Intelligent Particle Swarm Classifier

Seyed Hamid Zahiri and Seyed Alireza Seyedin

Abstract—**An Intelligent Particle Swarm classifier (IPS-classifier) is proposed in this paper. This classifier is described for finding the decision hyperplanes to classify patterns of different classes in the feature space using particle swarm optimization (PSO) algorithm. An intelligent fuzzy controller is designed to improve the performance and efficiency of proposed swarm intelligence based classifier by adapting three important parameters of PSO (i.e., *swarm size, neighborhood size, and constriction coefficient*). Three pattern recognition problems with different feature vector dimensions were used to demonstrate the effectiveness of the proposed classifier. They are the Iris data classification, the Wine data classification, and radar targets classification from backscattered signals.**

The experimental results show that the performance of the IPS-classifier is comparable to or better than the k-nearest neighbor (k-NN) and multi-layer perceptron (MLP) classifiers, which are two conventional classifiers.

Index Terms—**Decision hyperplanes, fuzzy controller, particle swarm optimization, pattern recognition.**

I. Introduction

PARTICLE swarm optimization (PSO) is a swarm intelligence technique developed by Kennedy and Eberhart in 1995 [1]. PSO inspired natural flocking and swarm behavior of birds and insects. It has been used in several tasks of optimization and engineering problems (e.g., [2] and [3]). In pattern recognition tasks some particle swarm clustering techniques are proposed (e.g., [4] and [5]), but a swarm intelligence based classifier using directly PSO to obtain the decision functions in the feature space has not been implemented in recent researches. In this article the concept of *intelligently controlling the search process of* PSO is integrated with a proposed *particle swarm classifier* to develop an *Intelligent Particle Swarm classifier* (*IPS-classifier).*

In fact, an IPS-classifier has an added intelligent controller for adaptation of the important parameters of PSO to increase its efficiency. It means convergence to better hyperplanes with a lower number of iterations. The most important parameters that should be controlled are: *swarm size, neighborhood size, and constriction coefficient*. Any kinds of intelligent controllers may be selected for efficiency of the classifier. In this article, a fuzzy structure has been chosen for aforementioned purpose and the IPS-classifier with this controller is called *fuzzy controlled particle swarm classifier (FCPS-classifier).*

The rules for designing the fuzzy controller have been extracted from logical linguistic descriptions on the effects of PSO parameters on its search process, which have been reported in previous researches [6]-[8].

Since PSO is a simple and powerful search technique in high dimensional spaces, an IPS-classifier has the potentiality to classify different classes successfully in *high dimensional* feature spaces, with a little a *priori* information.

In fact, the IPS-classifier searches in solution space and moves toward hyperplanes in such manner that the misclassified points are minimized.

Two common benchmark problems and a special problem in pattern recognition were considered for comparative experimental results. The Iris data and the Wine data classification are common problems in pattern recognition researches with low and medium feature space dimensions, and automatic target recognition in continuous wave radars is a special pattern recognition problem with high feature space dimensions. The performance of an IPS-classifier has been compared with k-nearest neighbor (k-NN) and multi-layer perceptron (MLP) classifiers, to show that the average of recognition scores of designed IPS-classifier are better than or comparable to those of the traditional classifiers. To see the effective role of intelligent fuzzy controller in the search process and correct steering the swarm toward the solution, some illustrative figures have been included.

In this paper, Section II explains a Particle Swarm classifier (PS-classifier). Intelligent particle swarm classifier is described in the next Section. Section IV considers implementation of the classifier and experimental results on three aforesaid pattern recognition problems. Finally, conclusion and discussion is presented in Section V.

II. Fundamentals of a Particle Swarm Classifier

A. PSO Algorithm

In the basic PSO proposed by Kennedy and Eberhart [1], many particles move around in a multi-dimensional space and each particle memorizes the position vector and velocity vector as well as the spot at which the particle has acquired the best fitness. Furthermore, respective particles can share data at the best-fitness spot for all particles. The velocity of each particle is updated with the best positions acquired for all particles over iterations, and the best positions are acquired by the respective particles over generations.

To improve the performance of the basic PSO, some new versions of this algorithm were proposed. At first, the concept of an *inertia weight* was developed to better control the exploration and exploitation in [9]. Then, the research done by Clerc [10] indicated that using a *constriction factor* may be necessary to insure convergence of the particle swarm algorithm. After these two important

Manuscript received November 11, 2004; revised March 30, 2005.

The authors are with the Department of Electrical Engineering, Faculty of Engineering, Ferdowsi University of Mashhad, Mashhad, Iran. P. O. Box 91755-1111, Mashhad, Iran (shzahiri@ yahoo.com, seyedin@ ferdowsi.un.ac.ir).

Publisher Item Identifier S 1682-0053(05)0323

modifications in the basic PSO, other researchers reported their works on PSO. For example, the *multi-phase particle swarm optimization (MPSO)* was introduced in [11]; in [12] the *particle swarm optimization with Gaussian mutation* combined the idea of swarm intelligence with the concepts of evolutionary algorithms; the *Quantum particle swarm optimization* was proposed in [13]; a modified PSO with *increasing inertia weight schedule* was proposed in [14]; the *Gaussian particle swarm optimization (GPSO)* was developed in [15] and the *guaranteed convergence PSO (GCPSO)*was introduced in [16].

In the present paper a PSO with constriction coefficient is used. The reason for this decision is that a good knowledge about the influence of constriction coefficient on the PSO search process is available [8]. Also, *"lbest"* strategy is considered in this paper. In this version, of PSO particles have information only of their own and their neighbors' bests, rather than of the entire swarm, which is called *"gbest"*.

In this approach updating is executed as follows:

$$V_i^{q+1} = \chi.(V_i^q + \phi_1(P_i - Y_i) + \phi_2(P_g - Y_i)) \qquad (1)$$

$$Y_i^{q+1} = Y_i^q + V_i^q \qquad (2)$$

$$\chi = \frac{2\kappa}{\phi - 2 + \sqrt{\phi^2 - 4\phi}} \qquad (3)$$

$$\phi > 4$$

where $i = 1,2,...,Pop$ and Pop is the swarm size, q is the generation counter, $V_i = (v_{i1}, v_{i2},..., v_{in})$ is the velocity vector of the i-th particle of the swarm, $P_i = (p_{i1}, p_{i2},..., p_{in})$ denotes the best position it has ever visited by the particle, $Y_i = (y_{i1}, y_{i2},..., y_{in})$ is the current position and $P_g = (p_{g1}, p_{g2},..., p_{gn})$ is the best particle among a neighborhood of the particles in the swarm. ϕ_1 and ϕ_2 are random numbers uniformly distributed in the range $(0, \phi/2)$, n is the dimension of space, and χ is the constriction coefficient. We set $\kappa = 1$, meaning that the space thoroughly searched before the swarm collapses into a point [8].

It is noteworthy that the above algorithm requires no explicit limitation as upper bound $V_{\max}$. However, from subsequent experiments and applications [17] it has been concluded that a better approach is to limit $V_{\max}$ to $Y_{\max}$, which is equal to the dynamic range of the variable.

B. Particle Swarm Classifier

A *particle swarm classifier* (PS-classifier) has three major parts including decision hyperplanes, fitness function definition, and its structure.

1) Decision Hyperplanes

A general hyperplane is in the form

$$d(X) = w_1 x_1 + w_2 x_2 + ... + w_n x_n + w_{n+1} \qquad (4)$$

where $X = (x_1, x_2,..., x_n, 1)$ and $W = (w_1, w_2,..., w_n, w_{n+1})$ are called the augmented feature and weight vector respectively and n is the feature space dimension.

In a general case, there are a number of hyperplanes ($\log_2^M$, is the minimum value, where M is the number of classes) that separate the feature space to different regions, which each region distinguishes an individual class

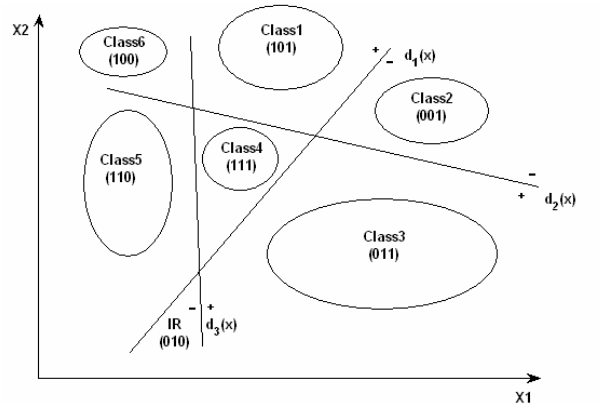


Fig. 1. Each region can identify an individual class by its code, which obtained from the sign of hyperplanes.

(Fig. 1). In Fig. 1 each class belongs to a region, which encoded by the sign of three hyperplanes (for two dimensional feature space). In this figure, IR denotes the indeterminate region. Some especial cases are described in [18].

The *PS–classifier* must find W_j $(j = 1,2,..., H)$ in solution space, where H is the necessary number of decision hyperplanes.

2) Fitness Function Definition

Fitness function is defined as

$$fit(P_i) = T - Miss(P_i) \qquad (5)$$

where T is the size of the total training data set and $Miss(P_i)$ is the number of misclassified training points by P_i. By maximizing the fitness function, minimum error of training points classification is achieved.

3) The Structure of Particle Swarm Classifier

According to the above descriptions, designing a PS-classifier has the following steps:

```
$q = 1$;
Swarm =Generate(Swarm_size);
Define (Neighborhood_ $h_i$ );
while (termination condition)
    Compute_fitness (Swarm);
    $P_g$ =Obtain ($P_g$) ;
    $P_i$ =Obtain ($P_i$) ;
    $\phi_1$ =rand $(0, \phi/2)$ ;
    $\phi_2$ =rand $(0, \phi/2)$ ;
    Calculate ( $\chi$ );
    for $i = 1$, Swarm_size
        Update_velocity ($V_i^q$) ;
        Update_position ($Y_i^q$) ;
    end for,
    $q = q + 1$;
end while;
```

In a PS-classifier each particle is selected randomly from the solution space and has the form of $P = [W_1, W_2,..., W_i,..., W_H]^T$ where $W_i = (w_{i1}, w_{i2},..., w_{in}, w_{in+1})$ is the weight vector of i-th hyperplane, and H is the pre-defined number of hyperplanes. Fitness function is defined as (5). Termination condition can be the best fitness value or a default maximum number of iterations. After enough iterations the particles converge to a solution that is the decision hyperplanes whose misclassified training points are minimized.

III. Intelligent Particle Swarm Classifier

It should be mentioned that PSO algorithm has some important parameters which play major roles in its search process (e.g., premature convergence, convergence rate, local capturing, exploitation, exploration, etc.). In the constriction *lbest* PSO defined in Section II.A, *swarm size, neighborhood size, and constriction coefficient* are these three important factors. In most researches reported on the applications of PSO, these parameters were obtained by running the PSO for several times with different sets of parameter to find a proper set.

On the other hand, since the search process of the PS-classifier is non-linear and very complicated, it is hard if not impossible, to mathematically model the search process to dynamically adjust the PS-classifier parameters. But over the years, some understanding of the PSO search process has been accumulated, and linguistic description of the search process is available. This understanding and linguistic description make a fuzzy system a good candidate for dynamically controlling the aforesaid PS-classifier parameters.

Already, in [19] a fuzzy system has been designed to adjust the inertia weight of PSO for the Rosenbrock function and in [6] this fuzzy system has been modified to fit a wide range of optimization problems.

In this paper we introduced another fuzzy system to control the swarm size, neighborhood size, and constriction coefficient to improve the efficiency and performance of the proposed PS-classifier. To extract some effective fuzzy rules, at first, a linguistic description about the effects of the PSO parameters on its search process is presented as a subsection.

A. Linguistic Description on the Effect of PSO Parameters on Its Search Process

1) Swarm Size

Swarm size has a significant effect on the search process of PSO. A large value of swarm size reduces the convergence rate considerably and slows down the algorithm; whereas a small value of swarm size causes a local minimum capturing and reduces the performance of PSO. In [7] a mathematical approach like the *selection* in evolutionary algorithms was proposed. We used similar approach but with some fuzzy rules in such manner that when the algorithm captures in a non-important local, the swarm size is increased to escape it from this local solution and by receiving a better solution, the swarm size is decreased to improve the convergence rate.

2) Neighborhood Size

In PSO, particles tend to be influenced by their success along their past history and also by the success of any particles in their neighborhood, i.e., with which they interact. To these "schemes of interactions" between particles, the authors termed *sociometric principles*. Particles can interact with each other in a number of ways. The simplest way is in the form that the particle interacts with its two nearest neighbors. Any number of nearest neighbors can be used. As it mentioned in Section II.A, if the number of nearest neighbors is less than the total number of particles in the swarm, then this sociometric principle is called *lbest*, else it is called *gbest*. Conceptually, *gbest* connects all the particles together, what means that their social interaction is maximal. In contrast, *lbest* results in a local neighborhood for the particle.

Early experiences (mainly by trial and error) led to neighborhood sizes about 15 percent of the swarm size being used for some applications, but it is not a restrict rule. Since the information is exchanged between neighboring particles in the topology, selection of the smaller neighborhood causes finding the best position more slowly. Increasing the neighborhood size increases the convergence rate, but in this case PSO might be trapped into a local optimum.

3) Constriction Coefficient

The constriction coefficient (χ) is controlled by the parameter ϕ using (3). In [8] an exploration was presented which indicates how the particle swarm algorithm works. Specifically, it has been proved that the application of constriction coefficient allows control over the dynamical characteristics of the particle swarm, including its exploration versus exploitation propensities. In fact, constriction coefficient prevents a buildup of velocity because of the effect of particle inertia. Without the constriction coefficient, particles with buildup velocities might explore the search space, but loose the ability to fine-tune a result. On the other hand, constriction the particle speed too much might damage the search space exploration. Thus the value of constriction coefficient affects the global versus local abilities of the PSO. It can be concluded from [8] that ϕ determines the value of attraction of particles by the best positions found previously by itself and by its neighborhood (this conclusion also is appeared in [7]). This means that the convergence characteristics of PSO can be controlled by ϕ .

As the fitness value of a particle increases, the part of search space, which the particle explores should be reduced. It means that ϕ should be increased above 4 to decrease χ and $\chi\phi$. The result is decreasing the inertia of the particle to emphasize the local search instead of global. A less improvement in the particle fitness causes a wider search space for the exploration. This means a decreasing should be happen on the value of ϕ , to increase the inertia of the particle. It results in emphasizing the global search instead of local.

B. The Fuzzy Controlled PS-Classifier

It is known that efficiency and complexity in most evolutionary classifiers have direct relationship. Normally, higher efficiencies are resulted from classifiers with higher complexities and vise versa. Similarly, in a PS-classifier, using an additional intelligent controller (herein fuzzy controller) increases the computation complexity. On the other hand, the use of such combination improves the efficiency of PS-classifier, especially in high dimensional feature space. The Fuzzy Controlled PS-classifier (FCPS-classifier) is a kind of IPS-classifiers, whose controller is a fuzzy structure with some fuzzy *inputs*, *outputs*, and *IF…THEN* fuzzy rules extracted from above linguistic descriptions.

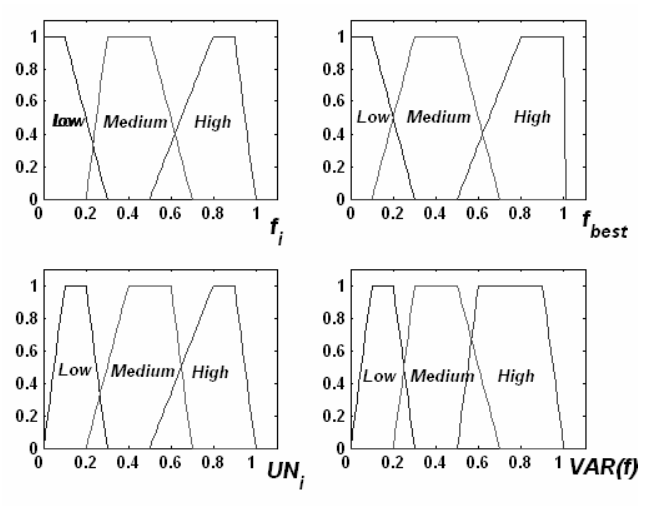


Fig. 2. Normalized inputs membership functions.

The fuzzy controller is constructed with three inputs and three outputs. The inputs are as follows:

- f_i: The fitness value of i-th particle ($fit\,(P_i)$) and is calculated from (5).
- f_{best}: The maximum fitness value of whole the swarm.
- UN_i: The number of iterations, whose fitness value of the neighborhood of h_i is unchanged.
- $\mathrm{VAR}(f)$: The variance of the fitness of the particles in each iteration.

UN_i is introduced as an input of fuzzy controller to know which neighborhood converged (or captured) to a local optimum and $\mathrm{VAR}(f)$ is introduced as a metric of *swarm diversity*. Obviously, large values of $\mathrm{VAR}(f)$ show large swarm diversity and vice versa.

The three outputs are:

- *Swarm_size*: The size of the swarm.
- h_i: The size of neighborhood around P_i.
- Cf_i: The controlling factor of constriction coefficient χ for P_i (ϕ_i in (3)).

The following eight fuzzy rules can be extracted from the linguistic description in Section III.A to control the search process of PS-classifier intelligently:

1- IF f_i is low, THEN h_i is high and Cf_i is low.
2- IF f_i is medium and UN_i is low, THEN h_i is medium and Cf_i is low.
3- IF f_i is medium and UN_i is medium, THEN h_i is medium and Cf_i is medium.
4- IF f_i is high and UN_i is high, THEN h_i is low and Cf_i is high.
5- IF f_{best} is low and $\mathrm{VAR}(f)$ is low, THEN *Swarm_size* is high and h_i is low.
6- IF f_{best} is medium and $\mathrm{VAR}(f)$ is high, THEN *Swarm_size* is high, h_i is low, and Cf_i is medium.
7- IF f_{best} is high and $\mathrm{VAR}(f)$ is medium, THEN *Swarm_size* is medium and h_i is medium.
8- IF f_{best} is high and $\mathrm{VAR}(f)$ is high, THEN *Swarm_size* is low, h_i is high, and Cf_i is low.

The fuzzy controller were designed with above fuzzy rules and its scaled inputs and outputs membership functions are shown in Figs. 2 and 3, respectively.

The increase or decrease of the *Swarm_size* is implemented by removing the worst particles and regenerating the best particles to receive a necessary swarm size.

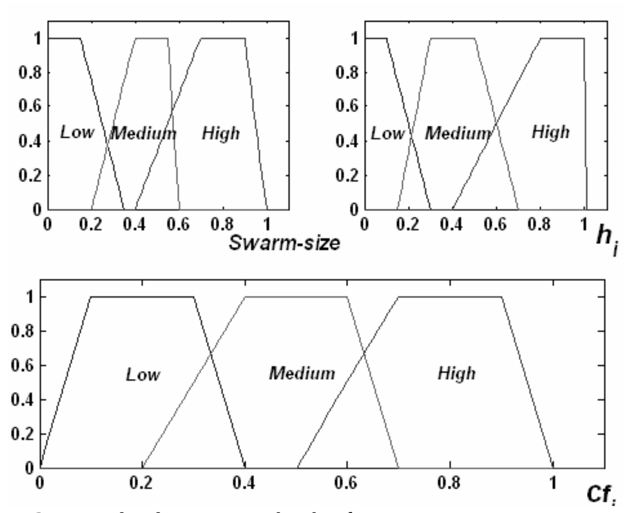


Fig. 3. Normalized outputs membership functions.

TABLE I
TEN TARGETS AS REFERENCE CLASSES

Number	Target	Application
1	V.F-3	Training
2	PC-7	Training
3	ANTONOV AN-12	Military
4	FFA AS ZZO118A	Training
5	BAE-248 SERIES 2B	Transportation
6	KJ 500-3S	Military
7	ROLLS ROYCE ALISON	Military
8	KUZNETSORNK-8-2	Transportation
9	TUMMANSKY R-11 F2S	Military
10	ROLLS ROYCE 535 E1 H4	Military

It must be mentioned that different kinds of inputs, outputs, membership function shapes, membership function locations, and fuzzy rules may be introduced and even these parameters can be optimized by another optimization algorithm. In this paper, the membership functions and their locations are selected and tuned manually.

IV. IMPLEMENTATION AND RESULTS

Three pattern recognition problems with different augmented feature vectors dimensions (5,14,129) were used to show the performance of the IPS-classifier. A description of the data sets is given here:

A. Data Sets

Iris data: The Iris data contains 50 measurements of four features from each three species Iris setosa, Iris versicolor, and Iris virginica [20]. Features are sepal length, sepal width, petal length, and petal width.

Wine Data: The Wine data contains the chemical analysis of wines grown in the same region in Italy but derived from different cultivars [21]. Thirteen continuous attributes are available for classification. The number of classes is three and the numbers of instances in each class are 59, 71, and 48, respectively.

Radar Targets: An application of pattern recognition is Automatic Target Recognition (ATR) for continuous wave radars. In this paper Jet Engine Modulations approach (JEM) is used for this purpose. In this way the modulation of the radar wave by rotating propellers and jet engine

TABLE II
RECOGNITION SCORES (%) FOR IRIS TEST DATA CLASSIFICATION WITH $H = 3$

	Total training points=15			Total training points=30			Total training points=45		
	IPS-classifier	MLP	k-NN	IPS-classifier	MLP	k-NN	IPS-classifier	MLP	k-NN
Class1	87.5	85.3	88.5	94.2	92.2	92.0	97.1	94.5	97.5
Class2	83.4	81.6	85.1	97.5	93.3	93.8	95.7	95.2	96.5
Class3	81.4	79.0	82.1	88.6	86.8	90.1	94.1	91.1	92.0
Overall	84.10	81.97	85.23	93.43	90.77	91.97	95.63	93.60	95.33

TABLE III
RECOGNITION SCORES (%) FOR WINE TEST DATA CLASSIFICATION WITH $H = 5$

	Total training points=25			Total training points=50			Total training points=75		
	IPS-classifier	MLP	k-NN	IPS-classifier	MLP	k-NN	IPS-classifier	MLP	k-NN
Class1	72.6	75.3	80.1	91.3	88.2	90.2	93.5	90.5	90.3
Class2	77.2	78.1	75.2	89.3	83.1	88.4	91.3	87.7	89.2
Class3	81.7	62.3	83.3	95.1	78.2	93.6	94.0	83.5	93.5
Class4	79.3	79.2	77.4	95.7	87.1	94.2	94.6	86.2	95.1
Class5	69.9	63.4	72.1	89.8	78.1	89.0	92.7	88.2	91.5
Overall	76.14	71.66	77.62	92.24	82.94	91.08	93.22	87.22	91.92

TABLE IV
AVERAGE RECOGNITION SCORES (%) WITH RESPECT TO DIFFERENT SNRS FOR RADAR TARGETS CLASSIFICATION WITH $H = 10$

SNR (dB)	-15	-10	-5	0	5	10	15
IPS-classifier	9.9	23.3	35.5	56.3	64.3	89.6	89.7
MLP	10.3	11.7	14.9	46.5	57.9	66.6	76.2
k-NN	10.5	7.7	17.0	50.5	62.5	79.0	79.5

blades of targets is considered [22]. Ten different flying objects were chosen as introduced in Table I for classification in 20º elevation angle. After sampling from backscattered signals and data reduction preprocess, we took 128 points FFT as feature vectors for each target.

B. Comparison with Existing Methods

The performance of proposed IPS-classifier is compared with the performance of MLP and k-NN classifiers to show that the average recognition scores of the designed PS-classifier are better than or comparable to those of the traditional classifiers. For MLP the [3,5,4] structure for Iris data, [8,5,4] for Wine data and [21,19,12,14,8,11] for radar targets are used and trained in MATLAB® 7.0. These structures were selected experimentally; no optimization technique is used because a *traditional* MLP classifier is considered for comparing the results. For these three structures the learning rate η is initially fixed at 2.0. This is decreased by a factor of 2, up to a pre-specified minimum value, if the mean squared error starts oscillating. In case the error decreases very slowly, then the learning rate is doubled. The reason is that most likely the algorithm has confronted a plateau in the error surface.

k-NN classifier is executed taking k equal to $\sqrt{T}$, where T is the number of training samples (it is known that as the number of training patterns T goes to infinity if the value of k and k/T can be made to approach infinity and 0, respectively, then k-NN classifier approaches the optimal Bayes classifier [18]. One such value of k for which the limiting conditions are satisfied is $\sqrt{T}$).

C. Experimental Results

The proposed IPS-classifier (i.e., FCPS-classifier), MLP and k-NN classifiers are tested on the data sets described in Section IV.A. Tables II and III present the results corresponding to Iris data and Wine data classifications, respectively. Different number of training samples $(t = 5,10,15)$ have been considered. The training samples were chosen randomly from each class. These experiments have been done using 3 hyperplanes for Iris $(H = 3)$ and 5 hyperplanes for Wine $(H = 5)$. The total number of training points is obtained by $t\times$*Number of Classes* (e.g., for Iris data when $t = 5$ the total number of training points is $T = 15$). The remained patterns in each case were considered as testing points and the reported results in Tables II and III are for testing patterns classification. Since the IPS-classifier starts with random initial condition, and it is possible to converge to different separating hyperplanes, the presented results for IPS-classifier are the average on ten times repetition.

For each problem the initial swarm size is set to 20 and the termination condition is considered as a maximum value of iterations, which is set to an experimentally obtained value of 250. Tables II and III show that as the number of training points increases, the performance of the IPS-classifier becomes better and better. It is comparable or better than MLP and k-NN classifiers. For example, if the total number of training points is equal to 30, for Iris data classification, the recognition score of class1 for IPS-classifier is 94.2%, which is better than both MLP and k-NN. In the case of Wine data classification for 50 training points the score is 91.3% that is better than MLP and k-NN results. The last rows of these tables show the overall recognition scores.

Although in some cases it can be seen that the recognition scores for MLP or k-NN is better than the IPS-classifier, but the differences between them is not significant. The maximum difference, for Iris data classification has been appeared for 15 training data points for class2, which is equal to 1.7%. The maximum difference, for Wine data classification has been appeared for 25 training data points of class1, about 7.5%. This is a significant difference, but overall recognition scores in this

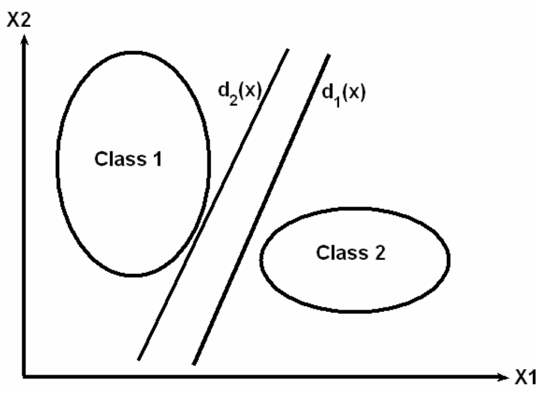


Fig. 4. $d_1(x)$ and $d_2(x)$ are two possible solutions that IPS-classifier may find.

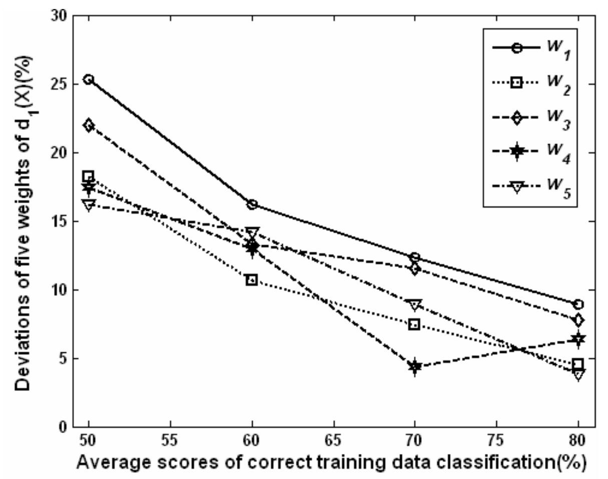


Fig. 5. The percent of deviations of normalized obtained weight vectors of the first decision hyperplane with respect to different performances of IPS-classifier on *training* data classification for Iris data.

number of training points is comparable for both IPS-classifier and k-NN classifier.

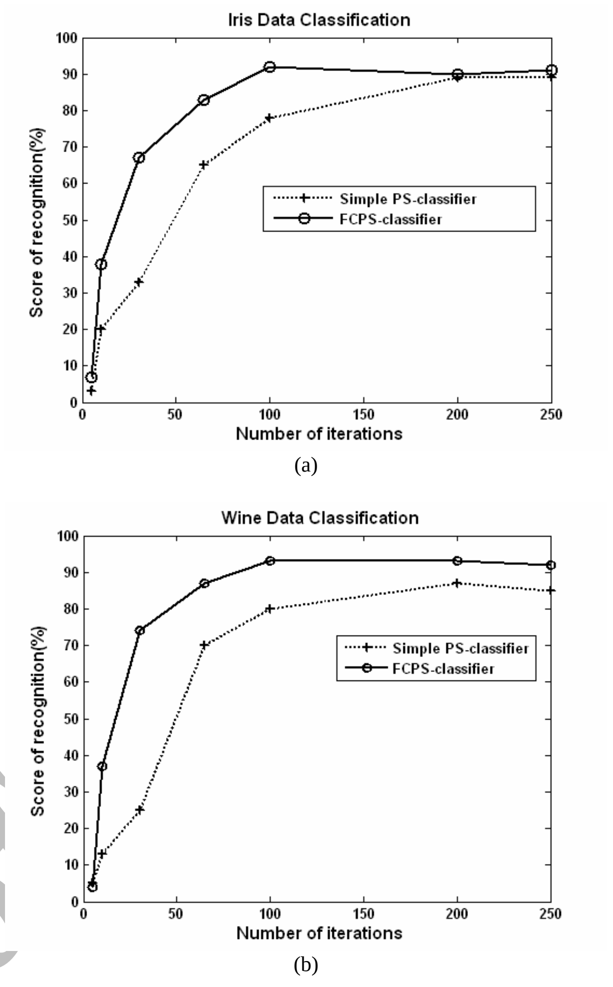
These values demonstrate the ability of IPS-classifier, as a new swarm intelligence based classifier, in comparison with two traditional classifiers, i.e., MLP and k-NN.

Radar targets classification is done by ten hyperplanes $(H = 10)$ and for ten training points from each target. In this experiment we waited for 300 iterations running for IPS-classifier in different signal to noise ratios (changing the variances of Gaussian noise produces different powers of noise). Table IV shows the results.

Only at SNR=-15dB the performance of IPS-classifier is worse than MLP and k-NN classifiers. Of course, the differences between them are such a small value that their performance can be considered comparable with others. In other SNRs the performance of IPS-classifier is better than other classifiers and as the SNR increases the improvement becomes more apparent.

Due to the kind of fitness function definition, IPS-classifier may obtain different hyperplanes each time it runs. For example, each $d_1(x)$ and $d_2(x)$ in Fig. 4 may be found by IPS-classifier in separate experiments.

Fig. 5 shows the percentage of deviations of normalized obtained weight vectors of the first decision hyperplane on *training* data classification for Iris data in ten repetitions of the experiment. By changing the value of $Miss(P_i)$ in fitness function definition (5) different values of performance on *training* data classification is obtained.



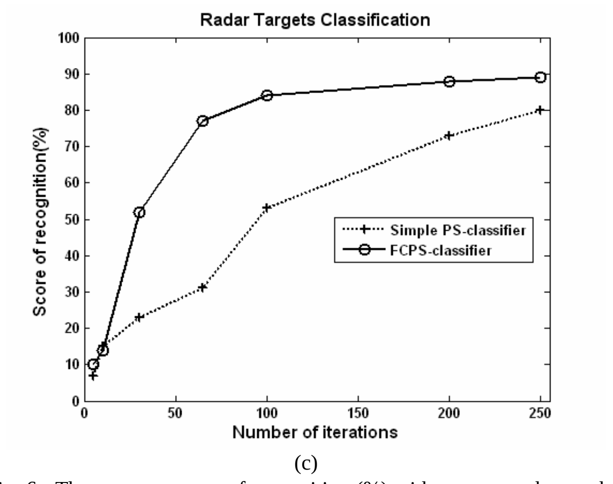


Fig. 6. The average scores of recognition (%) with respect to the number of iterations for (a) Iris data, (b) Wine data, and (c)Radar targets classification.

In this figure the total number of training data is 50, i.e., 10 points from each class.

D. Effectiveness of Fuzzy Controller

A meaningful concept to see how the designed fuzzy controller steers particles to converge toward the solution and escaping them from bad local points is the reduction values of the number of iterations while running the IPS-classifier (the number of fitness function evaluations can be obtained by: *Swarm_size*×*Number of iterations*).

In some individual experiments a simple PS-classifier, which has not any intelligent controller for adapting the *swarm size*, *neighborhood size* and *constriction factor* were designed and its performance in three aforesaid classification problems were evaluated for different number of iterations. Same experiments were done for an IPS-classifier (here FCPS-classifier). These experiments were repeated 10 times and then their average scores of test points recognition were compared with another. In setup of these experiments, the number of training data from each class is 10. The rest points are as test data. The swarm size of simple PS-classifier is set to 20 as an initial swarm size for FCPS-classifier and the initial value of ϕ is set to 4. The maximum value of iterations is the same as FCPS-classifier (250 iterations).

Fig. 6 shows the results. This figure is a good evidence to demonstrate that the designed fuzzy controller helps the PS-classifier to converge with lower iterations. This reduction is more significant at high recognition scores, because at the primary iterations, the designed intelligent controller has not enough time for playing its effective role in parameters adaptation in IPS-classifier. The iteration reduction at 90% average recognition score is about 150 iterations (60%) for Iris data classification. This improvement is approximately 50 iterations (50%) at 80% recognition score for Wine data classification. It is about 170 iterations (68%) reduction for 80% average score of recognition for radar targets classification (in this step the signal to noise ratio (SNR) is approximately 10 dB).

On the other hand the effectiveness of fuzzy controller is more apparent in more dimensional feature space. The final difference between the performances of simple PS-classifier and FCPS-classifier is low for Iris data classification with 4 dimensional feature vectors. It is about 7% for wine data classification, which has 13 features for each pattern and the difference is a significant value of 10% for radar targets classification.

Fig. 7 indicates the effects of variations of some of the parameters that exist in the fuzzy controller, (especially its outputs) for P_3 in Iris data classification. This figure clearly shows the effects of fuzzy controller in adaptation of the IPS-classifier parameters which improves the fitness of P_3. For example, the reduction of swarm size in high value of iterations due to a good improvement in fitness values is noteworthy.

V. Conclusion and Discussion

Powerfulness and effectiveness of PSO algorithm, specially in high dimensional spaces, are motivations to design a particle swarm based classifier (PS-classifier), which can obtain the decision hyperplanes in the feature space. Since swam-size, neighborhood size, and constriction factor are three important parameters, which have a great effects on the search process of PS-classifier, designing an intelligent fuzzy controller for adapting these parameters has been considered. The fuzzy controlled PS-classifier (FCPS-classifier) is proposed for this purpose. The rules of fuzzy controller were constructed based in a linguistic description of the roles of aforesaid parameters on the PSO search process. Experimental results on different kinds of data (with different dimensions of 4, 13 and 128) indicate that for a given value of H (the number of hyperplanes), the designed IPS-classifier is able to

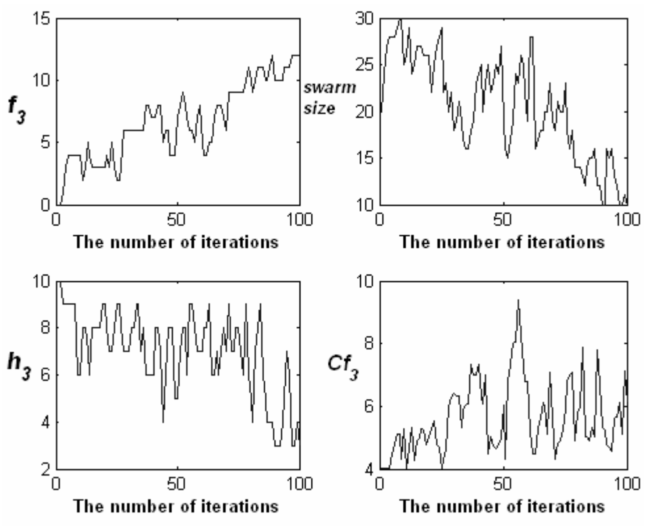


Fig. 7. Dynamically variations of f_3, *Swarm_size*, h_3 and Cf_3.

approximate efficiently the decision hyperplanes. The performance of the classifier is also found to be comparable to, sometimes better than, those of the k-NN and MLP classifiers.

Although in this paper a fuzzy structure has been selected as a candidate for development an IPS-classifier, but other kinds of intelligent controllers might be used to develop more efficient IPS-classifier instead of FCPS-classifier.

With regard to system complexity it may be noted that the complexity of an IPS-classifier is more than a k-NN classifier. Regarding the timing requirements, it may be noted that the IPS-classifier takes a large amount of time *during training* like a MLP classifier; however, the time taken *during testing is very small*, because the decision hyperplanes have been already obtained in training phase. On the contrary, the k-NN classifier takes *significant amount of time for testing*. In fact the proposed IPS-classifier is an efficient offline classifier and may be applied for offline usages (offline signature recognition, face recognition, voice classification, etc.)

A theoretical analysis of a PS-classifier and designing an effective strategy to reach to a performance comparable to the Bayes classifier, which is an optimal conventional classifier (but with some limitations in usage due to need to important *priori* knowledge), are topic tasks for future works.

References

[1] J. Kennedy and R. C. Eberhart, "Particle swarm optimization," in *Proc. IEEE Intl. Conf. on Neural Networks IV*, pp. 1942-1948, 1995.

[2] Z. He *et. al.*, "Extracting rules from fuzzy neural network by particle swarm optimization," *in Proc. IEEE Intl Conf. on Evolutionary Computation*, 1998.

[3] H. Yoshida, K. Kawata, Y. Fukuyama, and Y. Nakanishi, "A particle swarm optimization for reactive power and voltage control considering voltage security assessment," *IEEE Trans. on Power Systems*, vol. 15, no. 4, pp. 1232-1239, Nov. 2001.

[4] D. W. Van der Merwe and A. P. Engelbrecht, "Data clustering using particle swarm optimization," in *Proc. of the 2003 Congress on Evolutionary Computation*, pp. 215-220, 2003.

[5] Y. Yang and M. Kamel, "Clustering ensemble using swarm intelligence," in *Proc. of the 2003 IEEE Swarm Intelligence Symposium, SIS '03*, pp. 65-71, Apr. 2003.

[6] Y. Shi and R. C. Eberhart, "Fuzzy adaptive particle swarm optimization," in *Proc. Of the 2001 Congress on Evolutionary Computation*, pp. 101-106, 2001.

[7] W. Zhang, Y. Liu, and M. Clerc, "An adaptive PSO algorithm for reactive power optimization," in *Proc. of the Intl. Conf. on Advances Power System, Control, Operation and Management, APSCOM 2003*, pp. 302-307, Hong Kong, Nov. 2003.

[8] M. Clerc and J. Kennedy, "The particle swarm–explosion, stability, and convergence in a multidimensional complex space," *IEEE Trans. on Evolutionary Computation*, vol. 6, no .1, pp.58-73, Feb. 2002.

[9] Y. Shi and R. C. Eberhart, "A modified particle swarm optimizer," *In Proc. of the IEEE Intl. Conf. on Evolutionary Computation*, pp. 69-73, 1998.

[10] M. Clerc, "The swarm and the queen: towards a deterministic and adaptive particle swarm optimization," *In Proc. of the 1999 Congress on Evolutionary Computation*, pp. 1951-1957, 1999.

[11] B. Al-kazemi and C.K. Mohan, "Multi-phase generalization of the particle swarm optimization algorithm," in *Proc. of 2002 Congress on Evolutionary Computation*, vol. 1, pp. 489-494, May 2002.

[12] N. Higashi and H Iba, "Particle swarm optimization with Gaussian mutation," in *Proc. of the 2003 IEEE Swarm Intelligence Symposium, SIS '03*,pp. 72-79, Apr. 2003.

[13] S. Yang, M. Wang, and L. Jiao, "A quantum particle swarm optimization," in *Proc. of the 2004 Congress on Evolutionary Computation*, vol. 1, pp. 320-324, Jun. 2004.

[14] Y. Zheng, L. Ma, L. Zhang, and J. Qian, "On the convergence analysis and parameter selection in particle swarm optimization," in *Proc. of the Second Intl. Conf. on Machine Learning and Cybernetics*, pp. 1802-1807, Nov. 2003.

[15] B.R. Secrest and G.B. Lamont, "Visualizing particle swarm optimization – Gaussian particle swarm optimization," in *Proc. of the 2003 IEEE Swarm Intelligence Symposium, SIS'03*, pp. 198-204, Apr. 2003.

[16] F. Van den Berg, *An Analysis of Particle Swarm Optimizers*, Ph.D. Thesis, Department of Computer Science, University of Pretoria, South Africa, 2002.

[17] R. C. Eberhart and Y. Shi, "Comparing inertia weights and constriction factors in particle swarm optimization," in *Proc. of the 2000 Congress on Evolutionary Computation*, pp. 84-88, Jul. 2000.

[18] K. Fukunaga, *Introduction to Statistical Pattern Recognition*, New York: Academic, 1972.

[19] Y. Shi. and R. C. Eberhart, "Experimental study of particle swarm optimization," Presented at *4th Multiconference on Systems, Cybernetics and Informatics*, Orlando, Florida, US, Jul. 2000.

[20] R. A. Fisher, "The use of multiple measurements in taxonomic problems," *Ann. Eugen*, vol. 7, no. 2, pp. 179-188, 936.

[21] *Machine Learning Databases*, University of California, Irvine, via anonymous ftp ftp.ics.uci.edu/pub/machine-learning-databases.

[22] S. H. Zahiri, H. Zareie, and M.R. Agha-Ebrahimi, "Automatic target recognition using jet engine modulation on backscattered signals," in *the Communication Proc. of 8'th Iranian Conf. on Electrical Engineering*, Isfahan, pp. 296-303, 1996.

Seyed Hamid Zahiri was born in Iran. He received the B.Sc. and M.Sc. degrees in electronics engineering from the Sharif University of Technology, Tehran, Iran, and Tarbiat Modarres University, Tehran, Iran, in 1993 and 1995, respectively. He is currently a Ph.D. student in electronics engineering in the Ferdowsi University of Mashhad, Mashhad, Iran.

His research interests include pattern recognition, evolutionary algorithms, and swarm intelligence algorithms.

Seyed Alireza Seyedin was born in Iran. He received the B.Sc. degree in electronics engineering from Isfahan University of Technology, Isfahan, Iran in 1986 and the M.E. degree in control engineering from Roorkee University , Roorkee, India in 1992 and the Ph.D. degree in Electrical Engineering from the University of New South Wales, Sydney, Australia in 1996.

Since 1996, he has been with the Faculty of Engineering, Ferdowsi University of Mashhad, Mashhad, Iran as an Assistant Professor. He was the head of the department of the electrical engineering for the period 1998-2000. He is a member of Scientific Committee of the Iranian Conference on Machine Vision and Image Processing. His current research interests include image analysis, digital signal processing, robotics, machine vision, and pattern recognition.