-means Clustering Algorithm with Improved Initial center," in Second International Workshop on Knowledge Discovery and Data Mining (WKDD), pp. 790-792, 2009

[16] Huang, Z., 1997. A Fast Clustering Algorithm to Cluster Very Large Categorical Data Sets in Data Mining. Proc. SIGMOD Workshop on Research Issues on Data Mining and Knowledge Discovery. Tech. Report 97-07, Dept. of
CS, UBC.

[17] T. Zhang, R. Ramakrishnan, M. Livny: "BIRCH: An efficient data clustering method for very large databases." roc. ACM-SIGMOD int. Conf. Management of Data (SIGMOD 96), Montreal, Canada, June, Page: 103-114, 1996.

[18] William Peter, John Chiochetti, Clare Giardina, "New Unsupervised Clustering Algorithm for Large Datasets." Proc. Ninth ACM-SIGMOD int. conference on Knowledge discovery and data mining, Washington, D.C, Pages: 643 – 648, 2003.

[19] Wittman, T., 2005. MANIfold Learning Matlab Demo.
Http://www.math.umn.edu/~wittman/research.html

[20] Iris dataset: http://archive.ics.uci.edu/ml/machine-learning-databases/iris/

[21] Wine dataset: http://archive.ics.uci.edu/ml/machine-learning-databases/wine/

[22] Shuttle dataset: http://archive.ics.uci.edu/ml/machine-learning-databases/statlog/shuttle/

[23] Optdigits dataset: http://archive.ics.uci.edu/ml/machine-learning-databases/optdigits/

[24]Waveform Database Generator dataset: http://archive.ics.uci.edu/ml/machine-learning-databases/waveform/

[25] Pendigits dataset: http://archive.ics.uci.edu/ml/machine-learning-databases/pendigits/

[26] Yeast dataset: http://archive.ics.uci.edu/ml/machine-learning-databases/yeast/

[27] Balance scale dataset: http://archive.ics.uci.edu/ml/machine-learning databases/balance-scale/

[28] Image Segmentation dataset: http://archive.ics.uci.edu/ml/machine-learning databases/image/