International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:3, No:3, 2009

Performance Comparison of Particle Swarm Optimization with Traditional Clustering Algorithms used in Self-Organizing Map

Anurag Sharma and Christian W. Omlin

***Abstract*—**Self-organizing map (SOM) is a well known data reduction technique used in data mining. It can reveal structure in data sets through data visualization that is otherwise hard to detect from raw data alone. However, interpretation through visual inspection is prone to errors and can be very tedious. There are several techniques for the automatic detection of clusters of code vectors found by SOM, but they generally do not take into account the distribution of code vectors; this may lead to unsatisfactory clustering and poor definition of cluster boundaries, particularly where the density of data points is low. In this paper, we propose the use of an adaptive heuristic particle swarm optimization (PSO) algorithm for finding cluster boundaries directly from the code vectors obtained from SOM. The application of our method to several standard data sets demonstrates its feasibility. PSO algorithm utilizes a so-called U-matrix of SOM to determine cluster boundaries; the results of this novel automatic method compare very favorably to boundary detection through traditional algorithms namely k-means and hierarchical based approach which are normally used to interpret the output of SOM.

***Keywords*—**cluster boundaries, clustering, code vectors, data mining, particle swarm optimization, self-organizing maps, U-matrix.

I. Introduction

DATA mining simply refers to extracting or "mining" knowledge from large amounts of data. Data mining involves the use of data analysis tools to discover previously unknown, valid patterns and relationships in large data sets. There are number of data mining techniques that can be used for knowledge discovery in a database. For example Classification, Regression, Clustering, Summarization, Dependency Modeling etc [13].

Our main focus of data mining technique in this paper is clustering where no prior knowledge about the dataset is known. Clustering helps users to understand the natural grouping or structure in a data set. Clustering is the process of partitioning a set of data in a set of meaningful groups, called clusters. Since no prior knowledge or predefined classes are known that's why it is called unsupervised classification. Most of the existing clustering algorithms are based on either hierarchical or partitive approach [12]. Both approaches groups the data into classes or clusters so that objects within a cluster have high similarity in comparison to one another, but are very dissimilar to objects in other clusters. Similarity/dissimilarity can be defined as distances between individual data. Normally Euclidean distance is taken but there are other kinds of distances too such as Manhattan or Minkowski distance [14]. However, both of these approaches suffer from inability to produce good cluster in a dataset with complex data arrangements and in high dimensional datasets [6]. To deal with the curse of high dimensionality Self Organizing Map (SOM) was developed by Kohonen [24] which uses vector quantization to reduce the number of vectors and vector projection to reduce the dimensionality to either 2 or 3.

SOM is a tool to cluster the data and deliver them for further visual inspection. That means it also has some shortcomings. User has to either do manual inspection or apply traditional algorithms like hierarchical or partitive algorithms to find the cluster boundaries. Manual inspection is very tedious and not preferred and traditional algorithms might not take into account the distribution of code vectors [1]. SOM tool helps in understanding the data by providing visualization of data distribution. However, good postprocessing for auto interpretation of output of SOM is still not available. We have proposed a novel algorithm based on heuristic approach to find out cluster boundary automatically from output code vectors. This heuristic approach totally depends on the distribution of code vectors. We have implemented our algorithm as an objective function of a generic Particle Swarm Optimization algorithm. The algorithm tries to find the best cluster boundary based on reward-penalty scheme for locating the boundary which gives highest points. Our experiments show its competitiveness with traditional algorithms.

Some heuristic algorithms have already been used for clustering. Many fuzzy clustering are partitioning algorithms which aims to find the best partitions [15]. These fuzzy logics are mainly based on particular type of validity index to find the clusters. Validity index is used to get clusters with best partitioning. Validity index provides a measure of goodness of clustering on different partitions of a data set. The maximum value of this index, called the PBM-index, across the hierarchy provides the best partitioning [17]. [22] has used Genetic Algorithm, [23] has used Simulated Annealing and [19] has

A. Sharma is with the School of Computing, Information and Mathematical Sciences, the University of the South Pacific, Suva, Fiji (e-mail: sharma_au@usp.ac.fj).

C. W. Omlin is with the Department of Computer Engineering, Middle East Technical University, Northern Cyprus Campus, Kalkanli, Guzelyurt, KKTC, Mersin 10, Turkey (e-mail: omlin@metu.edu.tr).

used PSO to get the validity index heuristically. There are several types of validity indices which face problem in a noisy environment [18] and many times visualization are used to validate the clustering results [16].

We have not used any validity index as measurement of cluster quality. Our proposed algorithm has totally different approach which has been described in the next section.

II. Data Clustering

A. Clustering Through Partitioning

Given a database of n objects or data tuples, a partitioning method constructs k partitions of the data, where each partition represents a cluster and $k \leq n$. That is it classifies the data into k groups, which together satisfy the following requirements: (1) each group must contain at least one object, and (2) each object must belong to exactly one group. This method creates an initial partitioning. It then uses an iterative relocation technique that attempts to improve the partitioning by moving objects from one group to another. The general criterion of a good partitioning is that objects in the same cluster are "close" or related to each other, whereas objects of different clusters are "far apart" or very different. Some of the popular algorithms are k-means and k-medoids [14]. We have used k-means algorithm in this paper.

B. Hierarchical Clustering

A hierarchical method organizes data in a nested sequence of groups, which can be displayed in the form of a dendrogram or a tree structure. It divides the data into hierarchical structure such that data items in the same hierarchical level have similarity and groups in different hierarchy levels have dissimilarity. Similarity and dissimilarity can be measured by the branch height from one level to another. The greater the heights of a branch from one level to another, the greater the dissimilarity. Hence, we assign most dissimilar data items to different clusters and similar data item to a group that constitutes a cluster. A hierarchical method can be classified as being either agglomerative or divisive, based on how the hierarchical decomposition is formed. The agglomerative approach, also called the bottom-up approach, starts with each object forming a separate group. It successively merges the objects or groups close to one another until all of the groups are merged into one (the topmost level of the hierarchy), or until a termination condition holds. The divisive approach, also called the top-down approach, starts with all the objects in the same cluster. In each successive iteration, a cluster is split up into smaller clusters, until each object is in one cluster, or until a termination condition holds [14].

C. Self-Organizing Maps

The self-organizing map (SOM), developed by Kohonen [24] is a neural network model for the analysis and visualization of high dimensional data. The nonlinear statistical relationships between high-dimensional data are projected into simple topologies such as two-dimensional grids. The SOM thus reduces information while preserving the most important topological relationships of the data elements on the two-dimensional plane. SOMs are trained using unsupervised learning, i.e. no prior knowledge is available and no assumptions are made about the class membership of data.

Several techniques have been proposed for clustering the outputs of SOMs, i.e. their code vectors. Typically, clustering methods based on partitioning or hierarchical methods are applied to cluster the SOM code vectors; however, the solutions found normally do not reflect the clustering suggested by visual inspection of code vectors [12]. These methods are very sensitive to noise and outliers. Moreover, they are not suitable to discover clusters with *non-convex shapes* [6]. The difficulty in detection of cluster boundaries has resulted in problems of interpretability of trained SOMs which has limited its application to automatic knowledge discovery [1].

Data clustering with SOMs is typically carried out in two stages: first, the data set is clustered using the SOMs which provide code vector, i.e. prototype data points; then, the code vectors are clustered [7]. The use of code vectors in place of the raw data leads to significant gains in the speed of clustering. Our proposed method identifies cluster boundaries using particle swarm optimization on the distribution of code vectors. We have evaluated the performance and reliability of particle swarm optimization algorithm with other traditional algorithms that are used to automatically cluster the output of SOM.

The SOM training algorithm involves essentially two processes, namely vector quantization and vector projection [4]. Vector quantization creates a representative set of vectors, so-called output vectors or prototype vectors from input vectors. The second process, vector projection, projects output vectors onto a SOM of lower dimension (mainly 2 or 3); this can be useful for data visualization.

The basic SOM model is a set of prototype vectors with a defined neighborhood relation. This neighborhood relation defines a structured lattice, which may be linear, rectangular or hexagonal arrangement of map units. SOMs are trained in unsupervised, competitive learning process. This process is initiated when a winner unit is searched from map units, which minimizes the Euclidean distance measure between data samples x and the map units m_i. This unit is described as the best matching node, the code vector m_c:

$$\|x - m_c\| = \min\{\|x - m_i\|\} \quad (1)$$
$$where,\ c = \arg\min\{\|x - m_i\|\}$$

Then, the map units are updated in the topological neighborhood of the winner unit, which is defined in terms of the lattice structure. The update step can be performed by applying

$$m_i(t+1) = m_i(t) + h_{ci}(t)[x(t) - m_i(t)] \quad (2)$$

where t is an integer, the discrete-time coordinate and $h_{ci}(t)$ is the so-called neighborhood kernel defined over the lattice

points. The average width and form of h_{ci} defines the "stiffness" of the "elastic surface" to be fitted to the data points. The last term in the square brackets is proportional to the gradient of the squared Euclidean distance $d(x,m_i) = ||x-m_i||^2$. The learning rate $\alpha(t) \in [0,1]$ must be a decreasing function over time and the neighborhood function $h^c(t,i)$ is a non-increasing function around the winner unit defined in the topological lattice of map units. A typical choice is a Gaussian around the winner unit defined in terms of the coordinates **r** in the lattice of neurons [21], [9].

$$h^c(t,i) = \alpha(t).\exp\left(-\frac{\left\|r^i - r^c\right\|}{2\sigma(t)^2}\right) \quad (3)$$

Training is an iterative process which chooses a winner unit by the means of a similarity measure and updates the values of code vectors in the neighborhood of the winner unit. This iterative process is repeated until convergence is reached. All the output vectors are projected on to a 1 or 2 dimensional space, where each neuron corresponds to an output vector that is the representative of some input vectors [4].

SOMs will represent areas of high data densities with many map units whereas only few code vectors will represent sparsely populated areas. Thus, SOMs approximate the probability density function of the input data [9].

Self-organizing maps may be visualized by using a unified distance matrix (U-matrix) representation, which visualizes the clustering of the SOM by calculating distances between map units. An alternative choice for visualization is Sammon's mapping which projects high-dimensional map units onto a plane by minimizing the global distortion of inter point distances.

D. Unified Distance Matrices

A unified distance matrix (U-matrix) representation of SOMs visualizes the cluster structure by calculating the distances between map unit and mean/median distances from its neighbors. U-matrix representation of a sample data is shown in Fig. 1.

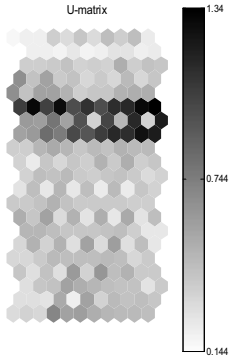

Fig. 1 U-matrix representation of SOM

The distance ranges are represented by different colors (or grey shades). Dark shades represent large distance, i.e. big gaps exist between the code vector values in the input space; light shades represent small distance, i.e. map units are tightly clustered together. U-matrices are useful tools for visualizing clusters in input data without having any priori information about the clusters [9].

III. Particle Swarm Optimization

A. Heuristic Algorithm

The Particle swarm Optimization (PSO) algorithm was originally designed to solve continuous and discrete problems of large domain [11]. It is a heuristic algorithm to solve optimizing engineering problems. PSO is an adaptive search algorithm which is inspired by the social behavior of bird flocking or fish schooling.

The analogy involves simulating social behavior among individuals (particles) "flying" through a multidimensional search space, each particle representing a single intersection of all search dimensions. The particles evaluate their positions relative to a goal (fitness) at every iteration, and particles in a local neighborhood share memories of their "best" positions; they use those memories to adjust their own velocities, and thus subsequent positions [2].

The original PSO formulae define each particle as potential solution to a problem in D-dimensional space. The position of particle i is represented as

$$X_i = (x_{i1}, x_{i2}, \ldots, x_{iD}) \quad (4)$$

Each particle also maintains a memory of its previous best position, represented as

$$P_i = (p_{i1}, p_{i2}, \ldots, p_{iD}) \quad (5)$$

A particle in a swarm is moving; hence, it has a velocity, which can be represented as

$$V_i = (v_{i1}, v_{i2}, \ldots, v_{iD}) \quad (6)$$

At each iteration, the velocity of each particle is adjusted so that it can move towards the neighborhood's best position known as *lbest* (P_i) and global best position known as *gbest* (P_g) attained by any particle present in the swarm [2].

After finding the two best values, each particle updates its velocity and positions according to (7) and (8) weighted by a random number c_1 and c_2 whose upper limit is a constant parameter of the system, usually set to value of 2.0 [11].

$$V_i(t+1) = V_i(t) + c_1 \cdot (P_i - X(t)) + c_2 \cdot (P_g - X_i(t)) \quad (7)$$

$$X_i(t+1) = X_i(t) + V_i(t+1) \quad (8)$$

All swarm particles tend to move towards better positions; hence, the best position (i.e. optimum solution) can eventually be obtained through the combined effort of the whole population.

B. Clustering with Particle Swarm Optimization

A heuristic approach is generally used to solve optimization problems. The proposed clustering algorithm is a problem of optimization by nature. In general, a cluster can have several cluster boundaries as shown in Fig. 2. It depends on how a particular clustering algorithm finds the best one. A generic

algorithm can be applied here to find the most suitable cluster boundary among all possible cluster boundaries. Particle Swarm Optimization which determines the optimum solution heuristically, could be applied in this kind of clustering approach provided the availability of an appropriate objective function to determine the optimum cluster boundary.

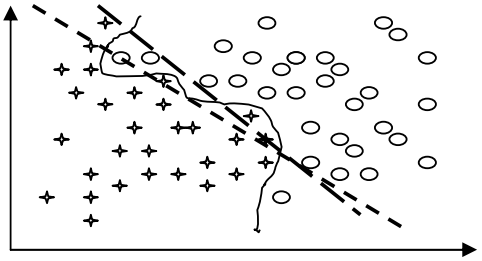

Fig. 2 Data Clustering: Two different clusters - one with star shaped data and one with circle shaped data – can be separated by straight lines or by a curved lines.

We have designed objective function for clustering with particle swarm optimization that works on the entries of U-matrix to locate the best cluster boundary. U-matrix is simply matrix of distances between code vectors where lower values indicate code vectors are closer and higher values indicate code vectors are far apart. Logically, clusters are formed where data density is relatively higher from near by region so we used the values from matrix to find the boundary with *maximum average* value so that optimum cluster boundary lies where density of code vectors starts decreasing. In some data sets we found that data that are far apart make the cluster of their own. To cater for this type of data distribution our method automatically determines if the region near cluster center has lower density then we do the opposite i.e. find the boundary with *minimum average* value so that optimum cluster boundary is where density of code vectors starts increasing i.e. relative lower values in U-matrix. We have named this proposed novel algorithm *Expanding Cluster Boundary Algorithm* which has been described below:

In reference [3] *Shrinking Cluster Boundary* approach has been used which is entailed with few shortcoming that has been tried to overcome in this new approach of *Expanding Cluster Boundary* algorithm. *Shrinking Cluster Boundary* does not perform well if a proper initial cluster boundary is not specified manually whereby the *Expanding Cluster Boundary* algorithm is insensitive to initializations because it does not require the initial cluster boundary to be specified.

This algorithm starts with automatic assignment of a unit square cluster boundary outside manually chosen cluster center anywhere in the region of the expected cluster using graphical U-matrix which grows until the optimum cluster boundary has found or the edge of the u-matrix has reached. There are some other existing methods too, to determine cluster centers automatically described in [20] and [5] that utilizes K-means algorithm with minimal user interference. The clustering algorithm proceeds as follows:

1) Initialization: After specifying the cluster center, PSO automatically initializes a unit principal cluster boundary i.e. closest square boundary outside cluster center along with a set of neighboring boundaries – inner and outer boundaries. Additional boundaries help in better acquisition of dispersion of data near the boundary. The cluster centers are approximated as middle point of denser regions through visual inspection of U-matrix. Now for each iteration of PSO:

2) Adjust ranks: PSO generates particles' positions (sequence/array of numbers) and changes their positions by picking 2 indices and swapping them. We call these indices ranks. These sequences have constant size; they are utilized as cluster boundary whose size changes iteratively. We simply arrange the ranks corresponding to the size of expanded cluster boundary i.e. mathematically:

*new_rank = ceil(rank*SOM_boundary_size/PSO_sequence_size)*

3) Expand cluster boundary: Expand the cluster boundary in the U-matrix gradually away from cluster center by picking both ranks (indices) and moving them one row and/or column outwards.

4) Calculate average weight of cluster boundaries (fitness function): Since the initial principal boundary is extracted from U-matrix, each weight unit is the distance between clusters or neighborhood distance. Additionally, some neighboring boundaries are calculated; this improves the search for cluster boundaries. If a matrix entry is promising to be good cluster boundary then we change the weight to make it closer to the maximum value of the matrix according to its quality and if the matrix entry is not promising then we change the weight towards the minimum value of the matrix. The higher the average weight the better the cluster boundary. For example if matrix entry has smaller weight *unit_weigh*t near the boundary then it is promising and we change it to:

unit_weight = max_val – unit_weight

where, *max_val* is the maximum value of the U-matrix and *unit_weight* is the weight of a single entry of U-matrix which is part of a cluster boundary.

The fitness function is simply the average weight of cluster boundaries. The objective function of clustering algorithm uses these *values of U-matrix* to find maximum average weight through PSO. If the weight unit of an outer boundary, i.e. a boundary that is further away from the corresponding cluster center than the principal boundary, is higher than the weight unit of the principal boundary, then we add to the weight since it is likely to be a cluster boundary. If weight units for the principal boundary is lower, then again we increase the corresponding weights. In some cases, we can also decrease weights; for example, if the weights for some outer boundary are lower than the corresponding values for the principle boundary, then the outer boundary is in fact not a boundary; instead, it belongs to the interior of another cluster. Similarly, inner boundaries (interior part of cluster) whose weights are higher than the corresponding values of the principal boundary in fact do not belong to the interior of the cluster; rather it is

part of the cluster boundary. Hence the fitness function is simply the average weight of the cluster boundaries which is depicted with mathematical equations below:

$$wt_i = \begin{cases} 2(max\_val - wt_i), \text{principal layer is promising} \\ max\_val + wt_i, \text{outer layer is promising} \\ -2max\_val, \text{penalty} \end{cases} \quad (9)$$

where, wt_i is the weight of i_{th} entry of U-matrix that is a part of cluster boundaries which can change from its original value to new value depend on the quality of its position and weight.

We need to calculate the weights of all entries of U-matrix that belong to cluster boundaries. Eq. 11 shows the calculation of average weight of all cluster boundaries after getting the total weight from Eq. 10.

$$wt\_total = \sum_{i=1}^{Psize} Pboundary_i + \sum_{i=1}^{Isize} Iboundary_i + \sum_{i=1}^{Osize} Oboundary_i \quad (10)$$

$$wt\_avg = wt\_total / (Psize + Isize + Osize) \quad (11)$$

where, $Pboundary_i$ is the weight (wt_i) of principal boundary of size *Psize* at i_{th} position,

$Iboundary_i$ is the weight (wt_i) of inner boundary of size *Isize* at i_{th} position,

$Oboundary_i$ is the weight (wt_i) of outer boundary of size *Osize* at i_{th} position and

wt_total is the total weight of cluster boundaries and wt_avg is the average weight of cluster boundaries.

Each particle of a swarm consists of the value of this fitness function which is simply the average weight of cluster boundaries.

5) Solution: Repeated application steps 2 to 4 results in a maximum average weight, which corresponds to the best cluster boundary.

The best boundary is the one which has highest average value of distances between points and their neighbors.

IV. Experiments

In this paper, clustering of data using novel algorithm is done in three steps:

i. Cluster the data with SOM which produces prototype code vectors.
ii. Use U-matrices to visualize output code vectors from SOM to approximate cluster center.
iii. Use PSO on the output of SOM i.e. U-matrix to automatically define cluster boundaries.

Our Experiment is comparison of efficiency and reliability of proposed novel PSO algorithm with traditional algorithms. These algorithms are *hierarchical* and *partitive* based algorithms which are currently used to interpret the output of SOM [8]. *k-means* algorithm is one of the most popular *partitive* algorithms. Our experiment is based on some standard data sets for machine learning. We have calculated accuracy and misclassification of each cluster. Accuracy is percentage of instances identified correctly and misclassification is percentage of instances either identified incorrectly or not identified at all on a given cluster. In our experiments we used symbols *C1, C2, ... Cn* to indicate different clusters. We made a 2-dimensional matrix *M* to show cluster *accuracy* and *misclassification*. For example Table 1 shows quality measurement of 2 clusters C1 and C2 shown in row and column headers. First 2 columns indicate expected cluster and first 2 rows indicate assigned cluster. For example row 1 and column 1 indicate data points expected in cluster C1 i.e. $M_{1,1}$ indicates data points correctly identified as C1. So this matrix entry indicates accuracy of algorithm in finding cluster C1. On the other hand, row 1 and column 2 indicates data points expected to be cluster C2 but cluster C1 is assigned so it indicates misclassifications.

Experiments with different datasets are as follows:

A. IRIS Dataset

Iris data is very well known database for machine learning. The dataset comprised of properties of 3 types of iris flowers namely Setosa, Versicolour and Virginica. Each type of flower has 50 samples hence 150 in total. Each sample consists of length and width of sepal and petal leaves. Fig. 3 shows output of SOM graphically. Fig. 3 (a) is a U-matrix diagram with 2 clusters indicated as C1 and C2, (b) is a labeled code vectors arranged in 11x6 hexagonal topology.

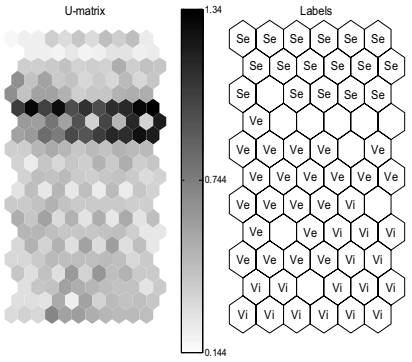

Fig. 3 (a) Graphical view of U-matrix of iris data indicates 2 clusters on top and bottom region. The distances between code vectors are represented by grey shades. The shaded scale in the middle shows the distance measurement. The darker the shades, the greater the distance between the code vectors. Hence darker region indicates cluster boundary and lighter region indicates cluster itself. (b) Prototype vectors of corresponding U-matrix with labels Se, Ve and Vi which represent Setosa, Versicolor and Virginica, respectivel

1) Hierarchical clustering

The Fig. 4 shows hierarchical cluster of 66 code vectors of iris data where 2 major clusters are obvious. The measurement of cluster quality is shown in Table 1.

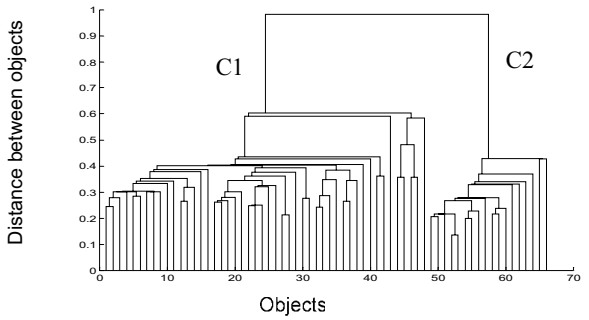

Fig. 4 Hierarchical clustering of 66 code vectors of iris data set. x-axis indicates objects itself and y-axis indicates distance between objects. C1 and C2 represent branches of two different clusters.

TABLE I
CLUSTER QUALITY MEASUREMENT OF IRIS DATASET WITH HIERARCHICAL ALGORITHM

	C1	C2	Total code vectors	Cluster size	Accu-racy	Misclas-sified
C1	17	0	17	17	100%	0%
C2	0	39	39	39	100%	0%
Average					100%	0%

2) Parititive Clustering (k-means) algorithm:

Fig. 5 (a) shows output of k-means algorithm where 2 clearly separated clusters identified by k-means algorithm are shown. The measurement of cluster quality is shown in Table 2.

TABLE II CLUSTER QUALITY MEASURMENT OF IRIS DATASET WITH K-MEANS ALGORITHM

	C1	C2	Total code vectors	Cluster size	Accu-racy	Misclas-sified
C1	17	0	17	17	100%	0%
C2	1	38	39	39	97.4%	2.6%
Average					98.7%	1.3%

3) Particle Swarm Optimization:

Fig. 5 (b) shows labels of code vectors identified by PSO algorithm. 1 and 2 indicates cluster C1 and C2 respectively. 0 indicates code vector has no label. The measurement of cluster quality is shown in Table 3.

It can be observed that all 3 algorithms reliability performance is almost same but hierarchical algorithm has shown the best results.

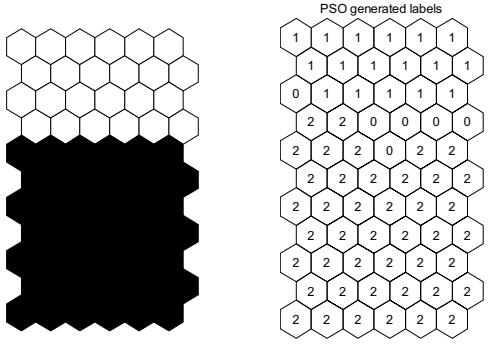

Fig. 5 (a) Cluster arrangement generated through k-means algorithm. (b) Labels of code vector found by PSO

TABLE III CLUSTER QUALITY MEASURMENT OF IRIS DATASET WITH PSO ALGORITHM

	C1	C2	Total code vectors	Cluster size	Accu-racy	Misclas-sified
C1	16	0	17	16	94.1%	5..9%
C2	0	38	39	38	97.4%	2.6%
Average					95.8	4.2%

B. Wine recognition dataset

These data are the results of a chemical analysis of wines grown in the region of Italy but derived from three different cultivars. The analysis determines the quantities of 13 constituents found in each of the three types of wines. It has total of 78 instances with 3 classes.

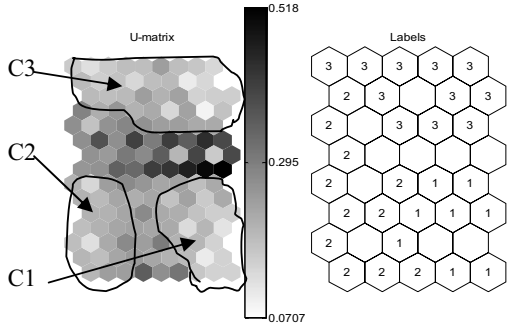

Fig. 6 (a) Graphical view of U-matrix of wine data indicates 3 clusters in top and bottom region. The distances between code vectors are represented by grey shades. The shaded scale in the middle shows the distance measurement. (b) Prototype vectors of corresponding U-matrix with labels.

Fig. 6 shows output of SOM graphically. (a) is a U-matrix diagram with 3 clusters indicated as C1, C2 and C3. (b) is a labeled code vectors arranged in 8x5 hexagonal topology. Results with different algorithms are shown below:

1) Hierarchical clustering

Fig. 7 shows dendrogram of 40 code vectors of wine data. By looking at the height of parent branches 2 clusters C1 and

C3 are clear but cluster C2 is not that clear. The measurement of cluster quality is shown in Table 4.

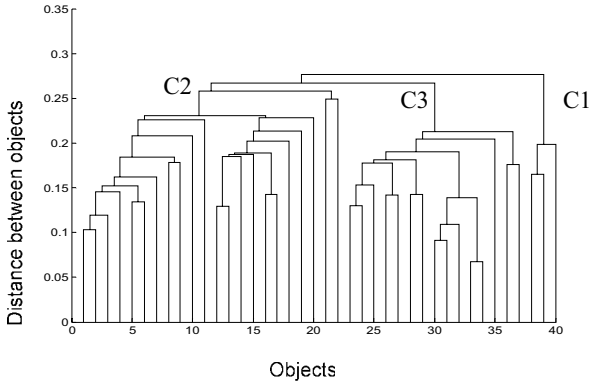

Fig. 7 dendrogram of wine data. Clusters can be distinguished with separate branches indicated by C1, C2 and C3.

TABLE IV CLUSTER QUALITY MEASURMENT OF WINE DATASET WITH HIERARCHICAL ALGORITHM

	C1	C2	C3	Total code vectors	Cluster size	Accu-racy	Misclas-sified
C1	8	0	0	8	22	36.4%	63.6%
C2	9	0	2	11	3	0%	100%
C3	0	0	11	11	15	73.3%	26.7%
Average						36.6%	63.4%

2) Paritive Clustering (k-means) algorithm:

Fig. 8 (a) shows 3 very clear clusters derived from k-means algorithm that matches with actual clustering arrangement shown in Fig. 6. The measurement of cluster quality is shown in Table 5.

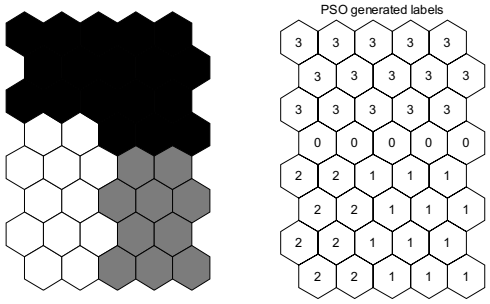

Fig. 8 (a) Cluster arrangement generated through k-means algorithm. (b) Labels of code vector found by PSO.

3) Particle Swarm Optimization:

Fig. 8 (b) shows labels of code vectors identified by PSO algorithm. 1, 2 and 3 indicates cluster C1, C2 and C3 respectively. 0 indicates code vector has no label. The measurement of cluster quality is shown in Table 6.

TABLE V CLUSTER QUALITY MEASURMENT OF WINE DATASET WITH K-MEANS ALGORITHM

	C1	C2	C3	Total code vectors	Cluster size	Accu-racy	Misclas-sified
C1	7	1	0	8	10	70%	30%
C2	1	8	2	11	12	66.7%	33.3%
C3	0	0	11	11	18	61.1%	38.9%
Average						65.9%	34.1%

TABLE VI CLUSTER QUALITY MEASURMENT OF WINE DATASET WITH PSO ALGORITHM

	C1	C2	C3	Total code vectors	Cluster size	Accu-racy	Misclas-sified
C1	8	0	0	8	12	66.7%	33.3%
C2	2	6	2	11	8	54.6	36.4%
C3	0	0	11	11	15	73.3%	26.7%
Average						64.9%	35.1%

In this experiment hierarchical algorithm has performed very poorly. PSO and k-means algorithms' reliability performance is quite similar but k-means has shown best results for this dataset.

C. Thyroid gland data set ('normal', hypo and hyper functioning)

This dataset consist of total of 215 data instances of thyroid gland experimental data values with three clusters namely, euthyroidism, hypothyroidism or hyperthyroidism. Fig. 9 shows output of SOM graphically. (a) is a U-matrix diagram with three clusters indicated as C1, C2 and C3. (b) is a labeled code vectors arranged in 12x6 hexagonal topology.

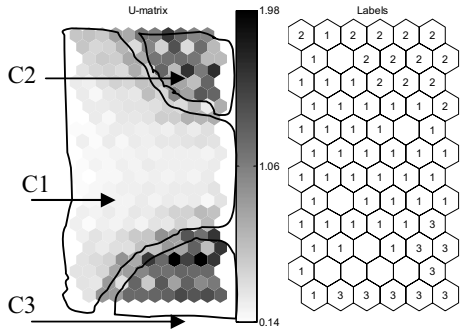

Fig. 9 (a) Graphical view of U-matrix of thyroid data where clusters are identified manually. (b) u-matrix's corresponding code vectors with labels shown as 1, 2 and 3. They represent clusters C1, C2 and C3 respectively.

1) Hierarchy clustering:

Fig. 10 shows dendrogram hierarchical view of code vectors for thyroid data. Clusters are not conspicuous through

dendrogram, hence the results are very poor. Table 7 shows the measurement of cluster quality.

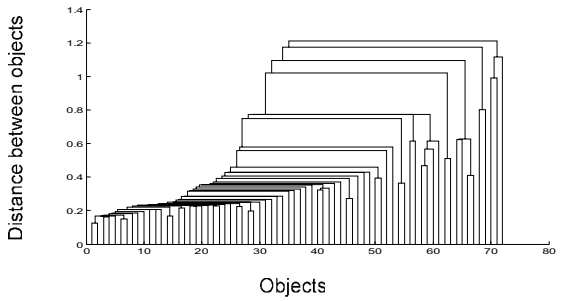

Fig. 10 dendrogram of thyroid data set.

TABLE VII CLUSTER QUALITY MEASURMENT OF THYROID DATASET WITH HIERARCHICAL ALGORITHM

	C1	C2	C3	Total code vectors	Cluster size	Accu-racy	Misclas-sified
C1	42	0	0	42	67	62.7	37.3
C2	13	0	0	13	2	0%	100%
C3	5	1	3	9	3	33.3%	66.7%
Average						32%	68%

2) Paritive Clustering (k-means) algorithm:

Fig 9 (a) shows three clusters identified by k-means algorithm closely matches with actual clustering produced though SOM as shown in Fig. 9. The measurement of cluster quality is shown in Table 8.

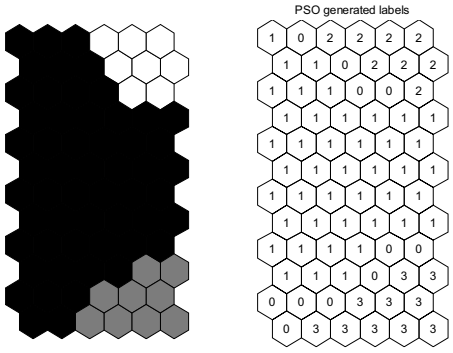

Fig. 11 (a) Clusters arrangement generated through k-means algorithm. (b) Labels of code vector found by PSO

3) Particle Swarm Optimization:

Fig. 11 (b) shows labels of code vectors identified by PSO algorithm. 1, 2 and 3 indicates cluster C1, C2 and C3 respectively. 0 indicates code vector has no label. The measurement of cluster quality is shown in Table 9.

TABLE VIII CLUSTER QUALITY MEASURMENT OF THYROID DATASET WITH K-MEANS ALGORITHM

	C1	C2	C3	Total code vectors	Cluster size	Accu-racy	Misclas-sified
C1	42	0	0	42	55	76.4%	0%
C2	5	8	0	13	8	61.5%	38.5%
C3	2	0	7	9	9	77.8%	22.2%
Average						71.9%	28.1%

TABLE IX CLUSTER QUALITY MEASURMENT OF THYROID DATASET WITH PSO ALGORITHM

	C1	C2	C3	Total code vectors	Cluster size	Accu-racy	Misclas-sified
C1	39	0	2	42	50	92.9%	4.8%
C2	5	11	0	13	11	84.6%	38.5%
C3	1	0	9	9	14	64.3%	11.1%
Average						80.6%	19.4%

To summarize the results shown by all three algorithms in terms of reliability it can be said that PSO has shown best results and K-means has also shown quite good results. Hierarchical algorithm has again showed poor results.

D. Image Segmentation dataset:

There are 210 instances of which 30 belong to cluster C1, 30 belong to cluster C2 and 150 belong to cluster C3. SOM has generated 72 codebook vectors in which C1 has 7, C2 has 4 and C3 has 40 samples. Fig. 12 shows output of SOM graphically. (a) is a U-matrix diagram with three clusters indicated as C1, C2 and C3. (b) is a labeled code vectors arranged in 12x6 hexagonal topology where 7 different types (classes) of images namely brickface (BF), sky, foliage (FL), cement (CT), window (WN), path and grass (GRS) are shown.

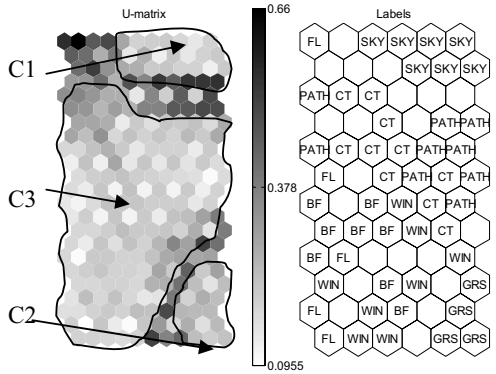

Fig. 12 (a) Graphical view of U-matrix of wine data indicates 3 clusters in top and bottom region. The distances between code vectors are represented by grey shades. The shaded scale in the middle shows the distance measurement. (b) Prototype vectors of corresponding U-matrix with labels of image types.

1) Hierarchy clustering:

Hierarchical cluster of image segment is shown in Fig. 13. The dendrogram can only identify one cluster properly that's

why the performance is very poor. Table 10 shows measurement of cluster quality.

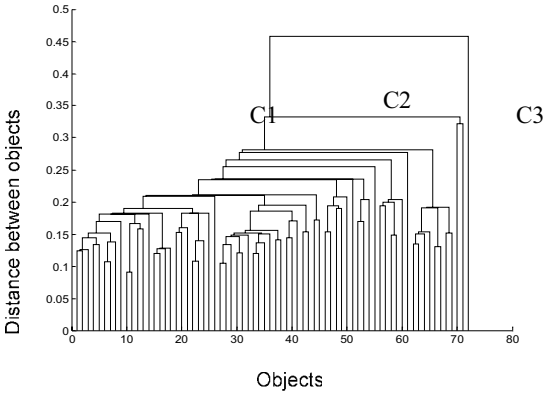

Fig. 13 Dendrogram of thyroid data set.

TABLE X CLUSTER QUALITY MEASUREMENT OF IMAGE DATASET WITH HIERARCHICAL ALGORITHM

	C1	C2	C3	Total code vectors	Cluster size	Accu-racy	Misclas-sified
C1	0	0	7	7	2	0%	100%
C2	0	0	4	4	1	0%	100%
C3	1	1	38	40	69	95%	5%
Average						31.7%	68.3%

2) Paritive Clustering (k-means) algorithm:

Fig. 14 (a) shows three clusters identified by k-means algorithm. Only cluster with label "SKY" has been identified correctly. The measurement of cluster quality is shown in Table 11.

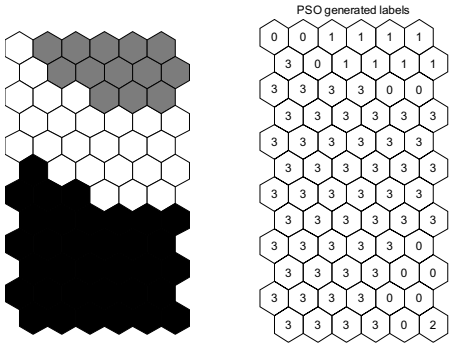

Fig. 14 (a) Cluster arrangement generated through k-means algorithm. (b) Labels of code vector found by PSO.

3) Particle Swarm Optimization:

Fig. 14 (b) shows labels of code vectors identified by PSO algorithm. 1, 2 and 3 indicates cluster C1, C2 and C3 respectively. 0 indicates code vector has no label. The measurement of cluster quality is shown in Table 12.

TABLE XI CLUSTER QUALITY MEASUREMENT OF IMAGE DATASET WITH K-MEANS ALGORITHM

	C1	C2	C3	Total code vectors	Cluster size	Accu-racy	Misclas-sified
C1	7	0	0	7	13	53.9%	56.1%
C2	0	4	0	4	34	11.8%	88.2%
C3	0	20	20	40	25	50%	50%
Average						38.6%	61.4%

TABLE XII CLUSTER QUALITY MEASUREMENT OF IMAGE DATASET WITH PSO ALGORITHM

	C1	C2	C3	Total code vectors	Cluster size	Accu-racy	Misclas-sified
C1	7	0	0	7	8	87.5%	12.5%
C2	0	4	4	4	7	57.1%	42.9%
C3	0	1	39	40	59	66.1%	33.9%
Average						70.2%	29.8%

Again PSO has shown best result in this data set.

E. Glass dataset

The study of classification of types of glass was motivated by criminological investigation. There are total of 214 instances of glass attributes and 7 different classes which are

building_windows_float_processed (label 1),
building_windows_non_float_processed (label 2),
vehicle_windows_float_processed (label 3),
vehicle_windows_non_float_processed (none in this database) (label 4), containers (label 5), tableware (label 6), headlamps (label 7), however SOM has identified just three classes separately.

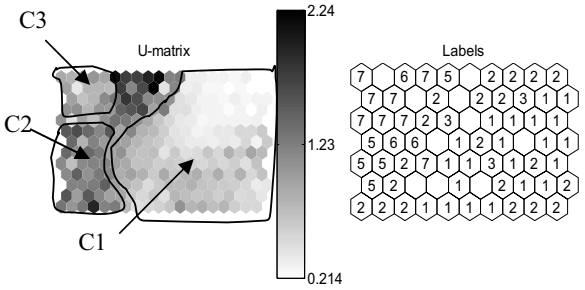

Fig. 15 (a) Graphical view of U-matrix of Glass data indicates 3 clusters as C1, C2 and C3. The distances between code vectors are represented by grey shades. The shaded scale in the middle shows the distance measurement. (b) Prototype vectors of corresponding U-matrix with labels from 1-7 that indicate different types of glasses.

1) Hierarchy clustering:

Hierarchical cluster of glass dataset is shown in Fig. 16. It is unable to define all three clusters clearly. Fig 14 shows just 2 clusters where all the groups except 70 make one cluster and group 70 makes another cluster of itself. Table 13 shows measurement of cluster quality.

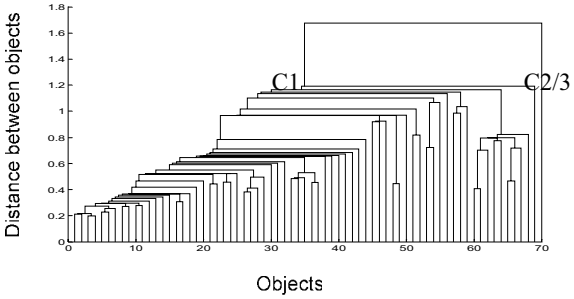

Fig. 16 hierarchical cluster of Glass dataset using dendrogram.

TABLE XIII CLUSTER QUALITY MEASUREMENT OF GLASS DATASET WITH HIERARCHICAL ALGORITHM

	C1	C2	C3	Total code vectors	Cluster size	Accu-racy	Misclas-sified
C1	41	0	0	41	68	60.3%	39.7%
C2	7	1	0	8	1	12.5%	87.5%
C3	7	0	1	8	1	12.5%	87.5%
Average						28.4%	71.6%

Cluster quality for C1 is good but C2 and C3 are very poor.

2) Paritive Clustering (k-means) algorithm

K-means algorithm is somewhat better than hierarchical cluster. Fig 15 (a) shows three clusters that match with actual clustering shown in Fig. 15. The measurement of cluster quality is shown in Table 14.

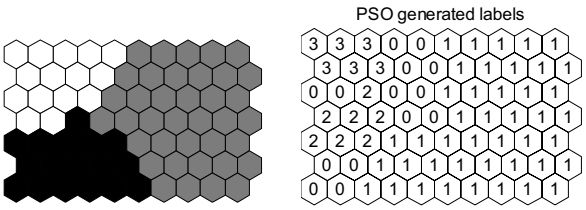

Fig. 17 (a) Cluster arrangement generated through k-means algorithm. (b) Labels of code vector found by PSO.

TABLE XIV CLUSTER QUALITY MEASUREMENT OF GLASS DATASET WITH K-MEANS ALGORITHM

	C1	C2	C3	Total code vectors	Cluster size	Accu-racy	Misclas-sified
C1	29	2	10	41	38	76.3%	23.7%
C2	0	7	1	8	15	46.7	53.3%
C3	0	4	4	8	17	23.5%	76.5%
Average						48.8%	51.2%

3) Particle Swarm Optimization:

Fig. 17 (b) shows labels of code vectors identified by PSO algorithm. 1, 2 and 3 indicates cluster C1, C2 and C3 respectively. 0 indicates code vector has no label. The measurement of cluster quality is shown in Table 15.

TABLE XV CLUSTER QUALITY MEASUREMENT OF GLASS DATASET WITH PSO ALGORITHM

	C1	C2	C3	Total code vectors	Cluster size	Accu-racy	Misclas-sified
C1	34	0	1	41	43	79.1%	20.9%
C2	1	5	3	8	8	62.5	37.5%
C3	0	1	5	8	9	55.6%	44.4%
Average						66.7	77.2%

PSO has again showed best results compared to other algorithms. K-means has also shown good results but hierarchical algorithm is not competitive enough where only one cluster has been identified.

V. ANALYSIS OF EXPERIMENTS

Fig. 18 - Fig. 21 compare the performances of PSO, k-means and hierarchical algorithms on five different datasets based on their ability/inability to correctly identify individual samples within their clusters. Please note that we have changed the notations of clusters from C1, C2, etc to their dataset names. For example C1 and C2 of iris data set have been changed to Iris1 and Iris2 to avoid ambiguity with other data sets' clusters' names. Fig. 18 shows the graph of reliability and accuracy of algorithms on individual clusters i.e. number of samples correctly identified within their clusters. x-axis shows names of different clusters within the datasets and y-axis indicates percentage accuracy. The higher the percentage, the better the performance of algorithm. Fig. 19 shows the graph of misclassification of algorithms on individual clusters .i.e. number of samples incorrectly identified within their clusters. x-axis shows names of different clusters within the datasets and y-axis indicates percentage misclassification. Here the lower the percentage, the better the algorithm as opposed to Fig. 18. Fig. 20 shows average percentage accuracy i.e. average of accuracies of all the clusters within data sets. Hence, we can conclude how well algorithms have performed on different data sets. The higher the percentage, the better the algorithm. Similarly Fig. 21 shows average percentage misclassification of all the clusters within given data sets. Here the lower the percentage, the better the results.

It is obvious from all the graphs shown in figures Fig. 18 that PSO is a very competitive algorithm and has showed superior results in general. The hierarchical algorithm generally shows poor results while the K-means algorithm has shown good results but it is not as consistent as PSO. It has performed very poorly for image dataset which indicates that k-means algorithm does not always follow the distribution of code vectors of SOM as mentioned by [1]. Another problem

with k-means algorithm is that it also includes all the outliers as part of a cluster whereas PSO normally does not include outliers as part of a cluster. Thus we conclude that PSO produces consistent results unlike other algorithms whose performance is highly dependent on the types of dataset. For some they show good results while for others they have poor results.

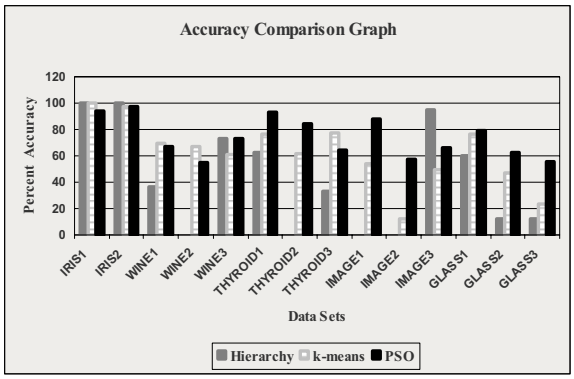

Fig. 18 Summary of Performance of Clustering Algorithms: Comparisons of three algorithms namely, hierarchical, k-means and particle swarm optimization on the basis of percentage of reliability and accuracy, i.e. number of samples correctly identified within their clusters. The higher the percentage, the better the algorithm.

We have presented new approach for clustering the data where we treat clustering as an optimization problem. One cluster can have several possible cluster boundaries. It depends on clustering algorithm how to find the best one. We have applied clustering as objective function to Particle Swarm Optimization algorithm which is an adaptive search heuristic algorithm

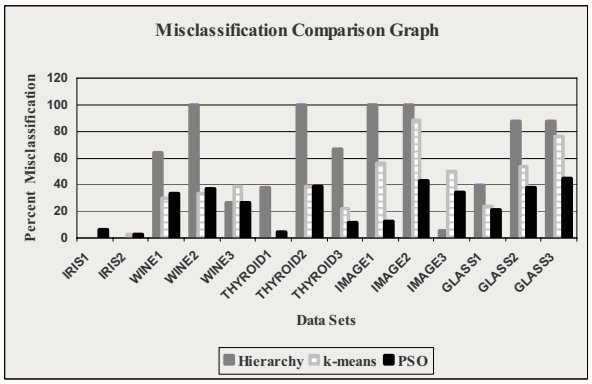

Fig. 19 Misclassification by Cluster Algorithms: Comparisons of three algorithms namely, hierarchical, k-means and particle swarm optimization on the basis of percentage of misclassification i.e. number of samples not identified/incorrectly identified within their clusters. x-axis indicates different clusters within the datasets and y-axis indicates percentage misclassification. The lower the percentage, the better the algorithm.

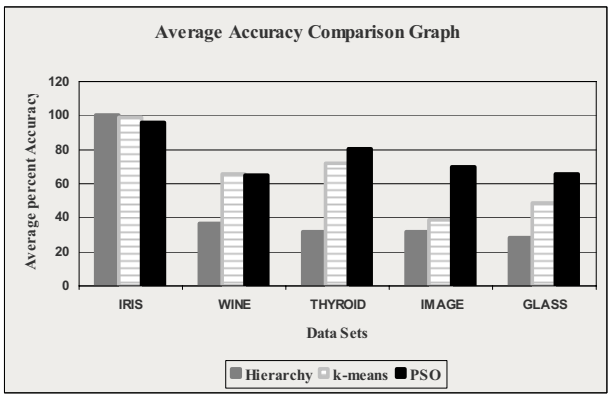

Fig. 20 Average Performance of Clustering Algorithms: Comparison of hierarchical, partitive (k-means) and particle swarm optimization algorithms on the basis of average percentage accuracy, i.e. average of accuracies of all the clusters within data sets. The higher the percentage, the better the algorithm.

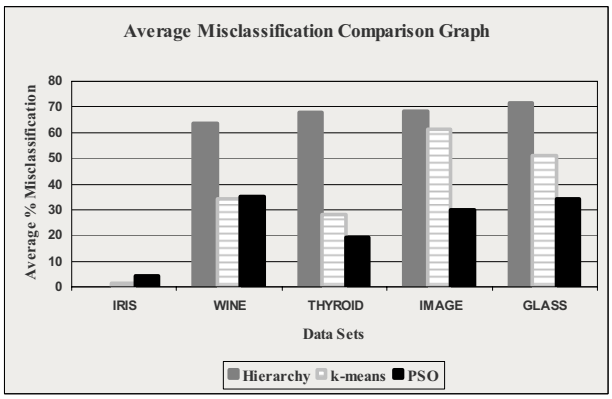

Fig. 21 Average Misclassification by Clustering Algorithms: Comparison of hierarchical, partitive (k-means) and particle swarm optimization algorithms on the basis of average percentage misclassification i.e. average of misclassifications of all the clusters within data sets. The lower the percentage, the better the algorithm

VI. Conclusion

We have proposed a solution for the automatic identification of cluster boundaries for SOMs' code vectors through particle swarm optimization (PSO). The proposed algorithm has been compared with well established k-means algorithm and hierarchical algorithm. Our results show the competitiveness of our novel algorithm which we have tested on five different sets of standard benchmark data. There are some known discrepancies for the k-means algorithm which our clustering algorithm overcomes. The k-means algorithm works well when the number of clusters is small, but its performance may decrease as the number of clusters increase [1]. Since PSO works individually with a particular cluster, hence it is insensitive to the number of clusters in the data set. However, when cluster size is very big with fuzzy boundaries then it may not give very good solution as we have seen in the *image* data set. The k-means algorithm is also very sensitive to noise and outlier data points because a small number of such

data can substantially influence the mean value [9]. It also includes all the outliers as part of a cluster but PSO normally does not include outliers as part of a cluster as our experimental results show. For example in the iris data set, we have seen that PSO has clearly discarded outliers from both clusters.

In some cases, choice of cluster centers affects the final result. Currently, we are approximating cluster center through visual inspection, but algorithms such as the k-means algorithm determine clusters centers automatically. Fuzzy logic approaches to automatically determine cluster centers have also been proposed [23], [17]. These algorithms can be combined with our novel algorithm to make the clustering fully automated.

It should not be misunderstood that PSO just works on the output of SOM i.e. U-matrix. PSO can be applied directly to raw data without any need for code vectors. Only a matrix of distances between data points (U-matrix) is required for finding optimum cluster boundaries. The remainder of the procedure is same.

REFERENCES

[1] A. Rauber and D. Merkl, Automatic Labelling of Self-Organizing Maps: Making a Treasure Maps Reveal Its Secrets, in *Proc. 4th Pacific-Asia Conference on Knowledge Discovery and Data Mining*, PAKDD99 (Beijing, China, 1999).

[2] A. Carlistle, and G. Dozier. Adapting Particle Swarm Optimization to Dynamic Environments [Online]. Available: http://www.CartistleA.edu (1998).

[3] A. Sharma, and W. C. Omlin, Determining cluster boundary using Particle Swarm Optimization, in Proc. World Academy of Science, Engineering and Technology, vol. 15 (2006) 250-254.

[4] B. Jiang, and L. Harrie, Selection of streets form a network using self-Organizing maps, *Transactions in GIS*, Vol. 8, No. 3 (2004) 335-350.

[5] 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 (2000) 1656-1659

[6] F. De la Torre Frade and T. Kanade, Discriminative Cluster Analysis, International Conference on Machine Learning, ACM Press, New York, NY, USA, Vol. 148 (June, 2006) 241 - 248.

[7] J. Vesanto, and E. Alhoniemi, Clustering of the Self-Organizing Map, *IEEE transaction on neural network*, Vol. 11, No. 3 (May 2000).

[8] J. Vesanto, and E. Alhoniemi, Clustering of the Self-Organizing Map, *IEEE Transaction on Neural Neetworks*, Vol. 11, No. 3 (2000) 586-600.

[9] J. Hollmén, Process Modelling using the Self-Organizing Map, M.Sc. thesis, Dept. Computer Science, Helsinki Univ. of Technology, Finland, 1996.

[10] J. Vesanto, J. Himberg, E. Alhoniemi, and J. Parhankangas, "SOM Toolbox for Matlab 5," ISBN 951-22-4951-0, ISSN 1456-2243 (Espoo, Finland, 2000).

[11] J. Kennedy, and R. C. Eberhart, The particle swarm: social adaptation in information processing systems, In Corne, D., Dorigo, M., and Glover, F., Eds., *New Ideas in Optimization* (London: McGraw-Hill, 1999) 379-387.

[12] J. Vesanto, and M. Sulkava, Distance matrix based clustering of the self-organizing map, in *Proc. International Conference on Artificial Neural Networks – ICANN 2002*, Lecture Notes in Computer Science, No. 2415 (Springer-Verlag, 2002) 951-956.

[13] J. Han and M. Kambler, Introduction, Data mining: concepts and techniques (Morgan Kaufmann Publishers, 2001).

[14] J. Han and M. Kambler, Cluster Analysis, Data mining: concepts and techniques, (Morgan Kaufmann Publishers, 2001).

[15] J. V. Oliveira and W. Pedrycz, Fundamentals of Fuzzy Clustering, Advances in Fuzzy Clustering and its Applications (John Wiley & Sons, Ltd, 2007).

[16] J. V. Oliveira and W. Pedrycz, Visualization of Clustering Results, Advances in Fuzzy Clustering and its Applications (John Wiley & Sons, Ltd, 2007).

[17] K. P. Malay, and S. Bandyopadhyay, and M. Ujjwal, "Validity index for crisp and fuzzy clusters," Pattern Recognition, Vol. 37, No. 3 (2004) 487-501.

[18] K. L. Wu, and M. S. Yang, A cluster validity index for fuzzy clustering, Pattern Recognition Letters, Vol 26, Issue 9 (2005).

[19] M. G. H. Omran, A. P. Engelbrecht, and A. Salman, Dynamic Clustering using Particle Swarm Optimization with Application in Unsupervised Image Classification, transactions on engineering, computing and technology, vol. 9 (2005).

[20] M. Kantardzic, Cluster Analysis, Data Mining – concepts, models, methods, and algrorithms (Wiley InterScience, 2003) 129-132.

[21] O. Abidogun, Data Mining, Fraud Detection and Mobile Telecommunication: Call pattern Analysis with Unsupervised Neural Networks, M.Sc. thesis, University of the Western Cape, Bellville, Cape Town, South Africa, 2004.

[22] R. J. Kuo, K. Chang, and S. Y. Chien, Integration of Self-Organizing Feature Maps and Genetic-Algorithm-Based Clustering Method for Market Segmentation, Journal of Organizational Computing and Electronic Commerce (2002).

[23] S. Bandyopadhyay, Simulated Annealing Using a Reversible Jump Markov Chain Monte Carlo Algorithm for Fuzzy Clustering, IEEE transactions on knowledge and data engineering, Vol. 17, No. 4 (2005) 479-490.

[24] T. Kohonen, Self-Organizing Maps. Springer-Verlag (Berlin, Germany, 2001).