The Journal of Systems and Software 85 (2012) 1333–1345

Contents lists available at SciVerse ScienceDirect

The Journal of Systems and Software

journal homepage: www.elsevier.com/locate/jss

An improved swarm optimized functional link artificial neural network (ISO-FLANN) for classification

Satchidananda Dehuri[a,*], Rahul Roy[b], Sung-Bae Cho[c], Ashish Ghosh[d]

[a] Department of Information and Communication Technology, Fakir Mohan University, Vyasa Vihar, Balasore 756 019, India
[b] Machine Intelligence Unit, Indian Statistical Unit, 203 B. T. Road, Kolkata 700 108, India
[c] Soft Computing Laboratory, Department of Computer Science, Yonsei University, 262 Seongsanno, Seodaemun-gu, Seoul 120-749, South Korea
[d] Center for Soft Computing Research, Indian Statistical Institute, 203 B. T. Road, Kolkata 700 108, India

ARTICLE INFO

Article history:
Received 13 October 2010
Received in revised form 11 January 2012
Accepted 13 January 2012
Available online 30 January 2012

Keywords:
Classification
Data mining
Functional link artificial neural networks
Multi-layer perception
Particle swarm optimization
Improved particle swarm optimization
SVM
FSN

ABSTRACT

Multilayer perceptron (MLP) (trained with back propagation learning algorithm) takes large computational time. The complexity of the network increases as the number of layers and number of nodes in layers increases. Further, it is also very difficult to decide the number of nodes in a layer and the number of layers in the network required for solving a problem a priori. In this paper an improved particle swarm optimization (IPSO) is used to train the functional link artificial neural network (FLANN) for classification and we name it ISO-FLANN. In contrast to MLP, FLANN has less architectural complexity, easier to train, and more insight may be gained in the classification problem. Further, we rely on global classification capabilities of IPSO to explore the entire weight space, which is plagued by a host of local optima. Using the functionally expanded features; FLANN overcomes the non-linear nature of problems. We believe that the combined efforts of FLANN and IPSO (IPSO + FLANN = ISO − FLANN) by harnessing their best attributes can give rise to a robust classifier. An extensive simulation study is presented to show the effectiveness of proposed classifier. Results are compared with MLP, support vector machine(SVM) with radial basis function (RBF) kernel, FLANN with gradiend descent learning and fuzzy swarm net (FSN).

1. Introduction

Classification is one the most challenging and frequently encountered decision making tasks of human activity. In classification, we are given a set of instances, called a training set, where each instances consists of several features or attributes. Attributes are either continues (coming from an ordered domain) or categorical (coming from an unordered domain). One of the attributes called the classifying attribute indicates the class to which each instance belongs. The aim of the classification problem is to build a model of classifying attribute based upon the other attributes. In other words, a classification problem occurs when an instance needs to be assigned into a pre-specified group or class based on a number of observed attributes related to that instance.

Most of the problem in marketing (Decker and Kroll, 2007), biological science (Pereira et al., 2009), industry and medicine (Marinakis et al., 2009) can be treated as classification problems. Examples include bankruptcy prediction, credit scoring, quality control, handwritten character recognition, and speech recognition. Additionally, stock market analysis may wish to characterize a group of stocks as buy, sell or hold; a cancer researcher may wish to categorize a list of tumors as benign or malignant; a mortgage analyst wish to categorize loans as good or bad. A major difficulty faced by such analyst is that the data to be classified can often be quite complex, with large number of records and features. The time and effort required to develop a model to solve accurately such classification problems can be significant.

There are number of classification methods (Dehuri et al., 2008; Hamamoto et al., 1997; Quinlan, 1993; Yager, 2006; Yung and Shaw, 1995; Zhang, 2000) developed by statistics, neural networks and machine learning researchers. However, the recent research activities in neural classification (Zhang, 2000) have established that neural networks are promising alternative to various conventional classification methods. The artificial neural networks (ANNs) are capable of generating complex mapping between input and the output space and thus they can form arbitrarily complex non-linear decision boundaries. Along the way, there are already several artificial neural networks, each utilizing a different form of learning or hybridization. For example, Kang and Brown (2008) has applied a model learning adaptive function neural network (ADFUNN) for classification combined with an unsupervised snap-drift without hidden neurons. Han et al. (2008) has proposed two modified

* Corresponding author. Tel.: +91 9668321 964.
E-mail addresses: satchi.lapa@gmail.com (S. Dehuri), link2rahulroy@gmail.com (R. Roy), sbcho@cs.yonsei.ac.kr (S.-B. Cho), ash@isical.ac.in (A. Ghosh).

0164-1212/$ – see front matter © 2012 Elsevier Inc. All rights reserved.
doi:10.1016/j.jss.2012.01.025

constrained learning algorithms by incorporating additional functional constraints into neural networks to obtain better generalization performance and faster convergence.

Pao (1989) and Pao et al. (1992) have given a direction that functional links neurons may be conveniently used for function approximation with faster convergence rate and lesser computational load than MLP. However they have not been applied to classification task. In Mishra and Dehuri (2007), we used FLANN with gradient descent method for classification task of data mining and achieved good results. Also in Mishra and Dehuri (2007) we suggest a different set of orthonormal basis function for feature expansion. Further, Dehuri and Cho (2010a,b) recently developed two FLANN based classifier combined with genetic algorithms (Goldberg, 1989) for enhancing the classification accuracy.

FLANN is basically a flat network with simple learning rule and without requiring hidden layers. The functional expansion effectively increases the dimensionality of the input vector and hence the hyper-planes generated by the FLANN provide greater discriminating capability of the input patterns. Although FLANN with gradient descent gives promising results, sometimes may be trapped in local optimal solutions. Moreover FLANN coupled with genetic algorithms may suffer with problems like heavy computation burdens, and large number of parameter tuning. We thus propose a swarm optimized functional link artificial neural network for classification (ISO-FLANN). The proposed method is a result of combination of best attributes of IPSO and FLANN. This method not only has gained an insight in the nature of the problem but also achieved good classification accuracy. The IPSO is improved by introducing two self adaptive mutations such as Gaussian and Cauchy to reduce the of getting stuck to local optima and an adaptive inertia weight. Although many types of neural network can be used for classification purpose (Zhang, 2000), we chose the multilayer perceptron (MLPs) as a bench mark method for comparison. Even though it has complex architecture and long training time, it is the most widely studied and used neural network for classification. Alongside, we chose support vector machine (SVM) (Hsu and Lin, 2002) with radial basis kernel and fuzzy swarm net (FSN) (Mishra et al., 2008) for data classification as other benchmark methods for comparison.

1.1. Related work

In the realm of the evolutionary neural network using particle swarm optimization, a number of methods have been proposed. Let us discuss a few of the potential proposals relevant to this work.

Yu et al. (2007) has proposed a new evolutionary ANN named IPSOnet based on an improved PSO to simultaneously evolve structure and weights. The improved PSO employs parameter automation strategy, velocity resetting, and crossover and mutation to significantly improve the performance of the IPSO in global search and fine tuning of the solutions. Liu et al. (2004) has investigated a variable neighborhood model in particle swarm search method for neural learning. Zhang et al. (2007) has used a hybrid particle swarm optimization-back propagation algorithm for feed-forward neural network training. Their method can overcome the problem of slow searching process of PSO around the global optimum. In Mazurowski et al. (2008) two methods of neural network training using PSO and BP learning for medical decision-making has been proposed by Mazurowski et al. their experimental results confer the BP is generally preferable over PSO for imbalanced training data, especially with small data samples and large number of features.

Ge et al. (2008) has proposed a modified particle swarm optimization for learning a dynamic recurrent Elman neural network. Their method can overcome some known shortcomings of ordinary gradient descent methods, namely (i) sensitivity to the selection of initial values and (ii) propensity to plague into local optimum.

Guerra and Coelho (2008) have proposed method for choosing thee centers and spread of Gaussian function and training the RBF-NN by using PSO and k-means clustering. Zhao and Yang (2009) have proposed a cooperative random learning particle swarm optimization (CRPSO) to train the single multiplicative neuron model for time series prediction.

Form the above discussion it is clearly that PSOs were successful in evolving ANNs. Thus far PSOs have been tried for evolving ANNs. However, no single effort have been found in the literature to evolve the higher order neural networks (HONs) particularly FLANN using PSO. Therefore, we believe that this effort can make a stepping stone for the researchers who are working on HONs. The organizational flow of the remaining part is as follows. In Section 2, we have discussed the background materials. Section 3 provides our proposed ISO-FLANN for classification. In Section 4 we have presented the experimental studies and comparative performance with other classifiers like MLP, SVM, FLANN with gradient descent and FSN (Mishra et al., 2008). Section 5 concludes the article.

2. Preliminaries

2.1. Computational model of a FLANN

The most popular model used to solve complex classification problems is multilayer neural network. There are many algorithms to train neural network models. However, for models being complex in nature, one single algorithm cannot be claimed to be the best for training to suite different scenarios of complexities of real life problems. Depending on the complexities of the problems, number of layers and number of neurons in the hidden layer need to be changed. As the number of layers and the number of neurons in the hidden layers increases, training the model becomes more complex.

To overcome the complexities associated with multi-layer neural network, a single layer neural network can be considered as an alternative approach. But the single layer neural network being linear in nature often fails to solve the complex non-linear problems. The classification task in data mining is highly non-linear in nature. Therefore, a single layer NN cannot solve this problem.

In order to bridge the gap between linearity in the single layer neural network and the highly complex and computationally intensive multi-layer neural network, the FLANN architecture with back propagation learning for classification was proposed (Mishra and Dehuri, 2007; Pao, 1989; Pao et al., 1992). FLANN architecture can be viewed as a non-linear network. In contrast to the linear weighting of the input pattern produced by the linear links of the artificial neural network, the functional link acts on an element of a pattern or on the entire pattern by generating a set of linearly independent functions, then evaluating these functions with the pattern as the argument. Thus class separability is more in the enhanced feature space. A simple FLANN model with a pattern of two features is shown in Fig. 1.

Let us consider a two dimensional input sample $\bar{x} = [x_1, x_2]^T$. This sample is mapped to a higher dimensional space by functional expansion using trigonometric functions

$$\Psi = [(x_1, \sin\pi x_1, \sin 2\pi x_1, \cos\pi x_1, \cos 2\pi x_1), \\ (x_2, \sin\pi x_2, \sin 2\pi x_2, \cos\pi x_2, \cos 2\pi x_2)]^T.$$

The weighted sum is defined by

$$\hat{y} = \sum_{i,j=1,2} w_i x_j + \sum_{i,j=1,2} w_i \sin i\pi x_j + \sum_{i,j=1,2} w_i \cos i\pi x_j. \quad (1)$$

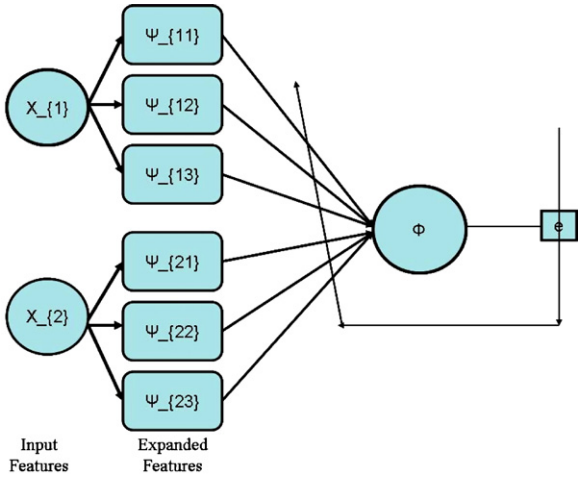


Fig. 1. FLANN for classification.

The FLANN obtains the solution for W iteratively using gradient descent algorithm based on the training samples.

Learning of FLANN may be considered as approximating or interpolating a continuous multivariate function $\phi(X)$ by an approximating function $\phi_w(X)$. In FLANN architecture, a set of basis functions Ψ, and a fixed number of weight parameters W are used to represent $\phi_w(X)$. With a specific choice of set of basis functions Ψ, the problem is then to find the weight parameters W that provide the best possible approximation of Ψ on the set of input–output samples. This can be achieved by iteratively updating W. The interested reader can refer to Dehuri and Cho (2010a,b) and Mishra and Dehuri (2007) for a detailed theory of FLANN for classification.

Let k training patterns denoted by $\langle X_i : Y_i \rangle$, $1 \le i \le k$ be applied to the FLANN and let the weight matrix be W. At the ith instant $1 \le i \le k$, the Q-dimensional input pattern and the FLANN output are given by $X_i = \langle x_{i1}, x_{i2}, \ldots, x_{iQ} \rangle$, $1 \le i \le k$ and $\hat{Y}_i = [\hat{y}_i]$, $1 \le i \le k$. Hence $X = [X_1, X_2, \ldots, X_k]^T$. The augmented matrix of Q-dimensional input pattern and the FLANN output are given by:

$$\langle X : \hat{Y} \rangle = \begin{pmatrix} x_{1,1} & x_{1,2} & . & x_{1,Q} & : & \hat{y}_1 \\ x_{2,1} & x_{2,2} & . & x_{2,Q} & : & \hat{y}_2 \\ . & . & . & . & : & . \\ x_{k,1} & x_{k,2} & . & x_{k,Q} & : & \hat{y}_k \end{pmatrix}$$

As the dimension of the input pattern is increased from Q to Q' by a set of basis functions ψ, given by

$$\psi(X_i) = [\psi_1(x_{i1}), \psi_2(x_{i1}), \ldots, \psi_1(x_{i2}), \psi_2(x_{i2}), \ldots, \psi_1(x_{iQ}), \psi_2(x_{iQ}), \ldots = [\psi_1(x_{i1}), \psi_2(x_{i2}), \ldots, \psi_Q(x_{iQ})].$$

The $k \times Q$ dimensional weight matrix is given by $W = [W_1, W_2, \ldots, W_k]^T$, where W_i is the weight vector associated with the ith output and is given by $W_1 = [w_{i1}, w_{i2}, \ldots, w_{iQ}]$. The ith output of the FLANN is given by $\hat{y}_i(t) = \varphi(\sum_{j=1}^{Q'} \psi_j(x_{ij}).w_{ij}) = \varphi(W_i \,.\, \psi^T(X_i))$, $\forall i$. The error associated with the ith output is given by $e_i(t) = y_i(t) - \hat{y}_i(t)$. Using adaptive learning, weights of the FLANN can be updated as:

$$\begin{aligned} w_{ij}(t+1) &= w_{ij}(t) + \mu.\Delta(t), \\ \Delta(t) &= \delta(t).[\psi(X_i)], \end{aligned} \tag{2}$$

where $\delta(t) = [\delta_1(t), \delta_2(t), \ldots, \delta_k(t)]$, $\delta_i(t) = (1 - \hat{y}_i^2(t)) \cdot e_i(t)$, μ is known as the learning parameter. The set of function considered for function expansion may not always be suitable for mapping the non-linearity of the complex task. In such cases a few more function may be incorporated to the set of functions considered for expansion of the input data. However, dimensionality of many problems are very high and further increasing the dimensionality to a large extent may not be an appropriate choice. So, it is advisable to chose a small set of alternative functions, which can map the function to the desired extent with significant improvement in output.

2.2. Basics of particle swarm algorithms

Particle swarm optimization (PSO) is a nature inspired algorithm invented by Kennedy and Eberhart (1995). Like Leonardo Da Vincis work modeling flying machines from watching bird flight, PSO received its inspiration from bird flocking, fish schooling, and herds of animals.

In PSO, a set of randomly generated solutions (initial swarm) propagates in the design space towards the optimal solution over a number of iterations (moves) based on large amount of information about the design space that is assimilates and shared by all members of the swarm. A complete chronicle of the development of the PSO algorithm from merely a motion simulator to heuristic optimization of the PSO algorithm is described in Kennedy and Eberhart (2001, 1995).

The standard PSO algorithm broadly consist of three computational steps:

1. generation of particles' positions and velocities;
2. updating the velocity of each particle;
3. updating the position of each particle.

Here, a particle refers to a potential solution to a problem. A particle $\vec{x}_k$ in d-dimensional design space is represented as $\vec{x}_k = \langle x_{k1}, x_{k2}, x_{k3}, \ldots, x_{kd} \rangle$, where $k = 1, 2, \ldots, N$, N is the number of particles in a swarm. Each particle has its own velocity and maintains a memory of its previous best position, $p_k = \langle p_{k1}, p_{k2}, p_{k3}, \ldots, p_{kd} \rangle$. Let the velocity $\vec{p}_g = \langle p_{g1}, p_{g2}, p_{g3}, \ldots, p_{gd} \rangle$ refer to the position found by the kth member of the neighborhood that has the best performance so far. The particle changes its position from iteration to iteration based on the velocity updates. In each iteration $\vec{p}_g$ and $\vec{p}_k$ of the current swarm are combined with some weighting coefficients to adjust the velocities of the particles in the swarm. The position of the velocity adjustment influenced by the particle's previous best position is considered as the cognition component, and the position influenced by the best in the neighborhood is the social component.

Without loss of generality, let us consider a minimization task and use symbol $f(.)$ to denote the objective function that is being minimized. The personal best of the kth particle and the global best position can be computed as follows.

$$\vec{p}_k(t+1) = \begin{cases} \vec{p}_k(t) & \text{if } f(\vec{x}_k(t+1)) \ge f(\vec{p}_k(t)) \\ \vec{x}_k(t+1) & \text{otherwise} \end{cases}, \forall k, k = 1, 2, \ldots, N \tag{3}$$

$$\vec{p}_g(t+1) = \begin{cases} \vec{p}_g(t) & \text{if } \forall k \quad f(\vec{x}_k(t+1)) \ge f(\vec{p}_g(t)) \\ \vec{x}_k(t+1) & \text{ifforanyj} \quad f(\vec{x}_j(t+1)) < f(\vec{p}_g(t)) \end{cases}, j = 1, 2, \ldots, N \tag{4}$$

In standard PSO algorithm (Kennedy and Eberhart, 1995), at iteration t, the velocity and the position can be updated using Eqs. (5) and (6), respectively.

$$\vec{v}_k(t+1) = w \otimes \vec{v}_k(t) + \vec{c}_1 \otimes \vec{r}_1(t) \otimes (\vec{p}_k(t) - \vec{x}_k(t)) + \vec{c}_2 \otimes \vec{r}_2(t) \otimes (\vec{p}_g(t) - \vec{x}_k(t)) \tag{5}$$

$$\vec{x}_k(t+1) = \vec{x}_k(t) + \vec{v}_k(t+1) \tag{6}$$

The symbol $\otimes$ denotes point by point vector multiplication. The inertia momentum factor w, $(0 < w \le 1)$, self-confidence factor c_1 and swarm confidence factor c_2 are non-negative real constants. Randomness (useful for good state space exploitation) is introduced via the vectors of random numbers $\vec{r}_1$ and $\vec{r}_2$. They are usually selected as a uniform random numbers in the range [0, 1].

The original PSO algorithm uses w=1, c_1=2, and c_2=2. Over years researchers have fined-tuned these parameters and found out a very standard optimized values (Jiang et al., 2007).

These three steps of velocity update, position update, and fitness computations are repeated until a desired convergence criterion is met. The stopping criterion is that the maximum change in the best fitness should be smaller than the specified tolerance for a specified number of iterations, I, as shown in Eq. (7). Alternatively, the algorithm can be terminated when the velocity updates are close to zero over a number of iterations.

$$|f(\vec{p}_g(t)) - f(\vec{p}_g(t-1))| \le \varepsilon, \quad t = 2, 3, \ldots, I \tag{7}$$

While empirical evidence has accumulated that the standard PSO algorithm works, e.g., it is a useful tool for global optimization, there has thus far been little insight into how it works. In order to address this, a generalized model has been proposed in Clerc and Kennedy (2002). Consequently, the convergence and the stability of the standard PSO has been proposed by many researches (Bergh and Engelbrecht, 2006; Clerc and Kennedy, 2002; Jiang et al., 2007; Trelea, 2003).

3. ISO-flann for classification

In this section we will discuss the proposed ISO-FLANN for classification. This section is divided into three subsections, namely the proposed improved PSO (IPSO), a general description of its architecture, and a high-level algorithm to measure the computational efficiency of the proposed architecture.

3.1. Improved PSO

The improved PSO algorithm is based on the standard global version of the PSO. Like the previous variants of PSO, the drawbacks of PSO with respect to inefficiency in fine tuning solutions, and a very slow searching around the global optimum inspired our modifications.

The IPSO can be described as follows:

$$\vec{v}_k(t+1) = \lambda \otimes \vec{v}_k(t) + \vec{c}_1 \otimes \vec{r}_1(t) \otimes (\vec{p}_k(t) - \vec{x}_k(t)) + \vec{c}_2 \otimes \vec{r}_2(t) \otimes (\vec{p}_g(t) - \vec{x}_k(t)) \tag{8}$$

$$\vec{x}_k(t+1) = \vec{x}_k(t) + \vec{v}_k(t+1) \tag{9}$$

where λ is the newly defined adaptive inertia weight. The algorithm, by adjusting the parameter λ, can make λ reduce gradually as the generation increases. In the searching process of the IPSO algorithm, the search space will reduce gradually as the generation increases. So the IPSO algorithm is more effective, because the search space is reduced step by step. The search step length for the parameter λ also reduces correspondingly. Similar to genetic algorithms (GAs) (Goldberg, 1989), after each generation, the best particle in the last generation will replaces the worst particle of the current generation, thus the better result can be achieved.

In the literature (Eberhart and Shi, 2000; Shi and Eberhart, 1999) several selection strategies of inertia weight λ have been given. Generally, in the initial stages of the algorithm, the initial weight λ should be reduced rapidly, while around the optimum, the initial weight λ should be reduced slowly. So in this paper, we adopted the following procedure:

Algorithm

Input: Initial inertia weight = λ_0; End point of linear section = λ_1;

Number of generations during which inertial weight is reduced linearly = *Gen1*;

Maximum generation = *Gen2*;

Reduced_λ()

```
Begin
  For i=1:Gen1
    λ1=λ0−((λ1/Gen1)× i);
  End
  For i=Gen1+1:Gen2
    λ1=(λ0−λ1)×exp(((Gen1+1)−i)/i);
  End
End
```

In particular the value of *Gen2* and *Gen1* is selected according to empirical knowledge.

Although PSO performs well for global search as it is capable of finding and exploring promising regions in the search space, quickly searching near global optimum is very slow. The self-adaptive evolutionary strategy (ES) is suited for local optimization due to its high probability of generating small Gaussian and Cauchy perturbation (Rudolph, 1997; Schwefel, 1981). Thus it is capable of fine-tuning those solutions found by PSO. When the global best position of PSO cannot be improved for some successive generations, the self-adaptive ES (Yang and Kao, 2001) is used an enhancement operation of $\hat{p}_i$ and $\hat{p}_g$. Thus the self adaptive ES facilitates the convergence of PSO towards global optima. In this study we adapted Schwefel's (1981) proposal to use self-adaptive ES (i.e., the self adaptive Gaussian and Cauchy mutations) for evolving weight parameters of FLANN.

Self-adaptive Gaussian mutation: Mutation is accomplished by first mutating the velocity and then the position of the particle.

$$v_{ki}(t+1) = v_{ki} \times exp(\gamma' \times N_{gi}(0,1)) + \gamma \times N_{ki}(0,1)), \tag{10}$$

$$x_{ki}(t+1) = x_{ki}(t) + v_{ki}(t+1), \tag{11}$$

where $N_{gi}(0, 1)$ is the standard Gaussian density function with respect to the ith dimension of the global best position of the particle. Similarly $N_{ki}(0, 1)$ is the standard Gaussian density function of the ith dimension of the best position found by the particle so far. For this work we follow Bäck and Schwefel (1993) in setting the values of $\gamma = 1/\sqrt{2n}$ and $= 1/\sqrt{2\sqrt{n}}$, respectively.

Self-adaptive Cauchy mutation: A random variable is said to have the Cauchy distribution ($C(t)$), if it has the following density function:

$$C(t) = \frac{t}{\pi(t^2 + x^2)}, \quad -\infty < x < +\infty. \tag{12}$$

We will define self adaptive Cauchy mutation as follows:

$$v_{ki}(t+1) = v_{ki} \times exp(\gamma\prime \times C_{gi}(0,1)) + \gamma \times C_{ki}(0,1)), \tag{13}$$

$$x_{ki}(t+1) = x_{ki}(t) + v_{ki}(t+1), \tag{14}$$

where $t = 1$. In practice, Cauchy mutation is able to make a larger perturbation than Gaussian mutation. This implies that the Cauchy mutation has a higher probability of escaping from the local minima than does Gaussian mutation.

3.2. ISO-FLANN method

ISO-FLANN is a typical three layer feed forward neural network consists of an input layer, a hidden layer and an output layer. The only difference from FLANN is that, the weight vector is evolved by the proposed IPSO during the training of the network. Even though many heuristic approaches exist (Goldberg, 1989) for optimizing the weight vector, we use IPSO because of its characteristics like rapid convergence to global solutions and less number of parameters to be optimized. In other words, here we are trying to reduce the local optimal solution of weight vector by IPSO.

The nodes between input and hidden layers are connected without weight vector, but the nodes between hidden layer and output layer are connected by weights. The signal of the output node is based on a function of the sum of the inputs to the node.

In ISO-FLANN architecture, there are d input nodes (i.e., equal to the number of features of the dataset) and m nodes in the hidden layer, where m is the number of functionally expanded node and one output neuron in the output layer. The connection between hidden layer and output layer is assigned with the weight vector.

In this work, we have used the orthonormal trigonometric function for mapping the input feature from one form to another form of higher dimension. However, one can use a function that is very close to the underlying distribution of the data, but it requires some prior domain knowledge. In this work we are taking five functions out of which four are trigonometric and one is linear (i.e., keeping the original form of the feature value). Out of the four trigonometric functions, two are *sine* and two are *cosine* functions. In the case of trigonometric functions the domain is the given feature values and range lies between [−1,1]. It can be written as

$$f : D \to R^{[-1,1]\cup\{x\}}, \tag{15}$$

where D= $\{x_{i1}, x_{i2},\ldots, x_{id}\}$, and d is the number of features.

In general let us take $f_1, f_2,\ldots, f_k$ as the number of functions to be used to expand each feature value of the pattern.

Therefore, each input pattern can now be expressed as

$$\vec{x}_i = \{x_{i1}, x_{i2}, \ldots, x_{id}\} \to \{\{f_1(x_{i1}), f_2(x_{i1}), \ldots, f_k(x_{i1})\}, \ldots, \{f_1(x_{id}), f_2(x_{id}), \ldots, f_k(x_{id})\}, = \{\{y_{11}, y_{21}, \ldots, y_{k1}\}, \ldots, \{y_{1d}, y_{2d}, \ldots, y_{kd}\}\}.$$

The weight vector between hidden layer and output layer is multiplied with the resultant sets of non-linear outputs and are fed to the output neuron as an input. Hence the weighted sum is computed as follows:

$$s = \sum_{j=1}^{m} y_{ij} \cdot w_j,\ i = 1, 2, \ldots, N \text{and } m \text{ be the total number of expanded features.} \tag{16}$$

The network has the ability to learn through training by IPSO. The training requires a set of training data, i.e., a series of input and associated output vectors. During the training, the network is repeatedly presented with the training data and the weights adjusted by IPSO from time to time till the desired input–output mapping occurs.

The estimated output is then computed by the following metric:

$$\hat{y}_i(t) = f(s_i),\ i = 1, 2, \ldots, N.$$

The error $e_i(t)$ = $y_i(t)$–$\hat{y}_i(t)$, i = 1, 2,. . ., N is the error obtained from the ith pattern of the training set.

Therefore the error criterion function can be written as,

$$E(t) = \sum_{i=1}^{N} e_i(t), \tag{17}$$

and our objective is to minimize this function with an optimal set of weights.

3.3. ISO-FLANN high-level algorithm

ISO-FLANN, a member of the family of higher order neural networks, is a computational model capable of learning through adjustment of internal weight parameters according to a training algorithm in response to some training examples.

There are many literatures exist (Carvalho and Ludermir, 2007; Chang et al., 2007; Da and Xiurun, 2005; Wu et al., 2006; John Paul et al., 2006) on PSO-based neural network training, but to the best of our knowledge IPSO-based FLANN training and its application to classification problem is the first effort in this direction.

In ISO-FLANN the weights between hidden and output layer is adaptively evolved by IPSO algorithm, starting from the parents' weights instead of randomly initialized weights, so this can preferably solve the problem of noisy fitness evaluation that can mislead the evolution.

The dataset is divided into two mutually exclusive sets: a training set and a test set (more details in Section 4). The training set is used to evolve the optimal model with optimal sets of weights using IPSO, and the fitness evaluation is based on the error criterion function E, which is already described in Section 3.2. In order to embed IPSO for weight evolution one could keep the following points in mind: the particle representation, and the objective function to measure the effectiveness of the particle.

3.3.1. Representation of a particle

For the evolutionary process, the length of each and every particle is m and it is fixed (i.e., the number of connection between expanded features of the hidden layer and the output neuron of the output layer), but one can go for variable length particle also. The variable length particle representation is highly useful for simultaneous evolution of architectures and weights. In this work our focus is on fixed length particle, the variable length particle is beyond the scope of our study. A particle can be represented as a vector of m weights, i.e., $\langle w_1, w_2, w_3,\ldots, w_m \rangle$.

In ISO-FLANN the weight values lie between [−1, 1]. Hence the velocity of the particle also lies between [−1, 1]. In case of extreme values like $w_i = 0$, one can believe that the connection between the expanded features and output node corresponding to w_i is not an informative one and is virtually deleted from the network.

3.3.2. Objective function

During evolution each particle measures its effectiveness by the error criterion function, using Eq. (17) mentioned in Section 3.2.

The major steps of EFLANN can be described as follows:

1. DIVISION OF DATASET
   Divide the dataset into two parts: training and testing.
2. MAPPING OF INPUT PATTERNS
   Map each pattern from lower dimension to higher dimension, i.e., expand each feature value according to predefined set of functions.
3. RANDOM INITIALIZATION
   Initialize each particle randomly with small values from the domain [−1, 1].
4. WHILE(Termination Criterion Not Met)
   FOR entire swarm
   FOR each particle in the swarm
   FOR each sample of training sample
   Calculate the weighted sum and feed as an input to the node of the output layer.
   Calculate the error and accumulate it.
   END
   Fitness of the particle is equal to the accumulated error.
   If fitness value is better than the best fitness value in history, set current value as the new personal best,
   END
   Choose the particle with best fitness value of all the particles as the global best.
   FOR each particle
   Call **Reduced**_λ() and calculate particle velocity according to Eq. (8).
   Update Particle position according to Eq. (9).
   END

END
MUTATION
Apply Cauchy and Gaussian mutation if the position of the global best solution is not improved for a successive number of pre-specified generations alternatively by using Eqs. (11) and (14).
5. WHILE END

This algorithm does not optimizing the weight vector only but also implicitly optimizing the required number of connections between hidden layer and output layer. Hence we can say this is a type of architecture optimization. However, in this work we are not considering this issue. Hence, instead of a multi-objective function optimization we are only optimizing the uni-objective, i.e., known as classification accuracy.

4. Experimental details

Even though the proposed algorithm is primarily intended for classification of datasets with large number of records and a moderate number of features (primarily for data mining), it can also be used very well on more conventional datasets. To exhibit this fact we evaluated our algorithm using a set of fifteen public domain datasets from the University of California at Irvine (UCI) machine learning repository (Blake and Merz, 2012).

We have compared the results of ISO-FLANN with other competing classification methods such as multi-layer perception (MLP) and the FLANN with gradient descent (and with the same set of orthonormal basis functions like ISO-FLANN), support vector machine (SVM) with radial basis kernel and FSN.

This section is divided into three subsections. Section 4.1 discusses the nature and characteristics of the datasets being classified. The environment, parameter setting of the proposed method along with the methods considered for comparative study and the performance of the model is demonstrated in Section 4.2 with a discussion. Finally a comparative performance is given in Section 4.3.

4.1. Description of the datasets

Let us briefly discuss the datasets, chosen for our experimental setup.

IRIS Datasets: This is the most popular and simple classification dataset based on multivariate characteristics of a plant species (length and thickness of petal and sepal) divided into three distinct classes(Iris Setosa, Iris Versicolor, and Iris Virginica) of 50 instances each. One class is linearly separable from the other two; the latter are not linearly separable from each other. In a nutshell, it has 150 instances and 5 attributes. Out of Five attributes, Four attributes are predicting attributes and one is goal attribute. All the predicting attributes are real values.

WINE Dataset: This dataset is resulted from a chemical analysis of wines grown in the same region in Italy but derived from three different cultivars. In classification context, this is a well-posed problem with well-behaved class structures. The total number of instances is 178 and it is distributed as 59 for class 1, 71 for class 2 and 48 for class 3. The number of attributes is 14 including class attribute and all 13 are continuous in nature. There are no missing attribute values in this dataset.

PIMA Indians Diabetes Data base: This database is a collection of all female patients of at least 21 years of PIMA Indian heritage. It contains 768 instances, 2 classes of positive and negative and 9 attributes including the class attribute. The attribute contains either integer or real values. There are no missing attribute values in the dataset.

BUPA Liver Disorders: This Dataset related to the diagnosis of liver disorders and created by BUPA Medical Research, Ltd. It consists of 345 records, 7 attributes including the class attributes. The class attribute is repeated with only two class values for entire database. The first 5 attributes are all blood tests, which are thought to be sensitive to liver disorders that might arise from excessive alcohol consumption. Each record corresponds to a single male individual.

Clevend Heart disease: This dataset is related to diagnoses of people with heart problems. It consists of 304 data instances, 5 attributes including the class and 2 classes.

Wisconsin Diagnostic Breast Cancer (WBC (D)): This dataset is related to diagnosis of people with breast cancer. Features of the dataset are computed from a digitized image of a fine needle aspirate (FNA) of a breast mass. They describe characteristics of the cell nuclei present in the image. The dataset has 569 instances, 32 attributes and 2 classes namely, benign and malignant.

Wisconsin prognostic Breast Cancer (WBC (D)): This dataset is related to diagnosis of people with breast cancer. Each record represents follow-up data for one breast cancer case. The dataset has 198 instances, 30 attributes and 2 classes namely, recurrent and non-recurrent. Out of the 198 instances 4 instances have missing values.

Page Block: This dataset is used for classifying all the blocks of the page layout of a document that has been detected by a segmentation process. This is an essential step in document analysis in order to separate text from graphic areas. Indeed, the five classes are: text (1), horizontal line (2), picture (3), vertical line (4) and graphic (5). The dataset consist of 5473 instances and 10 attributes. The attributes are combination of integer and float values. The dataset have no missing values.

Thyroid: This dataset is used to predict whether a patient's thyroid to the class euthyroidism (normal), hypothyroidism (Hypo) or hyperthyroidism (hyper). The diagnosis (the class label) was based on a complete medical record, including anamnesis, scan etc. The dataset had 215 instances with no missing values and 5 attributes. All the attributes are of continuous nature.

Metabolic Syndrome: The metabolic syndrome dataset (Park and Cho, 2006) is obtained from Yonchon County of Korea. The metabolic syndrome is a collection of metabolic disorder which includes hypertension, dyslipidemia, elevated blood glucose and obesity. The dataset have 1135 samples with 2 class labels. The dataset has 18 attributes and the class attribute determine the absence or presence of the disease. The attributes of the dataset have a combination of categorical and continuous values. The dataset doesn't have any missing values. 11 important attribute which are necessary for the prediction.

Dermatology: This dataset is obtained from the diagnosis of the patients of erythemato-squamous diseases. The diseases in this group are psoriasis, seboreic dermatitis, lichen planus, pityriasis rosea, cronic dermatitis, and pityriasis rubra pilaris. This dataset is obtained by evaluating clinically with 12 features. Afterwards, skin samples were taken for the evaluation of 22 histopathological features.

Hepatitis: This dataset is used for classification of hepatitis patients into two classes labeled as 'Die' and 'Live' based on the pathological analysis. The dataset has 20 attributes and 157 instances.

Parkinson: This dataset is composed of a range of biomedical voice measurements from 31 people, 23 with Parkinson's disease (PD). Each column in the table is a particular voice measure, and each row corresponds one of 195 voice recording from different individuals. The main aim of the data is to discriminate healthy people from those with PD, according to "status" column which is set to 0 for healthy and 1 for PD.

Vertebral column: This dataset is obtained from Group of Applied Research in Orthopaedics (GARO), Lyon, France. This dataset is

Table 1
Summarized view of characteristics of the dataset.

Dataset	Number of patterns	Number of features	Number of classes	Number of patterns in class 1	Number of patterns in class 2	Number of patterns in class 3	Number of patterns in class 4	Number of patterns in class 5	Number patterns in class 6
Page Block	5473	10	5	4913	329	28	88	115	
IRIS	150	5	3	50	50	50			
WINE	178	14	3	71	59	48			
Thyroid	215	5	3	150	35	30			
PIMA	768	9	2	500	268				
BUPA	345	7	2	145	200				
Clevend Heart	304	6	2	45	259				
WBC (D)	569	30	2	357	212				
WBC (P)	194	32	2	148	46				
Metabolic Syndrome	1135	18	2	612	523				
Dermatology	366	33	6	112	61	72	49	52	20
Hepatitis	155	19	2	32	123				
Parkinson	196	23	2	48	148				
Ionosphere	351	34	2	225	126				
Vertebral column	310	6	2	100	210				

used for classification task. It classifies patients into two categories labeled as 'normal' and 'abnormal' based on the analysis report of their vertebral column.

Ionosphere: This radar dataset is used for classifying the presence of electrons structure in the ionosphere. "Good" radar returns are those showing evidence of some type of structure in the ionosphere. "Bad" returns are those that do not; their signals pass through the ionosphere. The dataset consist of 34 of attributes which are continuous in nature.

Table 1 presents a summary of the main characteristics of the databases that have been used in this study. The first column of this table gives the name of the database, while other columns indicate, respectively, the number of instances, the number of attributes, number of classes, number of patterns in class 1, number of patterns in class 2 and number of patterns in class 3.

4.2. Environments, parameters and classification performance

4.2.1. Environments

The proposed method was implemented on a personal computer with an Intel Pentium IV, 2.40 GHz CPU, 1.00 GB RAM (the primary method), the Microsoft Windows XP Professional version 2002 operating system with Matlab 7.0.1 development environment. For evaluating all these algorithms, the following protocols related to dataset division were set.

The datasets are divided into 10 folds, and out of this 9-folds are used for training and 1 fold is used for testing the performance of the classifiers. However, 9-fold cross validation is carried out in the case of HEART disease database obtained from STATLOG project for comparing the cost estimation result of ISO-FLANN with other state-of-the-art algorithms presented in King et al. (1995) where 9-fold cross validation was reported.

4.2.2. Parameters

The quality of each particle is measured by the error criterion function E. It is also very important for the user to set *a priori* the values of the parameters of the proposed algorithm. These parameters are presented in Table 2.

In the literature different values of N have been used for swarm size. In this work we set $N = 10 \times d$ to avoid under-fit and over-fit during the training of the algorithm. The larger is the number of particles, more is the computation time. Length of the particle is fixed to m, where m depends on the functionally expanded features of the hidden layer.

Although the parameters are quite restricted and there are no such standard rule to assign systematic parameter values to $c1$, $c2$ and λ but in this experimental study the value of λ is restricted within the interval [1.8, 0.2]. The values of $c1$ and $c2$ are chosen as 2.8 $\times\lambda$ and 1.3 $\times\lambda$, respectively. These set of parameters are assigned after an extensive set of trial and error process.

The initial position and the velocity range of the particles in IPSO lies in the interval [−1, +1] and if the global best position p_g is not improved for successive generations (100) we run the self adaptive Gaussian and Cauchy mutation alternatively to escape from the false global solution (i.e., the equivalent of a local optimum in global optimization).

The next important question in the proposed IPSO is when to stop a run. Many researchers have used either the maximum number of generations (*Gen*2) or maximum number of function calls as stopping conditions in their experimental study (Blake and Merz, 2012). Liu et al. (2005) used $|f_{min} - f_{opt}| \leq \varepsilon$, where f_{min} is the best solution found so far. The stopping condition $|f_{min} - f_{opt}| \leq \varepsilon$ only applies if the optimal value of the problem under consideration is known. However, in many practical applications, the optimal value is not known. On other hand, the maximum number of iterations(or function calls) cannot be judged for an arbitrary decision boundary of a classification problem. This may lead to unnecessary function calls when the minimum is reached long before the maximum number of iterations (or function calls)—thus increasing computational costs. In this paper we use a combined approach as follows:

In each iteration we check the condition $|f_{min} - f_{opt}| \leq \varepsilon$, where $f_w = f(p_i)$ is the functional value of current worst personal best p_i in P and $f_b = f(p_g)$ is the functional value of the current global best p_g in P (since each particle in P is always updated with an improvement at each iteration, the set P will gradually contract) and *Gen*1 and *Gen*2. If this condition is reached earlier than *Gen*2, then stop. Otherwise continue till the value of *Gen*2 is reached. However, the Gaussian and Cauchy mutation is applied as usual and the inertia value is decreased accordingly. In this work we stop a run either the points in P are identical to an accuracy of three decimal places, i.e., $|f_{min} - f_{opt}| \leq \varepsilon = 10^{-3}$, or the maximum number of generations *Gen*2 = 3000.

Table 2
Parameters used in proposed algorithm.

Symbol	Name and purpose of the parameter
N	Size of the swarm (P)
λ	Inertia weight
λ_0	Inertial value of inertia weight
λ_1	Inertia weight value of the end point of linear section
*Gen*1	Generations that reduces linearly (i.e., *Gen*1 = *Gen*2 × 40%)
*Gen*2	Maximum generation of the algorithm
c_1	Cognitive parameter
c_2	Social parameter

Table 3
Results obtained by the ISO-FLANN model for the classification of the Heart Disease Database.

Dataset	Error in training set		Error in test set		Cost in training set	Cost in test set
	Class 1	Class 2	Class 1	Class 2		
heart1.dat	13/133	14/107	1/17	1/13	0.34583	0.2
heart2.dat	14/133	12/107	2/17	1/13	0.30833	0.23333
heart3.dat	13/134	15/106	4/16	2/14	0.36667	0.46667
heart4.dat	13/133	10/107	1/17	4/13	0.2625	0.7
heart5.dat	13/133	16/107	3/17	2/13	0.3875	0.43333
heart6.dat	13/134	14/106	6/16	0/14	0.3458	0.2
heart7.dat	15/133	13/107	0/17	3/13	0.3333	0.5
heart8.dat	18/133	17/107	1/17	0/13	0.42917	0.033333
heart9.dat	20/134	9/106	2/16	1/14	0.27083	0.23333
Average					0.33888	0.333333

However in the case of MLP, we have considered two hidden layers for all datasets. Each hidden layer contains $[Q \times n]$ number of neurons, where Q is the number of input neurons and n is the number of output neurons, Further, for all datasets we fixed 1000 epochs. For SVM, the number to support vectors is determined based on the minimum distance from the separating planes. The k-means algorithm finds a set of clusters. Each center is then associated with the kernel. An appropriated width can be computed by using nearest neighbor heuristics.

In case of FSN, initially we take a set of fuzzy nets, each net is treated as a particle and the set of fuzzy nets are treated swarm. Each net shares the same memory in the distributed environment. At any instance of time all the nets are supplied with one input record and the respective target. All the nets in the distributed environment are initialized to random weights in the range [0, 1].

After each iteration, we calculate the error for all the nets in the distributed environment. The net giving the minimum error is treated as the leader among all the nets. The nets also preserve the best value achieved by the respective nets during all the iterations in their local memory, which is treated as its personal best. Each net uses both personal best and global best values to update its weights for successive iterations. The stopping criterion may be allowing the nets to iterate till they converge to a single decision. However, in this case the net gets over training and lead to poor performance. Therefore, for different datasets a suitable range of iterations is fixed, as one range may not be suitable for all datasets. For case of IRIS dataset the range is [50–150], whereas all other datasets the range for iteration varies from 100 to 200.

4.2.3. Classification performance

As the improved PSO is a stochastic algorithm, 10 independent run for each algorithm were performed for every single fold.

The average values of 10 fold cross validation of each data set are used for comparisons with other classification methods (see Tables 4 and 5).

Now, we will explicitly examine the performance of the ISO-FLANN model by considering the heart dataset with the use of the 9-fold cross validation methodology. The reason for using 9-fold cross validation is that to compare the performance with the performance of the algorithms considered in StatLog Project (King et al., 1995). In 9-fold cross validation we partition the data set into nine subsets (heart1.dat, heart2.dat,. . ., heart9.dat), where eight subsets are used for training and the remaining one is used for training. The process is repeated nine times in such a way that each time a different subset of data is used for testing. Thus the dataset was randomly segmented into nine subsets with 30 elements each. Each subsets contains about 56% of samples from class 1 (without heart disease) and 44% of samples from class 2 (with heart disease).

The procedure makes use of a cost matrix, which is described as follows:

$$\text{Cost Matrix} = \begin{pmatrix} 0 & w_2 \\ w_1 & 0 \end{pmatrix}.$$

The purpose of each matrix is to penalize wrongly classified samples based on the weight of the penalty of the class. In general, the weight of the penalty for class 2 samples that are classified as class 1 samples is w_1, while the weight of the penalty for class 1 records that are classified as class 2 samples is w_2. Therefore, the metric used for measuring the cost of the wrongly classifying patterns in the training and test dataset is given by Eqs. (18) and (19).

$$C_{train} = \frac{(S_1 \times w_1 + S_2 \times w_2)}{S_{train}}, \tag{18}$$

$$C_{test} = \frac{(S_1 \times w_1 + S_2 \times w_2)}{S_{test}}, \tag{19}$$

where C_{train} is the cost of the training set; C_{test} is the cost of the test set; S_1 and S_2 denote the pattern that are wrongly classified which belong to class 1 and 2 respectively; S_{train} and S_{test} are the total number of training and test patterns respectively.

Table 3 presents the errors and the costs of the training and the sets for the ISO-FLANN model with a weight value of $w_1 = 5$ and $w_2 = 1$ subject to an average value of 10 independent runs. Upon closer inspection of Table 3, it may be observed that the configuration of the heart8.dat dataset as a test subset obtain lower errors and consequently a lower cost for the ISO-FLANN model. Note that these parameters totally depends on the designer keeping in mind that the classification accuracy should not be degraded.

4.3. Comparison with other models

The results obtain from the IRIS, WINE, PIMA Indians Diabetes Database, and BUPA Liver Disorders data sets were compared with the results obtain from the previous model FLANN with gradient decent, MLP, SVM with the radial basis kernel and FSN. Note that the result of the proposed algorithm and FSN are averaged over 10 independent runs. Since FLANN with gradient decent, MLP and SVM with radial basis kernal are not stochastic, only one run is performed for the classification,

Table 4 represents a summary of the comparative results of ISO-FLANN with MLP and FLANN, and Table 5 with SVM and FSN on both the training test and test set, respectively.

It clearly indicates that the classification accuracy of the same dataset (except PIMA dataset) could vary widely depending upon what kind of classification algorithm is applied on it.

The classification results found for the Heart disease data set were compared with the results found in the STATLOG project (King et al., 1995). Accordingly to the STATLOG project methodology, comparison consist of calculating the average cost produced by nine data subsets used for validation. Table 6 presents the

Table 4
Comparison of the average performance of ISO-FLANN and FLANN.

Dataset	ISO-FLANN		FLANN		MLP	
	HPT	HPS	HPT	HPS	HPT	HPS
IRIS	98.67	99.03	98.07	97.333	98.15	94.00
WINE	98.751	96.56	92.72	92.19	96.14	92.29
PIMA	80.62	79.63	79.20	78.82	76.61	77.19
BUPA	78.11	76.80	73.80	72.29	67.52	67.39
WBC (D)	97.68	97.36	97.96	91.54	91.62	92.65
WBC (P)	85.64	84.36	82.06	79.11	76.32	76.29
Thyroid	97.34	94.47	95.78	93.25	79.78	79.77
Clevend Heart	86.53	85.57	86.41	79.27	82.63	80.42
Page block	93.98	93.09	93.62	92.91	86.55	84.74
Metabolic Syndrome	80.61	72.32	71.27	68.72	71.35	62.92
Dermatology	97.018	94.43	96.36	92.45	86.78	80.63
Parkinson	89.67	85.94	84.67	80.97	77.04	73.75
Ionosphere	92.40	90.38	79.67	80.94	74.61	73.28
Hepatitis	80.39	75.72	73.57	70.60	60.42	60.83
Vertebral column	99.75	99.17	98.26	96.67	88.32	85.83

HPT: hit percentage in training set, HPS: hit percentage in test set.

Table 5
Comparison of the average performance of ISO-FLANN and SVM.

Dataset	ISO-FLANN		SVM		FSN	
	HPT	HPS	HPT	HPS	HPT	HPS
IRIS	98.67	99.03	91.69	91.70	97.182	96.00
WINE	98.75	96.56	79.06	73.66	97.87	93.69
PIMA	80.62	79.63	79.68	75.37	75.27	76.39
BUPA	78.11	76.80	74.57	68.53	65.19	65.00
WBC (D)	97.68	97.36	96.55	95.71	96.77	95.27
WBC (P)	85.64	84.38	80.07	76.32	82.46	79.42
Thyroid	97.34	95.47	90.70	90.76	96.74	94.39
Clevend Heart	86.53	85.57	85.20	84.19	85.19	84.86
Page block	93.98	93.09	92.70	91.39	92.83	92.09
Metabolic	80.61	72.32	79.49	71.12	78.48	71.47
Dermatology	97.018	94.43	95.49	87.65	96.28	90.65
Parkinson	89.67	85.94	88.32	82.50	88.43	83.69
Ionosphere	92.40	90.38	83.74	83.74	90.54	87.5
Hepatitis	80.39	75.72	76.27	63.18	76.57	72.52
Vertebral column	99.75	99.17	95.75	93.33	98.78	97.31

HPT: hit percentage in training set, HPS: hit percentage in test set.

Table 6
Comparison of the cost estimation of ISO-FLANN with other standard classification algorithms.

Classifier	ISO-FLANN	Naive Bayesian	Backpropagation	Kohonen	C4.5	RBF	CART
Cost in training	**0.3388**	0.351	0.381	0.429	0.439	0.303	0.463
Cost in testing	**0.3333**	0.374	0.574	0.693	0.781	0.781	0.452

average cost for the nine training and test subsets. The results of the ISO FLANN model is compared with six classification algorithms such as: statistical methods (Naive Bayes), neural network methods (backpropagation, Kohonen, and radial basis function (RBF)), and machine learning, methods (C4.5 and CART). This selection is due to their relative popularity and availability of performance data provided in the empirical study. The values of ISO FLANN are highlighted to show the significance of ISO-FLANN over other classification algorithms.

It is clearly indicates that in the proposed method classification error rate is very low compared to other neural network models like back-propagation, Kohonen, and radial basis function (RBF). In addition, the error rates for this particular dataset could vary widely among the classifiers.

4.4. Statistical performance evaluation

To evaluate the performance of the classifiers a non-parametric test is carried out with 10 different datasets. We consider the null hypothesis as

H_o :Alltheclassifiersareequivalent.

To test the null hypothesis, we start with a Friedman test (Demšar, 2006; García et al., 2010). In this test, we assign ranks r_i^j to the jth of k classifiers on the ith of N datasets based on their predictive accuracy. Then we obtain the Friedman statistics given by equation

$$F_F = \frac{(N-1)\chi_F^2}{N(K-1)-\chi_F^2} \tag{20}$$

where

$$\chi_F^2 = \frac{12N}{k(k+1)}\left[\sum_j R_j^2 - \frac{k(k+1)^2}{4}\right] \tag{21}$$

The Friedman statistic is distributed according to the F-distribution with $k-1$ and $(k-1)(N-1)$ degree of freedom.

In this case, Table 7 shows the ranks (shown in brackets) of each classifier on different datasets. Using Eq. (21), we find the χ_F^2 as 42.3708 and in turn the F_F as 33.6482. With the 5 classifiers and 15 datasets, F_F is distributed to the F distribution with $5-1=4$ and $(5-1)\times(15-1)=56$ degree of freedom. The critical value of $F(4,$

Table 7
Ranks of each classifiers on different dataset based on the average hits percentage on test set.

Dataset	ISO-FLANN	FLANN	FSN	MLP	SVM
IRIS	99.03 (1)	97.33 (2)	96.00 (3)	94.00 (4)	91.70 (5)
WINE	96.56 (1)	92.19(4)	93.69 (2)	92.29 (3)	73.66 (5)
PIMA	79.63 (1)	78.82 (2)	75.37 (5)	77.19 (3)	76.39 (4)
BUPA	76.80 (1)	72.29 (2)	68.53 (3)	67.39 (4)	65.00 (5)
WBC (D)	97.36 (1)	91.54 (5)	95.26 (3)	92.65 (4)	95.71 (2)
WBC (P)	84.38 (1)	79.11 (3)	79.421 (2)	76.29 (5)	76.32 (4)
Page block	93.09(1)	92.91 (2)	92.09 (3)	84.74 (5)	91.39 (4)
Thyroid	95.47 (1)	93.25 (3)	94.39 (2)	79.77 (5)	90.76 (4)
Clevend heart	85.57 (1)	79.27 (5)	84.86 (2)	80.42 (4)	84.19 (3)
Metabolic syndrome	72.32 (1)	68.72 (4)	71.47 (2)	62.92 (5)	71.12 (3)
Dermatology	94.28 (1)	92.45 (2)	90.65 (3)	80.63 (5)	87.65 (4)
Hepatitis	75.72 (1)	70.60 (3)	72.52 (2)	60.83 (5)	63.18 (4)
Parkinson	85.94 (1)	80.97 (4)	83.69 (2)	73.75 (5)	82.50 (3)
Ionosphere	90.38(1)	80.94(4)	87.5 (2)	63.28 (5)	83.74 (3)
Vertebral Column	99.17 (1)	96.67 (3)	97.31 (2)	85.83 (5)	93.33 (4)
Average	1	3.2	2.53	4.47	3.8

56) for $\alpha = 0.01$ as 3.674 which is less than the obtain F_F statistic. Thus, we can reject the null hypothesis H_o.

As the null hypothesis is rejected, we proceed with the post hoc test. We perform the Holm procedure (Demšar, 2006; García et al., 2010; Luengo et al., 2009) to evaluate performance of proposed model with other classifier. For carrying this test, we need to compute the z value using Eq. (22)

$$z = \frac{R_i - R_j}{SE} \tag{22}$$

where

$$SE = \sqrt{\frac{k(k+1)}{6N}} \tag{23}$$

This z value is used for calculating the probability p from the table of normal distribution, which then compared with an appropriate α. For this test, we consider the null hypothesis as

H_o :The pair of classifiers compared are equivalent.

We order the p values in increasing order of significance and then for Holm test, we compare the p_i value with $\alpha/(k-i)$. The value of p_i is less than the $\alpha/(k-i)$, thus, we reject the null hypothesis and proceed for comparison with other classifiers. The result of this test with ISO-FLANN and other classifiers is shown in Table 8. From this, it is clear that all the null-hypothesis are rejected. Thus ISO-FLANN has significantly better performance than all other classifiers.

4.5. Effect of parameters on classification accuracy

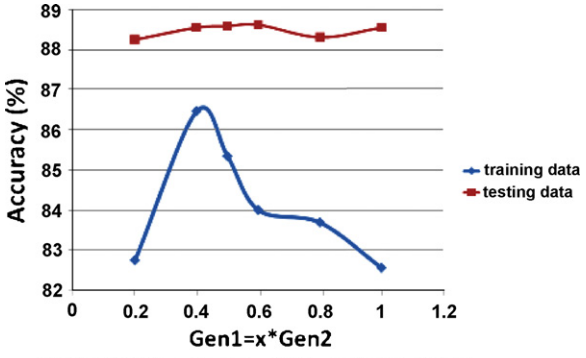
The parameters used in ISO-FLANN effects the classification accuracy of different datasets. So, a fine tuning of the parameters are important for attaining the optimal classification accuracy. The first parameter considered is the inertia weight which is adjusted using $Reduced_\lambda$ method. In the $Reduced_\lambda$ method, the initial inertia weight reduces rapidly for Gen1 and then inertia weight is reduces slowly for Gen2. The $Gen1$ is equal to $xofGen2$ where the x is determined empirically. Fig. 2 shows the effect of change in the x values. From the figure, it can be seen that on increasing the x, the training accuracy of clevend heart and metabolic dataset increases. But the testing accuracy increases for [0.4-0.5], but further increasing the value of x, i.e., if the inertia weight is reduced rapidly, the testing accuracy decreases. Almost for all the ten dataset, we have seen that if $Gen1 = [0.4 - 0.5] \times Gen2$, then maximum accuracy can be achieved.

We have seen similar effect of the cognitive learning factor (C_1) on classification accuracy. If the C_1 is considered in the range of [1.7–1.8], then a maximum classification accuracy can be achieved. Fig. 3 shows the impact of C_1 on classification accuracy. Here the training accuracy increases linearly for both the clevend heart and Wisconsin prognostic breast cancer dataset, however, the same cannot be seen in testing accuracy of the Wisconsin prognostic breast cancer dataset. So, we need to fine tune the C_1 parameter carefully.

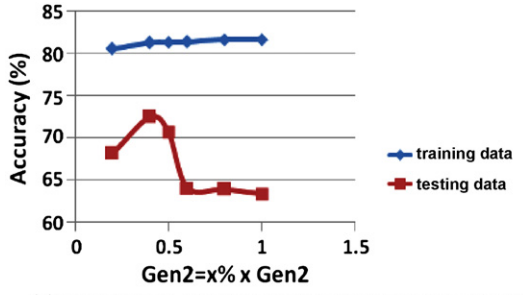
The impact of Social learning factorC_2 is same for almost all dataset. The classification accuracy increases till $C_2 = 2.5$, but on further increasing it the accuracy decreases. The same effect is seen in all the dataset. The representative sample of the effect is shown in Fig. 4.

4.6. Justification of trigonometric basis and asymptotic bound of ISO-FLANN

The proposed structure with functional expansion using trigonometric functions has the following advantages:



(a) Classification accuracy of Clevend heart dataset



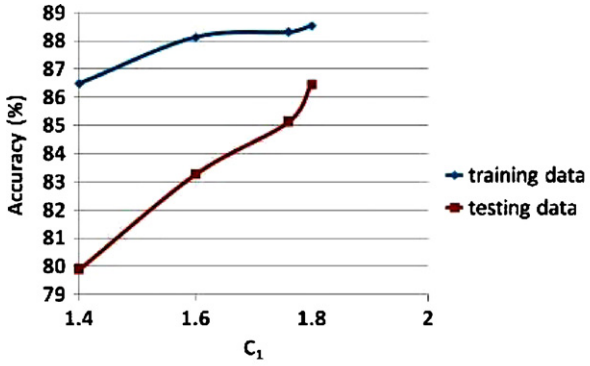
(b) Classification accuracy of Metabolic Syndrome dataset

Fig. 2. Impact of percentage change in Gen1 on classification accuracy. (a) Classification accuracy of Clevend heart dataset and (b) classification accuracy of Metabolic Syndrome dataset.

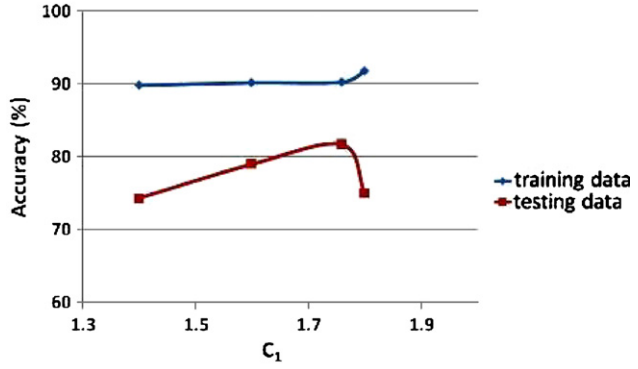
Table 8
Holm and Hochberg procedure.

i	Classifiers	z-values	p-values	$\alpha/(k-i)$
1	ISOFLANN-MLP	6.010	0.0000000008855	0.0125
2	ISOFLANN-SVM	4.850	0.0000005791	0.017
3	ISOFLANN-FLANN	3.845	0.00005710	0.025
4	ISOFLANN-FSN	2.650	0.003939	0.05

- The trigonometric basis function was chosen here because this basis forms a more compact representation than other possible functions like Gaussian and orthogonal polynomials (e.g., Legendre, Chebyshev, etc.). In addition the *sin* and *cos* functions can be computed quickly.
- It has been noted that if appropriated trigonometric polynomial is used for function expansion, the weight solution obtained by IPSO will approximate the terms in multi-dimension. In case of the proposed method the link acts on an element of a sample or on the entire sample itself by generating a set of independent functions. Then these functions are evaluated with the sample as the arguments. The functions are chosen as a subset of a complete set of orthonormal basis functions spanning an m-dimensional representation space, such as $\sin \pi x$, $\cos \pi x$, $\sin 2\pi x$, $\cos 2\pi x$ and so on. The net effect is to map the input sample from low to high dimensional spaces. However, when the outer product terms were used in combination with the functional expansion, good results were obtained on case of learning the network.



(a) Classification accuracy of Clevend heart dataset



(b) Classification accuracy of WBC (P) dataset

Fig. 3. Impact of cognitive learning C_1 on classification accuracy. (a) Classification accuracy of Clevend heart dataset and (b) classification accuracy of WBC (P) dataset.

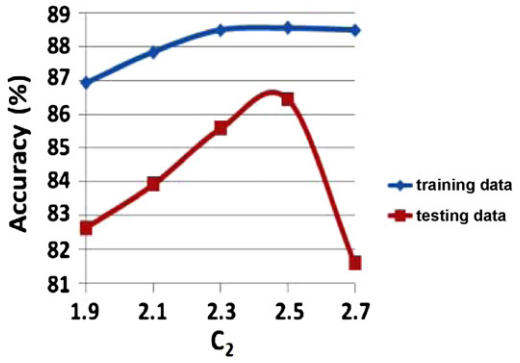


Fig. 4. Impact of social learning C_2 on classification accuracy.

Mathematical arguments: As we have already discussed above the learning procedure of the proposed method can be considered as approximating or interpolating a continuous, multivariate function $\phi(x)$ by an approximating function $\phi_w(x)$. Here an information mathematical argument is given to support how good the chosen basis function ψ and a fixed number of weight parameters W and used to represent the best approximator of $\phi_w(x)$ on the set of input output samples.

Let A be a compact and simply connected subset of $n\Re$ and $l_m(A)$ be the subset of Lebesgue measurable functions $\phi : A \subset n\Re \rightarrow m\Re$ such that the *sup* norm of ϕ, denoted by $\|\phi\|_A$ is bounded, i.e., $\|\phi\|_A = \sup_{x \epsilon A} |\phi(x)| \prec \infty$. The space of all continuous functions $\phi : A \rightarrow m\Re$ which is a subset of $l_m(A)$, and is denoted by $C_m(A)$. It is often desirable to parameterize ϕ in terms of discriminant $\sum_{i,j} w_{ij}.\psi_j(X_i)$ to identify a non-linear function $\phi : A \rightarrow m\Re$, for the selected trigonometric basis functions $\psi_i \epsilon l(A)$. By StoneWeierstrass theorem, there exists many sets of such functions that can uniformly approximate ψ by a discriminant, if $\psi(x)$ is a continuous function over a compact set.

Asymptotic bound: Here we present the asymptotic upper bound analysis of ISO-FLANN. As step 1 and step 2 are the preprocessing steps, these requires constant amount of time. The upper bound of step 3 requires $O(N \cdot m)$ time, where N is the size of the swarm and m is the length of the particle. However, without loss of generality, we ignore the length of the particle for subsequent steps.

Next we consider the heart of the algorithms such as while loop. Let us first consider a single iteration of this loop and later by the entire while loop. Let us first consider a single iteration of this loop and later by the entire while loop. For each particle it requires a single scan of the training set, so the time complexity is $O(T_r)$, T_r is the size of the training set. However, other steps are dominated asymptotically. Considering the entire swarm the time complexity will be $O(N.T_r)$. Finally for the entire for loop the asymptotic upper bound of the algorithm is $O(Gen2.N.T_r)$. Hence the asymptotic upper bound of the algorithm is $O(Gen2.N.T_r) + O(N.m) = O(Gen2.N.T_r)$, as $O(N.m)$ is dominated by $O(Gen2.N.T_r)$.

5. Conclusions and future path

In this paper, an improved PSO based Evolutionary Functional Link Artificial Neural Network(ISO-FLANN) model is proposed for the task of classification. Using the weight value obtained by IPSO and the set of optimized trigonometric basis functions chosen for the expansion of the feature vector, the method overcomes

the non-linearity of the classification problem. Further the self-adaptivity Gaussian and Cauchy mutation can further fine-tune the solutions found by the proposed algorithm. Experimental study demonstrated that the performance of ISO-FLANN for classification task is promising. In most cases, the result obtained with the EFLSNN model proved to be as good as or better than the best results found by the MLP, SVM, FLANN with gradient decent and FSN. The architectural complexity of the ISO-FLANN model is quite less compare to MLP, whereas it is the same or less as FLANN with gradient descent and FSN. This property of ISO-FLANN can attract the researches working in data mining for classification task.

Future research include simultaneous evolution of architecture and weights with a Pareto set of solutions. Mapping the input patterns form lower to higher dimension by other functions will constitute another part of study. A rigorous study on the convergence and stability analysis of the proposed method will also be made.

References

Bäck, T., Schwefel, H.P., 1993. An overview of evolutionary algorithms for parameter optimization. Evolutionary Computing 1, 1–23.

Bergh, F.V.D., Engelbrecht, A.P., 2006. A study of particle swarm optimization particle trajectories. Information Sciences 176 (8), 937–971.

Blake, C.L., Merz, C.J. UCI Repository of Machine Learning Databases. http://www.ics.uci.edu/ mlearn/MLRepository.html.

Carvalho, M., Ludermir, T.B., 2007. Particle swarm optimization of neural network architectures andweights. In: 7th International Conference on Hybrid Intelligent Systems, 2007, pp. 336–339.

Chang, C.G., Wang, D.W., Liu, Y.C., Qi, B.K., 2007. Application of particle swarm optimization based bp neural network on engineering project risk evaluating. In: Third International Conference on Natural Computation, 2007, ICNC 2007, volume 1, pp. 750–754.

Clerc, M., Kennedy, J., 2002. The particle swarm—explosion, stability, and convergence in a multidimensional complex space. IEEE Transactions on Evolutionary Computation 6 (1), 58–73.

Da, Yi, Xiurun, Ge, 2005. An improved PSO-based ANN with simulated annealing technique. Neurocomputing 63, 527–533.

Decker, R., Kroll, F., 2007. Classification in marketing research by means of LEM2-generated rules. In: Decker, R., Lenz, H.J. (Eds.), Advances in Data Analysis, Studies in Classification, Data Analysis, and Knowledge Organization. Springer Berlin Heidelberg, pp. 425–432.

Dehuri, S., Cho, S.B., 2010a. Evolutionarily optimized features in functional link neural network for classification. Expert System and Applications 37, 4379–4391.

Dehuri, S., Cho, S.B., 2010b. A hybrid genetic based functional link artificial neural network with a statistical comparison of classifiers over multiple datasets. Neural Computing Applications 19, 317–328.

Dehuri, S., Patnaik, S., Ghosh, A., Mall, R., 2008. Application of elitist multi-objective genetic algorithm for classification rule generation. Applied Soft Computing 8 (1), 477–487.

Demšar, J., 2006. Statistical comparisons of classifiers over multiple data sets. Journal of Maching Learning Research 7, 1–30.

Eberhart, R.C., Shi, Y., 2000. Comparing inertia weights and constriction factors in particle swarm optimization. In: Proceedings of the 2000 Congress on Evolutionary Computation 2000. volume 1, pp. 84–88.

García, S., Fernández, A., Luengo, J., Herrera, F., 2010. Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: Experimental analysis of power. Information Sciences 180, 2044–2064.

Ge, H.W., Qian, F., Liang, Y.C., Du, W.L., Wang, L., 2008. Identification and control of nonlinear systems by a dissimilation particle swarm optimization-based elman neural network. Nonlinear Analysis: Real World Applications 9 (4), 1345–1360.

Goldberg, David E., 1989. Genetic Algorithms in Search, Optimization and Machine Learning, 1st edition. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.

Guerra, F.A., Coelho, L.D.S., 2008. Multi-step ahead nonlinear identification of lorenz's chaotic system using radial basis neural network with learning by clustering and particle swarm optimization. Chaos. Solitons & Fractals 35 (5), 967–979.

Hamamoto, Y., Uchimura, S., Tomita, S., 1997. A bootstrap technique for nearest neighbor classifier design. IEEE Transactions onPattern Analysis and Machine Intelligence 19 (1), 73–79.

Han, F., Ling, Q.H., Huang, D.S., 2008. Modified constrained learning algorithms incorporating additional functional constraints into neural networks. Information Sciences 178, 907–919.

Hsu, C.W., Lin, C.J., 2002. A comparison of methods for multi-class support vector machines. IEEE Transactions on Neural Networks 13 (2), 415–425.

Jiang, M., Luo, Y.P., Yang, S.Y., 2007. Stochastic convergence analysis and parameter selection of the standard particle swarm optimization algorithm. Information Processing Letters 102 (1), 8–16.

John Paul, T., Yusiong, Prospero, C., Naval Jr, 2006. Training neural networks using multiobjective particle swarm optimization. In: ICNC (1), pp. 879–888.

Kang, M., Brown, D.P., 2008. A modern learning adaptive functional neural network applied to handwritten digit recognition. Information Sciences 178, 3802–3812.

Kennedy, J., Eberhart, R., 2001. Swarm Intelligence Morgan Kaufmann, 3rd edition. Academic Press, New Delhi, India.

Kennedy, J., Eberhart, R.C., 1995. Particle swarm optimization. In: Proceedings of the IEEE International Conference on Neural Networks, Perth, Australia, pp. 1942–1948.

King, R.D., Feng, C., Sutherland, A., 1995. Statlog: Comparison of classification algorithms on large real-world problems. Applied Artificial Intelligence 9 (3), 289–333.

Liu, H.B., Tang, Y.Y., Meng, J., Ji, Y., 2004. Neural networks learning using vbest model particle swarm optimisation. In: Proceedings of 2004 International Conference on Machine Learning and Cybernetics 2004, volume 5, pp. 3157–3159.

Liu, B., Wang, L., Jin, Y.-H., Tang, F., Huang, D.-X., 2005. Improved particle swarm optimization combined with chaos. Chaos, Solitons & Fractals 25 (5), 1261–1271.

Luengo, J., García, S., Herrera, F., 2009. A study on the use of statistical tests for experimentation with neural networks: Analysis of parametric test conditions and non-parametric tests. Expert System with Applications 36, 7798–7808.

Marinakis, Y., Marinaki, M., Dounias, G., Jantzen, J., Bjerregaard, B., 2009. Intelligent and nature inspired optimization methods in medicine: the pap smear cell classification problem. Expert Systems 26 (5), 433–457.

Mazurowski, M.A., Habas, P.A., Zurada, J.M., Lo, J.Y., Baker, J.A., Tourassi, G.D., 2008. Training neural network classifiers for medical decision making: the effects of imbalanced datasets on classification performance. Neural Networks 21 (2–3), 427–436.

Mishra, B.B., Dehuri, S., 2007. Functional link artificial neural network for classification task in data mining. Journal of Computer Science 3, 948–955.

Mishra, B.B., Dehuri, S., Panda, G., Dash, P.K., 2008. Fuzzy swarm net (FSN) for classification in data mining. The CSI Journal of Computer Science and Engineering 5 (2&4(b)), 1–8.

Pao, Y.H., 1989. Adaptive Pattern Recognition and Neural Networks. Addison Wesley.

Pao, Y.H., Phillips, S.M., Sobajic, D.J., 1992. neural-net computing and intelligent control systems. International Journal of Control Systems 56 (2), 263–289.

Park, H.S., Cho, S.B., 2006. An efficient attribute ordering optimization in Bayesian networks for prognostic modeling of the metabolic syndrome. In: ICIC (3)'06, pp. 381–391.

Pereira, M., Costa, V.S., Camacho, R., Fonseca, N.A., Sim oes, C., Brito, R.M., 2009. Comparative study of classification algorithms using molecular descriptors in toxicological databases. In: Proceedings of the 4th Brazilian Symposium on Bioinformatics: Advances in Bioinformatics and Computational Biology, BSB '09, Berlin, Heidelberg, Springer-Verlag, pp. 121–132.

Quinlan, J.R., 1993. C4. 5: programs for machine learning. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

Rudolph, G., 1997. Local convergence rates of simple evolutionary algorithms with cauchy mutations. IEEE Transactions on Evolutionary Computation 1 (4), 249–258.

Schwefel, H.P., 1981. Numerical Optimization of Computer Models. John Wiley & Sons, Inc., New York, NY, USA.

Shi, Y., Eberhart, R.C., 1999. Empirical study of particle swarm optimization. In: Proceedings of the 1999 Congress on Evolutionary Computation, 1999. CEC 99, volume 3, pp. 6–9.

Trelea, I.C., 2003. The particle swarm optimization algorithm: convergence analysis and parameter selection. Information Processing Letters 85 (6), 317–325.

Wu, J., Jin, L., Liu, M., 2006. Modeling meteorological prediction using particle swarm optimization and neural network ensemble. In: Wang, J., Yi, Z., Zurada, J., Lu, B.L., Yin, H. (Eds.), Advances in Neural Networks - ISNN 2006, volume 3973 of Lecture Notes in Computer Science. Springer, Berlin/Heidelberg, pp. 1202–1209.

Yager, R.R., 2006. An extension of the naive Bayesian classifier. Information Sciences 176 (5), 577–588.

Yang, J.M., Kao, C.Y., 2001. A robust evolutionary algorithm for training neural networks. Neural Computing & Applications 10, 214–230.

Yu, J., Xi, L., Wang, S., 2007. An improved particle swarm optimization for evolving feedforward artificial neural networks. Neural Processing Letters 26, 217–231.

Yung, Y., Shaw, M.J., 1995. Introduction to fuzzy decision tree. Fuzzy Set and Systems 69 (1), 125–139.

Zhang, G.P., 2000. Neural networks for classification: a survey. IEEE Transactions on Systems Man and Cybernetics. Part C: Applications and Reviews 30 (4), 451–462.

Zhang, J., Lok, T., Lyu, M., 2007. A hybrid particle swarm optimization-back-propagation algorithm for feedforward neural network training. Applied Mathematics and Computation 185 (2), 1026–1037.

Zhao, L., Yang, Y., 2009. Pso-based single multiplicative neuron model for time series prediction. Expert System Application 36, 2805–2812.

Further reading

Schutte, J.F., Groenwold, A.A., 2005. A study of global optimization using particle swarms. Journal of Global Optimization 31, 93–108.

Satchidananda Dehuri is a Reader and Head in P.G. Department of Information and Communication Technology, Fakir Mohan University, Vyasa Vihar, Balasore, Orissa. He received his M.Sc. degree in Mathematics from Sambalpur University, Orissa in 1998, and the M.Tech. and Ph.D. degrees in Computer Science from Utkal University, Vani Vihar, Orissa in 2001 and 2006, respectively. He completed his Post Doctoral Research in Soft Computing Laboratory, Yonsei University, Seoul, Korea under the BOYSCAST Fellowship Program of DST, Govt. of India. In 2010 he received Young Scientist Award in Engineering and Technology for the year 2008 from Odisha Vigyan Academy, Department of Science and Technology, Govt. of Odisha. He was at the Center for Theoretical Studies, Indian Institute of Technology Kharagpur as a Visiting Scholar in 2002. During May–June 2006 he was a Visiting Scientist at the Center for Soft Computing Research, Indian Statistical Institute, Kolkata. His research interests include Evolutionary Computation, Neural Networks, Pattern Recognition, Data Warehousing and Mining, Object Oriented Programming and its Applications and Bioinformatics. He has already published about 100 research papers in reputed journals and referred conferences, has published three text books for undergraduate and Post graduate students and edited five books, and is acting as an editorial member of various journals. He chaired the sessions of various International Conferences.

Rahul Roy is a research fellow in Machine Intelligence unit of Indian Statistical Institute, Kolkata. He did his M.Tech. student in School of Computer Engineering, KIIT University, Bhubaneswar, Orissa, India. He completed his Integrated M.Sc. degree in Computer science from Assam University, Silchar, India, in 2009. His research interest includes Evolutionary Computation; Bio inspired computing and Rule mining.

Sung-Bae Cho received the Ph.D. degrees in computer science from KAIST (Korea Advanced Institute of Science and Technology), Taejeon, Korea, in 1993. He was an Invited Researcher of Human Information Processing Research Laboratories at ATR (Advanced Telecommunications Research) Institute, Kyoto, Japan from 1993 to 1995, and a Visiting Scholar at University of New South Wales, Canberra, Australia in 1998. He was also a Visiting Professor at University of British Columbia, Vancouver, Canada from 2005 to 2006. Since 1995, he has been a Professor in the Department of Computer Science, Yonsei University. His research interests include neural networks, pattern recognition, intelligent man-machine interfaces, evolutionary computation, and artificial life. He is a Senior Member of IEEE and a Member of the Korea Information Science Society, the IEEE Computer Society, the IEEE Systems, Man, and Cybernetics Society, and the Computational Intelligence Society.

Ashish Ghosh is a Professor of the Indian Statistical Institute, Kolkata, India. He received the B.E. degree in Electronics and Telecommunication from the Jadavpur University, Kolkata, in 1987, and the M.Tech. and Ph.D. degrees in Computer Science from the Indian Statistical Institute, Kolkata, in 1989 and 1993, respectively. He received the prestigious and most coveted Young Scientists award in Engineering Sciences from the Indian National Science Academy in 1995; and in Computer Science from the Indian Science Congress Association in 1992. He has been selected as an Associate of the Indian Academy of Sciences, Bangalore, in 1997. He visited the Osaka Prefecture University, Japan, with a Post-doctoral fellowship during October 1995 to March 1997, and Hannan University, Japan as a Visiting Faculty during September to October, 1997 and September to October, 2004. He has also visited Hannan University, Japan, as Visiting Professor with a fellowship from Japan Society for Promotion of Sciences (JSPS) during February to April, 2005. During May 1999 he was at the Institute of Automation, Chinese Academy of Sciences, Beijing, with CIMPA (France) fellowship. He was at the German National Research Center for Information Technology, Germany, with a German Government (DFG) Fellowship during January to April, 2000, and at Aachen University, Germany in September 2010 with an European Commission Fellowship. During October to December, 2003 he was a Visiting Professor at the University of California, Los Angeles, and during December 2006 to January 2007 he was at the Computer Science Department of Yonsei University, South Korea. His visits to University of Trento and University of Palermo (Italy) during May to June 2004, March to April 2006, May to June 2007, 2008, 2009 and 2010 were in connection with collaborative international projects. He also visited various Universities/Academic Institutes and delivered lectures in different countries including Poland and the Netherlands.

His research interests include Pattern Recognition and Machine Learning, Data Mining, Image Analysis, Remotely Sensed Image Analysis, Video Image Analysis, Soft Computing, Fuzzy Sets and Uncertainty Analysis, Neural Networks, Evolutionary Computation and Bioinformatics. He has already published more than 120 research papers in internationally reputed journals and referred conferences, has edited 8 books and is acting as a member of the editorial board of various international journals.

He is a member of the founding team that established a National Center for Soft Computing Research at the Indian Statistical Institute, Kolkata, in 2004, with funding from the Department of Science and Technology (DST), Government of India, and at present is the In-Charge of the Center.