A FRAMEWORK FOR ENHANCING THE EFFICIENCY OF K-MEANS CLUSTERING ALGORITHM TO AVOID FORMATION OF EMPTY CLUSTERS

J. James Manoharan[1], S. Hari Ganesh[2]

[1]Asst. Professor in Dept. of Computer Applications, Bishop Heber College (Autonomous),Tiruchirappalli, India
[2]Asst. Professor & Head, Dept. of Computer Applications, Bishop Heber College (Autonomous),Tiruchirappalli,,India
Email: james_7676@yahoo.com, hariganesh17@gmail.com

Abstract

K-means algorithm is one of the most commonly used partition based clustering algorithms. K-means has recently been predictable as one of the best algorithms for clustering large data. The formation of empty clusters is one of the most important issues in K-means algorithm. This problem is considered insignificant when the data set is small and can be solved by executing the algorithm for a number of iterations. In some cases, the K-means is used as an essential part in some scientific applications like medical database; the empty cluster problem may affect the behavior of the system along with the performance of the algorithm. In this research article we propose a framework for enhancing the efficiency of K-means algorithm to avoid the formation of empty clusters using data structure. Experimental results show that the enhanced method can effectively improve the speed of clustering, accuracy and avoiding the formation of empty clusters.

Keywords: clustering; k-means Algorithm; Enhanced k-means Algorithm;

INTRODUCTION

Clustering algorithms are frequently useful in different fields like pattern recognition, data mining, learning theory etc[11]. The k-means algorithm is a well-known partition based unsupervised clustering algorithm [1]. It can be used to cluster very large data sets in many real world applications such as image processing, marketing research, etc. due its simplicity and efficiency. The *k*-means discover a locally optimal solution by minimizing a distance measure between each data and its nearest cluster center [2].

In this research article, we have proposed a framework for k-means algorithm which eliminates the problem of generation of empty clusters and increase the efficiency of traditional k-means algorithm. The framework is composed of 3 phases; choosing initial k-centroids phase, calculate the distance phase and recalculating cluster center phase. In short, in the choosing initial k-centroids phase the initial cluster centers have obtained using divide-and-conquer method. In calculate the distance phase the distance between each data objects and cluster centers in each iteration could be calculated using Linear data structure List. Finally, in the recalculating cluster center phase to modify the center vector updating procedure of the basic *k*-means that denies the possibility of empty clusters. Our main contributions in this article are:

1. Proposing an efficient clustering method that overcomes the known problems of traditional k-means clustering algorithm. The proposed technique uses a divide-and-conquer method to choose the best possible number of clusters and properly initializing the k centroids at the initial stage.
2. Developing an accurate clustering algorithm by applying *Euclidean distance* and Linear Data Structure List. The usage of Data structure List to reduce the execution time through less computation of the distance of each data object in clusters. We experimentally prove both the accuracy and efficiency the enhanced algorithm.
3. Employing a new method for calculating new clusters' centroids to improve the performance of our algorithm.

The remaining part of the article is described as follows: Section2 presents a related work of enhanced k-means clustering algorithm. Section 3 presents our proposed clustering algorithm. Section 4 shows our experimental results. Finally, section 5 concludes the paper.

II. RELATED WORK

Clustering analysis has become a good-looking research area and many well approaches have been projected [12]. The general definition of clustering is usually depending on the clustering goal and type of data to be clustered. Different and scalable clustering algorithms have been developed [7]. The k-means clustering method is a generally used clustering technique; there have been a number of research works that illustrate competitive algorithms for the k-means clustering problem [8–10].

In [1] author proposed an efficient algorithm to compute new cluster centers for each iterative step in the *k*-means clustering algorithm. After performing center initialization the data items are assigned to the concerned clusters. The process of re-initialization takes place when an empty cluster is found at this early stage and the procedure is repeated until all non-empty initial clusters are produced. The limitation is that this method are time consuming and may not be appropriate by keeping the *k*-means inherently simple structure and the improvements are based on the optimization generation of the problem and also, a novel iterative method.

In [7] a data clustering approach using customized *k*-means algorithm based on the enhancement of the sensitivity of initial center (seed point) of clusters is proposed. It divides the whole space into different division and calculates the occurrence of data point in each segment. Obtained division which illustrates highest frequency of data point will have the highest probability that include the centroid of cluster. The number of centroids (k) will be given by the user in the same manner like the traditional *k*-means algorithm and the number of partition will be k*k.

The maximum frequency of data point is same in different section and the higher bound of section crosses the threshold `k' then combining of various segments become compulsory and then take the highest k segment for calculating the initial centroid of clusters[6]. They also define a threshold distance for every cluster's centroid to evaluate the distance between data items and centroids with this threshold distance through which we can minimize the calculation effort during calculation of distance between cluster' centroids and data points. The time taken for numerical calculation and complexity of traditional k-means algorithm will decreased by using the enhanced algorithms. The advantage is that center and comparative weight variation is used as distance measure and the issues are vagueness, incompleteness and uncertainty.

The main purpose of the paper [8] is to propose a modified *k*-means algorithm with the aim of improving the efficiency of cluster value and to fix the best possible number of cluster. The proposed method works well for both the known number of clusters in advance and unknown number of clusters. In addition to suggesting user has the freedom to input the minimum number of clusters as required or fix the number of clusters. In the earlier case it works same as traditional *k*-means algorithm. The enhanced algorithm computes the new cluster centers in each iteration by incrementing the cluster counter by one until it satisfies the validity of cluster quality.

In this paper [9] select the number of clusters for the *k*-means algorithm has been proposed. This enhanced method can propose multiple values of *k* to users for cases while different clustering output could be find out with different required levels of detail. The method could be take more time for computation for large data sets because it requires a number of applications of the *k*-means algorithm before it can suggest a guide value for *k*.

Performance analysis of iterative k-means clustering algorithms which converge too many local minima highly depends on initial cluster centroids. In general initial cluster centers are selected randomly. In this paper [7] author suggest an algorithm to calculate initial cluster centroids for *k*-means clustering and the enhanced algorithm is highly depends on two observations that a number of of the patterns are very similar to each other and the same cluster membership irrespective to the preference of initial cluster centers. Furthermore, an individual attribute may give a number of information about initial cluster center. The initial cluster centers calculated using this method are found to be very close to the preferred cluster centers, for k-means iterative clustering algorithms. It is appropriate to clustering algorithms for continuous data items. However it is theoretically simple and quite simple to code, this algorithm complicated during execution which turn out to be an issue.

Authors in [4] proposed a method to seed the initial centers for k-means. Their design is based on the intuition of scattering the k initial cluster centroids not closer to each other, the first cluster center is chosen consistently at random from the data points that are being grouped, after that each subsequent centroid is selected from the remaining data points and its probability is proportional to its distance squared to the position closest cluster center. Adaptive k-means is a different proposed extension for the traditional k-means. Adaptive k-means clustering method aims to overcome the dependence of traditional k-means on the selection of the number of clusters, and on the initialization of the k centroids. The authors in [10] devise an algorithm to resolve both issues.

III. PROPOSED EXTENDED K-MEANS ALGORITHM

Clustering is a most important data mining task that aims to partition a given set of data items into groups such that data objects inside a cluster would have high level of similarity to each other and low similarity to data objects in other clusters [5].

In this research article, we propose a framework for avoiding the formation of empty clusters and enhance the efficiency of traditional k-means algorithm by using various Data structure concepts. The proposed approach is based on the widely used traditional k-means clustering algorithm and is composed of 3 phases; choosing initial centroids phase; calculate the distance phase and recalculating cluster center phase, which we discuss in more details in the remainder of this section. The enhanced framework for k-means clustering algorithm is depicted in figure 1.

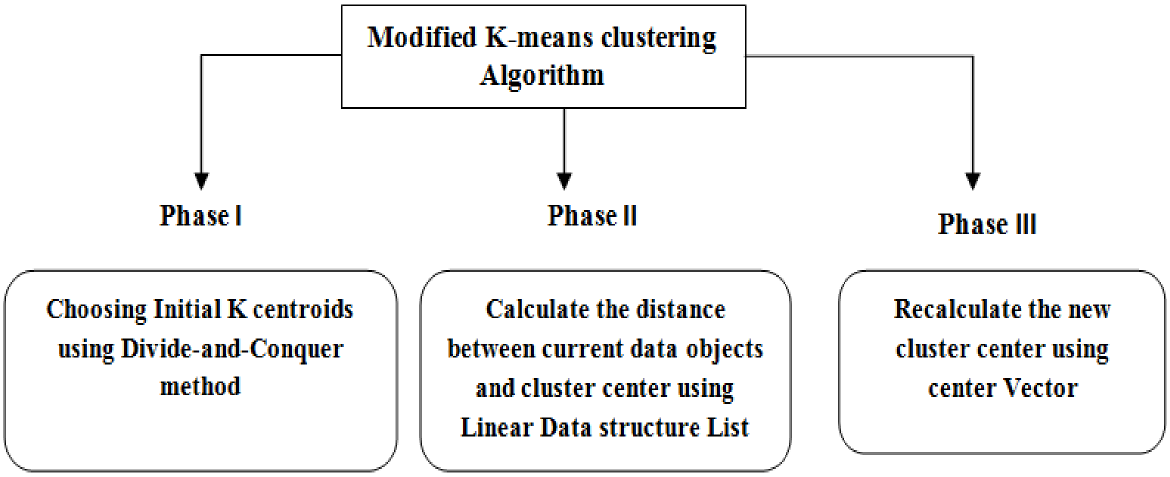

Modified K-means clustering Algorithm
Phase I
Phase II
Phase III
Choosing Initial K centroids using Divide-and-Conquer method
Calculate the distance between current data objects and cluster center using Linear Data structure List
Recalculate the new cluster center using center Vector

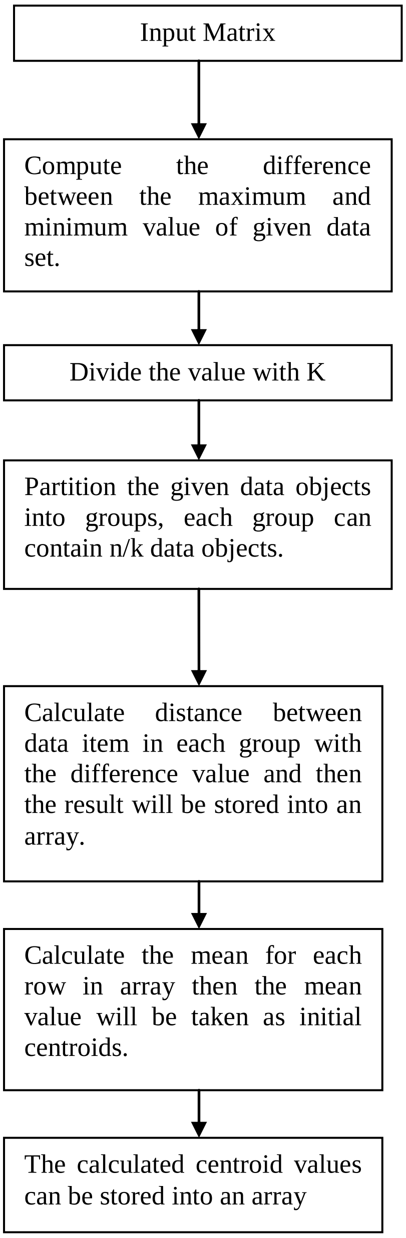
Fig. 1. An Enhanced Framework of K-means clustering algorithm

A. *Phase I (Choosing Initial K-Centroids):*

In this phase the initial k-centroids can be calculated by using Divide-and-Conquer method. Divide the given data objects into groups. Each group can contain items and then store each group into two dimensional arrays in row-wise order. Calculate the distance between data in each group and the value calculated and then the result will be stored into array. Finally calculate mean for each row in the array [14]. This value will be taken as initial centroids that can be stored into an array. Figure 2 illustrate the working principle of phase I.

B. *Phase II (Calculating the Distance):*

The distance between each data object and cluster centers in each iteration could be calculated using a linear data structure List, which store information about the data object cluster and its distance in each iteration. The stored value could be given as an input to the next iteration for comparisons. After that we compute the distance between the current data object and the new centroid, suppose if the calculated distance is lesser than or equal to the distance of the old centroid then the data item keep on same cluster. As a outcome, it is not required to calculate the distance from this data object to the other k-1centroids.In each iteration some data points still remain in the existing cluster that means some parts of the data points need not be calibrated and saving the total time of calculating the distance, thereby increasing the efficiency of the algorithm [13]. Figure 3 illustrate the working principle of phase II.



Input Matrix

Compute the difference between the maximum and minimum value of given data set.

Divide the value with K

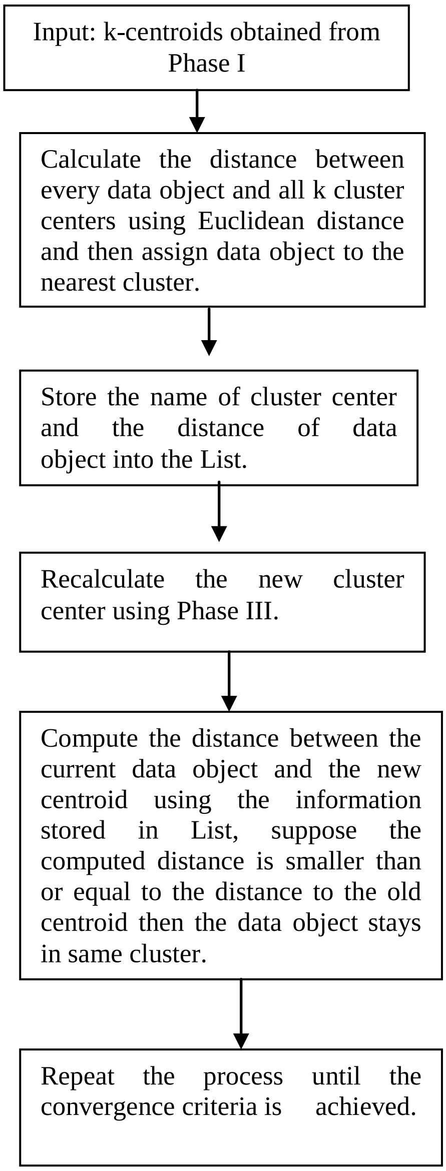
Partition the given data objects into groups, each group can contain n/k data objects.

Calculate distance between data item in each group with the difference value and then the result will be stored into an array.

Calculate the mean for each row in array then the mean value will be taken as initial centroids.

The calculated centroid values can be stored into an array

Fig. 2. Choosing initial k centroids using Divide-and-Conquer method



Input: k-centroids obtained from Phase I

Calculate the distance between every data object and all k cluster centers using Euclidean distance and then assign data object to the nearest cluster.

Store the name of cluster center and the distance of data object into the List.

Recalculate the new cluster center using Phase III.

Compute the distance between the current data object and the new centroid using the information stored in List, suppose the computed distance is smaller than or equal to the distance to the old centroid then the data object stays in same cluster.

Repeat the process until the convergence criteria is achieved.

Fig. 3. Calculating distance between data object and cluster center using Linear Data structure List

C. Phase III (Recalculating Cluster Center):

Formation of empty cluster can be avoided in this phase. In case of the enhanced *k*-means algorithm; the computation of new center vectors differs from that in the *k*-means clustering algorithm. Now, the old center vectors are assumed to be members of the concerned clusters with the allocated data items. In this scheme, we deny the formation of empty clusters. In the traditional *k*-means algorithm, iteration starts with a set of old center vectors $\mathbf{Z}_k{}^{(old)}$, the data items are distributed among clusters based on minimum Euclidean distance, after that a set of new cluster centers $Z_k{}^{(new)}$ is generated by averaging the data items. This center vector updating process in the traditional *k*-means algorithm can be mathematically illustrated as:

$$\mathbf{z}_k^{(new)} \leftarrow \frac{1}{n_k}\{\sum_{x_j \in C_k}(\mathbf{x}_j)\} \quad \text{--- (1)}$$

where, n_k is the number of elements in cluster C_k. If the new centers $\mathbf{z}_k^{(new)}$ do not match exactly with the old centers $\mathbf{z}_k^{(old)}$, the *k*-means algorithm enters into the next iteration assuming $\mathbf{z}_k^{(new)}$ as $\mathbf{z}_k^{(old)}$. The method of computing new centroids differ from traditional k-means algorithm, In enhanced algorithm, the old center vectors $\mathbf{z}_k^{(old)}$ are assumed to be members of the particular clusters along with the allocated data items. This method denies the formation of empty clusters. Therefore, center vector updating procedure in enhanced *k*-means can be written as :

$$\mathbf{z}_k^{(new)} \leftarrow \frac{1}{n_k+1}\{\sum_{x_j \in C_k}(\mathbf{x}_j)+\mathbf{z}_k^{(old)}\} \quad \text{----- (2)}$$

The equation (2) indicates that each cluster should always contain at least one element. One may estimate that including of old centroid point $\mathbf{Z}_k^{(old)}$ in computation of new centroid $\mathbf{Z}_k^{(new)}$ can affect the quality of clustering solutions and the rate of convergence to final cluster centers. Hence, within a few iterations the value of old cluster center $\mathbf{Z}_k^{(old)}$ almost immediately becomes well inside the concerned cluster . Figure 4 illustrate the working principle of phase III.

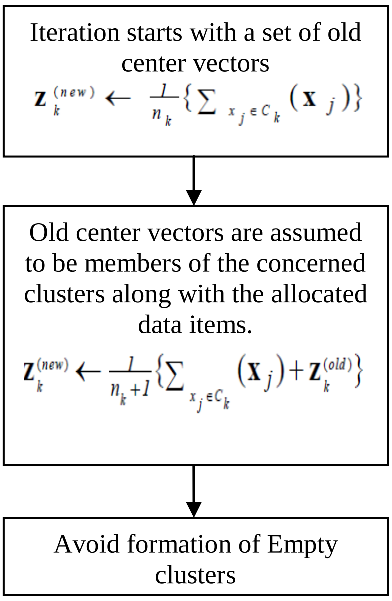

Iteration starts with a set of old center vectors
$\mathbf{z}_k^{(new)} \leftarrow \frac{1}{n_k}\{\sum_{x_j \in C_k}(\mathbf{x}_j)\}$
Old center vectors are assumed to be members of the concerned clusters along with the allocated data items.
$\mathbf{z}_k^{(new)} \leftarrow \frac{1}{n_k+1}\{\sum_{x_j \in C_k}(\mathbf{x}_j)+\mathbf{z}_k^{(old)}\}$
Avoid formation of Empty clusters

Fig. 4. Recalculating new cluster center

The execution steps of the enhanced *k*-means algorithm to form clusters are basically similar to those of the traditional *k*-means algorithm. The processors retain the cluster structures in its own local memory and iterates throughout the steps of the enhanced *k*-means algorithm to calculate a final set of cluster centers **Z**. The execution steps of enhanced k-means algorithm are summarized below.

D. Proof of Convergence for the enhanced k-means Algorithm:

As pointed out in Section III, including of old cluster center $\mathbf{Z}_k^{(old)}$ in computing the new cluster center $Z_k^{(new)}$ can affect the rate of convergence of the enhanced algorithm.

Let us consider that at iteration *t*, data objects are scattered among exact clusters, however center vectors are yet to be stable. Let, at this stage, a certain cluster to be described by $C = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m\}$ and the equivalent center is $\mathbf{Z}^{(t)} = X$ (say), where

$$\mathbf{x} \neq \frac{\mathbf{x}_1 + \mathbf{x}_2 + \cdots + \mathbf{x}_m}{m}$$

due to the initial center effect. So, the following is a sequence of center vectors obtained in the subsequent iterations:

$$Z^{(t)} = X$$

$$Z^{(t+1)} = \frac{\sum_{i=1}^{m} X_i + Z^{(t)}}{m+1} = \frac{\sum_{i=1}^{m} X_i}{m+1} + \frac{X}{m+1}$$

$$Z^{(t+2)} = \frac{\sum_{i=1}^{m} X_i + Z^{(t+1)}}{m+1} = \frac{\sum_{i=1}^{m} X_i}{m+1} + \frac{\sum_{i=1}^{m} X_i}{(m+1)^2} + \frac{X}{(m+1)^2}$$

$$Z^{(t+3)} = \frac{\sum_{i=1}^{m} X_i + Z^{(t+2)}}{m+1} = \frac{\sum_{i=1}^{m} X_i}{m+1} + \frac{\sum_{i=1}^{m} X_i}{(m+1)^2} + \frac{\sum_{i=1}^{m} X_i}{(m+1)^3} + \frac{X}{(m+1)^3}$$

```
Algorithm : The Enhanced K-Means Method
Require: D = {d1, d2, d3... dn} // Set of n data points. k // Number of desired clusters.
Ensure: A set of k clusters.
Steps:
1) Compute the difference between the maximum and minimum value of given data set.
2) Divide the difference value obtained in step 1 with K.
3) Partition the given data objects into G groups, each group can contain n/k data objects.
4) Calculate distance between data item in each group with the difference value obtained in step 2 and then the result
   will be stored into an array GD [i,j], where $1\leq i \leq k$ and $1\leq j\leq n/k$.
5) Calculate the mean for each row in GD then the mean value will be taken as initial centroids.
6) The calculated centroid values can be stored into an array CENT[i] ($1\leq i\leq k$).
7) Calibrate the distance between every data object di (1<=i<=n) and all k cluster centers cj (1<=j<=k) as Euclidean
   distance d ($d_i$, $c_j$) and assign data object $d_i$ to the nearest cluster.
8) For each data object $d_i$, find the nearest center $c_j$ and assign data object $d_i$ to cluster center $c_j$
9) Detect the name of cluster center and the distance of data object $d_i$ to the closest cluster. Then
   this information is stored in list Clu[ ] and the Dis[ ] separately.
        Set Clu[i]=j, j is the name of nearby cluster center.
        Set Dis[i ]=d($d_i$, $c_j$), d($d_i$, $c_j$) is the Euclidean
        distance to the nearest center.
10) Recalculate the cluster center;
          while measure center is not stable do
           begin
               Compute distance $d_{kj} = ||\mathbf{x}j - \mathbf{z}k||^2$ for each k, j where $1 \leq k \leq K$ and $1 \leq j \leq N$, and
               determine members of new K subsets based upon minimum distance to $z^k$ for $1 \leq k \leq K$;
               compute new center $\mathbf{z}^k$ for $1 \leq k \leq K$ using (2);
               Compute new center;
            end
11) Repeat
12) For each data object di (1<=i<=N) Compute its distance to the center of the current closest cluster;
        a)  If this distance <= Dist[i], then assign the data object $d_i$ in the initial cluster;
        b)  Else
               For every cluster center $c_j$, calculate the distance d ($d_i$, $c_j$) of each data object to all the centroid and
               then allocate the data object of data set $d_i$ to the nearest neighboring cluster center $c_j$.
                    Set Clu[i]=j;Set Dis[i]= d($d_i$, $c_j$);
13) Repeat the process until the convergence criteria is achieved.
14) Showing the clustering results.
```

Proceeding in a similar fashion, we get

$$z^{(t+u)}$$

$$= \frac{\sum_{i=1}^{m} x_i}{m+1} + \frac{\sum_{i=1}^{m} x_i}{(m+1)^2} + \cdots + \frac{\sum_{i=1}^{m} x_i}{(m+1)^u} + \frac{x}{(m+1)^u}$$

$$= \frac{\sum_{i=1}^{m} x_i}{m+1}\left[1 + \frac{1}{(m+1)} + \cdots + \frac{1}{(m+1)^{u-1}}\right] + \frac{x}{(m+1)^u}$$

$$= \frac{\sum_{i=1}^{m} x_i}{m+1} \times \frac{1-\left(\frac{1}{m+1}\right)^u}{1-\frac{1}{m+1}} + \frac{x}{(m+1)^u}$$

For large values of *m* or *u*, $1/(m+1)^u \rightarrow 0$. Hence,

$$z^{(t+u)} = \frac{\sum_{i=1}^{m} x_i}{m+1} \times \frac{m+1}{m} = \frac{\sum_{i=1}^{m} x_i}{m}$$

------------------(3)

This is the desired cluster center using traditional k-means algorithm and the additional *u* iterations needed for final result can be represented by the condition $1/(m+1)\,u \rightarrow 0$. For large data sets, even for a small *u* this condition holds. To facilitate, the effect of the initial centers toward convergence is insignificant.

E. Proof of Avoiding the Formation of Empty Clusters:

In the above section, we have shown that the *enhanced k*-means algorithm converges to the traditional *k*-means centers. In the following we illustrate that the *enhanced k*-means algorithm generates non-empty clusters only as opposed to the traditional *k*-means algorithm.

The problem of formation of empty clusters occurs when any two or more of the initial center vectors $Z_1^{(0)}$, $Z_2^{(0)}$, …. , $Z_k^{(0)}$ are either equal or very close to each other. In such a state, after allocating data items to particular clusters then the data items will be allocated to only one of the clusters with almost equal centers and the other cluster centers remain empty.

Let D = {x_1, x_2, …, x_m, y_1, y_2, …, y_n } be a data set with (*m*+*n*) data items forming two well separated clusters X and Y. If we apply serial *k*-means algorithm to dividing this data set into two clusters then we obtain the following center vectors

$$Z_1 = \frac{x_1 + x_2 + \cdots + x_m}{m} \text{ and } Z_2 = \frac{y_1 + y_2 + \cdots + y_n}{n}.$$

F. Execution using enhanced k-means:

In this case, the center vectors will become

$$Z_1^{(1)} = \frac{x_1 + x_2 + \cdots + x_m + y_1 + y_2 + \cdots + y_n + Z_1^{(0)}}{m+n+1} \text{ and } Z_2^{(1)} = Z_2^{(0)}.$$

Thus after the first iteration, $Z_1^{(1)} \neq Z_2^{(1)}$. Now, in a common situation, when the number of clusters *K* is greater than 2, we can arrive at the equivalent conclusion according to the following logic.

Let Z_1 , Z_2, …, Z_k be the *K* centroids, which are initialized to $Z_1^{(0)}$, $Z_2^{(0)}$, ……, $Z_k^{(0)}$ respectively. In the worst case, we can assume that all the clusters are initialized to the same value, i.e.,$Z_1^{(0)} = Z_2^{(0)} \ldots\ldots Z_k^{(0)} = Z^{(0)}$. Then, after the first iteration, we have

$$Z_1^{(1)} \neq Z^{(0)}$$

and

$$Z_2^{(1)} = Z_3^{(1)} = \ldots\ldots = Z_k^{(1)} = Z^{(0)}$$

because all the data items will be occupied by a single cluster. After the second iteration,

$$Z_1^{(1)} \neq Z_2^{(1)} \neq Z^{(0)}$$

and

$$Z_3^{(1)} = Z_4^{(1)} = \ldots\ldots = Z_k^{(1)} = Z^{(0)}$$

As a result, after the *K*th iteration, we will get *K* different center vectors and all non-empty clusters can be guaranteed.

In the previous instance, for a two cluster situation we have,

$$\mathbf{z}_1^{(0)} = \mathbf{z}_2^{(0)} = \frac{\mathbf{x}_1 + \mathbf{x}_2 + \cdots + \mathbf{x}_m + \mathbf{y}_1 + \mathbf{y}_2 + \cdots + \mathbf{y}_n}{m+n}$$

Let, in this case, all the elements are assigned to the first cluster keeping the other empty. Then, we have

This means that there is no change of the cluster centers even after the iteration is over. The same logic is equally appropriate for $K > 2$ also.

$$\mathbf{z}_1^{(0)} = \frac{\mathbf{x}_1 + \mathbf{x}_2 + \cdots + \mathbf{x}_m + \mathbf{y}_1 + \mathbf{y}_2 + \cdots + \mathbf{y}_n}{m+n}$$

$$\mathbf{z}_1^{(1)} = \frac{\mathbf{x}_1 + \mathbf{x}_2 + \cdots + \mathbf{x}_m + \mathbf{y}_1 + \mathbf{y}_2 + \cdots + \mathbf{y}_n + \mathbf{z}_1^{(0)}}{m+n+1}$$

$$= \frac{\mathbf{x}_1 + \mathbf{x}_2 + \cdots + \mathbf{x}_m + \mathbf{y}_1 + \mathbf{y}_2 + \cdots + \mathbf{y}_n}{m+n} = \mathbf{z}_1^{(0)}$$

IV. EXPERIMANTAL RESULTS

In this section we provide the performance comparison of the traditional *k*-means and enhanced *k*-means in terms of the quality of the solution and rate of convergence. The efficiency of the enhanced *k*-means algorithm is then compared with respect to traditional *k*-means algorithm regarding formation of empty clusters.

We tested original K-means and proposed K-means algorithm for the data sets with known clustering Iris, New Thyroid, Height-Weight and Diggle. Both the algorithms require number of clusters as an input. In additional, for the existing k-means clustering algorithm the empty clusters can be generated but in the proposed method the generation of empty clusters can be avoided.

The proposed method requires only the data values and number of clusters as inputs and it does not take any other inputs like threshold values. In each experiment the time and accuracy was calculated. Table 1 depicts the comparison of the formation of empty clusters, accuracy, number of iterations and accuracy in traditional k-means and proposed k-means algorithms.

Table 1. The comparative analysis of running time, no. of iterations and accuracy in traditional k-means and enhanced k-means algorithm

Data Set	**Running Time in ms**		**No. of Iterations**		**Accuracy (%)**	
	Standard K-means	**Enhanced K-means**	**Standard K-means**	**Enhanced K-means**	**Standard K-means**	**Enhanced K-means**
Fisher's Iris	**0.096**	**0.080**	**8**	**3**	**86.3**	**87.5**
Height-Weight	**0.081**	**0.069**	**14**	**6**	**78.9**	**82.5**
Echocardiogram	**0.097**	**0.081**	**9**	**4**	**91.3**	**96.54**
Diggle	**0.092**	**0.070**	**13**	**5**	**79.3**	**90.50**

Table 2. The comparative analysis of formation of empty clusters using iris data in traditional k-means and enhanced k-means algorithm

Number of Clusters	Formation of Empty Clusters	
	Traditional K-means	Enhanced K-means
2	0	0
3	0	0
4	1	0
5	1	0
6	2	0
7	2	0
8	1	0
9	3	0
10	3	0

The results also illustrated with the help of bar charts in the Figure 5,6, and 7. The comparative results illustrate that the enhanced framework of k-means algorithm is producing accurate clustering results than traditional k-means algorithm and also guaranteed that formation of non-empty clusters.

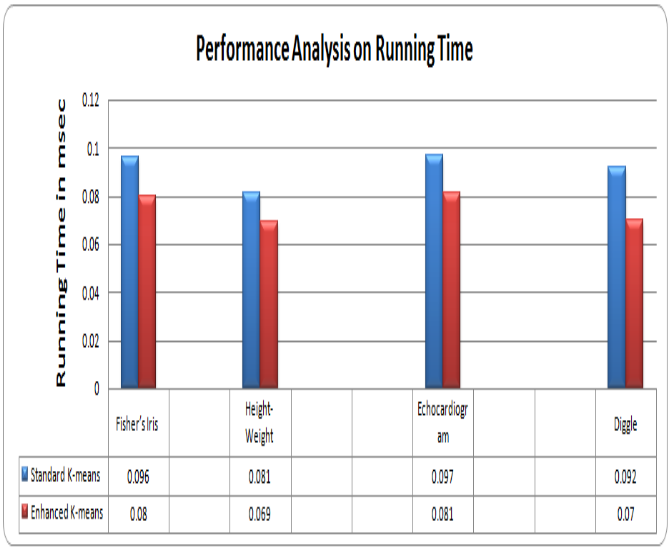


Fig. 5. Comparison of traditional K-means clustering Algorithm and enhanced K-means clustering Algorithm on the basis of running time

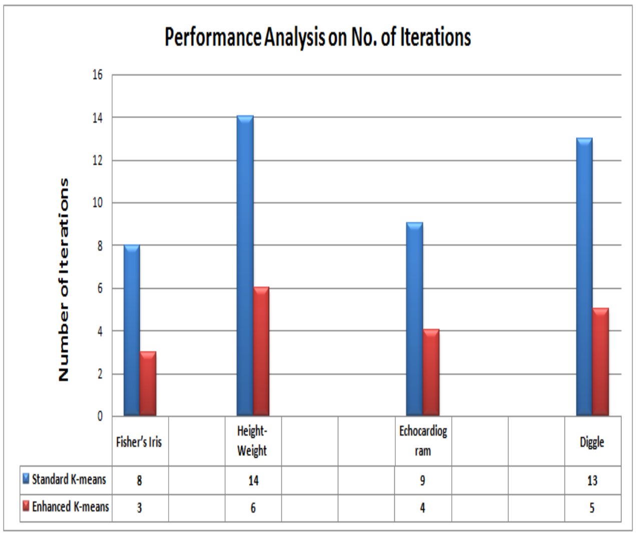


Fig. 6. Comparison of traditional K-means clustering Algorithm and Proposed K-means clustering Algorithm on the basis of Number of Iterations

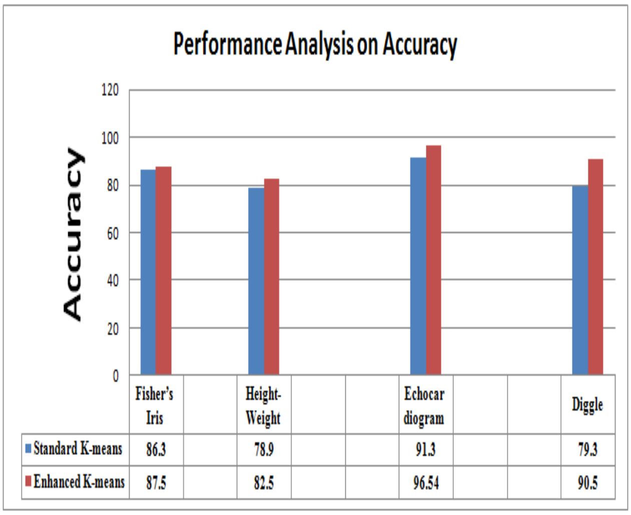


Fig. 7. Comparison of traditional K-means clustering Algorithm and Proposed K-means clustering Algorithm on the basis of Accuracy.

V. CONCLUSION

This article proposes a Framework for *k*-means clustering algorithm. The enhanced Framework preserve all important features of the traditional *k*-means and at the same time eliminates the possibility of formation of empty clusters and enhance the efficiency and accuracy of K-means clustering algorithm. A detailed comparison of this enhanced algorithm with the traditional *k*-means has been reported. Experimental results demonstrate that the enhanced clustering design is able to solve the empty cluster problem without any significant performance degradation.

REFERENCES

[1]. Wei Li, “Modified k-means clustering algorithm”, IEEE computer society Congress on Image and Signal Processing, 2008, pp. 618-621.

[2]. Jiawei Han and Micheline Kamber, “Data Mining: Concepts and Techniques”, Morgan Kaufmann Publishers, second Edition, 2006.

[3]. Hatamlou, “In search of optimal centroids on data clustering using a binary search algorithm”, Pattern Recognit. Lett., vol. 33, **(2012)**, pp. 1756-1760.

[4]. F. Cao, J. Liang and G. Jiang, “An initialization method for the K-Means algorithm using neighborhood model”, Comput. Math. Appl., vol. 58, **(2009)**, pp. 474-483.

[5]. R. C. De Amorim and P. Komisarczuk, “On initializations for the minkowski weighted K-means”, proceedings of the 11th International Conference on Advances in Intelligent Data Analysis XI, **(2012)** October 25-27, 2012, Helsinki, Finland, pp. 45-55.

[6]. Changqing Zhou, Dan Frankowski, Pamela Ludford, Shashi Shekhar,Loren Terveen, “Discovering personally meaningful places: An interactive clustering approach”, Volume 25, Issue 3 ACM Transactions on Information Systems (TOIS), July 2007.

[7]. Ran Vijay Singh and M.P.S Bhatia, “Data Clustering with Modified k-means Algorithm”, IEEE International Conference on Recent Trends in Information Technology, ICRTIT 2011, 2011, pp 717-721.

[8]. Ahamed Shafeeq B M and Hareesha K S “Dynamic Clustering of Data with Modified k-Means Algorithm” International Conference on Information and Computer Networks, ICICN 2012, pp 221-225.

[9]. Komarasamy G and Amitabh Wahi, “An Optimized k-means Clustering Technique Using Bat Algorithm”, European Journal of Scientific Research, ISSN 1450-216X Vol.84 No.2, August 2012, pp.263 – 273.

[10]. D T Pham, S S Dimov, and C D Nguyen “Selection of k in k-means clustering”, Mechanical Engineering Science, 2004, pp. 103-119.

[11]. R. Xu and D. Wunsch, II, “Survey of clustering algorithms”, IEEE Trans. Neural Networks., vol. 16, no. 3, 2005, pp. 645– 678.

[12]. Baolin Yi, Haiquan Qiao, Fan Yang, Chenwei Xu, “An Improved Initialization Center Algorithm for Kmeans Clustering,” IEEE 2010.

[13]. J.James Manoharan and Dr.S.Hari Ganesh, “Improved K-means Clustering Algorithm using Linear Data Structure List to Enhance the Efficiency”, International Journal of Applied Engineering Research, ISSN 0973-4562 Vol. 10 No.20 (2015).

[14]. J.James Manoharan and Dr.S.Hari Ganesh, “Initialization of Optimized K-means Centroids using Divide-and-Conquer Method”,ARPN Journal(Under review).