ACADEMIC
World Academic Union

ISSN 1746-7659, England, UK
Journal of Information and Computing Science
Vol. 9, No. 4, 2014, *pp*. 277-284

Enhanced *K*-Means Clustering Algorithm using A Heuristic Approach

Vighnesh Birodkar [1] and Damodar Reddy Edla [1,*]

[1] Department of Computer Science and Engineering
National Institute of Technology Goa – 403 401, India.
Email: vighneshbirodkar@gamil.com, dr.reddy@nitgoa.ac.in

(*Received March 31, 2014, accepted October 1, 2014*)

Abstract. *K*-means algorithm is one of the most popular clustering algorithms that has been survived for more than 4 decades. Despite its inherent flaw of not knowing the number of clusters in advance, very few methods have been proposed in the literature to overcome it. The paper contains a fast heuristic algorithm for guessing the number of clusters as well as cluster center initialization without actually performing *K*-means, under the assumption that the clusters are well separated in a certain way. The proposed algorithm is experimented on various synthetic data. The experimental results show the effectiveness of the proposed approach over the existing.

Keywords: partitional clustering, K-means, unsupervised learning, cluster center, synthetic data

1. Introduction

Clustering is a widely used approach in data-mining. It has a variety of application, including Medicine, [1],Feature Detection [2] ,Geology [3] and Robotics [4].The process of grouping a set of physical or abstract objects into classes of similar objects is called clustering[5] . Clustering is broadly divided into partitioning, hierarchical, density-based, grid-based, model-based and constraint-based methods.

Partitioning method divides *n* objects into *k* groups such that each group has at least one object and each object belongs to only one group [5]. *K*-Means is one of the most widely used partition-based clustering algorithms due to its simplicity. It divides a set of objects into a fixed number of groups called clusters. Let $P = \{p_i \mid 1 \le i \le N\}$ be a set of objects. *K*-Means divides *P* into *K* clusters. If $C = \{C_i \mid 1 \le i \le K\}$ is a set of *K* centers *K*-Means tries to minimize the sum of Euclidean distances of objects from its cluster centers, which can be given by

$$sum = \sum_{i=1}^{n} dist\left(x_i - c_k\right)$$

where $dist(p_i, c_k)$ is the Euclidean distance of object p_i from its cluster center. *K*-Means attempts to find a local minima for sum, and hence requires that *K* (number of clusters) to be known in advance, the global minima being $sum = 0$ when $K = N$. The problem of finding clusters so as to minimize the Euclidean Sum of Squares is *NP*-Hard [6] and *K*-Means suffices only as a heuristic

2. A Review

K-means is the most popular clustering technique of this model developed by MacQueen [6] in 1967. However, it is sensitive to the random selection of initial cluster centers. In addition to that, a prior knowledge of the number of clusters is necessity to input to K-means. Many researches proposed various methods [7], [8] to overcome these problems. Kanungo et al. [9] proposed a novel initialization method for K-means using kd-tree. This scheme does not pass information from one stage to its next. Du et al. [10] developed an initialization scheme for K-means clustering called *PK*-means to cluster the gene expression data. The convergence rate of this technique is fast and the computational load is less. A novel clustering algorithm called modified filtering algorithm (MFA) has been proposed in [11]. It is the improvement of the algorithm in [12]. A fast *K*-means clustering algorithm named FKMCUCD was proposed in [13] using cluster center displacement. This method is significant for high-dimensional large data. Zalik [14] proposed an efficient algorithm named K'-means to enhance the K-means algorithm by exploiting a cost function. This scheme fails when the clusters are of various shapes such as elliptical. Redmond et al. [15] proposed a novel seed selection algorithm using kd-tree [16]. This scheme is unable to deal with the noise. Cao et al. [17] proposed an algorithm by defining the cohesion degree of the neighborhood of a given point and the coupling degree between neighborhoods of the points. This algorithm has quadratic time complexity. Khan et al. [18] designed an algorithm called CCIA. This method first develops $k'(>k)$ cluster centers from which the desired k centers are chosen. Lu et al. [19] contributed with a hierarchical initialization approach in which the clustering problem has treated as a weighted clustering problem. A genetic clustering algorithm named GAGR [20] has been proposed to cluster the genome data using *K*-means. It uses the genetic algorithm with gene rearrangement process. Ahmad et al. [21] proposed an enhanced K-means clustering algorithm for mixed numeric and categorical data based on co-occurrence of the values. An algorithm called KGA [22] was proposed using the genetic algorithm. This method may not produce fine results whenever the number of clusters is unknown. An improved version of K-means called *K**-means has been developed in [23]. It is unable to deal with the noisy data. Likas et al. [24] proposed a global *K*-means clustering algorithm in which the clusters are formed using a global search procedure. A recursive method is proposed by Duda and Hart [25]. Milligan [26] developed an enhanced algorithm based on Ward's hierarchical method [27] that helps in finding the initial cluster centers. The algorithm proposed by Fisher [28] generates good seeds by constructing initial hierarchical clustering based on [29]. Both Higgs et al., [30] and Snarey et al. [31] developed a method using MaxMin algorithm to choose a subset of the original database as initial cluster centers. Bradley et al., [32] formed the initial clusters based on the bilinear program. Tou and Gonzales [33] presented a method which entirely depends on the order of the points and the threshold value. Linde et al., [34] proposed a method based on Binary Splitting (BS). Here, the clusters quality depends on the selection of a random vector. Kaufman and Rousseeuw [35] developed a method based on the reduction in the Distortion. Babu and Murty [36] proposed a technique for the near optimal seed selection based on genetic programming. This is not robust for large data bases. Huang and Harris [37] projected a method called Direct Search Binary Splitting (DSBS) based on the Principal Component Analysis (PCA) and the vector of Linde et al., [35]. Thiesson et al., [38] designed an algorithm that depends on the mean value of the given data. Bradley and Fayyad [39] proposed an initialization approach for K-means using the Forgy's method [40].

3. Proposed Algorithm

Let P = $\{p_i \mid 1 \le i \le N\}$ be a set of *N* objects. Let *ClusterSet* = $\{C_i \mid 1 \le i \le K\}$ be the set of K desired Clusters. *Let* $x \in C_i$ and $y \in C_j$. $dist(x, y) \mid \mathrm{i = j}, \ \forall \mathrm{x} \ \forall \mathrm{y} < \mathrm{dist(x, y)} \mid \mathrm{i} \neq \mathrm{j}, \ \forall \mathrm{x} \ \forall \mathrm{y}$

The algorithm first finds *K* objects from *K* different clusters. Assume that at a certain stage *ClusterPoints* is a set of *m* objects (where $m < K$) from *m* different clusters. To find the $(m + 1)^{th}$ object we find the Minimum Euclidean Distance of all objects in P from any object in *ClusterPoints* and assign it as *minDist.* Under the assumption that the clusters are Well-separated all the objects belonging to the same clusters as any object in *ClusterPoints* will have lesser *minDist* than an object not in the same cluster as any object in *ClusterPoints.* If we select the object q with the Maximum *minDist* it is assured that it does not belong to the same cluster as any of the objects in *ClusterPoints*. We add the *q* in *ClusterPoints* and increment *m* by 1 and the process is repeated. To start the algorithm a random object is chosen and added to *ClusterPoints* and m is set to 1.The algorithm thus assures that the first *K* objects chosen are from different clusters.

Upon successive iterations of the above process there will be a state where $m > K$, particularly $m = K + 1$. We need to identify this state for the algorithm to stop and return *K*. To achieve this the proposed algorithm relies on a heuristic based on Euclidean Distance. If 2 objects belong to the same they will have similar Euclidean Distances from objects from other clusters. Algorithm 1 takes 2 arrays of length *N* and outputs a real number which is significantly lower for arrays of Euclidean Distances of two points belonging to the same cluster. At each iteration the algorithm calls Algorithm 1 and halts when the value is less than R times all the previously computed values by Algorithm 1

Diff Algorithm:

```
Input
A – Array of objects
B - Array of objects
N – Length of Arrays

Output
ans – A Positive Real Number
sum = 0
for $i = 1 \rightarrow N$
sum = sum + $\left( \frac{2*(A[i] - B[\mathrm{i}])}{A[i] + B[i]} \right)^2$
end
```

```
return $\sqrt{\frac{sum}{N}}$
```

FindK Algorithm

```
Variables Used:
ClusterPoints – Set of objects each belonging to a different cluster
minDist – Array of Length N, minimum distance of objects from CluterPoints
diffSet – Set of Values returned by diff
newDist,closeDist – Arrays of length N

Functions Used:
maxIndex(array) – Returns index of Maximum Value in array
diff(array1,array2) - Described in Algorithm 1
dist(obj1,obj2) - Returns Euclidian Distance between obj1 and onj2
max(S) - Returns Maximum Value in set S
closest(obj,S) – Returns closest object to obj from set S.

Input:
P - Array of N objects
R – Positive real number (< 1)

Output:
K – Number of Clusters
ClusterPoints – Set of K objects ( Initial cluster centers )

Initialize all values in minDist to ∞
Initialize ClusterPoints as EmptySet
Initialize DiffSet as EmptySet
Choose a random object newPoint from P
m=0
repeat
m = m + 1
Add newPoint to ClusterPoints
  for i = 1 → N do
    if(dist(newPoint[i],P[i]) < minDist[i]) then
      minDist[i] = dist(newPoint[i],P[i])
    end
  end
  i = maxIndex(minDist)
  newPoint = P[i]
  closePoint = closest(newPoint,clusterPoints)
  for i = 1 → N do
    newDist[i] = dist(newPoint,P[i])
    closeDist[i] = dist(closePoint,P[i])
  end
  newDiff = diff(newDist,closrDist,N)
until newDiff > R*max(diffSet)
return m,clusterPoints
```

If the running time of Algorithm 2 is T(N) it can be clearly seen that T(N) = θ(N)

4. Experimental results

The algorithm was applied to various synthetic data sets with $R = 0.4$ where number of clusters were correctly identified. Some of the results are shown below. The Python Programming Language was used, along with *wxPython* for the GUI elements. The synthetic data sets were generated using *Python Imaging Library*. All programs were executed with *Python 2.7 32-bit on Windows 7 Professional 64-bit* running on

Intel 2450M Cpu @ 2.50 GHz , with *4096 MB of RAM* and *Intel Sandy Bridge Chipset.* The clusters and the corresponding values returned by *Diff* algorithm in every iteration are shown above. The y-Axis represents the values returned by *Diff* algorithm, x-Axis represents *k*-value In Figure 1. The algorithm halts during the 5th iteration. It adds 4 points to cluster Points (Empty Circles) and the 5th new point is rejected (Dotted Circle). Similarly in Figure 2, the algorithm halts during the 3rd iteration. It adds 3 points to cluster points (Empty Circles) and the 4th new point is rejected (Dotted Circle). *Figure 3. and Figure 4* can be similarly interpreted.

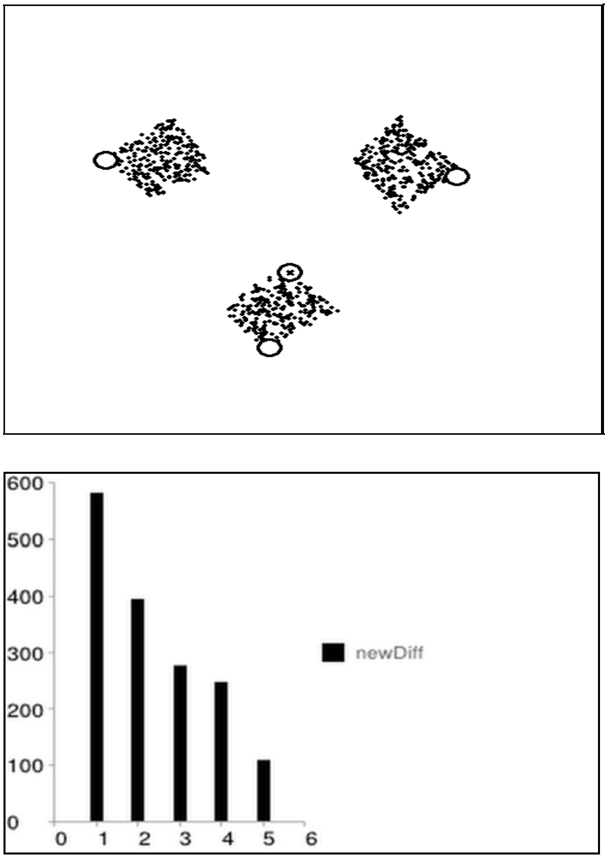

Fig. 1: *max(diffSet)* = 577.4 , *newDiff* = 106.0

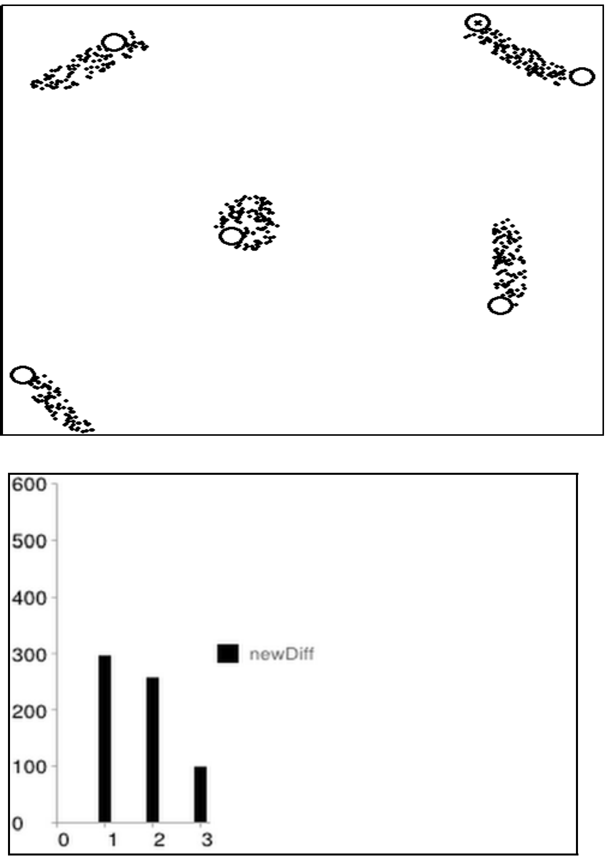

Fig. 2: *max(diffSet)* = 298.5 , *newDiff* = 94.6

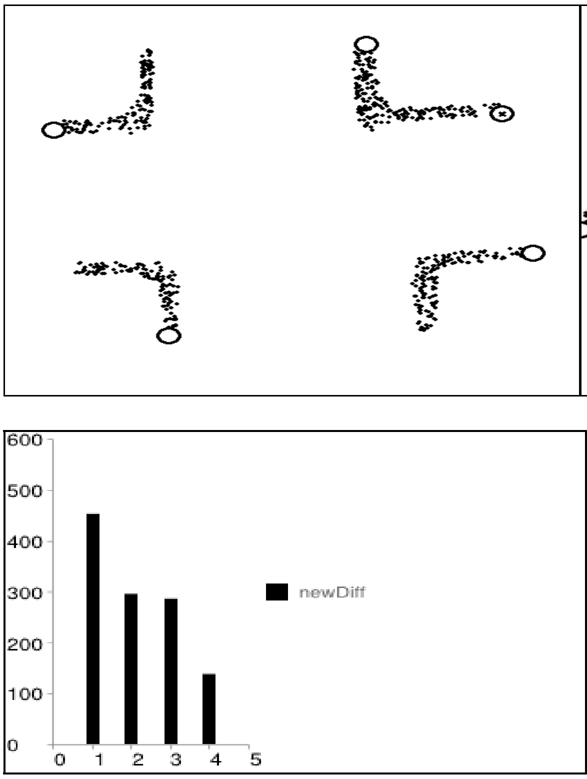

Fig. 3: *max(diffSet)* = 488.0 , *newDiff* = 98.6

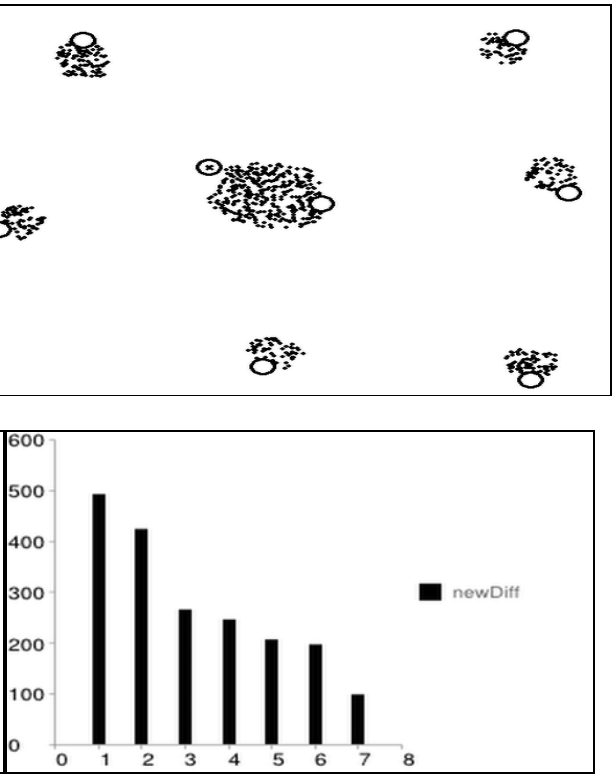

Fig. 4. *max(diffSet)* = 450.0 , *newDiff* = 141.4

5. Conclusion

The above sections have presented a fast heuristic method to guess the number of clusters from a set of objects as well as provide the initial cluster centers for K-Means, provided that the clusters are relatively well spaced out as compared to their size. The Algorithm does not rely on performing K-Means Clustering during any stage and runs in Linear Time. However, since the halting condition is a heuristic, the correctness can't be proven. Moreover, clusters in Real World Data might not always well spaced out. Future efforts will be made to eliminate the heuristic and to indicate when the clusters are not well spaced out as per the requirements.

The task of halting the Algorithm depends on a heuristic, which can't be proven to work for all cases. Future attempts would be to eliminate all heuristics and to develop a technique to halt the Algorithm which can be proved. The assumption of clusters being Well Separated might not be encountered frequently when real-world Data Sets are considered. Future efforts will also be made to indicate whether the Data Set satisfies the assumption or not. The obvious solution of checking all possible pairs of objects will require O(N2) running time. Future efforts will focus on speeding up the process of checking for Well Separated Clusters.

Acknowledgements

The authors would like to thank Prof. GRC Reddy, Director, National Institute of Technology Goa, India for his encouragement and support towards our research work.

6. References

[1] N. Srinivasan and V. Vaidehi, "Application of cluster computing in medical image processing," in *Broadband Networks, BroadNets 2005. 2nd International Conference* on, 2005, pp. 1007–1010 Vol. 2, 2005.

[2] L. Suyi, Z. Hua, J. Jianping, and L. Bing, "Feature recognition for underwater weld images," *in Control Conference (CCC), 2010 29th Chinese,* 2010, pp. 2729–2734.

[3] L. Vibha, G. HarshaVardhan, S. Prashanth, P. Deepa Shenoy, K. Venu-gopal, and L. Patnaik, "A hybrid clustering and classification technique for soil data mining," in *Information and Communication Technology in Electrical Sciences (ICTES 2007), 2007. ICTES. IET-UK International Conference on*, 2007, pp. 1090–1095.

[4] M. Castelnovi, P. Musso, A. Sgorbissa, and R. Zaccaria, "Surveillance robotics: analyzing scenes by colors analysis and clustering," *in Com- putational Intelligence in Robotics and Automation, 2003. Proceedings.2003 IEEE International Symposium on*, vol. 1, 2003, pp. 229–234 vol.1.

[5] J. Han and M. Kamber, *Data Mining: Concepts and Techniques, Second Edition*. Morgan Kaufmann, 2006.

[6] J.B. MacQueen. Some methods for clustering and analysis of multivariate observations. In 5*th* *Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, University of California Press, Berkeley, CA, USA, pages 281-297, 1967.

[7] L. Galluccio O. Michel P. Comon and A.O. Hero-III. Graph based *K*-means clustering. *Signal Processing*, 92(9):1970–1984, September 2012.

[8] A. K. Jain. Data clustering: 50 years beyond *K*-means*. *Pattern Recognition Letters*, 31(8):651-666, June 2010.

[9] T. Kanungo D.M. Mount N.S. Netanyahu C.D. Piatko R. Silverman and A.Y. Wu. An efficient *K*-means clustering algorithm: analysis and implementation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(7):881-892, 2002.

[10] Z. Du Y. Wang and Z. Ji. *PK*-means: a new algorithm for gene clustering. *Computational Biology and Chemistry*, 32(4):243-247, 2008.

[11] J.Z.C. Lai and Y.C. Liaw. Improvement of the *K*-means clustering filtering algorithm. *Pattern Recognition*, 41(12):3677-3681, December 2008.

[12] T. Kanungo D.M. Mount N.S. Netanyahu C.D. Piatko R. Silverman and A.Y. Wu. An efficient *K*-means clustering algorithm: analysis and implementation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(7):881-892, 2002.

[13] J.Z.C. Lai T.J. Huang and Y.C. Liaw. A fast *K*-means clustering algorithm using cluster center displacement. *Pattern Recognition*, 42(11):2551-2556, November 2009.

[14] K.R. Zalik. An efficient *K′*-means clustering algorithm. *Pattern Recognition Letters*, 29(9):1385-1391, July 2008.

[15] S.J. Redmond and C. Heneghan. A method for initializing the *K*-means clustering algorithm using *kd*-trees. *Pattern Recognition Letters*, 28(8):965-973, June 2007.

[16] M.D. Berg O. Cheong M.V. Kreveld and M. Overmars. *Computational Geometry*: *Algorithms and Applications*. 3rd edition, Springer-Verlag, Berlin Heidelberg, New York, 2008.

[17] F. Cao J. Liang and G. Jiang. An initialization method for the *K*-means algorithm using neighborhood model. *Computers and Mathematics with Applications*, 58(3):474-483, August 2009.

[18] S.S. Khan and A. Ahmad. Cluster center initialization algorithm for *K*-means clustering. *Pattern Recognition Letters*, 25(11):1293-1302, August 2004.

[19] J.F. Lu J.B. Tang Z.M. Tang and J.Y. Yang. Hierarchical initialization approach for *K*-means clustering. *Pattern Recognition Letters*, 29(6):787-795, April 2008.

[20] D.X. Chang X.D. Zhang and C.W. Zheng. A genetic algorithm with gene rearrangement for *K*-means clustering. *Pattern Recognition*, 42(7):1210-1222, July 2009.

[21] A. Ahmad and L. Dey. A *K*-mean clustering algorithm for mixed numeric and categorical data. *Data* & *Knowledge Engineering*, 63(2):503-527, November 2007.

[22] S. Bandyopadhyay and U. Maulik. An evolutionary technique based on *K*-means algorithm for optimal clustering in R^N. *Information Sciences*, 146(1-4):221-237, October 2002.

[23] Y.M. Cheung. K^*-Means: A new generalized *K*-means clustering algorithm. *Pattern Recognition Letters*, 24(15):2883-2893, November 2003.

[24] A. Likas N. Vlassis and J.J. Verbeek. *The global K-means clustering algorithm*. *Pattern Recognition*, 36(2): 451-461, February 2003.

[25] R.O. Duda and P.E. Hart. *Pattern Classification and Scene Analysis*. John Wiley and Sons, New York, 1973.

[26] G.W. Milligan. An Examination of the effect of six types of error perturbation on fifteen clustering algorithms. *Psychometrika*, 45(3):325-342, September 1980.

[27] J.H. Ward-Jr. Hierarchical grouping to optimize an objective function. *Journal of the American Statistical Association*, 58(301):236-244, March 1963.

[28] D. Fisher. Iterative optimization and simplification of hierarchical clusterings. *Journal of Artificial Intelligence Research*, 4(1):147-179, January 1996.

[29] D.H. Fisher. Knowledge acquisition via incremental conceptual clustering. *Machine Learning*, 2(2):139-172, September 1987.

[30] R.E. Higgs K.G. Bemis I.A. Watson and J.H. Wikel. Experimental designs for selecting molecules from large chemical databases. *Journal of Chemical Information and Computer Sciences*, 37(5):861-870, September 1997.

[31] M. Snarey N.K. Terrett P. Willet and D.J. Wilton. Comparison of algorithms for dissimilarity-based compound selection. *Journal of Molecular Graphics and Modeling*, 15(6):372-385, December 1997.

[32] P.S. Bradley O.L. Mangasarian and W.N. Street. Clustering via concave minimization. 10*th* *Annual Conference on Advances in Neural Information Processing System*, USA, volume 9, pages 368-374, December 2-5, 1996.

[33] J. Tou and R. Gonzales. *Pattern Recognition Principles*. Massachusetts, Addison Wesley, 1974.

[34] Y. Linde A. Buzo and R.M. Gray. An algorithm for vector quantizer design. *IEEE Transactions on Communications*, 28(1):84-95, January 1980.

[35] L. Kaufman and P.J. Rousseeuw. *Finding Groups in Data-An Introduction to Cluster Analysis*. Canada, Wiley, 1990.

[36] G.P. Babu and M.N. Murty. A near-optimal initial seed value selection in *K*-means algorithm using a genetic algorithm. *Pattern Recognition Letters*, 14(10):763-769, October 1993.

[37] C. Huang and R. Harris. A comparison of several codebook generation approaches. *IEEE Transactions on Image Process*, 2(1):108-112, 1993.

[38] B. Thiesson B. Meck C. Chickering and D. Heckerman. Learning mixtures of Bayesian networks. *Microsoft Technical Report* (MSR-TR-97-30), 1997.

[39] P.S. Bradley and U.M. Fayyad. Refining initial points for *K*-means clustering. In 15^{th} *International Conference on Machine Learning* (ICML-1998), Wisconsin, USA, pages 91-99, July 24-27, 1998.

[40] E. Forgy. Cluster analysis of multivariate data: efficiency vs interpretability of classifications. *Biometrics*, 21(3):768-769, 1965.