Automatic image clustering using a swarm intelligence approach

Salima Ouadfel[1], Mohamed Batouche[2] and Abdlemalik Ahmed-Taleb[3]

[1] University of Batna Computer Science department
Batna 33000, Algeria

[2] COEIA – CCIS, King Saud University,
Riyadh, Kingdom of Saudi Arabia

[3] LAMIH UMR CNRS UVHC
8530Valenciennes, France

Abstract

In order to implement clustering under the condition that the number of clusters is not known a priori, we propose in this paper ACPSO a novel automatic image clustering algorithm based on particle swarm optimization algorithm. ACPSO can partition image into compact and well separated clusters without any knowledge on the real number of clusters. ACPSO used a novel representation scheme for the search variables in order to determine the optimal number of clusters. The partition of each particle of the swarm evolves using evolving operators which aim to reduce dynamically the number of clusters centers. Experimental results on real images demonstrate the effectiveness of the proposed approach.

Keywords: *Image clustering, swarm intelligence, Particle swarm optimization, automatic clustering.*

1. Introduction

Image segmentation is an important technology for image processing, and also is a fundamental process in many image, video, and computer vision applications. The goal of image segmentation is to cluster pixels into salient image regions, such as regions corresponding to individual surfaces, objects, or natural parts of [1]. Clustering process aims to partition the image into clusters such that the pixels within a cluster are as homogenous as possible whereas the clusters among each other are as heterogeneous as possible with respect to some similarity measure.

Several clustering methods are provided in the literature [2]. They fall into two categories: hierarchical and partitioning methods. Hierarchical methods proceed by stages producing a sequence of partitions, where each partition corresponds to a different number of clusters. A hierarchical algorithm yields a tree representing the nested grouping of patterns. Partitioning methods obtain a single partition of the pixels by moving pixels iteratively from one group to another, starting from an initial partition. An extensive survey of various clustering techniques can be found in [2]. The focus of this paper is on the partitional clustering algorithms.

Hard or crisp partitional clustering [3] and fuzzy partitional clustering [4] are two partitioning clustering algorithms such that hard clustering assigns each data point to only one cluster while fuzzy clustering assigns each data point to several clusters with varying degrees of memberships. The most widely used hard partitioning algorithm is the iterative K-means approach [5, 6, 7]. In the K-means algorithm, pixels with similar features like gray levels or colors are grouped in the same cluster. The clustering is obtained by iteratively minimizing a cost function that is dependent on the distance of the pixels to the cluster centers. The major problem with this algorithm like most of the existing clustering algorithms is that its result is sensitive to the selection of the initial partition, it may converge to local optima and it requires the a priori specification of the number of clusters *K*.

To deal with the limitations existing in the traditional partition clustering methods, a number of new clustering algorithms have been proposed with the inspiration coming from observations of natural processes [8].

In order to remedy the drawbacks of K-means, this paper proposes a new automatic image clustering algorithm based on a modified version of particle swarm optimization. The proposed algorithm, called by us the ACPSO (Automatic Clustering with PSO) effectively search for both the optimal cluster centers positions and the number of effective clusters, and this with minimal user interference. ACPSO has the following characteristics: (1) particles can contain different cluster number in a range defined by minimum and maximum cluster number, (2) Particles are initialized randomly to process different cluster numbers in a specified range, (3) The goal of each

particle is to search the optimum number of clusters and the optimum cluster centers, (4) Three new evolving operators are introduced to evolve dynamically the partitions encoded in the particles.

The paper is organized as follows. Section 2 defines the clustering problem in a formal language and gives a brief overview of a previous works done in the field of unsupervised partitional clustering. Section 3 presents a description of PSO algorithm. Section 4 outlines the proposed ACPSO algorithm. In section 5, we present the experimental results as well as a comparative study. Finally, conclusion is drawn.

2. Scientific background

2.1 Problem definition

The clustering problem can be formally defined as follows. Given a data set $Z = \{z_1, z_2, ...z_n\}$ where z_i is a data item and n is the number of data items in *Z*. The clustering aims to partitioning *Z* into *K* compacts and well separated clusters.

Compactness means that members of a cluster are all similar and close together. One measure of compactness of a cluster is the average distance of the cluster instances compared to the cluster center.

$$compactness(c_j) = \frac{1}{n_j} \sum_{z_i \in C_j} \left(z_i - m_j\right)^2 \quad (1)$$

where m_j is the center of the *j*th cluster c_j and n_j is its cardinal. Lower value of *compactness* (c_j) is better.

Thus, the overall compactness of a particular grouping of *K* clusters is just the sum of the compactness of the individual clusters

$$compactness = \frac{1}{n} \sum_{j=1}^{K} \sum_{z_i \in C_j} \left(z_i - m_j\right)^2 \quad (2)$$

Separability means that members of one cluster are sufficiently different from members of another cluster (cluster dissimilarity). One measure of the separability of two clusters c_i and c_j is their squared distance.

$$separability(c_i, c_j) = \left\| m_i - m_j \right\| \quad (3)$$

where m_i and m_j are the center of the *i*th and *j*th cluster respectively.

The separability of the partition of *K* clusters could be defined as following:

$$Separability = \sum_{i=1}^{K} \min_j separability(c_i, c_j) \quad (4)$$

The bigger the distance, the better the separability, so we would like to find groupings where separability is maximized.

2.1 Unsupervised clustering algorithms

Clustering can be formally considered as a particular kind of NP-hard grouping problem [9]. This assumption has stimulated much research and use of efficient approximation algorithms.

One of the most frequently used clustering algorithms is the iterative K-means algorithm [10, 11]. The K-means algorithm starts with *K* cluster centers randomly selected using some heuristics. Each data item in the data set is then assigned to the closest cluster center according to a distance measure. The centers are updated by using the mean of the associated items. The process is repeated until some stopping criterion is verified. Although the k-means algorithm has been widely used due to its easy implementation, it has two major drawbacks: it is too sensitive to the initial clusters centers and it needs to specify the number of clusters in advance. However, in many practical cases, it is impossible to determine the exact cluster number in advance. Under these circumstances, the k-means algorithm often leads to a poor clustering performance.

In the literature, many approaches to finding dynamically the number of clusters has been proposed. In [12], the ISODATA (Iterative Self-Organizing data Analysis technique) was proposed. Like the K-means algorithm, ISODATA assigns each item to the closest cluster center; however, it adds division of a cluster c_i into two clusters if the cluster standard deviation of c_i exceeds a user-specified threshold th_{div}, and processing of fusion of two clusters if the distance between their centers is smaller than another user-specified threshold th_{merg}. Using this variant, the optimal partition starting from any arbitrary initial partition can be obtained. However, it requires many parameters to be specified by the user. In [13], the authors proposed SYNERACT, which combines K-means with hierarchical descending approaches. In [14] Rosenberger and Chehdi introduced a new improvement to the K-means algorithm. During each step of the clustering process, from a set of K clusters, a cluster with the higher intra_cluster distance is chosen for splitting into two clusters. Next, the K-means algorithm is applied to the (K+1) clusters. The iterative procedure is repeated until a valid partition of the data items is obtained. Pelleg and Moore [15] proposed X-means algorithm which is based on the classical K-means algorithm with the model selection. Hamerly [16] proposed G-means algorithm which splits clusters that not fit a Gaussian distribution.

Since the problem of data clustering can be easily viewed as a complex optimization problem [17], several optimization algorithms have been used for optimizing the cost function and to find the optimal number of clusters. For example, in [18], the authors proposed a nonparametric Variable string length genetic algorithm (VGA), with real encoding of the cluster centers in the chromosome. In [19] a novel variable length GA (VLIGA) algorithm which is an improvement version of VGA was proposed with a modified mutation function. In [20], authors proposed an evolutionary-fuzzy clustering algorithm for automatically grouping the pixels of an image into different homogeneous regions. The algorithm does not require a prior knowledge of the number of clusters. The fuzzy clustering task in the intensity space of an image is formulated as an optimization problem. An improved variant of the differential evolution (DE) algorithm has been used to determine the number of naturally occurring clusters in the image as well as to refine the cluster centers. Bandyopadhyay proposed in [21] a Variable String Length Simulated Annealing (VFC-SA) algorithm, which applied a simulated annealing algorithm to the fuzzy c-means clustering technique and used a cluster validity index measure as the energy function. Tseng and Yang [22] proposed a genetic algorithm based approach for the clustering problem. The proposed method can search for a proper number of clusters and classify non overlapping objects into these clusters. Lin et al. [23] presented an automatic genetic clustering algorithm based on a binary chromosome representation. Lai [24] adopted the hierarchical genetic algorithm to solve the clustering problem. In the proposed method, the chromosome consists of two types of genes, control genes and parametric genes. The control genes are coded as binary digits. The parametric genes are coded as real numbers to represent the coordinates of the cluster centers. The total number of "1" represents the number of clusters. In [25] authors proposed an algorithm to determine the optimal number of clusters by applying SA to cluster microarray data. In their method, first the fuzzy k-means algorithm is used to minimize the sum of within-cluster distance, then, the optimal number of clusters is obtained from the SA algorithm. In [26], authors proposed a dynamic clustering algorithm based on a modified version of classical Particle Swarm Optimization (PSO) algorithm, known as the Multi-Elitist PSO (MEPSO) model. A new particle representation scheme has been adopted for selecting the optimal number of clusters from several possible choices. It also employs a kernel-induced similarity measure instead of the conventional sum-of-squares distance. In [27] a new fuzzy clustering algorithm is proposed by combining the possibility clustering and ISODATA clustering algorithm. This new algorithm not only can determine the number of clusters dynamically with the degree of possibility of each date point, but also can reduce the number of input parameters of ISODATA algorithm. In [28] an approach for solving the automatic clustering of the Gene Ontology is proposed by incorporating cohesion-and-coupling metric into a hybrid algorithm consisting of a genetic algorithm and a split-and-merge algorithm. In [29] authors address the problem of cluster number selection by using a k-means approach that exploits local changes of internal validity indices to split or merge clusters. The split and merge k-means issues criterion functions to select clusters to be split or merged and fitness assessments on cluster structure changes. In [30] authors propose a Bacterial Evolutionary clustering algorithm, which can partition a given dataset automatically into the optimal number of groups. Experiments were done with several synthetic as well as real life data sets including a remote sensing satellite image data. The results establish the superiority of the proposed approach in terms of final accuracy. In [31] Omran et al. presented dynamic clustering PSO (DCPSO), which is, in fact, a hybrid clustering algorithm where binary PSO is used to determine the number of clusters while the traditional K-means method performs the clustering operation with this number of clusters. In [32], Abraham et al. combined the Fuzzy clustering algorithm with the multielitist PSO (MEPSO) to find automatically the number of clusters. In [33] authors proposed an evolutionary particle swarm optimization for data clustering. The proposed algorithm is based on the evolution of swarm generations. After each generation, the swarm dynamically adjusts itself in order to reach optimal position.

3. Particle swarm optimization

Particle swarm optimization (PSO) is a population-based evolutionary computation method first proposed by Kennedy and Eberhart [34]. It originated from the computer simulation of the individuals in a bird flock or fish school, which basically show a natural behavior when they search for some target (e.g., food). The PSO algorithm is initialized with a swarm of n particles randomly distributed over the search area with a random velocity and a random position. Each particle encodes a potential solution to the optimization problem. Particles flies through the search space and aims to converge to the global optimum of a function attached to the problem.

Each particle x_i in the swarm is represented by the following characteristics: the current position of the particle (p_i) and the current velocity of the particle (v_i). Its movement through the search space is influenced dynamically according to its personal best position *Pbest*, which is the best solution that it has so far achieved and its neighbors' best position P_g. At each iteration t, the

particle's new position and its velocity are updated as follows:

$$p_i(t) = p_i(t-1) + v_i(t) \quad (5)$$

$$v_i(t) = wv_i(t-1) + c_1 \times rand_1(p_{best} - p_i(t-1)) + c_2 \times rand_2(p_g - p_i(t-1)) \quad (6)$$

The parameter *w* is an inertia weight and it is equivalent to a temperature schedule in the simulated annealing algorithm and controls the influence of the previous velocity: a large value of *w* favors exploration, while a small value of *w* favors exploitation [35]. As originally introduced, w decreases linearly during the run from w_{min} to w_{max}. c_1 and c_2 are two constants which control the influence of the social and cognitive components such that $c_1 + c_2 = 4$. $rand_1$ and $rand_2$ are random values in the range [0,1].

Two topologies of neighborhoods exist in the literature: the *gbest* model and the *lbest* model. The *gbest* model maintains only a single best solution, called the global best particle, across all the particles in the swarm. This particle acts as an attractor, pulling all the particles towards it. The *gbest* offers a faster rate of convergence at the expense of robustness. The *lbest* model tries to prevent premature convergence by maintaining multiple attractors. In fact, *gbest* model is actually a special case of the *lbest* model. Experiments have shown that *lbest* algorithm converges somewhat more slowly than the *gbest* version, but it is less likely to become trapped in an inferior local minimum.

4. ACPSO algorithm

In this section, we describe an automatic image clustering algorithm based on a new version of particle swarm optimization algorithm, called ACPSO.

Let $Z = \{z_1, z_2, ...z_n\}$ be the image with *n* number of pixels. The ACPSO maintains a swarm of particles, where each particle represents a potential solution to the clustering problem. Each particle encodes an entire partition of the image Z. ACPSO tries to find an optimal partition $C = \{c_1, c_2, ...c_k\}$ of *K* optimal number of compactness and well separated clusters. In ACPSO, both the numbers of clusters as well as the appropriate clustering of the data are evolved simultaneously using the search capability of particle swarm optimization algorithm.

4.1 Particle representation

The initial population $P = \{X_1, X_2, X_3, ...X_{pop\_size}\}$ is made up of *pop_size* possible particles (solutions). For a user-defined maximum cluster number $K_{max,}$, a single particle x_i is a vector of K_{max} binary numbers 0 and 1 (flags) and K_{max} real numbers that represents the K_{max} cluster centers.

For a particle x_i, each probably cluster center m_{ij} (j=1…K_{max}) is associated with a binary flag $\gamma_{ij}(j = 1....K_{max})$. The cluster center m_{ij} is valid and so selected to clustering the image pixels, if it's corresponding flag $\gamma_{ij} = 1$ and invalid if $\gamma_{ij} = 0$. The total number of "1" implicitly represents the number of clusters encoded in a particle.

If due to the update of the position of a particle some flags in a particle exceed 1, it is fixed to 1 or zero, respectively. However, if it is found that no flag could be set to one in a particle (all cluster centers are invalid and so no selected), two random flags are selected and we re-initialize them to 1. Thus the minimum number of possible clusters is always 2.

Two examples of the particle structure in the proposed approach are shown in Figure 1.

$$[\overbrace{0,0,1,1,0}^{actvation-clusters-part}, \overbrace{12.5,\ 45.7,\ 36.5,\ 22.5,\ 66.3}^{cluster-centers-part}]$$

Particle *i* represents 2 clusters, and the associated cluster centers are 36.5 and 22.5.Cluster centers 12.5, 45.7 and 66.3 are invalid and not used to clustering the image.

$$[\overbrace{1,0,1,0,1}^{actvation-clusters-part}, \overbrace{39.5,\ 45.7,\ 26.5,\ 40.3,\ 33.3}^{cluster-centroids-part}]$$

Particle *i* represents 3 clusters, and the associated cluster centers are 39.5, 26.5 and 33.3. Cluster centers 45.7, 26.5 and 40.3 are invalid.

Figure1. Two examples of the particle structure in the ACPSO algorithm.

4.2 Population initialization

To generate the initial population of particles, we use in this paper the random generation strategy until all particles in a population are created. For a particular particle x_i , K_i cluster centers are randomly selected points from the given data set and K_i flags are randomly generated. Note that if the number of valid centers contained in a particle is less than two, then its flags are reinitialized.

4.3 Fitness evaluation

The fitness of a particle indicates the degree of goodness of the solution it represents. In this work, the fitness function of a particle is based on the Ray and Turi's validity criterion [36] proposed to color image segmentation using the $\frac{\text{int}ra}{\text{int}er}$ ratio with a multiplier function to avoid the selection of low cluster numbers. The criterion is defined as:

$$V(K) = \left(c \times N(2,1) + 1\right) \times \frac{\text{int}ra}{\text{int}er} \qquad (7)$$

where c=25 is a constant multiplier, K is the number of clusters found by the clustering algorithm and $N(2,1)$ is a Gaussian function with mean 2 and standard deviation of 1. The intra and inter cluster distances represent respectively the compactness and the separability measures of clusters and are defined by Eq. (2) and Eq. (4).

A lower value of $V(K)$ indicates a better quality of the clustering.

The Ray and Turi's measure based fitness function (to be maximized) for the particle x_i encoding K_i clusters is given by:

$$Fitness_i = \frac{1}{V(K_i) + eps} \qquad (8)$$

where eps is a small bias term equal to $2\text{x}10^{-4}$ and prevents the denominator of Eq. (8) from being equal to zero. When the algorithm converges, the particle that has the maximum Fitness value will be the optimal particle.

4.4 Evolving operators

The evolving operators are specifically designed to allow the number of the clusters of the particles to be changed dynamically. In the following, we describe each evolving operator.

4.4.1 Perturb operator

A valid cluster c_{ij} of the configuration encoded in a particle x_i is chosen randomly to be perturbed. The centre m_{ij} of the selected cluster c_{ij} is then modified as follows:

$$m_{ij}^{new} = m_{ij}^{old} + \delta * m_{ij}^{old} \qquad (9)$$

where m_{ij}^{new} and m_{ij}^{old} represent the new and the old cluster centre of the cluster c_{ij} . δ is a random number between [-1, 1].

Thus the cluster encoded by the particle is reconfigured, although the number of clusters belonging to it remains unaltered.

4.4.2 Split operator

For a particle x_i, we compute the compactness measure for each valid cluster according to Eq. (1). Let S the set of clusters c_{ij} $(j = 1...k_i)$ with the compactness measure higher than a threshold th_{split}. The threshold th_{split} is defined as the global compactness measure (see Eq. (2)) divided by the number of clusters of the particle x_i .

A cluster c_{ij} from the set S is selected for splitting into two new valid clusters, with the probability P_{split} defined as follows:

$$P_{split}(c_{ij}) = \frac{\frac{1}{n_j}\sum_{z_k \in c_{ij}} \left\| z_k - m_{ij} \right\|^2}{\sum_{s=1}^{K_i} \frac{1}{n_s} \sum_{z_k \in c_{is}} \left\| z_k - m_{is} \right\|^2} \qquad (10)$$

That is, the *sparser* cluster c_{ij}, the more possibly it is selected as the cluster for the split operator and vice versa.

The resulting number of clusters is K_i +1 and must be lower than K_{max}, otherwise, the split operator terminates.

4.4.3 Merge operator

For a particle x_i, first the pairwise separation distances D_{jl} between all distinct pairs of valid clusters (c_{ij},c_{il}) are calculated according to Eq. (3). Let S the set of pairs of valid cluster with the distance D_{jl} lower than a threshold th_{merge} . The threshold th_{merge} is defined as the average distance of D_{jl} for all distinct pairs of valid clusters.

A pair of distinct clusters (c_{ij},c_{il}) of S is selected for the merge operator with the probability P_{merge} defined as follows:

$$P_{merge}(c_{ij}, c_{il}) = 1 - \frac{D_{jl}}{\max(D_{jl})} \qquad (11)$$

where $\max(D_{jl})$ is the maximum pairwise separation distance between all distinct pairs of valid cluster centers from the set S.

The final number of clusters must be greater than 2, otherwise, the merge operator terminates.

Any one of the above-described evolving operators is applied for a particle if it is selected. The particle is selected with an adaptive probability P_e as in [37]. Let *gbest* the global best fitness of the current iteration; $\overline{Pbest}$ be the average fitness value of the population and $Pbest_i$ be the fitness value of the solution (particle) to be evolved. The expression for probability, P_e is given below:

$$P_e = \begin{cases} k_2 \times \dfrac{(gbest - Pbest_i)}{(gbest - \overline{Pbest})} & \text{if } Pbest_i > \overline{Pbest} \\ k_4 & \text{if } Pbest_i \le \overline{Pbest} \end{cases}$$

Here, values of k_2 and k_4 are kept equal to 0.5 [37]. This adaptive probability helps PSO to avoid getting stuck at local optimum.

The value of P_e increases when the fitness of the particle is quite poor. In contrast when the fitness of the particle is a good solution, P_e will be low so as to reduce the likelihood of disrupting good solution by evolving operators.

The framework of the ACPSO algorithm is given as follows:

1. Initialize the maximum cluster number K_{max} and all the constant parameters;
2. Initialize each particle x_i with random $k_i \in \{2,3,..,K_{\max}\}$, randomly selected cluster centers, flags and initial velocities.
3. Initialize for each particle x_i the $Pbest_i$
4. Initialize the *gbest*
5. For each particle x_i
   Calculate the fitness value $Fitness_i$ using Eq. 7.
   Set $Pbest_i = Fitness_i$.
   If ($Pbest_i > gbest$) then set $gbest = Pbest_i$.
6. Update the position and the velocity of each particle according to Eqs. (5) and (6)
7. Apply randomly the evolving operators to alter the clusters centers of each particle
8. If termination criterion is satisfied go to step 9 else go to step 5
9. Segment the image using the optimal number of clusters and the optimal clusters centers given by the best global particle.

5. Experimental results

In order to evaluate the ability of our algorithm ACPSO to find the optimal clusters, we have tested it using natural images with varying range of complexity.

The performance of three dynamic clustering algorithms, ACPSO, DCPSO and ISODATA, were compared.

The parameter settings of DCPSO and ISODATA algorithm were determined by both referring to original papers and performing empirical studies. In Table 1, we report an optimal set-up of the parameters that gives the best results.

Table1. Parameter setup of the clustering algorithms for the image segmentation problem

DCPSO		ACPSO		ISODATA	
parameter	value	parameter	value	parameter	value
Pope size	100	Pop size	50		
Inertia	0.72	W_{min} W_{max}	0.4 0.9	Threshold for split clusters	10
C_1, C_2 ξ_{ini}	1.494 0.75			Threshold for merge clusters	1
K_{max} K_{min}	20 2	K_{max} K_{min}	20 2	K_{max} K_{min}	20 2

The clustering algorithms used in the experimental tests have been run several times for each test image. The optimal number of clusters has not been provided to any of the three optimization algorithm. Table 2 and Table 3 report the experimental results obtained over the grayscale images in terms of the mean and standard deviations of the number of clusters found and the final Turi measure reached by the three clustering algorithms. The results have been stated over 40 independent runs in each case.

Table 2. Number of clusters found by the clustering algorithms for real grayscale images.

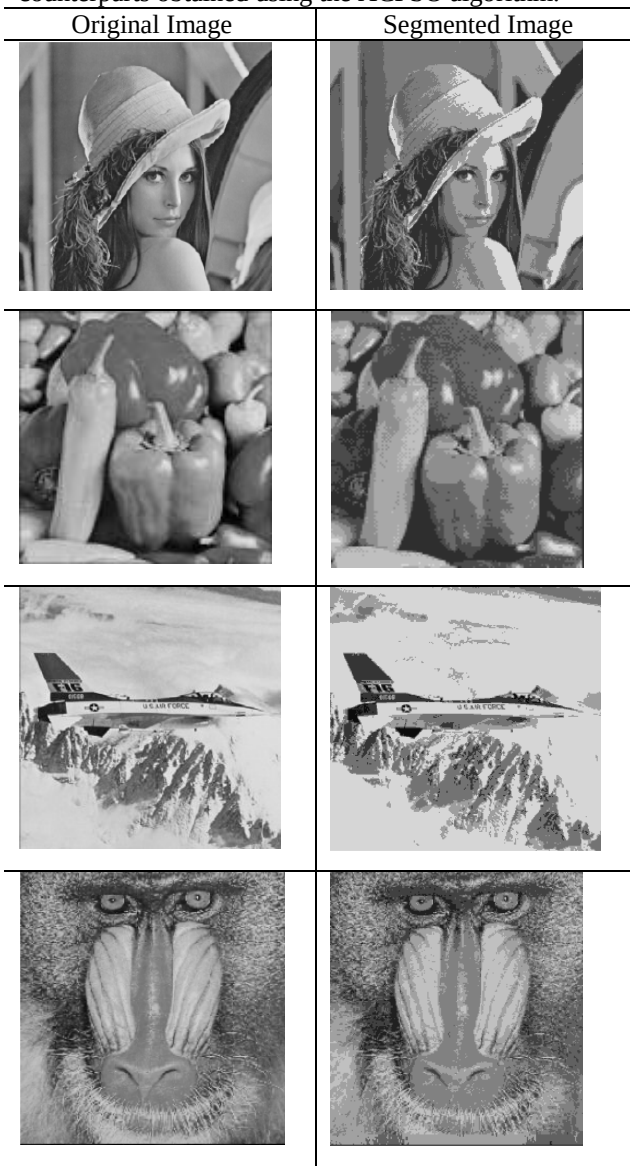
Image	Optimal number of clusters	ISODATA	DCPSO	ACPSO
LENA	7	6.79 ± 0034	6.65 ± 0.134	7.02 ± 0.234
MANDRILL	6	6.95 ± 0.004	6.25 ± 0.345	6.05 ± 0.456
CAMERAMAN	5	6 ± 0.010	5.3 ± 0.082	5.06 ± 0.0767
PEPPERS	7	6.581 ± 0.703	6.85 ± 0.064	7.190 ± 0.230
CLOUDS	4	3.667 ± 0.307	4.50 ± 0.132	4.290 ± 0.148
ROSE	3	4.50 ± 0.007	3.70 ± 0.637	3.2 ± 0.024
ROBOT	3	4.839 ± 1.926	2.30 ± 0.012	3.613 ± 0.146
JET	5	5.40 ± 0.967	5.6 ± 0.043	5.05 ± 0.023

Table 3. Automatic clustering result over real grayscale images using the Turi based fitness function over 40 independent runs.

	(Turi index)$^{-1}$		
Image	ISODATA	DCPSO	ACPSO
LENA	0.19	0.16	**0.12**
MANDRILL	0.14	0.12	**0.10**
CAMERAMAN	0.097	0.089	**0.086**
PEPPERS	0.16	0.12	**0.10**
CLOUDS	0.094	0.074	**0.070**
ROSE	0.107	0.097	**0.084**
ROBOT	0.19	0.067	**0.052**
JET	0.098	0.070	**0.057**

From Tables 2-3 we can see that the proposed algorithm ACPSO outperforms the state of-the-art DCPSO and ISODATA algorithms for the present images related problems. The proposed algorithm is able to find the optimal number of clusters with better clustering result in term of the Turi cluster validity index.

Figure 2 shows the original images and their segmented counterparts obtained using the ACPSO algorithm.



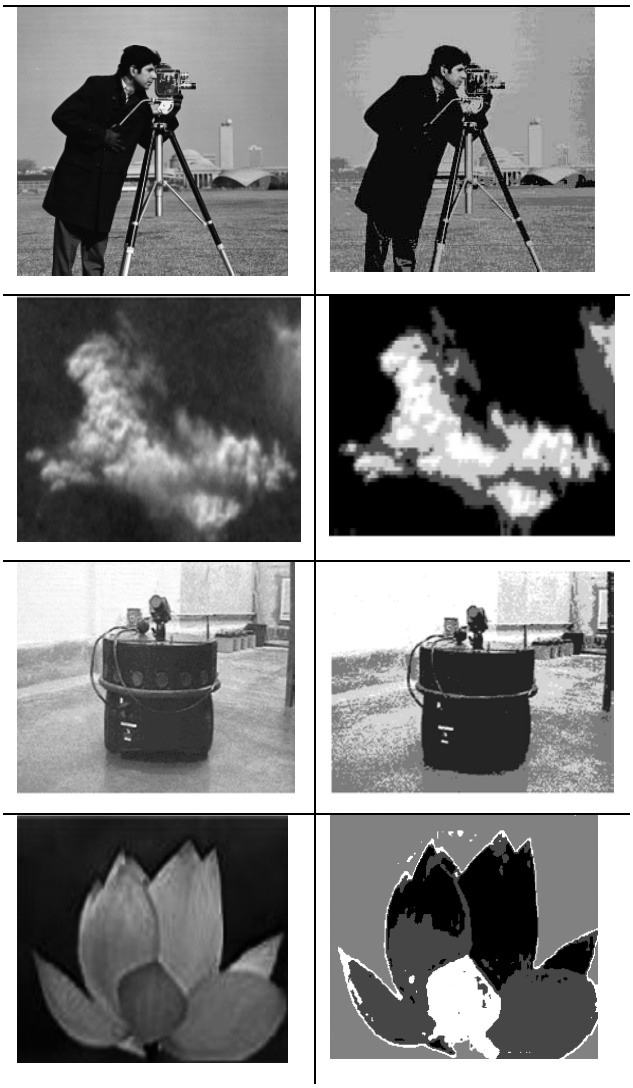


Figure 2. Samples of segmented images resulting from ACPSO

6. Conclusion

In this paper we have presented a new particle swarm optimization based method for automatic image clustering. ACPSO, in contrast to most of the existing clustering techniques, requires no prior knowledge of the data to be classified. ACPSO used a novel representation scheme for the search variables in order to determine the optimal number of clusters. Each particle encoded a partition of the image with a number of clusters chosen randomly from the set of the maximum number of clusters. The partition of each particle of the swarm evolves using evolving operators which aim to reduce dynamically the number of clusters centers. Superiority of the new method has been demonstrated by comparing it with ISODATA algorithm

and a recently developed partitional clustering technique based on Particle Swarm Optimization (PSO) algorithm.

References

[1] Monga O. and Wrobel B. Segmentation d'images: vers une méthodologie, Traitement du Signal, 1987, 4(3), 169-193.
[2] Jain, A.K., Murty, M.N., Flynn, P.J. :Data clustering: a review. ACM Computing Surveys 1999, 31(3), 264–323.
[3] Duda, R.O., Hart, P.E.: Pattern Classification and Scene Analysis. John Wiley and Sons, ManChichester (1973)
[4] Bezdek, J.C., Keller, J., Krishnampuram, R., Pal, N.R. Fuzzy Models and Algorithms for Pattern Recognition and Image Processing. Kluwer Academic Publishers, Dordercht (1999)
[5] Forgy, E.W. Cluster Analysis of Multivariate Data: Efficiency Versus Interpretability of classification. Biometrics 1965, 21, 768–769.
[6] Al-Sultan, K. S. and Khan, M. M. Computational experience on four algorithms for the hard clustering problem. Pattern Recogn. Lett.1996, 17(3), 295–308.
[7] Hartigan, J.A. Clustering Algorithms, John Wiley and Sons,724 Inc., New York, NY, 1975.
[8] Theodoridis, S., Koutroumbas, K.: Pattern Recognition, 2nd edn. Elsevier Academic Press, Amsterdam 2003.
[9] Hruschka, E.R.; Campello, R.J.G.B.; Freitas, A.A. & de Carvalho, A.C.P.L.F. A Survey of Evolutionary Algorithms for Clustering. IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews. 2009, 39(2), pp.133-155, March 2009, ISSN 1094-6977
[10] MacQueen, J.: Som methods for classification and analysis of multivariate observations. In: Proceedings of the Fifth Berkely Symposium on Mathematical Statistics and Probability, 1967, pp. 281–297.
[11] Selim, S.Z., Ismail, M.A. K-means type algorithms: a generalized convergence theorem and characterization of local optimality, *IEEE Transactions on Pattern Analysis and Machine Intelligence* 6 (1984) 81–87.
[12] Ball, G., Hall, D. A clustering technique for summarizing multivariate data. Behavioral Science 12, 1967, 153–155..
[13] Huang, K. A synergistic automatic clustering technique (Syneract) for multispectral image analysis. Photogrammetric Engineering and Remote Sensing 2002, 1(1), 33–40
[14] Rosenberger, C., Chehdi, K. Unsupervised clustering method with optimal estimation of the number of clusters: Application to image segmentation. In: Proc. IEEE International Conference on Pattern Recognition (ICPR), vol. 1, Barcelona, 2000, pp. 1656– 1659.
[15] Pelleg, D., Moore, A. X-means Extending K-means with efficient estimation of the number of clusters. In: Proceedings of the 17th International Conference on Machine Learning, 2000, pp. 727–734.
[16] Hamerly, G Learning structure and concepts in data using data clustering, PhD Thesis, University of California, San Diego, 2003.
[17] Halkidi, M., Batistakis, Y., Vazirgiannis, M. On clustering validation techniques. *Journal of Intelligent Information Systems* (JIIS) 2001, 17(2-3), 107–145.
[18] Bandyopadhyay, S., Maulik, U. Genetic clustering for automatic evolution of clusters and application to image classification. *Pattern Recognition* 2002, 35, 1197–1208.
[19] Venkatesh K., Satapathy, S. C, JVR Murthy, PVGD Prasad Reddy Hybridized Improved Genetic Algorithm with Variable Length Chromosome for Image Clustering. In *IJCSNS* , 2007, 7(11), 21-131.
[20] Das, S. and Konar, A. Automatic image pixel clustering with an improved differential evolution *Applied Soft Computing* 2009, (1) p. 226-236
[21] Bandyopadhyay, S. Simulated Annealing for Fuzzy Clustering: Variable Representation, Evolution of the Number of Clusters and remote Sensing Applications, 2003 unpublished, private communication
[22] Tseng, L. Y. and Yang, S. B. A genetic approach to the automatic clustering algorithm, *Pattern Recognition*, 2001, 34(2), pp. 415-424.
[23] Lin H. J., Yang, F. W. and Kao, Y. T. An efficient GA-based clustering technique, *Tamkang Journal of Science and Engineering*, 2005, 8(2), pp. 113-122.
[24] Lai, C. C., A novel clustering approach using hierarchical genetic algorithms, *Intelligent Automation and Soft Computing*, 2005, 11(3), pp. 143-153.
[25] Alexander V. Lukashin and Rainer Fuchs, Analysis of temporal gene expression profiles: Clustering by simulated annealing and determining the optimal number of clusters, *Bioinformatics*, 2000, 17(5), pp. 405–414.
[26] Das, S, Abraham, A and Konar, A Automatic Kernel Clustering with Multi-Elitist Particle Swarm Optimization Algorithm, *Pattern Recognition Letters*, Elsevier Science, 2008, 29, pp. 688-699.
[27] Liu, Wemping., Chih. Cheng. Hung, Bor. Chen. Kuo, and Tommy. Coleman, An Adaptive Clustering Algorithms Based on the Possibility Clustering and ISODATA for Multispectral Image Classification, proceedings of the International Society for Photogrammetry and Remote Sensing (ISPRS – XXI Congress), Beijing, China, 2008, July 3 – 11.
[28] Othman, R.M., Deris, S., Illias, R.M., Zakaria, Z. and Mohamad, S.M. Automatic clustering of gene ontology by genetic algorithm, Int'l J. Information Technology, 2006 3(1) pp. 37-46.
[29] Markus, M. and Grani, M. Automatic Cluster Number Selection Using a Split and Merge K-Means Approach 2009 20th International Workshop on Database and Expert Systems Application p. 363-367
[30] Das, S, Chowdhury,A, and Abraham, A. A Bacterial Evolutionary Algorithm for automatic data clustering Evolutionary Computation 2009 CEC 09 IEEE Congress on p. 2403-2410.
[31] Omran, M.G., Engelbrecht, A.P., Salman, A. Dynamic clustering using particle swarm optimization with application in image classification. In *Pattern Analysis and Application*. 2006, 332-344.
[32] Abraham, A., Das, S., and Roy, S. Swarm intelligence algorithms for data clustering in Soft Computing for Knowledge Discovery and Data Mining Book. New York: Springer-Verlag, , 2007, 279–313.
[33] Alam, S., Dobbie, G., Riddle, P. An Evolutionary Particle Swarm Optimization algorithm for data clustering. In Swarm Intelligence Symposium, 2008, 1-6.
[34] Kennedy, J., Eberhart, R.C. Particle swarm optimization. In: Proc. of the IEEE Int. Conf. on Neural Networks, Piscataway, NJ, 1995 1942–1948.

[35] Shi, Y., and Eberhart, R. C, A modified particle swarm optimizer. In Prc. IEEE Congr. Evol. Comput., 1998, 69–73.
[36] Turi, R. Clustering-based colour image segmentation, Ph.D. Thesis, Monash University, Australia, 2001.
[37] Srinivas, M. and Patnaik, L.M. Adaptive probabilities of crossover and mutation in genetic algorithms, *IEEE Transactions on Systems, Man and Cybernetics* 1994, 24 (4) , 656–667.