Available online at www.sciencedirect.com
ScienceDirect
Fuzzy Sets and Systems 159 (2008) 2890–2909
FUZZY sets and systems
www.elsevier.com/locate/fss

An efficient immune-based symbiotic particle swarm optimization learning algorithm for TSK-type neuro-fuzzy networks design

Cheng-Jian Lin*

Department of Electrical Engineering, National University of Kaohsiung, Kaohsiung 811, Taiwan, ROC

Received 23 October 2006; received in revised form 31 August 2007; accepted 11 January 2008
Available online 8 February 2008

Abstract

In this paper, we propose a new learning algorithm that can be used to design TSK-type neuro-fuzzy networks. Though there has been a great deal of interest in the use of immune algorithms (IAs) for computer science and engineering, in terms of fundamental methodologies, they are not dramatically different from other algorithms. In order to enhance the IA performance, we propose the immune-based symbiotic particle swarm optimization (ISPSO) for use in TSK-type neuro-fuzzy networks for solving the prediction and skin color detection problems. The proposed ISPSO embeds the symbiotic evolution scheme in an IA and utilizes particle swarm optimization (PSO) to improve the mutation mechanism. In order to avoid trapping in a local optimal solution and to ensure the search capability of a near global optimal solution, mutation plays an important role. Therefore, we employed the advantages of PSO to improve the mutation mechanism and used a method that introduces chaotic mapping with certainty, ergodicity and the stochastic property into PSO to improve global convergence. Unlike the IA that uses each individual in a population as a full solution to a problem, symbiotic evolution assumes that each individual in a population represents only a partial solution to a problem. Complex solutions combine several individuals in the population.

Keywords: TSK-type neuro-fuzzy networks; Immune algorithm; Particle swarm optimization; Symbiotic evolution; Prediction; Skin color detection

1. Introduction

Fuzzy systems and neural networks are complementary technologies that can be used in the design of intelligent systems. Each method has its own strengths and weaknesses. Neural networks are essentially low-level computational structures and algorithms that offer good performance in dealing with sensory data, while fuzzy system technologies often deal with issues such as reasoning on a high-level than neural networks. However, since fuzzy systems do not have much learning capability, it is difficult for a human operator to tune the fuzzy rules and membership functions from the training data. Also, because the internal layers of neural networks are opaque to the user, the mapping rules in the networks are not visible and are difficult to understand. Thus, a promising approach for reaping the benefits of both fuzzy systems and neural networks is to merge or fuse them into an integrated system [29].

Neuro-fuzzy networks [27,40,13–15,30,5,22] are gaining research interest. They not only have attracted considerable attention in recent years due to their diverse applications in fields such as pattern recognition, image processing, and

* Fax: +886 43742375.
E-mail address: cjlin@nuk.edu.tw.

0165-0114/$ - see front matter © 2008 Elsevier B.V. All rights reserved.
doi:10.1016/j.fss.2008.01.020

control, but they can also handle imprecise information through linguistic expressions. However, these models often have a serious drawback; that is, they use the backpropagation (BP) learning algorithm [5,14]. Using the steepest descent optimization technique in BP training could minimize the error function, allowing the algorithm to reach the local minima very fast, but never finds a global solution. In addition, BP training performance depends on the initial system parameter values. For different network topologies one must derive new mathematical expressions for each network layer.

Considering the aforementioned disadvantages, suboptimal performance occurs, even for a suitable neuro-fuzzy network topology. Hence, technologies capable of training the system parameters and finding the global solution while optimizing the overall structure are needed. Lee and Takagi [23] proposed a design method of the fuzzy system using genetic algorithms (GAs). Since GAs are heuristic and stochastic, they are less likely to get stuck at the local minimum, and they are based on populations made up of individuals with specific behaviors similar to certain biological phenomena. These common characteristics have led to the development of evolutionary computation as an increasingly important field.

Recently, there has been a great deal of interest in the use of immune systems and algorithms in computer science and engineering [17,24,41,43,7]. Kalinli [17] proposed an artificial immune algorithm (IA) for an infinite impulse response (IIR) filter design. Liao [24] embedded chaos search capability in immune GAs to solve short-term thermal generating unit commitment problems. Liao's method uses fuzzy systems to determine the rate of crossover and mutation mechanism. Wen [41] proposed an immune evolutionary algorithm for sphericity error evaluation. A self-adaptive mutation operator was constructed to divide the mutation step size of every antibody according to its environment. Zhou's proposal [43] was based on the immune recognition principle to predict the performances of hot-rolled steel bars. Chun [7] employed an IA as the search method for the shape optimization of an electromagnetic device. Chun's search method improved the global search performance of the GA by using the immune network theory. However, these approaches in the fundamental methodologies are not dramatic in terms of overall performance.

This study proposes the immune-based symbiotic particle swarm optimization (ISPSO) for use in TSK-type fuzzy networks to solve the prediction and skin color detection problems. The proposed ISPSO embeds the symbiotic evolution scheme in an IA and utilizes particle swarm optimization (PSO) to improve the mutation mechanism.

The PSO algorithm, proposed by Kennedy and Eberhart [19], has proved to be very effective for solving global optimization. It is not only a recently invented high-performance optimizer that is easy to understand and implement, but it also requires little computational bookkeeping and generally only a few lines of code [3]. In order to avoid trapping in a local optimal solution and to ensure the search capability of a near global optimal solution, mutation plays an important role in ISPSO. Therefore, we employed the advantages of PSO to improve the mutation mechanism. However, the parameters (ω, r_1 and r_2) of the traditional PSO are the key factors that affect convergence [37,32,16], where ω, r_1 and r_2 are called the coefficient of the inertia term, the cognitive term and the society term, respectively. In fact, the parameters of the traditional PSO cannot ensure the optimization's ergodicity entirely in phase space because they are absolutely random. Therefore, we use a method that introduces chaotic mapping with certainty, ergodicity and the stochastic property into PSO so as to improve global convergence.

In addition to modifying the PSO, we improve the IA structure. Unlike the IA that uses each individual (antibodies) in a population as a full solution to a problem, symbiotic evolution [16,31] assumes that each individual in a population represents only a partial solution to a problem. Complex solutions combine several individuals in the population. Our proposed method improves the search ability and greatly increases the convergence speed in simulations.

This paper is organized as follows. Section 2 describes the structure of the TSK-type neuro-fuzzy network. The proposed ISPSO is presented in Section 3. In Section 4, the proposed ISPSO method is evaluated, and its performances are benchmarked against other structures. Finally, conclusions on the proposed method are given in the last section.

2. Structure of the TSK-type neuro-fuzzy network

This section describes TSK-type neuro-fuzzy networks [36]. A fuzzy logical system is a knowledge-based system characterized by a set of rules that determine the relationship between the input and the output. The reasoning process is defined by means of the inference method, aggregation operators, and fuzzy connectives. The fuzzy knowledge base contains the definition of fuzzy sets, which is stored in a fuzzy database, and a collection of fuzzy rules. The definition of fuzzy sets and the collection of fuzzy rules constitute the fuzzy rule base.

Fuzzy rules are defined by their antecedents and consequents, which relate an observed input state to a desired output. Most fuzzy systems employ the inference method proposed by Mamdani in which the consequent parts are defined by fuzzy sets [29]. A Mamdani-type fuzzy rule has the form:

$$\textbf{IF } x_1 \textbf{ is } A_{1j}(m_{1j},\sigma_{1j}) \textbf{ and } x_2 \textbf{ is } A_{2j}(m_{2j},\sigma_{2j}) \ \ldots\ \textbf{ and } x_n \textbf{ is } A_{nj}(m_{nj},\sigma_{nj})$$
$$\textbf{THEN } y' \textbf{ is } B_j(m_j,\sigma_j), \tag{1}$$

where m_{ij} and σ_{ij} represent a Gaussian membership function with mean and deviation, respectively, of the *i*th dimension and the *j*th rule node. The consequent B_j of the *j*th rule is aggregated into one fuzzy set for the output variable y'. The crisp output is obtained through defuzzification, which calculates the centroid of the output fuzzy set. In addition, the most common fuzzy inference method proposed by Mamdani and Takagi–Sugeno–Kang (TSK) introduced a modified inference scheme [36]. The first two parts of the fuzzy inference process, fuzzifying the inputs and applying the fuzzy operator, are exactly the same. A TSK type model employs different implication and aggregation methods than the standard Mamdani model. Instead of fuzzy sets being used, the conclusion part of a rule is a linear combination of the crisp inputs, as follows:

$$\textbf{IF } x_1 \textbf{ is } A_{1j}(m_{1j},\sigma_{1j}) \textbf{ and } x_2 \textbf{ is } A_{2j}(m_{2j},\sigma_{2j}) \ \ldots\ \textbf{ and } x_n \textbf{ is } A_{nj}(m_{nj},\sigma_{nj}),$$
$$\textbf{THEN } y' = w_{0j} + w_{1j}x_1 + \cdots + w_{nj}x_n, \tag{2}$$

where m_{ij} and σ_{ij} represent a Gaussian membership function with mean and deviation, respectively, of the *i*th dimension and the *j*th rule node. Since the consequent of a rule is crisp, the defuzzification step becomes obsolete in the TSK inference scheme. Instead, the model output is computed as the weighted average of the crisp rule outputs. This computation is less expensive than calculating the center of gravity.

In this paper, we adopt a TSK-type neuro-fuzzy network with ISPSO to solve the prediction and skin color detection problems. The structure of a TSK-type neuro-fuzzy network is shown in Fig. 1, where n and R are, respectively, the number of input dimensions and the number of rules. It is a five-layer network structure. We shall describe the operation functions of the nodes in each layer. In the following description, $u^{(l)}$ denotes the output of a node in the lth layer.

Layer 1 (*Input node*). No computation is done in this layer. The node only transmits input values to layer 2 directly

$$u_i^{(1)} = x_i. \tag{3}$$

Layer 2 (*Membership function node*). Nodes in this layer correspond to one linguistic label of the input variables in layer 1; that is, the membership value specifying the degree to which an input value belongs to a fuzzy set is calculated in layer 2. For an external input x_i, the following Gaussian membership function is used:

$$u_{ij}^{(2)} = \exp\left(-\frac{[u_i^{(1)} - m_{ij}]^2}{\sigma_{ij}^2}\right), \tag{4}$$

where m_{ij} and σ_{ij} are, respectively, the mean and the deviation of a Gaussian membership function of the *j*th term of the *i*th input variable x_i.

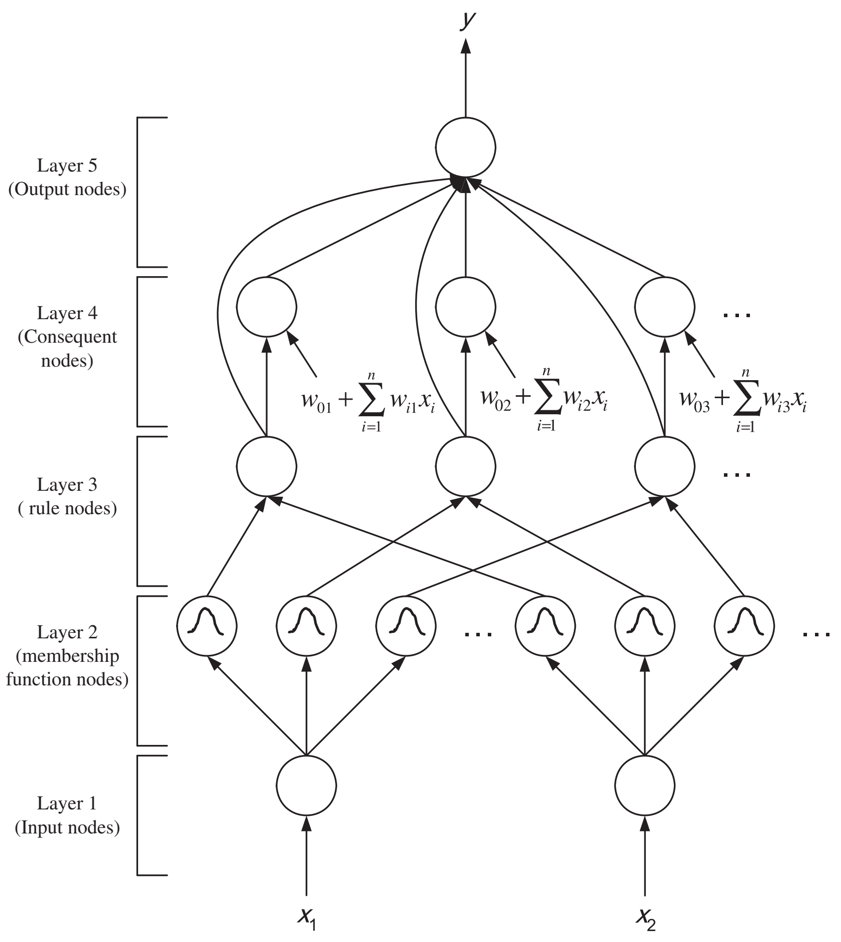
Layer 3 (*Rule node*). The output of each node in this layer is determined by the fuzzy AND operation. Here, the product operation is utilized to determine the firing strength of each rule. The function of each rule is

$$u_j^{(3)} = \prod_i u_{ij}^{(2)}. \tag{5}$$

Layer 4 (*Consequent node*). Nodes in this layer are called the consequent nodes. The input to a node in layer 4 is the output delivered from layer 3, and the other inputs are the input variables from layer 1, as depicted in Fig. 1. For this kind of node, we have

$$u_j^{(4)} = u_j^{(3)}\left(w_{0j} + \sum_{i=1}^{n} w_{ij}x_i\right), \tag{6}$$

where the summation is over all the inputs, and w_{ij} are the corresponding parameters of the consequent part.



y
Layer 5
(Output nodes)
Layer 4
(Consequent nodes)
$w_{01}+\sum_{i=1}^{n} w_{i1}x_i$
$w_{02}+\sum_{i=1}^{n} w_{i2}x_i$
$w_{03}+\sum_{i=1}^{n} w_{i3}x_i$
Layer 3
(rule nodes)
Layer 2
(membership function nodes)
Layer 1
(Input nodes)
x_1
x_2

Fig. 1. The structure of the TSK-type neuro-fuzzy network.

Layer 5 (*Output node*). Each node in this layer corresponds to a single output variable. The node integrates all the actions recommended by layers 3 and 4 and acts as a defuzzifier with

$$y = u^{(5)} = \frac{\sum_{j=1}^{R} u_j^{(4)}}{\sum_{j=1}^{R} u_j^{(3)})} = \frac{\sum_{j=1}^{R} u_j^{(3)}\left(w_{0j} + \sum_{i=1}^{n} w_{ij}x_i\right)}{\sum_{j=1}^{R} u_j^{(3)}}, \tag{7}$$

where R is the number of fuzzy rules and n is the number of input variables.

3. The immune-based symbiotic particle swarm optimization (ISPSO)

The IA is a global search algorithm and is based on the characteristics of the biological immune system. It is an adaptive, distributed, and parallel intelligent system that has the capability to control a complex system. The immune system is aimed at protecting living bodies from the invasion of various foreign substances such as viruses, bacteria, and other parasites (called antigens), and eliminating debris and malfunctioning cells. When an animal is exposed to an antigen, some subpopulation of B cells responds by producing antibodies. Antibodies are molecules attached primarily to the surface of B cells whose aim is to recognize and bind to antigens. B cells, in addition to proliferating and differentiating into plasma cells, can differentiate into long-lived B memory cells. Memory cells circulate through the blood, lymph, and tissues and, when exposed to a second antigenic stimulus, commence to differentiate into plasma cells capable of producing high-affinity antibodies, preselected for the specific antigen that had stimulated the primary response. To perform this function, the immune system has to be able to recognize all cells (or molecules) within

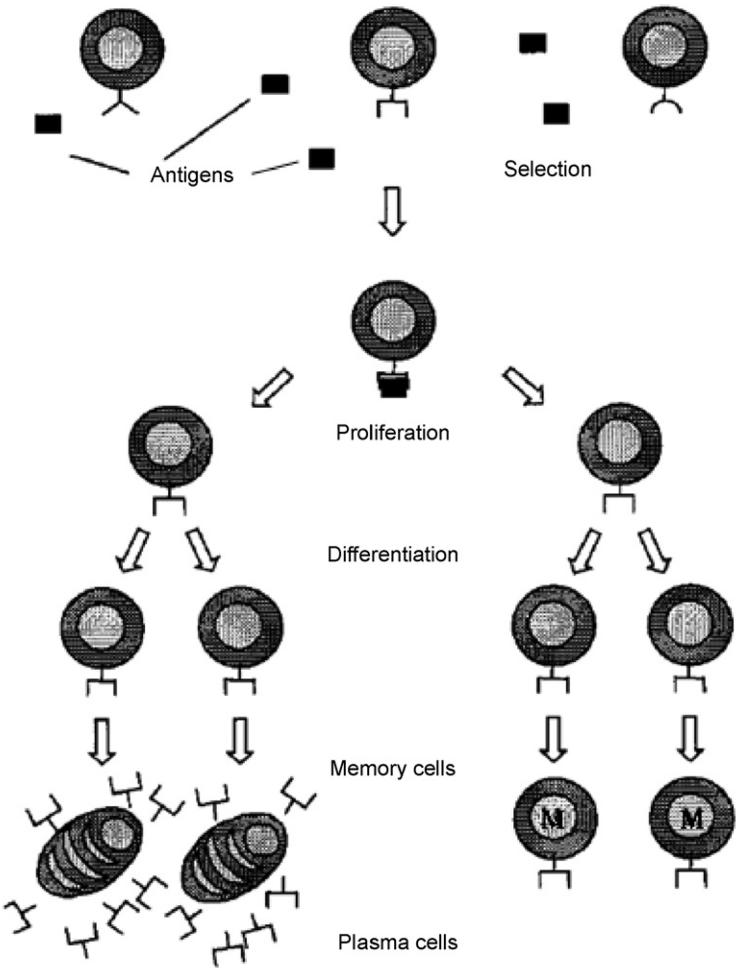


Fig. 2. Clonal selection principle.

the body and categorize these cells as self or non-self. Fig. 2 depicts a model representing the relationship among components in an immune system [4].

Analogous to the biological immune system, the proposed algorithm has the capability of seeking feasible solutions while maintaining diversity. The proposed ISPSO embeds the symbiotic evolution scheme in an IA and utilizes PSO to improve the mutation mechanism to perform parameter learning.

The PSO algorithm, proposed by Kennedy and Eberhart [19], has proved to be very effective for solving global optimization. It is not only a recently invented high-performance optimizer that is very easy to understand and implement, but it also requires little computational bookkeeping and, generally, only a few lines of code. Recently, an adaptation of PSO specifically designed for biomedical image registration was proposed by Wachowiak et al. [39]. In order to avoid trapping in a local optimal solution and to ensure the search capability of a near global optimal solution, mutation plays an important role in ISPSO. Therefore we employed the advantages of PSO to improve the mutation mechanism.

However, the parameters (ω, r_1 and r_2) of the traditional PSO are the key factors that affect convergence [37,32,16]. In fact, parameters of the traditional PSO cannot ensure the optimization's ergodicity entirely in phase space because they are absolutely random. Therefore, we use a method that introduces chaotic mapping with certainty, ergodicity and the stochastic property into PSO to improve global convergence.

In addition to modifying the PSO, we improve the IA structure. In a normal evolution algorithm, a single individual is responsible for the overall performance, with an affinity value assigned to that individual according to its performance. The traditional IA uses each antibody to represent a population as a full solution to a problem. Symbiotic evolution

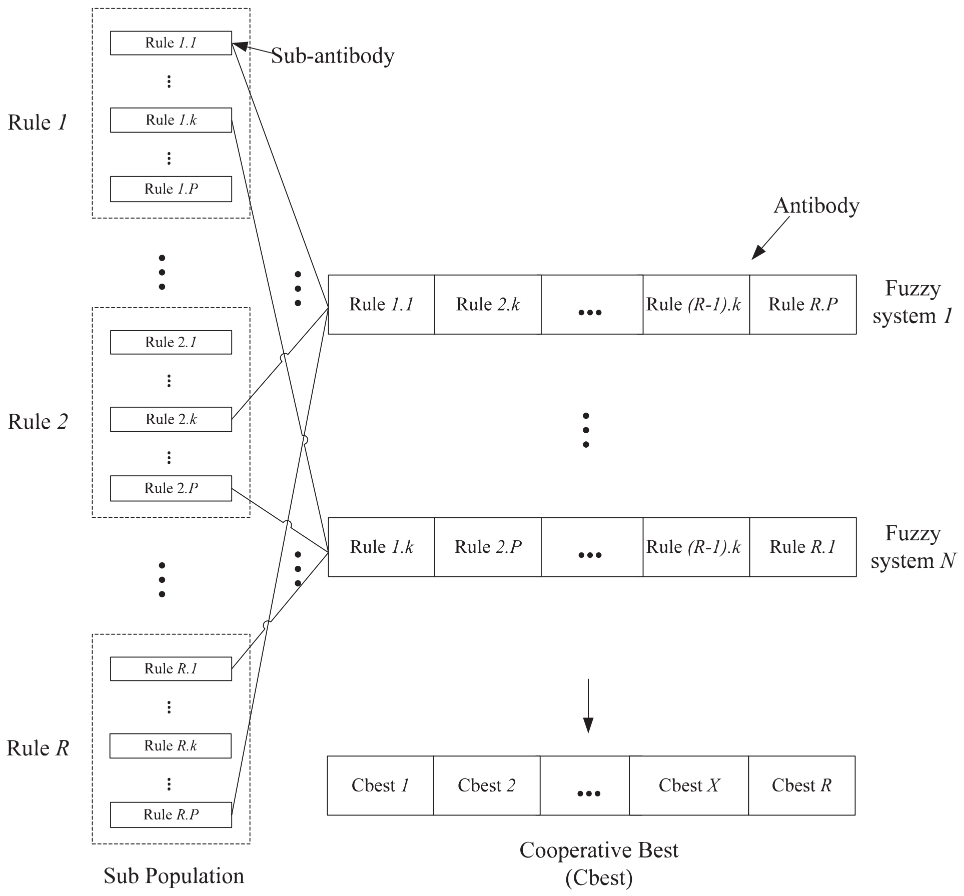


Fig. 3. A representation of a fuzzy system using ISPSO.

assumes that each individual (sub-antibody) in a population represents only a partial solution to a problem. Complex solutions combine several individuals in the population. The goal of each individual is to form a partial solution that can be combined with other partial solutions currently in the population to build an effective full solution. The structure of the antibodies in the symbiotic evolution is shown in Fig. 3. In this figure, we will reserve the best parameters of fuzzy system (i.e., the best sub-antibodies) by the cooperative best (Cbest).

As shown in [16,31], partial solutions can be characterized as *specializations*. The specialization property ensures diversity and prevents a population from converging to a suboptimal solution. A single partial solution cannot occupy a population since there must be other specializations present.

A detailed ISPSO of the TSK-type neuro-fuzzy network is presented in Fig. 4. The whole learning process is described step-by-step below.

3.1. Initial population production

The coding step is concerned with the membership functions and the corresponding parameters of the consequent part of a fuzzy rule that represent sub-antibodies suitable for ISPSO. We will discuss this first. The initialization step assigns the population values before the evolution process begins. This step in ISPSO codes a rule of a TSK-type neuro-fuzzy network into a sub-antibody. Fig. 5 shows an example of a rule of the TSK-type neuro-fuzzy network coded into a sub-antibody, where *i* and *j* represent the *i*th dimension and the *j*th rule, respectively. In this paper, a Gaussian membership function is used with variables representing the mean and deviation of the membership function. Each fuzzy rule in Fig. 1 has the form in Eq. (2), where m_{ij} and σ_{if} represent a Gaussian membership function with mean and deviation of the *i*th dimension and *j*th rule node.

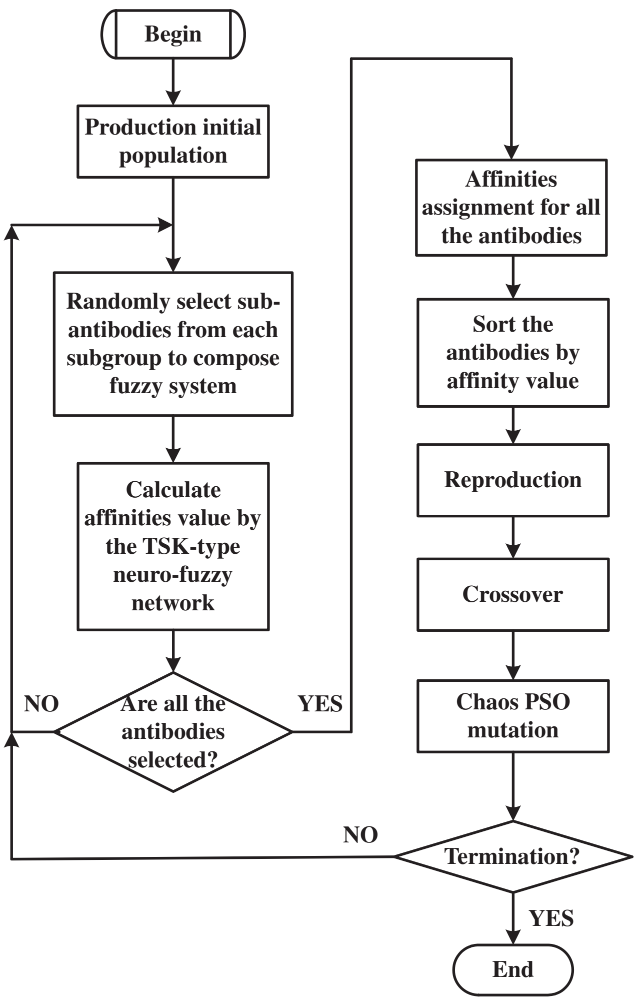

Begin
Production initial population
Randomly select sub-antibodies from each subgroup to compose fuzzy system
Calculate affinities value by the TSK-type neuro-fuzzy network
Are all the antibodies selected?
NO
YES
Affinities assignment for all the antibodies
Sort the antibodies by affinity value
Reproduction
Crossover
Chaos PSO mutation
Termination?
NO
YES
End

Fig. 4. Flowchart of the proposed ISPSO designs method.

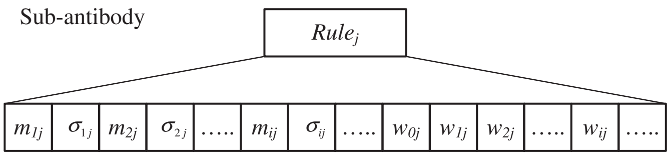

Sub-antibody
$Rule_j$
m_{1j} σ_{1j} m_{2j} σ_{2j} ….. m_{ij} σ_{ij} ….. w_{0j} w_{1j} w_{2j} ….. w_{ij} …..

Fig. 5. Coding a fuzzy rule into a sub-antibody in the ISPSO method.

The number of fuzzy rules and approximate estimates of the corresponding parameters, such as means and variances, describing the fuzzy term sets in the precondition parts need to be extracted from given input–output pairs. Thus, the choice of clustering technique in neuro-fuzzy networks is an important consideration. This is due to the use of partition-based clustering techniques, such as fuzzy *C*-means (FCM) [12], possibilistic *C*-means (PCM) [21], linear vector quantization (LVQ) [11], fuzzy Kohonen partitioning (FKP), and pseudo FKP [2], to perform cluster analysis. However, such clustering techniques require prior knowledge such as the number of clusters present in a data set. To solve the above problem, online-based cluster techniques were proposed [18,38]. But there still is a problem with these methods; namely, the clustering methods [18,38] only consider the total variations of the mean and variance in all dimensions of the input data. This is because the cluster numbers increase quickly.

In this paper, we use the self-clustering algorithm (SCA) [28,26] to partition the input space to create fuzzy rules (i.e., generate first initial sub-antibody). It is an online and distance-based connectionist clustering algorithm, which is unlike the traditional clustering techniques [12,21,11,2,18,38]. The tendency of the traditional clustering techniques is to consider the total variations in all dimensions of the input data that will cause clusters to extend too fast. According to the above-mentioned problems, the SCA method considers the variation of each dimension for the input data. The online SCA is a fast, one-pass algorithm that dynamically estimates the number of clusters in a set of data and finds the current centers of clusters in the input data space. In any cluster, the maximum distance between an example point and the cluster center is less than a threshold value. This clustering algorithm sets clustering parameters and affects the number of clusters to be estimated.

In the clustering process, the data examples come from a data stream. The clustering process starts with an empty set of clusters. The clusters will be updated and changed depending on the position of the current example in the input space. In the last step, the SCA generates the first initial sub-antibody of each rule utilizing a real variable string, and the other sub-antibodies of a population of each rule are generated to add a random value by the first sub-antibody.

3.2. Affinity value assignment step

The affinity value of each sub-antibody is computed by summing up the affinity values of all the feasible combinations of that rule with all other randomly selected rules and then dividing the sum by the total number of combinations. The details for assigning the affinity value are described step-by-step as follows:

Step 1: Randomly select R fuzzy rules (sub-antibodies) from each group of rules with size ps to build a fuzzy system.

Step 2: Evaluate every fuzzy system that is generated from step 1 to obtain an affinity value. The affinity value is designed according to the follow formulation:

$$\textit{Affinity value} = \frac{1}{1+\sqrt{\frac{1}{N_t}\sum_{k=1}^{N_t}(y_k - y_k^d)^2}}, \tag{8}$$

where y_k represents the kth model output, y_k^d represents the desired output, and N_t represents the number of the training data.

Step 3: Divide the affinity value by R and accumulate the divided affinity value to the selected R rules with their affinity value records that were initially set to zero.

Step 4: Repeat the above steps until each rule (sub-antibody) in each group of rules has been selected a sufficient number of times. Record the number of fuzzy systems in which each sub-antibody has participated.

Step 5: Divide the accumulated affinity value of each sub-antibody by the number of times it has been selected.

Step 6: Sort these sub-antibodies in each group of rules according to decreasing affinity values.

3.3. Reproduction step

Reproduction is a process in which individual strings are copied according to their affinity values. In this thesis, we use the roulette-wheel selection method [9]—a simulated roulette is spun—for this reproduction process (see Fig. 6). In Fig. 6, the intermediate population is P', which is generated from identical copies of a sub-antibody sampled by spinning the roulette wheel a sufficient number of times. The better performing sub-antibody (of higher affinity value) in the top-half of the population advances to the next generation (see Fig. 7). The other half of the population is generated by the top-half of the parent generation to perform crossover, which is chosen by tournament selection and chaos PSO mutation operations. A flowchart of the reproduction, crossover, and mutation steps is shown in Fig. 7.

3.4. Crossover step

Reproduction directs the search toward the best existing sub-antibodies but does not create any new ones. In nature, an offspring has two parents and inherits genes from both. In this paper, we use tournament selection [9], which selects the top-half of the best performing sub-antibodies [16]. The process is that we randomly pick three sub-antibodies and select the sub-antibody with the best affinity value. This process is repeated until we have two different sub-antibodies

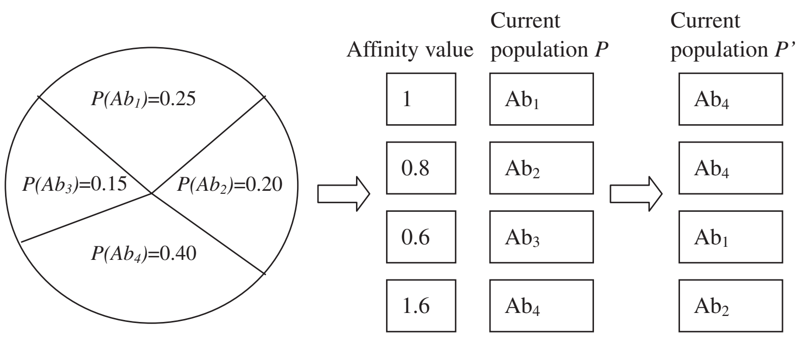


Fig. 6. The roulette-wheel selection.

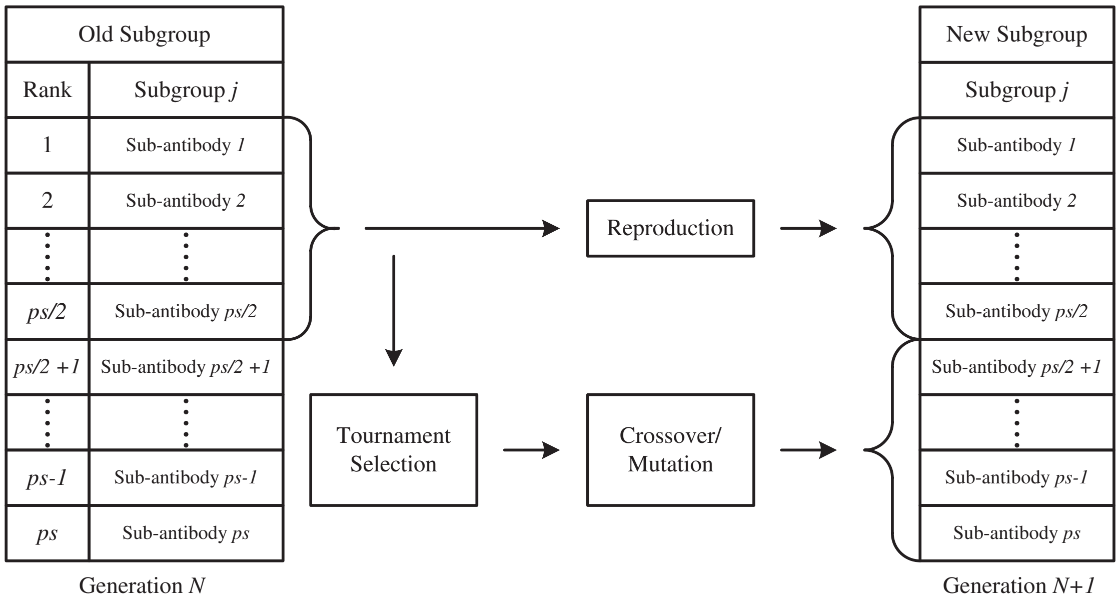


Fig. 7. A flowchart of the reproduction, crossover and mutation steps.

from the top-half of the best performing group of rules for the crossover step. The sub-antibodies are crossed and separated using a two-point crossover; that is, the new sub-antibodies are created by exchanging the site's values with the selected sites of the old sub-antibodies. After this operation, the sub-antibodies with poor performances are replaced by the newly produced offspring.

3.5. Mutation step

Although reproduction and crossover will produce many new strings, they do not introduce any new information to all the groups of rules at the site of the sub-antibodies. The parameters (ω, r_1 and r_2) of the traditional PSO are the key factors that affect convergence. In fact, parameters of the traditional PSO cannot ensure the optimization's ergodicity entirely in phase space because they are absolutely random. We use a method that introduces chaotic mapping with certainty, ergodicity and the stochastic property into PSO to improve global convergence. The parameters are modified by the following equation:

$$r_i(t+1) = \mu * r_i(t) * (1 - r_i(t)), \quad 0 \leqslant r(0) \leqslant 1, \quad i = 1, 2, 3, \tag{9}$$

where μ is the control parameter. Although the above equation is deterministic, it exhibits chaotic dynamics when $\mu = 4$ and $r(0) \notin \{0, 0.25, 0.5, 0.75, 1\}$. That is, it exhibits sensitive dependence on initial conditions, which is a basic characteristic of chaos.

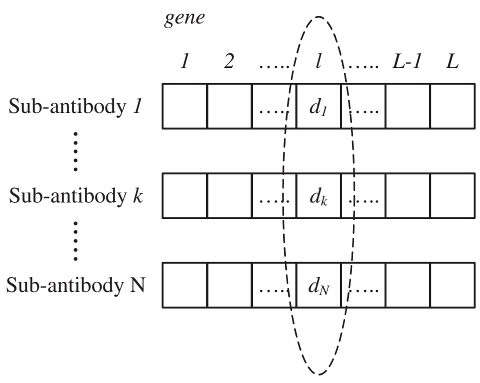


Fig. 8. The coding of a sub-antibody population.

On the other hand, the weight scale operator ω will affect the performance of the algorithms. A large scale contributes to the search for the global optimal solution in an expansive area. However, its precision is not that good for rough search. A small scale improves the precision of the optimal solution, but the algorithm may be trapped in a local optimization [6]. Thus, Shi and Eberhart [35] significantly improved the performance of the traditional PSO with a linearly varying weight over the generations. The improvement linearly varies from 0.9 at the beginning of the search to 0.4 at the end. Therefore, we use an adaptive weight scale, which can be expressed as follows:

$$\omega = \omega_{\max} - \frac{\omega_{\max} - \omega_{\min}}{iteration_{\max}} \times iteration_{\mathrm{N}}, \tag{10}$$

where $\omega_{\max}$ and $\omega_{\min}$ denote the maximum and minimum of ω, respectively, $iteration_{\max}$ denotes the total number of evolution generations, and $iteration_{\mathrm{N}}$ denotes the current number of evolution generations.

In recent years, many researchers [42,8] have proposed carrying out stability analysis of the dynamics of PSO in order to understand how it searches for a globally optimal solution and to devise a strategy to tune its parameters. In this paper, we propose the new velocity update function $\vec{v}_i(k+1)$ to improve performance. We hope it will accelerate every sub-antibodies in a direction. The velocity of the sub-antibody at the $(k+1)$th iteration is redefined in the following equation:

$$\begin{aligned}\vec{v}_i(k+1) = \omega \times \vec{v}_i(k) + \phi_1 \times rand() \times (Lbest - \vec{x}_i(k))\\ +\phi_2 \times rand() \times (Gbest - \vec{x}_i(k)) + \phi_3 \times rand() \times (Cbest - \vec{x}_i(k)),\end{aligned} \tag{11}$$

where ω is the coefficient of inertia, ϕ_1 is the cognitive study, ϕ_2 is the society study, and ϕ_3 is the group study.

3.6. Promotion and suppression of sub-antibodies

In order to eliminate antigens and keep diversity to a certain degree, we use information entropy theory to measure the diversity of sub-antibodies. If the affinity between two sub-antibodies is greater than the suppression threshold Th_{aff}, these two sub-antibodies are similar, and the sub-antibody of lower affinity value is eliminated. The sub-antibodies with high antigenic affinity are transformed into memory cells; others with low antigenic affinity are eliminated. Fig. 8 shows the IA composed of N sub-antibodies having L genes.

From information entropy theory, we get

$$IE_l(N) = \sum_{i=l}^{N} -P_{il} \log P_{il}, \tag{12}$$

where P_{il} is the probability that the *i*th allele comes out at the *l*th gene. The diversity of the genes is calculated using Eq. (12). The average entropy value $IE(N)$ of diversity can be also computed as follows:

$$IE(N) = \frac{1}{N}\sum_{l=1}^{L} IE_l(N), \tag{13}$$

where L is the size of the gene in a sub-antibody. There are two kinds of affinities in ISPSO. One explains the relationship between a sub-antibody and an antigen using Eq. (8). The other accounts for the degree of association between the *j*th sub-antibody and the *k*th sub-antibody and measures how similar these two sub-antibodies are. It can be calculated by using

$$Affinity\_Ab_{jk} = \frac{1}{1 + IE(2)}. \tag{14}$$

4. Simulation results

In order to demonstrate the performance of the proposed ISPSO approach, experiments were conducted on the prediction and skin color detection problems using two different data sets. The initial parameters before training are given in Table 1. All the programs were developed using Visual C + + 6.0, and each problem was simulated 50 times on a Pentium IV 3.2 GHz desktop computer.

4.1. Example 1: prediction of the chaotic time series

We used the Mackey–Glass equation to generate time series data. The chaotic time series is a non-linear, delay-differential equation whose dynamics exhibit chaotic behavior. The Mackey–Glass chaotic time series $x(t)$ in consideration here is generated from the following delay-differential equation:

$$\frac{dx(t)}{dt} = \frac{0.2x(t-\tau)}{1 + x^{10}(t-\tau)} - 0.1x(t). \tag{15}$$

Crowder [9] extracted 1000 input–output data pairs $\{x, y^d\}$ which consist of four past values of $x(t)$, i.e.,

$$[x(t-18), x(t-12), x(t-6), x(t); x(t+6)], \tag{16}$$

where $\tau = 17$ and $x(0) = 1.2$. There are four inputs to the TSK-type neuro-fuzzy network, corresponding to the values of $x(t)$, and one output representing the value $x(t + \Delta t)$, where Δt is a time prediction into the future. For the simulation, three cross-validation [1,33] groups of training and testing data are used. They are CV1, CV2, and CV3. The training and testing windows are labeled as such in Fig. 9.

In this example, through the SCA, the number of fuzzy rules we can get is seven. The number of sub-antibodies for a group of rules is set 20. Fifty experiments were run. Each experiment ran 500 epochs. In order to demonstrate the performance of the ISPSO method, we used ISPSO method to update the parameter ϕ with chaos PSO (i.e., Eq. (9)) and

Table 1
The initial parameters before training

Parameters	Value
Coding type	Real number
$[\omega_{\min},\omega_{\max}]$	[0.4,0.9]
ϕ_1, ϕ_2	1
ϕ_3	2
Weight value (ω)	0.6
Suppression threshold (Th_{aff})	0.9
Crossover rate	0.5
Mutation rate	0.2

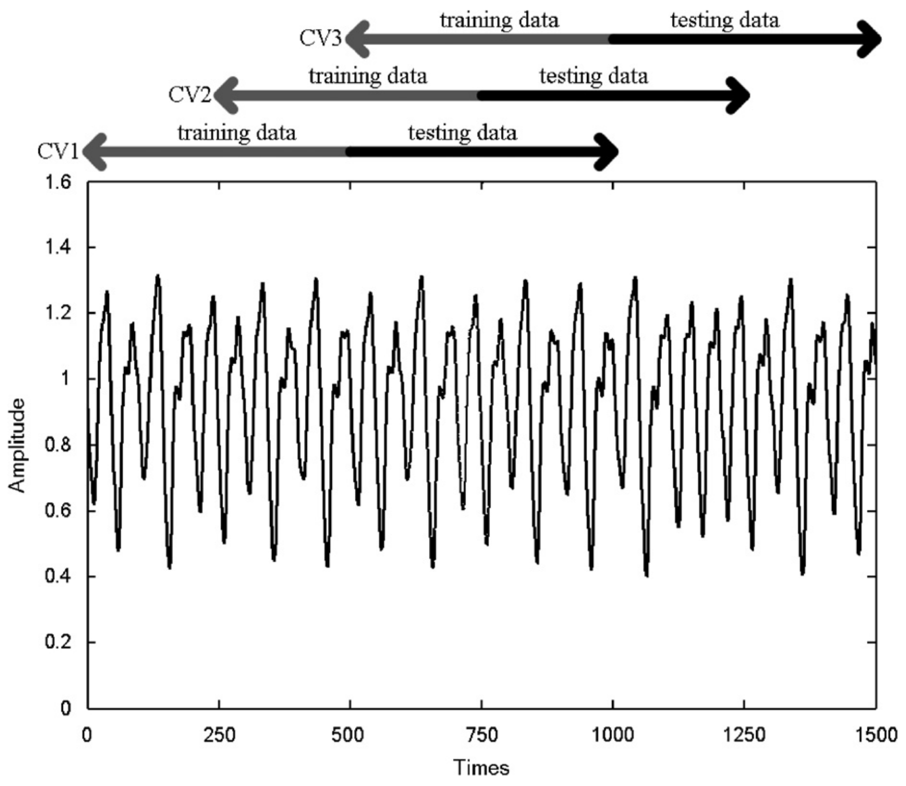


Fig. 9. Three cross-validation groups of training and testing data.

Table 2
Comparison results of various updated formula of ISPSO

		RMS error			
		CV1	CV2	CV3	Average
ISPSO	With chaos PSO and adaptive weight (ω) value	**0.0056**	**0.0043**	**0.0052**	**0.0051**
	With traditional PSO and adaptive weight (ω) value	0.007	0.0061	0.0066	0.0066
	With traditional PSO and fix weight (ω) value	0.0072	0.007	0.0068	0.007

adaptive weight (ω) value (i.e., Eq. (10)), to update the parameter ϕ of traditional PSO and adaptive weight (ω) value, and to update the parameter ϕ of traditional PSO and fix weight (ω) value, respectively. However, the comparison results are tabulated in Table 2.

In this example, we compared a TSK-type neuro-fuzzy network using the proposed ISPSO method, the PSO method [19], and the IA method [7], respectively. The prediction outputs of the chaotic time series are shown in Fig. 10(a). The prediction errors between the proposed model and the desired output are shown in Fig. 10(b). Figs. 10(c) and (d) show the prediction results and the prediction errors of the IA method [7]. Figs. 10(e) and (f) show the prediction results and the prediction errors of the PSO method [19]. The learning curves of the proposed ISPSO method, the IA method, and the PSO method are shown in Fig. 11. Furthermore, we also compared our proposed method with TGA [23], GFPE [20], GEFREX [34], and ANFIS [13]. The results of comparison with other existing methods are tabulated in Table 3. We find that the proposed method obtains better prediction results than other existing methods.

4.2. Example 2: skin color detection problem

Current human recognition methods, such as fingerprinting, retinal scanning, and voice detection have become mature technologies. The aforementioned detection processes generally require a cooperative subject, views from

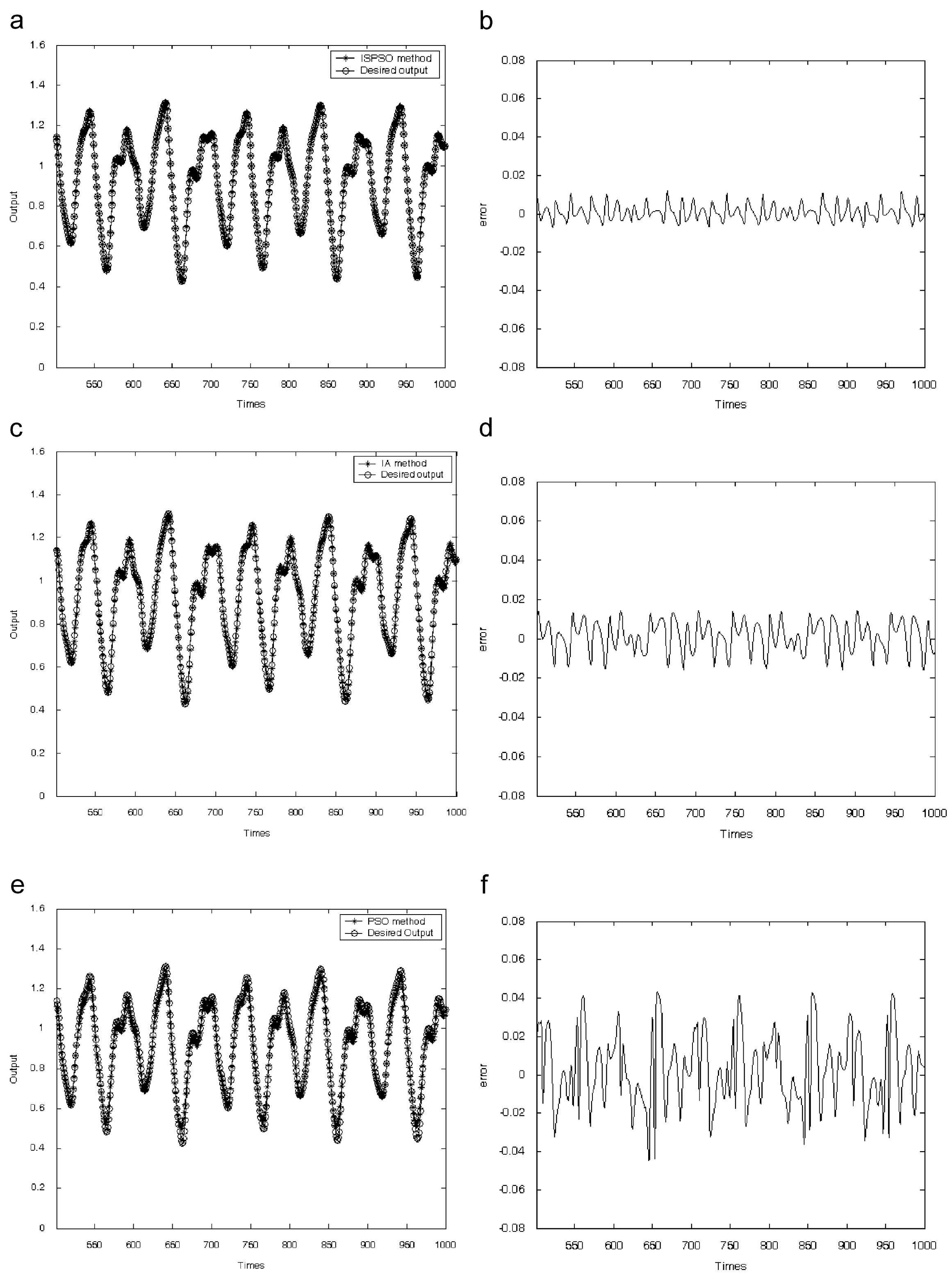


Fig. 10. (a) The prediction results of the ISPSO method in CV1. (b) The prediction errors of the ISPSO method. (c) The prediction results of the IA method [7] in CV1. (d) The prediction errors of the IA method [7]. (e) The prediction results of the PSO method [19] in CV1. (f) The prediction errors of the PSO method [19].

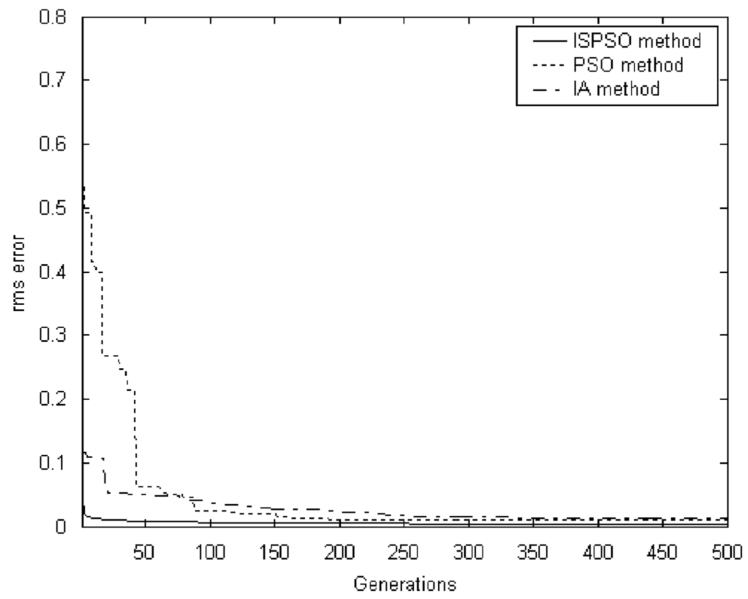


Fig. 11. The learning curves of the proposed ISPSO method, the IA method, and the PSO method.

Table 3
Performance comparison of various existing methods

Method	RMS error
ISPSO	0.0051
PSO [19]	0.039
IA [7]	0.034
TGA [23]	0.036
GFPE [20]	0.026
GEFREX [34]	0.0061
ANFIS [13]	0.07

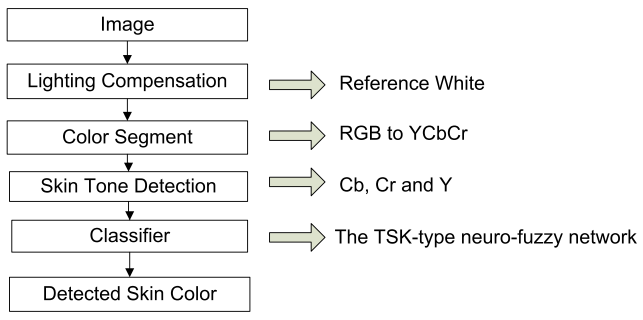


Fig. 12. A flowchart of the skin color detection system.

certain aspects, and physical contact with close subject proximity. Often people feel that these detection systems violate their privacy. Skin color detection is a non-contact detection method.

In skin color detection, users pass through detection areas, with a computer system scanning the face in a non-intrusive manner. Skin color detection plays an important role in applications such as face recognition and face image database management, and it is also a crucial step to detect faces in identification applications.

Table 4
Experimental results using cross-validation model

	CV1 (%)	CV2 (%)	CV3 (%)	Average (%)
RMS error (training)	98.89	97.63	95.72	97.41
RMS error (testing)	93.47	91.83	92.11	92.47

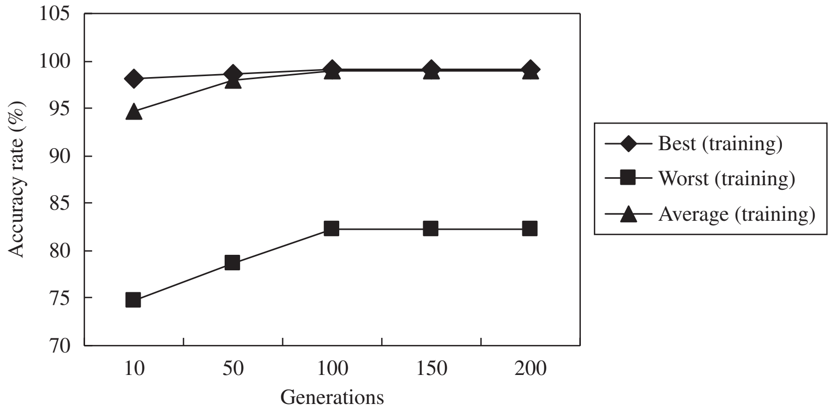


Fig. 13. The accuracy rate of training data with different generations for 50 runs.

Table 5
The best, worst, and average accuracy rate of training data with different generations for 50 runs

	Generations				
	10 (%)	50 (%)	**100** (%)	150 (%)	200 (%)
Best accuracy rate (training)	98.12	98.57	**99.12**	99.12	99.13
Worst accuracy rate (training)	74.80	78.67	**82.21**	82.23	82.24
Average accuracy rate (training)	94.66	97.91	**98.89**	98.89	98.90

Fig. 12 shows a flowchart of a skin color detection system. The detection process is as follows. We take a color image and perform lighting compensation to delete many lighting conditions caused by chrominance. Because the RGB color space cannot clearly segment the skin tone regions, we transform the RGB color space into the YCbCr color space. Then, using skin tone cluster as a characteristic in the YCbCr color space, we get the Y, Cb, and Cr information of the color image as the features into the classifier. After classifier computation, the output consists of skin or non-skin information used to detect the facial region.

We used the California Institute of Technology (CIT) face database on http://www.vision.caltech.edu/Image_Datasets/faces/. The database has 450 color images, the size of each being $320 * 240$ pixels, and contains 27 different people and a variety of lighting, backgrounds and facial expressions.

Three input dimensions (Y, Cb, and Cr) were used in this experiment. We chose 6000 training data and 6000 testing data. We used the CIT database to produce both the training data and the testing data. We chose 3000 skin and 3000 non-skin pixels as the training data in the color images. Also, we chose other 3000 skin and 3000 non-skin pixels as the testing set. The three cross-validation groups are denoted as CV1, CV2, and CV3, respectively. Each cross-validation group consists of training and test sets that are randomly generated. We set four rules constituting a TSK-type neuro-fuzzy network. The number of antibodies for a group of rules was set to 10. The experimental results are tabulated in Table 4 using cross-validation model. With the same initial condition, the accuracy rate of training data with different generations for 50 runs is shown in Fig. 13 and tabulated in Table 5. Therefore, we observe that 100 generations for training are enough.

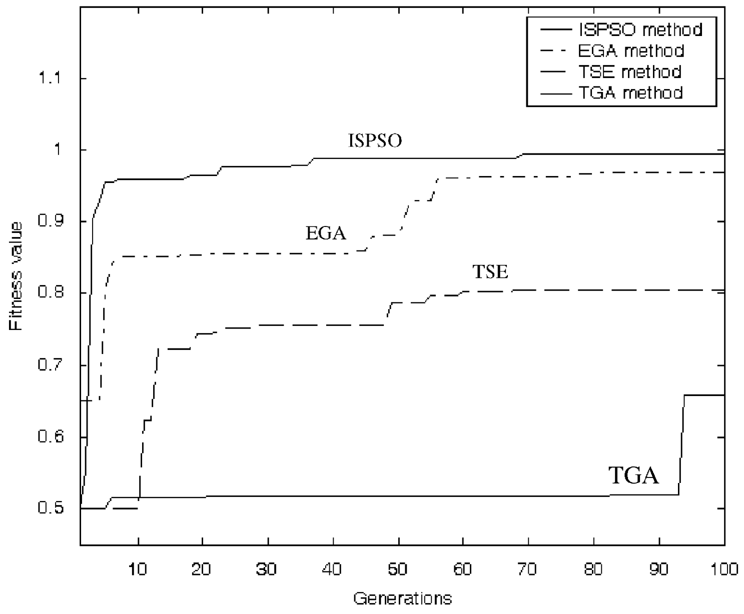


Fig. 14. The learning curves of the four methods using the CIT database.

Table 6
Performance comparison with various existing models from the CIT database

Method	**ISPSO**		EGA [25]		TSE [16]	
Training data	**6000**		6000		6000	
Best and worst accuracy rate (training)	**Best**	**Worst**	Best	Worst	Best	Worst
	99.12%	**81.97%**	96.65%	80.1%	83.73%	73.25%
Average training accuracy rate	**98.89%**		93.33%		79.05%	
Best and worst accuracy rate (testing)	**Best**	**Worst**	Best	Worst	Best	Worst
	99.38%	**89.93%**	95.43%	85.9%	79.78%	67.3%
Average testing accuracy rate	**93.47%**		90.19%		74.33%	
Generations	**100**		100		100	
Method	TGA [23]		PSO [19]		IA [7]	
Training data	6000		6000		6000	
Best and worst accuracy rate (training)	Best	Worst	Best	Worst	Best	Worst
	83.28%	61.38%	81.13%	65.27%	82.51%	68.71%
Average training accuracy rate	70.4%		71.33%		73.68%	
Best and worst accuracy rate (testing)	Best	Worst	Best	Worst	Best	Worst
	79.48%	60.23%	77.68%	63.71%	79.25%	65.4%
Average Testing accuracy rate	65.34%		67.01%		70.47%	
Generations	100		100		100	

We also compared our method with other methods. The total 100 generations for training are set. The learning curves are shown in Fig. 14. In Fig. 14, we find that the performance of the proposed ISPSO method is superior to the other methods. In this example, the performance of the ISPSO method is compared with the PSO method [19], and the IA method [7], the efficient genetic algorithms (EGA) [25], traditional symbiotic evolution (TSE) [16], and traditional genetic algorithm (TGA) [23]. The EGA method [25] exploited the sequential-search-based efficient generation method to generate an initial population and determines the most efficient mutation points. The TSE method [16] was based upon symbiotic evolution which complements the local mapping property of a fuzzy rule. The TGA method [23] was proposed for automatically designing complete fuzzy systems using a GA and a penalty strategy to

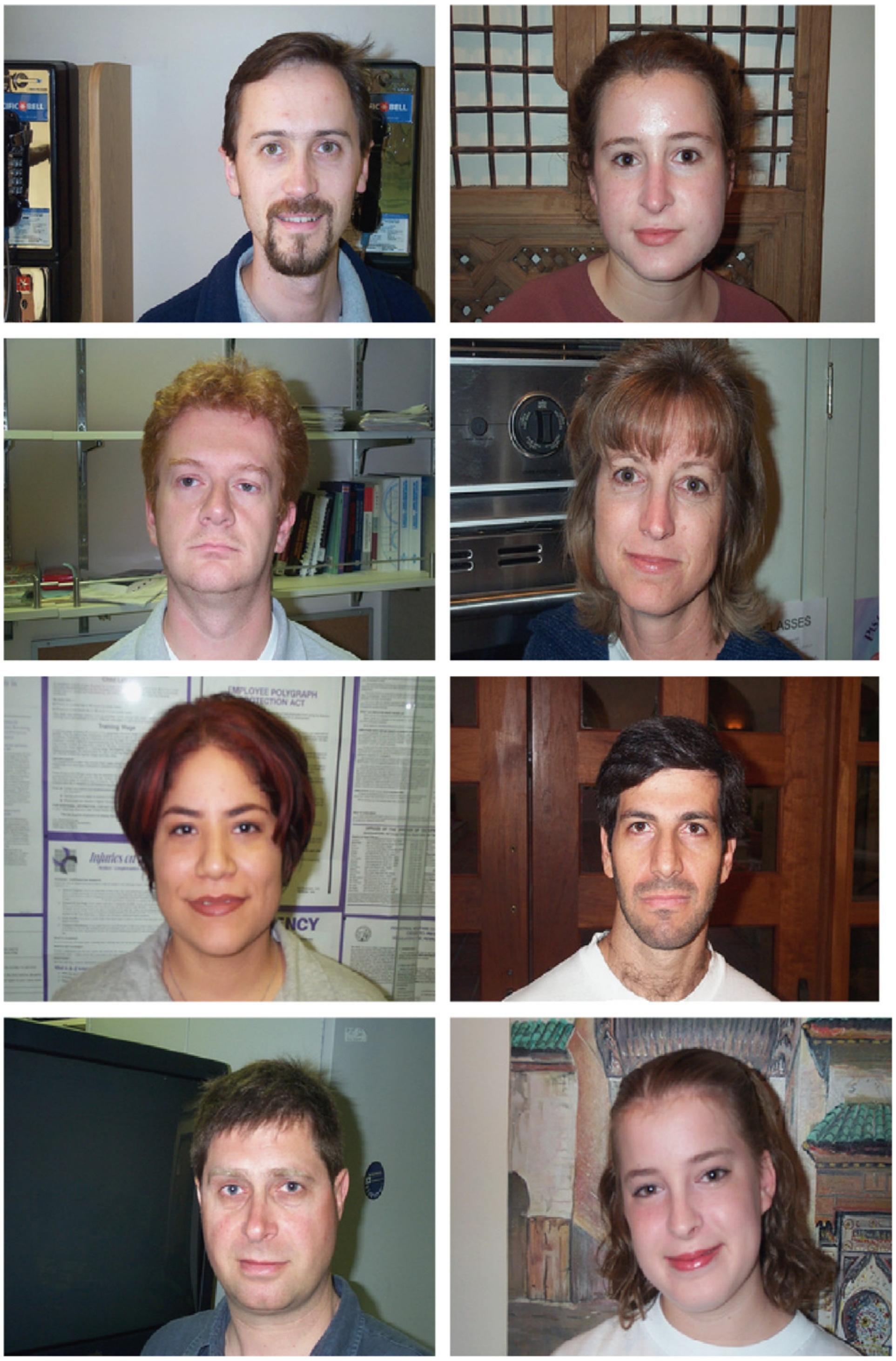


Fig. 15. Original face database from California Institute of Technology (CIT).

determine membership function shape and position, number of fuzzy rules, and consequent parameters simultaneously. The comparison items include the training and testing accuracy rates (e.g. best, worst and average situations), training and testing errors, and cost. The comparison results with various existing models are tabulated in Table 6.

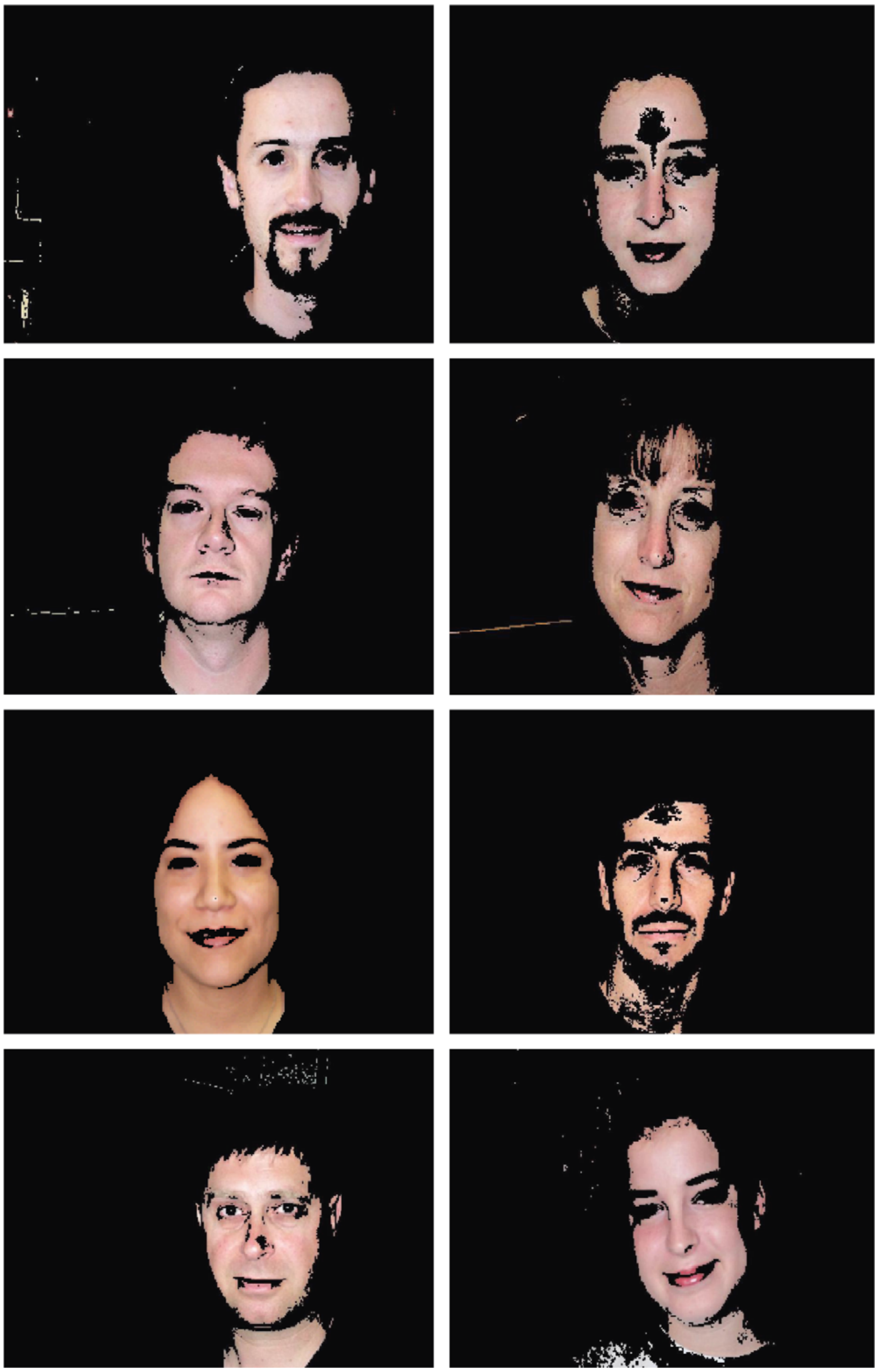


Fig. 16. Results of skin color detection with three dimension input (Y, Cb, Cr).

The CIT face database consists of complex backgrounds and diverse lighting. Hence, from the comparison data listed in Table 6, the average of the test accuracy rate is 90.19% for EGA, 74.33% for TSE, 65.34% for TGA, 67.01% for PSO, 70.47% for IA and 93.47% for the proposed ISPSO. This demonstrates that the CIT database is more complex and does not lead to a decrease in the accuracy rate. The proposed ISPSO method maintains a superior accuracy rate. The color images from the CIT database are shown in Fig. 15. A well-trained network can generate binary outputs (1/0 for skin/non-skin) to detect a facial region. Fig. 16 shows that our approach accurately determines a facial region.

5. Conclusion

This paper proposed a new learning algorithm that can be used to design TSK-type neuro-fuzzy networks. We proposed the immune-based symbiotic particle swarm optimization (ISPSO) for use in TSK-type neuro-fuzzy networks for solving the prediction and skin color detection problems. In order to avoid trapping in a local optimal solution and to ensure the search capability of a near global optimal solution, mutation plays an important role in ISPSO. The proposed ISPSO embeds the symbiotic evolution scheme in an IA and utilizes PSO to improve the mutation mechanism. The advantages of the proposed method are summarized as follows: (1) we employed the advantages of PSO to improve the mutation mechanism; (2) the complicated problems with symbiotic evolution can be better solved; (3) there is more of a likelihood to get a global optimum compared to heuristic methods; (4) the experimental results show that our method achieves faster learning and a higher design accuracy rate in many problems. The proposed learning algorithm has great potential to be further applied to many ill-conditioned problems.

References

[1] S. Amari, N. Murata, K.R. Muller, M. Finke, H.H. Yang, Asymptotic statistical theory of overtraining and cross-validation, IEEE Trans. Neural Networks 8 (5) (1997) 985–996.

[2] K.K. Ang, C. Quek, M. Pasquier, POPFNN-CRIS(S): pseudo outer product based fuzzy neural network using the compositional rule of inference and singleton fuzzifier, IEEE Trans. Systems Man Cybernet. 33 (6) (2003) 838–849.

[3] D.W. Boeringer, D.H. Werner, Particle swarm optimization versus genetic algorithms for phased array synthesis, IEEE Trans. Antennas and Propagation 52 (3) (2004) 771–779.

[4] L.N. de Castro, F.J. Von Zuben, Learning and optimization using the clonal selection principle, IEEE Trans. Evolutionary Comput. 6 (3) (2002) 239–251.

[5] L. Chen, D.H. Cooley, J. Zhang, Possibility-based fuzzy neural networks and their application to image processing, IEEE Trans. Systems Man Cybernet. 29 (1) (1999) 119–126.

[6] J. Chuanwen, E. Bompard, A self-adaptive chaotic particle swarm algorithm for short term hydroelectric system scheduling in deregulated environment, Energy Conversion Manage. 46 (17) (2005) 2689–2696.

[7] J.S. Chun, M.K. Kim, H.K. Jung, Shape optimization of electromagnetic devices using immune algorithm, IEEE Trans. Magnetics 33 (2) (1997) 1876–1879.

[8] M. Clerc, J. Kennedy, The particle swarm-explosion, stability, and convergence in a multidimensional complex space, IEEE Trans. Evolutionary Comput. 6 (1) (2002) 58–73.

[9] O. Cordon, F. Herrera, F. Hoffmann, L. Magdalena, Genetic Fuzzy Systems Evolutionary Tuning and Learning of Fuzzy Knowledge Bases, Advances in Fuzzy Systems—Applications and Theory, World Scientific Publishing, NJ, 2001.

[10] R.S. Crowder, Predicting the Mackey–Glass time series with cascade-correlation learning, in: Proc. Connectionist Models Summer School, 1990, pp. 117–123.

[11] J. He, L. Liu, G. Palm, Speaker identification using hybrid LVQ-SLP networks, in: Proc. IEEE Internat. Conf. Neural Networks, 1995, pp. 2052–2055.

[12] M.C. Hung, D.L. Yang, The efficient fuzzy *c*-means clustering technique, in: Proc. IEEE Internat. Conf. Data Mining, 2001, pp. 225–232.

[13] J.-S.R. Jang, ANFIS: adaptive-network-based fuzzy inference system, IEEE Trans. Systems Man Cybernet. 23 (3) (1993) 665–685.

[14] Q. Jishuang, W. Chao, W. Zhengzhi, Structure-context based fuzzy neural network approach for automatic target detection, in: Proc. IEEE Internat. Geoscience and Remote Sensing Symp., 2003, pp. 767–769.

[15] C.F. Juang, C.T. Lin, An online self-constructing neural fuzzy inference network and its applications, IEEE Trans. Fuzzy Systems 6 (1) (1998) 12–31.

[16] C.F. Juang, J.Y. Lin, C.T. Lin, Genetic reinforcement learning through symbiotic evolution for fuzzy controller design, IEEE Trans. Systems Man Cybernet. 30 (2) (2000) 209–302.

[17] A. Kalinli, N. Karabogab, Artificial immune algorithm for IIR filter design, Eng. Appl. Artif. Intell. 18 (8) (2005) 919–929.

[18] N.K. Kasabov, Q. Song, DENFIS: dynamic evolving neural-fuzzy inference system and its application for time-series prediction, IEEE Trans. Fuzzy Systems 10 (2) (2002) 144–154.

[19] J. Kennedy, R. Eberhart, Particle swarm optimization, in: Proc. IEEE Internat. Conf. on Neural Networks, 1995, pp. 1942–1948.

[20] D. Kim, C. Kim, Forecasting time series with genetic fuzzy predictor ensemble, IEEE Trans. Fuzzy Systems 5 (4) (1997) 523–535.

[21] R. Krishnapuram, J.M. Keller, The possibilistic *c*-means algorithm: insights and recommendations, IEEE Trans. Fuzzy Systems 4 (3) (1996) 385–393.

[22] B. Kusumoputo, P. Irwanto, W. Jatmiko, Optimization of fuzzy-neural structure through genetic algorithm and its application in artificial odor recognitions, in: Asia-Pacific Conf. on Circuits and Systems, Vol. 2, 2002, pp. 47–51.
[23] M.A. Lee, H. Takagi, Integrating design stages of fuzzy systems using genetic algorithms, in: Proc. IEEE Internat. Conf. on Fuzzy Systems, 1993, pp. 612–617.
[24] G.C. Liao, T.P. Tsao, Application embedded chaos search immune genetic algorithm for short-term unit commitment, Electric Power Systems Res. 71 (2) (2004) 135–144.
[25] C.J. Lin, H.C. Chuang, Y.J. Xu, Face detection in color images using efficient genetic algorithms, Optical Eng. (2007), accepted publication.
[26] C.J. Lin, I.F. Chung, C.H. Chen, An entropy-based quantum neuro-fuzzy inference system for classification applications, Neurocomputing 70 (13–15) (2007) 2502–2516.
[27] C.J. Lin, C.T. Lin, An ART-based fuzzy adaptive learning control network, IEEE Trans. Fuzzy Systems 5 (4) (1997) 477–496.
[28] C.J. Lin, Y.J. Xu, A self-adaptive neural fuzzy network with group-based symbiotic evolution and its prediction applications, Fuzzy Sets and Systems 157 (8) (2006) 1036–1056.
[29] C.T. Lin, C.S.G. Lee, Neural Fuzzy Systems: a Neuro-Fuzzy Synergism to Intelligent System, Prentice-Hall, Englewood Cliffs, NJ, 1996.
[30] F.J. Lin, C.H. Lin, P.H. Shen, Self-constructing fuzzy neural network speed controller for permanent-magnet synchronous motor drive, IEEE Trans. Fuzzy Systems 9 (5) (2001) 751–759.
[31] D.E. Moriarty, R. Miikkulainen, Efficient reinforcement learning through symbiotic evolution, Mach. Learn. 22 (1996) 11–32.
[32] S. Naka, T. Genji, T. Yura, Y. Fukuyama, A hybrid particle swarm optimization for distribution state estimation, IEEE Trans. Power Systems 18 (1) (2003) 60–68.
[33] M.N. Nguyen, D. Shi, C. Quek, FCMAC-BYY: fuzzy CMAC using Bayesian Ying–Yang learning, IEEE Trans. Systems Man Cybernet. 36 (5) (2006) 1180–1190.
[34] M. Russo, Genetic fuzzy learning, IEEE Trans. Evolutionary Comput. 4 (3) (2000) 259–273.
[35] Y. Shi, R.C. Eberhart, Empirical study of particle swarm optimization, in: Proc. IEEE Congress on Evolutionary Computation, 1999, pp. 6–9.
[36] T. Takagi, M. Sugeno, Fuzzy identification of systems and its applications to modeling and control, IEEE Trans. Systems Man Cybernet. 1 (1) (1985) 116–132.
[37] I.C. Trelea, The particle swarm optimization algorithm: convergence analysis and parameter selection, Inform. Process. Lett. 85 (6) (2003) 317–325.
[38] W.L. Tung, C. Quek, GenSoFNN: a generic self-organizing fuzzy neural network, IEEE Trans. Neural Networks 13 (5) (2002) 1075–1086.
[39] M.P. Wachowiak, R. Smolikova, Y. Zheng, J.M. Zurada, A.S. Elmaghraby, An approach to multimodal biomedical image registration utilizing particle swarm optimization, IEEE Trans. Evolutionary Comput. 8 (3) (2004) 289–301.
[40] L.X. Wang, J.M. Mendel, Generating fuzzy rules by learning from examples, IEEE Trans. Systems Man Cybernet. 22 (6) (1992) 1414–1427.
[41] X. Wen, A. Song, An immune evolutionary algorithm for sphericity error evaluation, Internat. J. Mach. Tools Manufacture 44 (2004) 1077–1084.
[42] K. Yasuda, A. Ide, N. Iwasaki, Adaptive particle swarm optimization, in: Proc. IEEE Internat. Conf. on Systems Man Cybernetics, 2003, pp. 1554–1559.
[43] Y. Zhou, D. Zheng, Z.L. Qiu, G.Y. Dong, The application of RBF networks based on artificial immune algorithm in the performance prediction of steel bars, in: Proc. Third Internat. Conf. on Machine Learning and Cybernetics, 2004, pp. 3439–3443.