International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:1, No:9, 2007

Dynamic Clustering using Particle Swarm Optimization with Application in Unsupervised Image Classification

Mahamed G.H. Omran, Andries P Engelbrecht, and Ayed Salman

***Abstract*—**A new dynamic clustering approach (DCPSO), based on Particle Swarm Optimization, is proposed. This approach is applied to unsupervised image classification. The proposed approach automatically determines the "optimum" number of clusters and simultaneously clusters the data set with minimal user interference. The algorithm starts by partitioning the data set into a relatively large number of clusters to reduce the effects of initial conditions. Using binary particle swarm optimization the "best" number of clusters is selected. The centers of the chosen clusters is then refined via the K-means clustering algorithm. The experiments conducted show that the proposed approach generally found the "optimum" number of clusters on the tested images.

***Keywords*—**Clustering Validation, Particle Swarm Optimization, Unsupervised Clustering, Unsupervised Image Classification.

I. Introduction

DATA clustering is the process of identifying natural groupings or clusters, within multidimensional data, based on some similarity measure (e.g. Euclidean distance) [1],[2]. Clustering algorithms are used in many applications, such as data mining [3], compression [4], image segmentation [5]-[7], machine learning [8], etc. A cluster is usually identified by a cluster center (or *centroid*) [9]. Data clustering is a difficult problem as the clusters in data may have different shapes and sizes [2]. Furthermore, it is usually not known how many clusters should be formed [10].

Most clustering algorithms are based on two popular techniques known as hierarchical and partitional clustering [11],[12]. In hierarchical clustering, the output is "a tree showing a sequence of clustering with each clustering being a partition of the data set" [12]. Such algorithms have the following advantages [11]:

- The number of clusters need not be specified *a priori*, and
- they are independent of the initial conditions.

Manuscript received July 12, 2005.

Mahamed Omran is with the Faculty of Computing & IT, Arab Open University, Kuwait (email: mjomran@engineer.com).

Andries Engelbrecht is with the Computer Science Department, University of Pretoria, South Africa (phone:+27-12-420-3578; fax: +27-12-362-5188; email:engel@cs.up.ac.za).

Ayed Salman is with the Computer Engineering, Department, Kuwait University, Kuwait (email:ayed@eng.kuniv.edu.kw).

However, hierarchical clustering techniques suffer from the following drawbacks:

- They are static, i.e. data points assigned to a cluster cannot move to another cluster.
- They may fail to separate overlapping clusters due to a lack of information about the global shape or size of the clusters.

On the other hand, partitional clustering algorithms partition the data set into a specified number of clusters. These algorithms try to minimize certain criteria (e.g. a square error function) and can therefore be treated as optimization problems. The advantages of hierarchical algorithms are the disadvantages of the partitional algorithms and *vice versa*. Partitional clustering techniques are more popular than hierarchical techniques in pattern recognition [2], hence, this paper will concentrate on partitional techniques.

Partitional clustering aims to optimize cluster centers, as well as the number of clusters [10]. Most clustering algorithms require the number of clusters to be specified in advance [9],[10]. Finding the "optimum" number of clusters in a data set is usually a challenge since it requires *a priori* knowledge, and/or ground truth about the data, which is not always available. The problem of finding the optimum number of clusters in a data set has been the subject of several research efforts [13],[14], however, despite the amount of research in this area, the outcome is still unsatisfactory [15].

This paper proposes a new approach called Dynamic Clustering using a Particle Swarm Optimization algorithm (DCPSO). The approach uses some of the ideas presented by Kuncheva and Bezdek [16]. DCPSO automatically determines the "optimum" number of clusters and simultaneously clusters the data set with minimal user interference. The algorithm starts off by partitioning the data set into a relatively large number of clusters in order to reduce the effects of initial conditions. Using binary Particle Swarm Optimization (PSO), the "optimal" number of clusters is selected. The centers of the chosen clusters are then refined using the K-means clustering algorithm. The binary PSO is applied again on the cluster centers to find a new "optimal" number of clusters in the data set. This process is repeated until convergence is reached.

The remainder of the paper is organized as follows: Section II surveys related work. The DCPSO algorithm is presented in Section III, while an experimental evaluation of DCPSO in the area of unsupervised image classification is provided in

Section IV, using natural images. Furthermore, DCPSO is compared to other clustering algorithms. Finally, Section V concludes the paper.

II. Related Work

A. Unsupervised Clustering Algorithms

ISODATA, proposed by Ball and Hall [17], is an enhancement of the K-means algorithm, with addition of the possibility of merging classes and splitting elongated classes. An alternative approach to ISODATA is SYNERACT [18]. SYNERACT combines K-means with hierarchical descending approaches. According to Huang [18], SYNERACT is faster than and almost as accurate as ISODATA. Furthermore, it does not require the number of clusters and initial location of centroids to be specified in advance. Another improvement to the K-means algorithm is proposed by Rosenberger and Chehdi [15], which automatically finds the number of clusters in an image set by using intermediate results. Furthermore, Pelleg and Moore [19] proposed a K-means based algorithm, called X-means, that uses model selection. X-means searches over a range of values of K and selects the value with the best Bayesian Information Criterion (BIC) [20] score. Recently, Hamerly and Elkan [10] proposed another wrapper around K-means called G-means. G-means starts with a small value for K, and with each iteration splits up the clusters whose data do not fit a Gaussian distribution. Between each round of splitting, K-means is applied to the entire data set in order to refine the current solution. According to [10],[21], G-means works better than X-means, however, it works only for data having spherical and/or elliptical clusters. G-means is not designed to work for arbitrary-shaped clusters [21]. A program called *snob* [22],[23] uses various methods to assign objects to classes in an intelligent manner [24]. After each assignment, the Wallace Information Measure [25],[26] is calculated and based on this calculation the assignment is accepted or rejected. Hence, snob can split/merge and move points between classes. Bischof *et al*. [27] proposed another algorithm based on K-means which uses a minimum description length (MDL) framework. The algorithm starts with a large value for K and proceeds to remove centroids when this results in a reduction of the description length. K-means is used between the steps that reduce K.

Gath and Geva proposed an unsupervised clustering algorithm based on the combination of FCM and fuzzy maximum likelihood estimation [28]. Lorette *et al.* [29] proposed an algorithm based on fuzzy clustering to dynamically determine the number of clusters in a data set. This approach, however, requires a parameter to be specified, which has a profound effect on the number of clusters generated (i.e. not fully unsupervised). Similarly, Boujemaa [30] proposed an algorithm, based on a generalization of the competitive agglomeration clustering algorithm introduced by Frigui and Krishnapuram [31].

The algorithms found within the above paragraph try to modify the objective functions of FCM. These approaches are still sensitive to initialization and other parameters [32]. Frigui and Krishnapuram proposed a robust competitive clustering algorithm, based on the process of competitive agglomeration. Initialization does, however, have a significant effect on the result of this algorithm [32].

Kohonen's Self Organizing Maps (SOM) [33]-[35] can be used to automatically find the number of clusters in a data set. SOM combines competitive learning (in which different nodes in the Kohonen network compete to be the winners when an input pattern is presented) with a topological structuring of nodes, such that adjacent nodes tend to have similar weight vectors (this is done via lateral feedback) [34],[35]. SOM suffers from being dependent on the order in which the data points are presented. To overcome this problem, the choice of data points can be randomized during each epoch [35].

Lee and Antonsson [9] used an evolution strategy (ES) to dynamically cluster a data set. The proposed ES implemented variable length genomes to search for both the centroids and K.

B. Clustering Validation Techniques

The main subject of cluster validation is "the evaluation of clustering results to find the partitioning that best fits the underlying data" [13]. Hence, cluster validity approaches are approaches used to quantitatively evaluate the result of a clustering algorithm [13]. These approaches have representative indices, called validity indices. The traditional approach to determine the "optimum" number of clusters is to run the algorithm repetitively using different input values and select the partitioning of data resulting in the best validity measure [36].

Two criteria that have been widely considered sufficient in measuring the quality of partitioning a data set into a number of clusters, are [13]

- *Compactness*: samples in one cluster should be similar to each other and different from samples in other clusters. An example of this would be the variance of a cluster.
- *Separation*: clusters should be well-separated from each other. An example of this criterion is the Euclidean distance between the centroids of clusters.

There are several relative validity indices; for a thorough survey in this field refer to Halkidi *et al*. [13]. Recently, Turi [24] proposed an index incorporating a multiplier function (to penalize the selection of a small number of clusters) to the ratio between intra-cluster and inter-cluster distances, with some promising results.

C. Particle Swarm Optimization

Particle swarm optimizers (PSO) are population-based optimization algorithms modeled after the simulation of social behavior of bird flocks [37],[38]. PSO is generally considered to be an evolutionary computation (EC) paradigm. Other EC paradigms include genetic algorithms (GA), genetic programming (GP), evolutionary strategies (ES), and evolutionary programming (EP) [39]. These approaches simulate biological evolution and are population-based. In a

PSO system, a swarm of individuals (called *particles*) fly through the search space. Each particle represents a candidate solution to the optimization problem. The position of a particle is influenced by the best position visited by itself (i.e. its own experience) and the position of the best particle in its neighborhood (i.e. the experience of neighboring particles). When the neighborhood of a particle is the entire swarm, the best position in the neighborhood is referred to as the global best particle, and the resulting algorithm is referred to as a *gbest* PSO. When smaller neighborhoods are used, the algorithm is generally referred to as a *lbest* PSO [40]. The performance of each particle (i.e. how close the particle is from the global optimum) is measured using a fitness function that varies depending on the optimization problem.

Each particle in the swarm is represented by the following characteristics:

$\boldsymbol{x}_i$: The *current position* of the particle;
$\boldsymbol{v}_i$: The *current velocity* of the particle;
$\boldsymbol{y}_i$: The *personal best position* of the particle.

The personal best position of particle *i* is the best position (i.e. one resulting in the best fitness value) visited by particle *i* so far. Let *f* denote the objective function. Then the personal best of a particle at time step *t* is updated as

$$\boldsymbol{y}_i(t+1) = \begin{cases} \boldsymbol{y}_i(t) & \text{if } f(\boldsymbol{x}_i(t+1)) \geq f(\boldsymbol{y}_i(t)) \\ \boldsymbol{x}_i(t+1) & \text{if } f(\boldsymbol{x}_i(t+1)) < f(\boldsymbol{y}_i(t)) \end{cases} \quad (1)$$

Two main approaches to PSO exist, namely *lbest* and *gbest*, the difference being the neighborhood topology used to exchange experience among particles. For *gbest*, the best particle is determined from the entire swarm. If the position of the global best particle is denoted by the vector $\hat{\boldsymbol{y}}$, then

$$\hat{\boldsymbol{y}}(t) \in \{\boldsymbol{y}_0, \boldsymbol{y}_1, .., \boldsymbol{y}_s\} = \min\{f(\boldsymbol{y}_0(t)), f(\boldsymbol{y}_1(t)), .., f(\boldsymbol{y}_s(t))\} \quad (2)$$

where *s* denotes the size of the swarm. For the *lbest* model, a swarm is divided into overlapping neighborhoods of particles. For each neighborhood N_j, a best particle is determined with position $\hat{\boldsymbol{y}}_j$. This particle is referred to as the *neighborhood best* particle, defined as

$$\hat{\boldsymbol{y}}_j(t+1) \in \{N_j \mid f(\hat{\boldsymbol{y}}_j(t+1)) = \min\{f(\boldsymbol{y}_i(t))\}, \forall \boldsymbol{y}_i \in N_j\} \quad (3)$$

where

$$N_j = \{\boldsymbol{y}_{i-l}(t), \boldsymbol{y}_{i-l+1}(t), .., \boldsymbol{y}_{i-1}(t), \boldsymbol{y}_i(t), \boldsymbol{y}_{i+1}(t), .., \boldsymbol{y}_{i+l-1}(t), \boldsymbol{y}_{i+l}(t)\} \quad (4)$$

Neighborhoods are usually determined using particle indices [41], however, topological neighborhoods can also be used [41]. It is clear that *gbest* is a special case of *lbest* with $l = s$; that is, the neighborhood is the entire swarm. While the *lbest* approach results in a larger diversity, it is still slower than the *gbest* approach.

For each iteration of a PSO algorithm, The velocity $\boldsymbol{v}_i$ update step is specified for each dimension $j \in 1..N_d$, where N_d is the dimension of the problem. Hence, v_{ij} represents the j^{th} element of the velocity vector of the i^{th} particle. Thus the velocity of particle *i* is updated as using the following equation:

$$v_{i,j}(t+1) = w v_{i,j}(t) + c_1 r_{1,j}(t)(y_{i,j}(t) - x_{i,j}(t)) + c_2 r_{2,j}(t)(\hat{y}_j(t) - x_{i,j}(t)) \quad (5)$$

where *w* is the inertia weight [42], c_1 and c_2 are the acceleration constants and $r_{1,j} \sim U(0,1)$ and $r_{2,j} \sim U(0,1)$. Equation 5 consists of three components, namely

- The *inertia weight* term, *w*, which serves as a memory of previous velocities. The inertia weight controls the impact of the previous velocity: a large inertia weight favors exploration, while a small inertia weight favors exploitation [40].
- The cognitive component, $\boldsymbol{y}_i(t) - \boldsymbol{x}_i$, which represents the particle's own experience as to where the best solution is.
- The social component, $\hat{\boldsymbol{y}}(t) - \boldsymbol{x}_i(t)$, which represents the belief of the entire swarm as to where the best solution is. Different social structures have been investigated [43],[44], with the star topology being used most.

The position of particle *i*, $\boldsymbol{x}_i$, is then updated using the following equation:

$$\boldsymbol{x}_i(t+1) = \boldsymbol{x}_i(t) + \boldsymbol{v}_i(t+1) \quad (6)$$

The reader is referred to Van den Bergh [45] and Van den Bergh *et al*. [46] for a study of the relationship between the inertia weight and acceleration constants, in order to select values which will ensure convergent behavior. Velocity updates can also be clamped through a user defined maximum velocity, V_{max}, which would prevent them from exploding, thereby causing premature convergence [45].

The PSO algorithm performs the update equations above, repeatedly, until a specified number of iterations have been exceeded, or velocity updates are close to zero. The quality of particles is measured using a fitness function which reflects the optimality of a particular solution.

Kennedy and Eberhart have adapted PSO to search in binary space [47]. In binary PSO, the component values of $\boldsymbol{x}_i$ and $\boldsymbol{y}_i$ are restricted to the set {0, 1}. The velocity, $\boldsymbol{v}_i$ is interpreted as a probability to change a bit from 0 to 1, or from 1 to 0 when updating the position of particles. This can be done using a sigmoid function, defined as

$$sig(x) = \frac{1}{1+e^{-x}} \tag{7}$$

Hence, the equation for updating positions (Eq. 6) is replaced by the probabilistic update equation, namely [45]:

$$x_{i,j}(t+1) = \begin{cases} 0 & \text{if } r_j(t) \geq sig(v_{i,j}(t+1)) \\ 1 & \text{if } r_j(t) < sig(v_{i,j}(t+1)) \end{cases} \tag{8}$$

Where $r_j(t) \sim U(0,1)$. This paper makes use of a binary PSO to optimize the number of clusters.

III. The DCPSO Algorithm

This section presents the DCPSO algorithm. For this purpose define the following symbols:

- N_c is the maximum number of clusters.
- N_d is the dimension of the data set.
- N_p is the number of data vectors to be clustered.
- $\boldsymbol{Z} = \{z_{j,p} \in \Re \mid j = 1,\ldots, N_d \text{ and } p = 1,\ldots, N_p\}$ is the set of data points.
- $\boldsymbol{M} = \{m_{j,k} \in \Re \mid j = 1,\ldots, N_d \text{ and } k = 1,\ldots, N_c\}$ is the set of N_c cluster centroids.
- $\boldsymbol{S} = \{\boldsymbol{x}_1,\ldots, \boldsymbol{x}_i,\ldots, \boldsymbol{x}_s\}$ is the swarm of s particles such that $\boldsymbol{x}_i$ indicates particle i, with $x_{i,k} \in \{0,1\}$ for $k = 1,\ldots, N_c$ such that if $x_{i,k} = 1$ then the corresponding $\boldsymbol{m}_k$ in $\boldsymbol{M}$ has been chosen to be part of the solution proposed by $\boldsymbol{x}_i$. Otherwise, if $x_{i,k} = 0$ then the corresponding $\boldsymbol{m}_k$ in $\boldsymbol{M}$ is not part of the solution proposed by $\boldsymbol{x}_i$.
- n_i is the number of clusters used by the clustering solution represented by particle $\boldsymbol{x}_i$ such that

$$n_i = \sum_{k=1}^{N_c} x_{i,j} \text{, with } n_i \leq N_c.$$

- $\boldsymbol{M}_i$ is the clustering solution represented by particle $\boldsymbol{x}_i$ such that $\boldsymbol{M}_i = (\boldsymbol{m}_k)\ \forall\ k$: $x_{i,k} = 1$ with $\boldsymbol{M}_i \subseteq \boldsymbol{M}$.
- n_τ is the number of clusters used by the clustering solution represented by the global best particle $\hat{\boldsymbol{y}}$ such that

$$n_\tau = \sum_{k=1}^{N_c} \hat{y}_k \text{, with } n_\tau \leq N_c.$$

- $\boldsymbol{M}_\tau$ is the clustering solution represented by $\hat{\boldsymbol{y}}$ such that $\boldsymbol{M}_\tau = (\boldsymbol{m}_k)\ \forall\ k$: $\hat{y}_k = 0$ with $\boldsymbol{M}_\tau \subseteq \boldsymbol{M}$.
- $\boldsymbol{M}_r$ is the set of centroids in $\boldsymbol{M}$ which have not been chosen by $\hat{\boldsymbol{y}}$ such that $\boldsymbol{M}_r = (\boldsymbol{m}_k)\ \forall\ k$: $\hat{y}_k = 0$ with $\boldsymbol{M}_r \subseteq \boldsymbol{M}$ (i.e. $\boldsymbol{M}_r \cap \boldsymbol{M}_\tau = \varnothing$ and $\boldsymbol{M}_r \cup \boldsymbol{M}_\tau = \boldsymbol{M}$).
- p_{ini} is a user-specified probability defined in [16], which is used to initialize a particle position, $\boldsymbol{x}_i$, as follows:

$$x_{i,k}(t) = \begin{cases} 0 & \text{if } r_k(t) \geq p_{\text{ini}} \\ 1 & \text{if } r_k(t) < p_{\text{ini}} \end{cases} \tag{9}$$

where $r_k(t) \sim U(0,1)$. Obviously a large value for p_{ini} results in selecting most of the centroids in $\boldsymbol{M}$.

The algorithm works as follows: A pool of cluster centroids, $\boldsymbol{M}$, is randomly chosen from $\boldsymbol{Z}$. The swarm of particles, $\boldsymbol{S}$, is then randomly initialized. Binary PSO is then applied to find the "best" set of cluster centroids, $\boldsymbol{M}_\tau$, from $\boldsymbol{M}$. K-means is applied to $\boldsymbol{M}_\tau$ in order to refine the chosen centroids. $\boldsymbol{M}$ is then set to $\boldsymbol{M}_\tau$ plus $\boldsymbol{M}_r$ which a randomly chosen set of centroids from $\boldsymbol{Z}$ (to add diversity to $\boldsymbol{M}$). The algorithm is then repeated using the new $\boldsymbol{M}$. When the termination criteria are met, $\boldsymbol{M}_\tau$ will be the resulting "optimum" set of cluster centroids and n_τ will be the "optimum" number of clusters in $\boldsymbol{Z}$.

The DCPSO algorithm is summarized below:

1. Select $\boldsymbol{m}_k \in \boldsymbol{M}\ \forall\ k = 1,\ldots, N_c$ where $1 < N_c < N_p$, randomly from $\boldsymbol{Z}$
2. Initialize the swarm $\boldsymbol{S}$, with $x_{i,k} \sim U\{0,1\}\ \forall\ i = 1,\ldots, s$ and $k = 1,\ldots, N_c$ using Eq. 9.
3. Randomly initialize the velocity $\boldsymbol{v}_i$ of each particle i in $\boldsymbol{S}$ such that $v_{i,k} \in [-5,5]\ \forall\ i = 1,\ldots, s$ and $k = 1,\ldots, N_c$. The range of [-5,5] was set empirically.
4. **For** each particle $\boldsymbol{x}_i$ in $\boldsymbol{S}$
   a. Partition $\boldsymbol{Z}$ according to the centroids in $\boldsymbol{M}_i$ by assigning each data point $\boldsymbol{z}_p$ to the closest (in terms of the Euclidean distance) cluster in $\boldsymbol{M}_i$.
   b. Calculate the clustering validity index, *VI*, using one of the clustering validity indices as defined in Section III.A to measure the quality of the resulting partitioning of $\boldsymbol{Z}$.
   c. $f(\boldsymbol{x}_i) = VI$
5. Apply the binary PSO velocity and position update equations (Eqs. 5 and 8) on all particles in $\boldsymbol{S}$.
6. Repeat steps 4) and 5) until the termination criteria are met.
7. Adjust $\boldsymbol{M}_\tau$ by applying the K-means clustering algorithm.
8. Randomly re-initialize $\boldsymbol{M}_r$ from $\boldsymbol{Z}$.
9. Set $\boldsymbol{M} = \boldsymbol{M}_r \cup \boldsymbol{M}_\tau$.
10. Repeat steps 2) through 9) until termination criteria are met.

The termination criteria can be a user-defined maximum number of iterations or a lack of progress in improving the best solution found so far for a user-specified consecutive number of iterations, *TC*. In this paper, the latter approach is used with $TC = 50$ for Step 6 and $TC = 2$ for step 10. These values for *TC* were set empirically.

N_c and s are user defined parameters. Large values for N_c and s are recommended to find a good solution.

A. Validity Index

One of the advantages of the DCPSO is that it can use any validity index. Therefore, the user can choose the validity index suitable for his/her data set. In addition, any new index

can easily be integrated with DCPSO. The validity indix proposed by Turi [24] is briefly described below. In the following, K is the number of clusters and $\boldsymbol{C}_k$ is the k^{th} cluster.

$$V = (c \times \mathrm{N}(2,1) + 1) \times \frac{\text{intra}}{\text{inter}} \tag{10}$$

where c is a user specified parameter and N(2,1) is a Gaussian distribution with mean 2 and standard deviation of 1. The *intra* term is the average of all the distances between each data point and its cluster centroid which is defined as

$$\text{intra} = \frac{1}{N_p} \sum_{k=1}^{K} \sum_{\boldsymbol{u} \in \boldsymbol{C}_k} || \boldsymbol{u} - \boldsymbol{m}_k ||^2$$

This term is used to measure the compactness of the clusters. The *inter* term is the minimum distance between the cluster centroids which is defined as

$$\text{inter} = min\left\{ || \boldsymbol{m}_k - \boldsymbol{m}_{kk} ||^2 \right\} \forall\ k = 1,2,..., K-1 \text{ and } kk = k+1,..., K$$

This term is used to measure the separation of the clusters.

IV. Experimental Results

Experiments were conducted using natural images. The following well known images were used as examples of natural images: *Lenna*, *mandrill*, *jet* and *peppers*. Furthermore, one MRI and one satellite image of Lake Tahoe (shown in Table I) have been used to show the wide applicability of the proposed approach.

The results reported in this section are averages and standard deviations over 20 simulations. In addition, we start with a *lbest* implementation of the PSO (with zero-radius neighborhood) and linearly increase the neighborhood radius until a *gbest* implementation of the PSO is reached. In this paper, this approach is referred to as *lbest-to-gbest*-PSO. This hybrid approach is used in order to initially avoid being trapped in local optima, by using a *lbest* approach [41]. The algorithm then attempts to converge into the best solution found by the initial phase by using a *gbest* approach. Furthermore, if the best solution has not been improved after a user-specified number of iterations (50 iterations was used for all the experiments conducted) then the algorithm will terminate (Step 6 of the algorithm, Section 3). For the index proposed by Turi [24], the parameter, c, was set to 25 in all experiments as recommended by [24]. The DCPSO parameters were empirically set as follows: $N_c = 20$, $p_{\text{ini}} = 0.75$ and $s = 100$ for all experiments conducted. In addition, the PSO parameters were set as follows: $w = 0.72$, $c_1 = c_2 = 1.49$ and $V_{\max} = 255$. For the SOM, all implementation issues where set as in [35], using a Kohonen network of 5×4 nodes.

Table II shows the results of DCPSO using the the validity index described in Section III. These results are compared with the results of SOM. In addition, the results of snob for the images of Lenna, mandrill, jet and peppers are copied from [24]. The optimal range for the number of clusters for the images of Lenna, mandrill, jet and peppers are also taken from [24] which was based on a visual analysis survey conducted by a group of ten people. Similarly, the optimal range for the MRI and Lake Tahoe images were estimated by the authors using a group of three people. It appears from the table that results of SOM and snob were poor. DCPSO using V always found a solution within the optimal range. These results clearly show the efficiency of DCPSO.

V. Conclusion

This paper presented DCPSO, a new dynamic clustering algorithm based on PSO with application to unsupervised image classification. DCPSO clusters a data set without requiring the user to specify the number of clusters in advance. This is an important feature since knowing the number of clusters in advance is often not easy. DCPSO uses a validity index to measure the quality of the resultant clustering. DCPSO has been applied on natural images (including MRI and satellite images), and has been compared with other unsupervised clustering techniques. From these experiments it can be concluded that DCPSO using the validity index proposed by Turi [24] has outperformed other approaches. In general, DCPSO successfully found the optimum number of clusters on the tested images.

References

[1] A.K. Jain, M.N. Murty, P.J. Flynn, Data Clustering: A Review, ACM Computing Surveys, vol. 31(3), 264-323,1999.
[2] A.K. Jain, R. Duin, J. Mao, Statistical Pattern Recognition: A Review, IEEE Transactions on Pattern Analysis and Machine Intellgence, vol. 22 (1), 4-37, 2000.
[3] D. Judd, P. Mckinley, A.K. Jain, Large-scale Parallel Data Clustering, IEEE Transactions on Pattern Analysis and Machine Intellgence, vol. 20 (8), 871-876, 1998.
[4] H.M. Abbas, M.M. Fahmy, Neural Networks for Maximum Likelihood Clustering, Signal Processing, vol. 36(1), 111-126, 1994.
[5] G.B. Coleman, H.C. Andrews, Image Segmentation by Clustering, Proc. IEEE, vol. 67, 773-785, 1979.
[6] A.K. Jain, R.C. Dubes, Algorithms for Clustering Data, New Jersey, Prentice Hall, 1988.
[7] S. Ray, R.H. Turi, Determination of Number of Clusters in K-Means Clustering and Application in Colour Image Segmentation, Proceedings of the 4th International Conference on Advances in Pattern Recognition and Digital Techniques (ICAPRDT'99), Calcutta, India, 137-143, 1999.
[8] C. Carpineto, G. Romano, A Lattice Conceptual Clustering System and Its Application to Browsing Retrieval, Machine Learning, vol. 24(2), 95-122, 1996.
[9] C.-Y. Lee, E.K. Antonsson, Dynamic Partitional Clustering Using Evolution Strategies, In The Third Asia-Pacific Conference on Simulated Evolution and Learning, 2000.
[10] G. Hamerly, C. Elkan, Learning the K in K-means, 7th Annual Conference on Neural Information Processing Systems, 2003.
[11] H. Frigui and R. Krishnapuram, A Robust Competitive Clustering Algorithm with Applications in Computer Vision, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 21(5), 450-465, 1999.
[12] Y. Leung, J. Zhang, Z. Xu, Clustering by Space-Space Filtering, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 22(12), 1396-1410, 2000.
[13] M. Halkidi, Y. Batistakis, M. Vazirgiannis, On Clustering Validation Techniques, Intelligent Information Systems Journal, Kluwer Pulishers, vol. 17(2-3), 107-145, 2001.
[14] S. Theodoridis and K. Koutroubas, Pattern Recognition, Academic Press, 1999.

[15] C. Rosenberger and K. Chehdi, Unsupervised Clustering Method with Optimal Estimation of the Number of Clusters: Application to Image Segmentation, International Conference on Pattern Recognition (ICPR'00), vol. 1, 1656-1659 (2000).
[16] L. Kuncheva and J. Bezdek, Nearest Prototype Classification: Clustering, Genetic Algorithms, or Random Search?, IEEE Transactions on Systems, Man, and Cybernetics-Part C: Applications and Reviews, vol. 28(1), 160-164, 1998.
[17] G. Ball and D. Hall, A Clustering Technique for Summarizing Multivariate Data, Behavioral Science, vol. 12, 153-155, 1967.
[18] K. Huang, A Synergistic Automatic Clustering Technique (Syneract) for Multispectral Image Analysis, Photogrammetric Engineering and Remote Sensing, vol. 1(1), 33-40, 2002.
[19] D. Pelleg, A. Moore, X-means: Extending K-means with efficient estimation of the number of clusters, Proceedings of the 17th International Conference on Machine Learning, 727-734, Morgan Kaufmann, San Francisco, CA, 2000.
[20] R. Kass, L. Wasserman, A reference Bayesian test for nested hypotheses and its relationship to the Schwarz criterion, Journal of the American Statistical Association, vol. 90(431), 928-934, 1995.
[21] G. Hamerly, Learning structure and concepts in data using data clustering, PhD Thesis, University of California, San Diego, 2003.
[22] C.S. Wallace, D.L. Dowe, Intrinsic classification by MML – the snob program, Proceedings 7th Australian Joint Conference on Artificial Intelligence, UNE, Armidale, NSW, Australia, 37-44, 1994.
[23] C.S. Wallace, An improved program for classification, Technical Report No. 47, Department of Computer Science, Monash University, Australia, 1984.
[24] R.H. Turi, Clustering-Based Colour Image Segmentation, PhD Thesis, Monash University, Australia, 2001.
[25] C.S. Wallace, D.M. Boulton, An information measure for classification, The Computer Journal, vol. 11, 185-194, 1968.
[26] J.J. Oliver, D. Hand, Introduction to minimum encoding inference, Technical Report No. 94/205, Department of Computer Science, Monash University, Australia, 1994.
[27] H. Bischof, A. Leonardis, A. Selb, MDL principle for robust vector quantization, Pattern analysis and applications, 2, 59-72, 1999.
[28] I. Gath, A. Geva, Unsupervised Optimal Fuzzy Clustering, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 11(7), 773-781, 1989.
[29] A. Lorette, X. Descombes, J. Zerubia, Fully Unsupervised Fuzzy Clustering with Entropy Criterion, International Conference on Pattern Recognition (ICPR'00), vol. 3, 3998-4001, 2000.
[30] N. Boujemaa, On Competitive Unsupervised Clustering, International Conference on Pattern Recognition (ICPR'00), vol. 1, 1631-1634, 2000.
[31] H. Frigui and R. Krishnapuram, Clustering by Competitive Agglomeration, Pattern Recognition Letters, vol. 30(7), 1109-1119, 1997.
[32] H. Frigui and R. Krishnapuram, A Robust Competitive Clustering Algorithm with Applications in Computer Vision, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 21(5), 450-465, 1999.
[33] T. Kohonen, Self-Organizing Maps, Springer Series in Information Sciences, 30, Springer-Verlag, N.Y., 1995.
[34] K. Mehrotra, C. Mohan, Rakka, Elements of Artificial Neural Networks, MIT Press, 1997.
[35] A. Pandya, R. Macy, Pattern Recognition with Neural Networks in C++, CRC Press, 1996.
[36] M. Halkidi, M. Vazirgiannis, Clustering Validity Assessment: Finding the Optimal Partitioning of a data set, Proceedings of ICDM Conference, CA (USA), Nov. 2001.
[37] J. Kennedy, R. Eberhart, Particle Swarm Optimization, Proceedings of IEEE International Conference on Neural Networks, Perth, Australia, vol. 4, 1942-1948, 1995.
[38] J. Kennedy, R. Eberhart, Swarm Intelligence, Morgan Kaufmann, 2001.
[39] A. Engelbrecht, Computational Intelligence: An Introduction, John Wiley and Sons, 2002.
[40] Y. Shi, R. Eberhart, Parameter Selection in Particle Swarm Optimization, Evolutionary Programming VII: Proceedings of EP 98, 591-600, 1998.
[41] P. Suganthan, Particle Swarm Optimizer with Neighborhood Optimizer, Proceedings of the Congress on Evolutionary Computation, 1958-1962, 1999.
[42] Y. Shi, R. Eberhart, A Modified Particle Swarm Optimizer, Proceedings of the IEEE International Conference on Evolutionary Computation, Piscataway, NJ, 69-73, 1998.
[43] J. Kennedy, Small Worlds and Mega-Minds: Effects of Neighborhood Topology on Particle Swarm Performance, Proceedings of the Congress on Evolutionary Computation, 1931-1938, 1999.
[44] J. Kennedy, R. Mendes, Population Structure and Particle Performance, Proceedings of the IEEE Congress on Evolutionary Computation, Honolulu, Hawaii, 2002.
[45] F. Van den Bergh, An Analysis of Particle Swarm Optimizers, PhD Thesis, Department of Computer Science, University of Pretoria, 2002.
[46] F. van den Bergh, A.P. Engelbrecht, A New Locally Convergent Particle Swarm Optimizer, Proceedings of the IEEE Conference on Systems, Man, and Cybernetics, Hammamet, Tunisia, 2002.
[47] J. Kennedy, R. Eberhart, A Discrete Binary Version of the Particle Swarm Algorithm, Proceedings of the Conference on Systems, Man, and Cybernetics, 4104-4109, 1997.

TABLE I
MRI AND LAKE TAHOE IMAGES

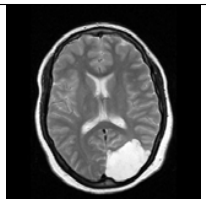
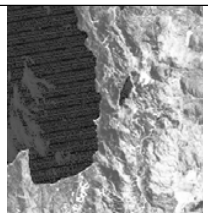
Image name	Image
MRI	
Tahoe	

TABLE II
EXPERIMENTS ON NATURAL IMAGES

Image	Optimal range	DCPSO using V	SOM	snob
Lenna	5 to 10	6.85 ± 0.477	20	31
Mandrill	5 to 10	6.25 ± 0.433	20	42
Jet	5 to 7	5.3 ± 0.459	14	22
peppers	6 to 10	6 ± 0	20	39
MRI	4 to 8	5 ± 0	19	-
Tahoe	3 to 7	6.1 ± 0.539	4	-