Recognition of Handwritten Digits Using Optimized Adaptive Neuro-Fuzzy Inference Systems and Effective Features

Amir Bahador Bayat

Tafresh University, Iran

amirbahador87@ymail.com

***Abstract*-Automatic recognition of handwritten characters has long been a goal of many research efforts in the pattern recognition field. This paper investigates the design of a high efficient system for recognition of handwritten digits. First it proposes an efficient system that includes two main modules: the feature extraction module and the classifier module. In the feature extraction module, seven sets of discriminative features are extracted and used in the recognition system. In the classifier module, as the first time in this area, the adaptive neuro-fuzzy inference system (ANFIS) is investigated. Experimental results show that the proposed system has good Recognition Accuracy (RA). However, the results show that in ANFIS training, the vector of radius has very important role for its recognition accuracy. At the second fold, it proposes an intelligence system in which a novel optimization module, i.e., improved bees algorithm (IBA) is proposed for finding the best parameters of the classifier. In test stage, 3-fold cross validation method was applied to the MNIST handwritten numeral database to evaluate the proposed system performances. Simulation results show that the proposed system has high recognition accuracy.**

Keywords- Handwritten Digits; ANFIS; MNIST; IBA; Optimization

I. INTRODUCTION

In recent years, handwriting recognition has become one of the hottest and challenging directions in the field of image processing and pattern recognition [1]. New technologies and methods have been proposed continuously. With the development of the smart phone operation system, the application in handwritten recognition has aroused more and more attention from researchers.

In general, handwritten character recognition is classified into two types of offline and online recognition methods [2]. Taking the need of actual application into consideration, our study only aims at offline handwritten digits recognition.

In areas of automatic document analysis and recognition, the correct interpretation of handwritten digits is very important. Automatic recognition of handwritten digits is difficult due to several reasons, including different writing styles of different persons, different writing devices, and the context of the digit. This leads to digits of different sizes and skews, and strokes that vary in width and shape.

An optical character recognition (OCR) system with a good recognition performance needs to maintain a very high recognition rate, and at the same time, to obtain a very high reliability, or a very low error rate [3, 4]. Recent developments on classifiers and feature extraction have significantly increased the recognition accuracy of handwritten digit recognition systems.

Among popular handwritten digit databases (e.g. MNIST database, NIST Special Database 19, CENPARMI database and CEDAR isolated digit database), the MNIST database has been widely used in recent years as a benchmark for evaluating new classifiers or testing new feature extraction methods.

In [5], a recognition system for the unconstrained hand printed numerals was proposed, which used topological, geometrical and local measurements to identify the character or to reject the character as unrecognizable. The recognition system yielded a recognition rate of 97% with a substitution error rate of 0.3% and a rejection rate of 2.7%.

In Stringa [6], a pattern recognition system was applied to the unconstrained alphanumeric character recognition. The recognition system was designed to allow hierarchical re-description of the input images and the phrase-structure grammars were developed. The experiments conducted on handwritten digits indicated that the recognition rates were comparable to the best OCR system at that time, but with a considerable reduction in computing time. In Suen et al. [7], four experts for the recognition of handwritten digits were proposed. Mitchell and Gillies [8] used the tools of mathematical morphology to extract cavity features as the starting input for their specialized digit recognizers. A classification system was implemented by a symbolic model matching process. Le Cun et al. [9] achieved excellent results with the convolutional neural networks, which were specifically designed to deal with the variability of two dimensional (2-D) shapes. For the recognition of handwritten numerals, the recognition rate with this method could be as high as 99.18% on the MNIST database. In [10], authors expanded the training set of the MNIST dataset by adding a new form of distorted data, and the convolutional neural networks were better suited for classification purposes. The recognition rate was achieved at 99.60%. Shi et al. [11] proposed a handwritten digit recognition system using the gradient and curvature of the gray character image in order to improve the accuracy of

handwritten numeral recognition. The experiments were conducted on IPTP CDROM1, NIST SD3, and SD7 databases. The recognition rates could reach from 98.25% to 99.49%. In [12] authors proposed a handwritten digit recognition system based on a biological vision model. The features were empirically extracted by the model, which could linearly separate over a large training set (MNIST). The high recognition rate was reported, where the error rate was 0.59%. In [13] it proposed a handwritten digit recognition system where the prior knowledge about invariance of a classification problem was incorporated into the training procedure. Support Vector Machines (SVMs) were used as classifiers. The system achieved a low error rate of 0.56% when using this procedure with the MNIST dataset.

Recently, many handwritten digit recognition systems with very high recognition rates have emerged. These recognition systems were conducted on the well-known MNIST database. Here are some examples:

- 99.58% of SVCs on gradient features [14],
- 99.41% of LIRA_grayscale [15],
- 99.46% of Trainable Feature Extractor and Support Vector Machine (TFE-SVM)with affine transformations for increasing the training set [16],
- 99.56% of Image Recognition Systems with Permutative Coding [17],
- 99.63% of Support Vector Machine (VSVM) [18, 19],

A comprehensive survey on handwritten numeral recognition by using different feature extraction methods, and different classifiers on CENPARMI, CEDAR, MNIST databases has been reported in [20]. The classifiers included one k-nearest classifier, three neural classifiers, a learning vector quantization classifier, a discriminative learning quadratic function classifier and two support vector classifiers. On the MNIST test dataset, 80 recognition results were given by combining eight classifiers with ten feature vectors. The error recognition rates were between 1.50 and 0.61.

Based on the published papers, there exist some important issues in the design of automatic OCR system which if suitably addressed, lead to the development of more efficient recognizers. One of these issues is the extraction of the features. In this paper for obtaining the compact set of features which capture the prominent characteristics of the handwritten digits in a relatively small number of the components, seven sets of discriminative features are extracted. These features are presented in Section 2.

Another issue is related to the choice of the classification approach to be adopted. The developed method uses fuzzy rules for recognition task. In the proposed method, an expert system has been developed which has fuzzy rules obtained by ANFIS. ANFIS represents the promising new generation of information processing systems. Adaptive network based fuzzy inference systems are good at tasks such as pattern matching and classification, function approximation, optimization and data clustering, while traditional computers, because of their architecture, are inefficient at these tasks, especially pattern-matching tasks [21- 23].

In ANFIS training process, the vector of radius has high efficiency on the performance of system. In order to increase the accuracy of proposed system, we intend to find the optimum vector of radius using the optimization algorithm. In the proposed method improved bees algorithm (IBA) is used for finding the optimum vector of radius because it has more robust performance than other intelligent optimization methods have proved. The computational simulations reveal very encouraging results in terms of the quality of solution and the processing time required [24, 25].

The rest of paper is organized as follows. Section 2 explains the feature extraction. Section 3 presents the classifier. Section 4 presents the optimization method. Section 5 shows some simulation results and finally Section 6 concludes the paper.

II. FEATURE EXTRACTION

A. Seven sets of Features

Feature extraction is a vital step in pattern recognition. In this section, seven sets of features are extracted. We use MNIST handwritten digit database, which includes 60,000 training samples and 10,000 testing samples. All the digit images in the MNIST database are grayscale images with 28×28 sizes. In the preprocessing, each 28×28 grayscale digit image is binarized and normalized to a size of 32×32. These feature sets and the methods of extracting them are summarized below:

Feature set 1: gradient-based wavelet features. We use Kirsch nonlinear edge enhancement algorithm to extract statistical features from the characters and apply wavelet transform on these statistical features to form original features. The directional-based feature extraction is implemented as follows: firstly, the Kirsch nonlinear edge enhancement algorithm is applied to a 32×32 character image to extract horizontal, vertical, right-diagonal and left-diagonal directional feature images and the global feature image; then a two dimensional (2D) wavelet transform is used to filter out the high frequency components of each directional feature image and the global feature image, and to convert the feature matrix into a 4×4 matrix, respectively. In total, $16\times5=80$ features can be extracted from each character. More details regarding the algorithm can be found in [26].

Feature set 2: MAT-based directional features. MAT is a method of highlighting the center skeleton of the character strokes with maximum grayscale values and keeping the stroke information and local information, which has richer information for image processing and pattern recognition. Our iterative MAT algorithm is implemented as follows:

- Design the structure matrix of erosion as a 3×3 matrix E with all elements being set to 1 and set the initial iteration number as 1;
- Erode a 32 × 32 character image Im by the morphological erosion operator E, and the value of the eroded pixel in the character image is set equal to the current iteration number; 1(c) Increase the iteration number by 1, then repeat step 2 until no more new eroded pixels are created.

After getting the MAT transformed images, we can use the following steps to extract MAT gradient-based features:

- Normalize the MAT image with its pixel values from 0.0 to 1.0;
- Convolute the normalized character image Iz with a 3 × 3 Sobel operator to generate the amplitudes and phases of the gradient image.

The X-gradient character image can be calculated by:

$$Ix = Iz * Sx$$

and the Y -gradient character image is calculated by

$$Iy = Iz * Sy$$

The gradient magnitude and phase are then obtained by

$$r(i,j) = \sqrt{Ix^2(i,j) + Iy^2(i,j)}$$

$$\theta(i,j) = \tan^{-1}\frac{Iy^2(i,j)}{Ix^2(i,j)}$$

- Count the gradient direction of each pixel of the convoluted image with non-zero gradient magnitude values as a direction feature.

In order to generate a fixed number of features, each gradient direction is quantized into one of eight directions at $\pi/4$ intervals. Each normalized gradient image is divided into 16 sub-images. The number in each direction of each sub-image is counted as a feature. In total, the number of features is 4 × 4 × 8 = 128.

Feature set 3: complex wavelet features. A 2D complex wavelet transform (2D-CWT) not only keeps the wavelet transform's properties of multi-resolution and perfect reconstruction, etc., but it also adds new merits such as insensitivity of its magnitude to small image shifts, and a multiple directional selectivity [27]. A feature extraction scheme using CWT was proposed in Ref. [28]. In this case, the number of extracted features = 4 × 4 (for each subband image) * 3 (high frequency subband images for each tree) * 2 (trees) +4 × 4 (for each subband image) * 2 (trees) * 2 (parts: real and imaginary) =160.

Feature set 4: binary gradient directional features. This feature extraction method is the same as that of MAT-based directional features except that no MAT transform is needed. The gradient directional features are extracted directly from each binary character image. In total, a feature vector consisting of 128 (features) was extracted for each handwritten character image.

Feature set 5: median filter gradient features. The following steps are used to extract the median filter gradient features: firstly, filter the character image by a 2D median filter; then apply Robert operator [29] on the resulting image to generate the amplitudes and phases; finally, apply the gradient feature extraction method, which was described in feature set 2, to extract 128 gradient features.

Feature set 6: image thinning distance features. In Feature set VI, a 32×32 character image is thinned and scaled into an 8×8 array. The thinned image is scanned both horizontally and vertically. In the horizontal scanning, for each pixel in the 8×8 thinned image, if the value of the pixel is 0 (black), then the distance is 0; otherwise, it is the distance from that pixel to the nearest black pixel in both horizontal directions on the same scanning line. In the vertical scanning, the same algorithm is applied. There were 128 extracted features.

Feature set 7: geometrical features. In this feature extraction method, in order to explore character geometric features, we used concave features on the character's four profiles, middle line features, horizontal segment features on the left and right profiles, character width features in the top five rows, middle 10 rows, and bottom five rows, as well as endpoint and crossing point features as geometrical features [30]. These features were encoded as 20 features. Conceptually speaking, the first six feature sets consisted of statistical features. The last one was a structural feature set.

B. Feature Rank with a Divergence Criterion

In multi-class pattern recognition, there are C classes, and each class is represented as y_i. The domain of the multi-class case can be noted as: Y= (y1, y2, y3,..., y_c).

Each classy_j has M_j samples in the training database. A feature vector consists of d features: X=(x1, x2, x3,..., x_d). For each feature x_j, we can extract M_j feature values for each of the C classes, which are denoted by $X_j^n=(X_{j,1}^n, X_{j,2}^n, \ldots, X_{j,Mj}^n)$. j=1, 2, 3,..., d, where d is the total number of features; n: n=1, 2, 3,..., C, where C is the total number of classes; M_j: number of training samples in the jth class. The mathematical expectation and variance of each sub-feature vector for each class is denoted as follows:

$$U_{j,n}= E[x_j^n] = \frac{1}{Mj}\sum_{k=1}^{M_j} x_{j,k}^n$$

$$\delta_{j,n}^2 = E\left[\left(x_j^n - U_{j,n}\right)^2\right] = \frac{1}{Mj}\sum_{k=1}^{M_j}(x_{j,k}^n - U_{j,n})^2$$

In the above two equations, u_j, $\delta_{j,n}^2$ and represent the expected value and variance of the jth feature in the nth class obtained from the training set. According to our analysis, we can calculate the divergence (coefficient) for feature mbased on C classes as follows:

$$D(m) =\sum_{l=1}^{c}\sum_{r=1}^{c} D_{l,r}(m) * (1 - P_{l.r}(err))$$

$$D_{l,r}=\frac{|U_{m,l}-U_{m,r}|^2}{2\delta_{r,m}^2} + \frac{|U_{m,r}-U_{m,l}|^2}{2\delta_{l,m}^2}$$

Where $P_{l.r}(err)$ is the misrecognition rate of the lth class, being recognized as the rth class. We can obtain $P_{l.r}(err)$ by training classifiers using the training samples, and testing the classifiers on the test set without any feature selection beforehand. Through Eqs. above, less weight is put on those class pairs, which are inclined to cause more errors. As a result, those classes which are more easily misrecognized will have less power to dominate the divergence for feature ranking, thereby decreasing the recognition errors and improving the recognition performance.

We can re-rank the seven feature sets I–VII to form seven newly ranked feature sets according to each feature's divergence coefficient calculated through Eq. above; then, three new random feature sets are constructed by randomly choosing feature components, which have larger divergence values, from the seven newly ranked feature sets. The three new random feature sets are called random feature set I (200), random feature set II (218) and random feature set III (240). The number in the parentheses is the number of dimensions.

III. ANFIS

The ANFIS represents a useful neural network approach for the solution of function approximation problems. Data driven procedures for the synthesis of ANFIS networks are typically based on clustering a training set of numerical samples of the unknown function to be approximated. Since introduction, ANFIS networks have been successfully applied to classification tasks, rule-based process controls, pattern recognition problems and the like. Here a fuzzy inference system comprises of the fuzzy model [31- 32] proposed by Takagi, Sugeno and Kang to formalize a systematic approach to generate fuzzy rules from an input output data set.

A. ANFIS Structure

For simplicity, it is assumed that the fuzzy inference system under consideration has two inputs and one output. The rule base contains two fuzzy if-then rules of Takagi and Sugeno's type [33] as follows:

$$If \text{ x is A and y is B then z is f(x,y)}$$

Where *A* and *B* are the fuzzy sets in the antecedents and *z=f(x,y)* is a crisp function in the consequent. *z=f(x,y)* is usually a polynomial for the input variables x and y. But it can also be any other function that can approximately describe the output of the system within the fuzzy region as specified by the antecedent. When *z=f(x,y)* is a constant, a zero order Sugeno fuzzy model is formed, which may be considered to be a special case of Mamdani fuzzy inference system [34] where each rule consequent is specified by a fuzzy singleton. If *z=f(x,y)* is taken to be a first order polynomial a first order Sugeno fuzzy model is formed. For a first order two-rule Sugeno fuzzy inference system, the two rules may be stated as:

$$\text{Rule 1}: If \text{ x is } A_1 \text{ and y is } B_1 \text{ then } f_1=p_1x+q_1y+r_1$$

$$\text{Rule 2}: If \text{ x is } A_2 \text{ and y is } B_2 \text{ then } f_1=p_2x+q_2y+r_2$$

Here type-3 fuzzy inference system proposed by Takagi and Sugeno [35] is used. In this inference system the output of each rule is a linear combination of input variables added by a constant term. The final output is the weighted average of each rule's output. The corresponding equivalent ANFIS structure is shown in Fig. 1.

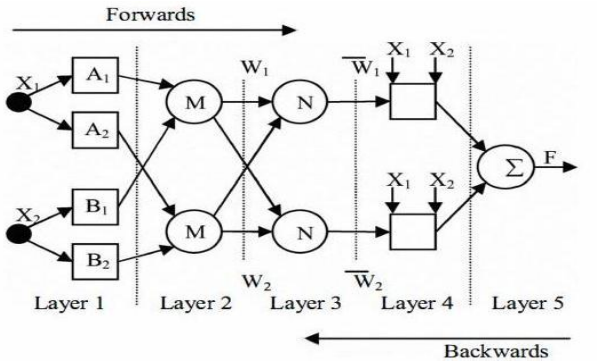

Fig. 1 ANFIS structure

The individual layers of this ANFIS structure are described below:

Layer 1: Every node i in this layer is adaptive with a node function

$$O_i^1 = \mu_{A_i}(x) \tag{1}$$

where x is the input to node i, A_i the linguistic variable associated with this node function and μ_{A_i} is the membership function of A_i. Usually $\mu_{A_i}(x)$ is chosen as

$$\mu_{A_i}(x) = \frac{1}{1+[(x-c_i)/a_i)^2]^{b_i}} \tag{2}$$

Or

$$\mu_{A_i}(x) = \exp\left\{-\left(\frac{x-c_i}{a_i}\right)^2\right\} \tag{3}$$

where x is the input and $\{a_i, b_i, c_i\}$ is the premise parameter set.

Layer 2: Each node in this layer is a fixed node which calculates the firing strength w_i of a rule. The output of each node is the product of all the incoming signals to it and is given by

$$O_i^2 = w_i = \mu_{A_i}(x) \times \mu_{B_i}(x), \quad i=1, 2 \tag{4}$$

Layer 3: Every node in this layer is a fixed node. Each *i*th node calculates the ratio of the ith rule's firing strength to the sum of firing strengths of all the rules. The output from the *i*th node is the normalized firing strength given by

$$O_i^3 = \overline{w} = \frac{w_i}{w_1 + w_2}, \quad i=1,2 \tag{5}$$

Layer 4: Every node in this layer is an adaptive node with a node function given by

$$O_i^4 = \overline{w_i} f_i = \overline{w_i}(p_i x + q_i y + r_i) \tag{6}$$

where $\overline{w}_i$ is the output of Layer 3 and $\{p_i, q_i, r_i\}$ is the consequent parameter set.

Layer 5: This layer comprises of only one fixed node that calculates the overall output as the summation of all incoming signals, i.e.

$$O_i^5 = overall\ \text{output} = \sum_i \overline{w}_i f_i = \frac{\sum_i w_i f_i}{\sum_i w_i} \tag{7}$$

B. Learning Algorithm

From the proposed ANFIS structure, it is observed that given the values of premise parameters, the final output can be expressed as a linear combination of the consequent parameters. The output f in Fig. 1 can be written as

$$\begin{aligned} f &= \frac{w_1}{w_1 + w_2} f_1 + \frac{w_2}{w_1 + w_2} f_2 = \overline{w}_1 f_1 + \overline{w}_2 f_2 = \\ &(\overline{w}_1 x) p_1 + (\overline{w}_1 y) q_1 + (\overline{w}_1) r_1 + (\overline{w}_2 x) p_2 + (\overline{w}_2 y) q_2 + (\overline{w}_2) r_2 \end{aligned} \tag{8}$$

f is linear in the consequent parameters $\{p_1, q_1, r_{1,} p_2, q_2, r_{2,}\}$.

In the forward pass of the learning algorithm, consequent parameters are identified by the least squares estimate. In the backward pass, the error signals, which are the derivatives of the squared error with respect to each node output, propagate backward from the output layer to the input layer. In this backward pass, the premise parameters are updated by the gradient descent algorithm [36–38].

C. Derivation of the Initial Fuzzy Model

As described earlier, in ANFIS based system modeling for a set of rules with fixed premises, identification of an optimal fuzzy model with respect to the training data reduces to a linear least squares estimation problem. A fast and robust method for identification of fuzzy models from input–output data was proposed by Chiu [39, 40]. This method selects the important input variables when building fuzzy model from data by combining cluster estimation method with a least squares estimation algorithm. The method follows in two steps: (i) first step involves extraction of an initial fuzzy model from input output data by using a cluster estimation method incorporating all possible input variables; (ii) in the next step the important input variables are identified by testing the significance of each variable in the initial fuzzy model.

D. Extracting the Initial Fuzzy Model

To start the modeling process, an initial fuzzy model has to be derived. This model is required to find the number of inputs, number of linguistic variables and hence the number of rules in the final fuzzy model. The initial model is also required to select the input variables for the final model and also the model selection criteria, before the final optimal model can be derived. As a first step towards extracting the initial fuzzy model the subtractive clustering technique [39] is applied to the input-output data pairs, which are obtained from the system which is to be modeled. The cluster estimation technique helps in locating the cluster centers of the input output data pairs. This in turn helps in the determination of the rules which are scattered in input output space, as each cluster center is an indication of the presence of a rule. In addition to this it also helps to determine the values of the premise parameters. This is important because an initial value, which is very close to the final value, will eventually result in the quick convergence of the model towards its final value during the training session with neural network. In this clustering technique the potentials of all the input output data points are calculated as functions of their Euclidian distances from all the other data points. The points having a potential above a certain preset value are considered as cluster centers. After the cluster centers are ascertained the initial fuzzy model can be subsequently extracted as the centers will also give an indication of the numbers of linguistic variables. More details regarding the extracting of the initial fuzzy model can be found in Refs. [39- 41].

E. Analysis of an Ensemble Classifier

According to the principle of divide and conquer, a complex task can be solved by dividing it into a number of computationally simpler tasks. The simpler tasks can be achieved by distributing the tasks to a number of experts. For example, one way is to divide the input space into a set of subspaces. Each expert works on an individual subspace. The combination of experts is said to constitute an ensemble classifier. The responses of several experts are combined to produce an overall output.

Fig. 2 shows the block diagram of an ensemble classifier. For simplicity, it is assumed that input xi, (i=1, 2….n) is either an individual feature component or a feature vector, whose outputs are somehow combined to produce an overall output y.

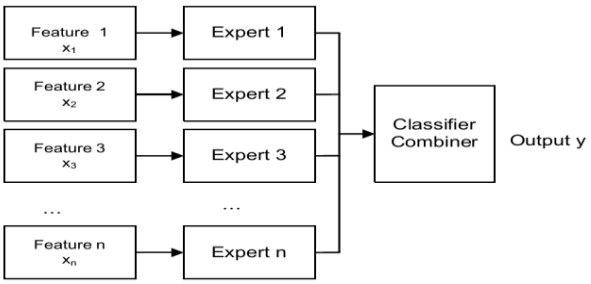

Fig. 2 Block diagram of an ensemble classifier

Generally speaking, in a classification problem, the goal of the classification is to predict the output value Y (where Y is a label vector $[y_1, y_2,\ldots, y_c]^T$ with Clements, which denotes C classes with one corresponding to the correct class, and all others corresponding to zero), given the values of a set of input features $X=\{x_1, x_2,\ldots, x_n\}$ simultaneously measured on the same system.

IV. OPTIMIZATION METHOD

A. Original Bees Algorithm (BA)

BA is an optimization algorithm inspired by the natural foraging behavior of honey bees to find the optimal solution. Fig. 3 shows the pseudo-code for the algorithm in its simplest form. The algorithm requires a number of parameters to be set, namely: Number of scout bees (*n*), number of sites selected out of n visited sites (*m*), number of best sites out of m selected sites (*e*), number of bees recruited for beste sites (*nep*), number of bees recruited for the other (*m-e*) selected sites (*nsp*), initial size of patches (*ngh*) which includes site and its neighborhood and stopping criterion. The algorithm starts with the n scout bees being placed randomly in the search space. The fitnesses of the sites visited by the scout bees are evaluated in step 2.

1. Initialise the solution population.
2. Evaluate the fitness of the population.
3. While (stopping criterion is not met)
//Forming new population.
4. Select sites for neighbourhood search.
5. Recruit bees for selected sites (more bees for the best *e* sites) and evaluate fitnesses.
6. Select the fittest bee from each site.
7. Assign remaining bees to search randomly and evaluate their fitnesses.
8. End While

Fig. 3 Pseudo code

In step 4, bees that have the highest fitnesses are chosen as "selected bees" and sites visited by them are chosen for neighborhood search. Then, in steps 5 and 6, the algorithm conducts searches in the neighborhood of the selected sites, assigning more bees to search near to the best e sites. The bees can be chosen directly according to the fitnesses associated with the sites they are visiting. Alternatively, the fitness values are used to determine the probability of the bees being selected. Searches in the neighborhood of the best e sites which represent more promising solutions are made more detailed by recruiting more bees to follow them than the other selected bees. Together with scouting, this differential recruitment is a key operation of the BA.

However, in step 6, for each patch only the bee with the highest fitness will be selected to form the next bee population. In nature, there is no such a restriction. This restriction is introduced here to reduce the number of points to be explored. In step 7, the remaining bees in the population are assigned randomly around the search space scouting for new potential solutions. These steps are repeated until a stopping criterion is met. At the end of each iteration, the colony will have two parts to its new population representatives from each selected patch and other scout bees assigned to conduct random searches [24].

B. Improved Bees Algorithm (IBA)

In order to improve the convergence velocity and accuracy of the BA, this article recommends an IBA. In BA ngh defines the initial size of the neighborhood in which follower bees are placed. For example, if X is the position of an elite bee in the ith

dimension, follower bees will be placed randomly in $X_{ie} \pm ngh$ in that dimension at the beginning of the optimization process. As the optimization advances, the size of the search neighborhood gradually decreases to facilitate fine tuning of the solution.

For each of the m selected sites, the recruited bees are randomly placed with uniform probability in a neighborhood of the high fitness location marked by the scout bee. This neighborhood (flower patch) is defined as an n-dimensional hyper box of sides $a_1,....,a_n$ that is centered on the scout bee. For each flower patch, the fitness of the locations visited by the recruited bees is evaluated. If one of the recruited bees lands in a position of higher fitness than the scout bee, that recruited bee is chosen as the new scout bee. At the end, only the fittest bee of each patch is retained. The fittest solution visited so-far is thus taken as a representative of the whole flower patch. This bee becomes the dancer once back at the hive.

In BA, the size of a patch is kept unchanged as long as the local search procedure yields higher points of fitness. If the local search fails to bring any improvement in fitness, the size a is decreased. The updating of the neighborhood size follows the following heuristic formula

$$\text{ngh (t+1)} = 0.8 \times \text{ngh(t)} \tag{9}$$

where t denotes the tth iteration of the BA main loop.0.8 is an experimental value. Thus, following this strategy, this number is the optimal one for other handwritten datasets.

Thus, following this strategy, the local search is initially defined over a large neighborhood, and has a largely explorative character. As the algorithm progresses, a more detailed search is needed to refine the current local optimum. Hence, the search is made increasingly exploitative, and the area around the optimum is searched more thoroughly.

Since the search process of BA is nonlinear and highly complicated, linearly and nonlinearly decreasing size of patch with no feedback taken from the elite bees fitnesses cannot truly reflect the actual search process. In the beginning of the search process, the bees are far away from the optimum point and hence a big patch size is needed to globally search the solution space. Conversely, when the best solution found by the population improves greatly after some iteration, i.e., the bees find a near optimum solution, only small movements are needed and patch size must be set to small values. Based on this, in this study, we proposed IBA in which the patch size is set as a function of elite bees fitness during the search process of BA as follows:

$$ngh_i^t = \frac{1}{t} \times \frac{ngh}{1+\exp(-F(elite_i^t))} \tag{10}$$

where $F(elite_i^t)$ is the fitness of ith elite bee in tth iteration and ngh_i^t is the ith elite bees size of the neighborhood in tth iteration. In this case, patch size changes according to the rate of elite bee fitness improvement.

According to Eq. (10), during the search of IBA, while the fitness of an elite bee is far away from the real global optimal, value of patch size will be large resulting in strong global search abilities and locating the promising search areas. Meanwhile, when the fitness of an elite bee is achieved near the real global optimal, the patch size will be set small, depending on the nearness of its fitness to the optimal value, to facilitate a finer local explorations and hence accelerate convergence.

The main difference between BA and IBA is in the patch size definition. First, in IBA, the patch size is associated with fitness value (Eq. (10)). Second, in BA patch size is the same for all elite bees; meanwhile in IBA every elite bee has its own patch size [25].

C. IBA-ANFIS

The ANFIS model was developed using MATLAB Fuzzy Logic Toolbox (2009). A subtractive fuzzy clustering was generated to establish a rule base relationship between the input and output parameters. The data were divided into groups called as clusters using the subtractive clustering method to generate fuzzy inference system. In this study, the Sugeno-type fuzzy inference system was implemented to obtain a concise representation of a system's behavior with a minimum number of rules. The linear least square estimation was used to determine each rule's consequent equation. A radius value was given in the MATLAB program to specify the cluster center's range of influence to all data dimensions of both input and output. If the cluster radius was specified a small number, then there will be many small clusters in the data that results in many rules. In contrast, specifying a large cluster radius will yield a few large clusters in the data resulting in fewer rules [22]. For example, if the data dimension is 3 (e.g., input has two columns and output has one column), radii = [0.5 0.4 0.3] specifies that the ranges of influence in the first, second, and third data dimensions (i.e., the first column of input, the second column of input, and the column of output) are 0.5, 0.4, and 0.3 times the width of the data space, respectively. Therefore in this study IBA-ANFIS is proposed to find the optimum vector of radius.Fig.4shows a sample bee. In this figure p denotes the number of input-output variables.

$$bee = [radius_1, radius_2, ..., radius_p]$$

Fig. 4 Sample of bee

Based on the above descriptions, the flowchart of the IBA-ANFIS algorithm used in this paper is shown in Fig. 5. Detailed description of each step is given below:

Step 1: Unprocessed data

- For this purpose we have used the MNIST database.

Step 2: Feature extraction

- For this purpose we have used the features that described in section 2.

Step 3: Determine the optimum vector of radius

For this purpose IBA was used as optimization algorithm.

- *Initialization*

Randomly generate a position for each candidate in [0, 1].

- *Fitness evaluation*
- *Local search*
- *Global bests*
- *Check the termination criteria*

If the termination condition is not satisfied, go to step 3-2, otherwise stop the algorithm.

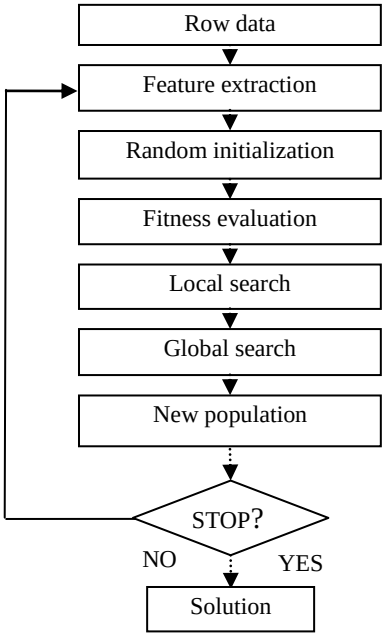

Fig. 5 Flowchart of the proposed method

V. SIMULATION RESULT

In this section we evaluate the performance of proposed recognizer. The MNIST database was used to train and test the proposed system. The MNIST database included 60,000 training samples and 10,000 testing samples. In order to compare the performance of classifiers, the k-fold cross validation technique is used. The k-fold cross validation technique proposed by salzberg [42] was employed in the experiments, with k=3. The data set was thus split into three portions, with each part of the data sharing the same proportion of each class of data. Two data portions were used in the training process, while the remaining part was used in the testing process. The ANFIS training methods were run three times to allow each slice of the data to take turn as a testing data. The classification accuracy rate is calculated by summing the individual accuracy rate for each run of testing, and then dividing of the total by three. All the obtained results are the average of 50 independent runs.

A. Performance Without Optimization

First we have evaluated the performance of the recognizer without optimization. Table 1 shows the RA of different systems. From Table 1 it can be seen that ANFIS with unprocessed data achieves 96.34% recognition accuracy. Its performance increases with using proposed features value up to 97.74%.

TABLE I RECOGNITION ACCURACY OF THE RECOGNIZER WITHOUT OPTIMIZATION

Classifier	Input	Recognition accuracy (%)		
		Mean	Min	Max
ANFIS	Unprocessed data	96.34	94.65	96.86
ANFIS	**Proposed features**	**97.74**	**96.21**	**97.95**

B. Performance with Optimization

Next, we apply IBA to find the optimum vector of radius. Table 2 compares the performance of (IBA-ANFIS) model using row data and that using the proposed features. Combining the IBA with the ANFIS (IBA-ANFIS), we demonstrate a significantly improved performance relative to the stand-alone ANFIS model. The highest recognition accuracy (99.52%) is achieved with only 56 fuzzy rules. The reduction in the number of features also contributes in the reduction of fuzzy rules in the developed fuzzy model from approximately 342 rules to 56 rules; this contributes to reducing the computational complexity of the overall system.

TABLE II RECOGNITION ACCURACY OF THE RECOGNIZER WITH OPTIMIZATION

Classifier	Input	Recognition accuracy (%)		
		Mean	Min	Max
IBA-ANFIS	Unprocessed data	96.43	95.66	97.33
IBA-ANFIS	**Proposed features**	**99.52**	**99.56**	**99.47**

C. Confusion Matrix

In order to indicate the details of the recognition for each pattern, the confusion matrix of the recognizer is shown by Table 3. The values in the diagonal of confusion matrix show the correct performance of recognizer for each pattern. In other words, these values show that how many considered patterns are recognized correctly by the system. The other values show the mistakes of system. For example, look at the fourth row of this matrix. The value of 99.50% shows the percentage of correct recognition of "**3**" pattern and the value of 0.50% shows that this type of pattern is wrongly recognized with "**8**" pattern. In order to achieve the recognition accuracy (RA) of system, it is needed to compute the average value of that appears in diagonal.

TABLE III CONFUSION MATRIX FOR BEST RESULT (99.56%)

	0	1	2	3	4	5	6	7	8	9
0	100	0	0	0	0	0	0	0	0	0
1	0	99.5	0	0	0	0	0	0	0	0
2	0	0	99.5	0	0	0	0	0	0	0
3	0	0	0	99.5	0	0	0	0	0.50	0
4	0	0	0	0	100	0	0	0	0	0
5	0	0	0	0	0	100	0	0	0	0
6	0	0	0	0	0	0	100	0	0	0
7	0	0	0	0	0	0.50	0	99.5	0	0
8	0	0	0	0.50	0	0	0	0	99.5	0
9	0	0	0.50	0	0	0	0.50	0	0	99

D. Performance Evaluation with Optimization in Different Runs

In this sub-section, for evaluating the performance of the IBA, five different runs have been performed. Fig. 6 shows a typical increase of the fitness (classification accuracy) of the best individual fitness of the population obtained from proposed system for different runs. As indicated in this figure, its fitness curves gradually improved from iteration 0 to 100, and exhibited no significant improvements after iteration 40 for the five different runs. The optimal stopping iteration to get the highest validation accuracy for the five different runs was around iteration 30–40. In Fig. 7, the accuracy and the speed of bees algorithm and improved bees algorithm are compared. The achieved diagrams show the mean of 50 different runs for both algorithms. As depicted in this figure, the improved bee algorithm has higher accuracy and speed of convergence compared with bees algorithm.

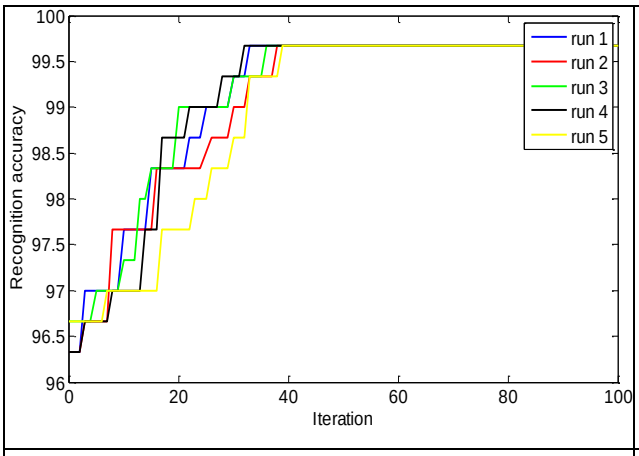

Fig. 6 Evolution of recognition accuracy for different runs

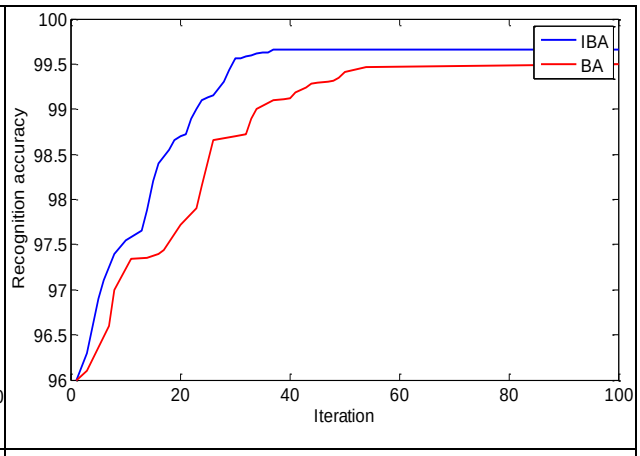

Fig. 7 Performance comparisons of BA and IBA

In order to compare the performance of improved bees algorithm (IBA) with another nature inspired algorithm, we have used several nature inspired algorithms such as genetic algorithm (GA) [43], imperialist competitive algorithm (ICA) [44], particle swarm optimization (PSO) [45] and BA to evolve the ANFIS. Table 4 shows the obtained results. It can be seen that the success rates of IBA-ANFIS is higher than the performance of other systems.

TABLE IV COMPARISON AMONG THE PERFORMANCE OF GA-ANFIS, ICA-ANFIS, BA-ANFIS, PSO-ANFIS AND IBA-ANFIS.

Classifier	Recognition accuracy (%)
GA-ANFIS	99.17
ICA-ANFIS	99.26
PSO-ANFIS	99.31
BA- ANFIS	99.33
IBA-ANFIS	**99.52**

E. Comparison With Different Classifier

The performance of the proposed classifier has been compared with other classifiers for investigating the capability of the proposed classifier, as indicated in Table 5. In this respect, probabilistic neural networks (PNN) [46], radial basis function neural network (RBFNN) [47] and Multilayered perceptron (MLP) neural network with different training algorithms such as: Back propagation (BP) learning algorithm [48] and with resilient propagation (RP) learning algorithm [49] are considered. They comprise parameters which should be readjusted in any new classification. Furthermore, those parameters regulate the classifiers to be best fitted in for classification task. In most cases, there is no classical method for obtaining the values of them and therefore, they are experimentally specified through try and error. It can be seen from Table5 that the proposed method has better recognition accuracy than other classifiers.

TABLE V COMPARISON THE PERFORMANCE OF PROPOSED CLASSIFIER (IBA-ANFIS) WITH OTHER CLASSIFIERS.

Classifier	Recognition accuracy (%)
PNN	98.15
RBF	99.18
MLP (BP)	97.32
MLP (RP)	99.29
IBA-ANFIS	**99.52**

F. Classification Speed

We conducted an experiment to calculate the cascade ensemble classifier system's speed. For example, if we used ANFIS as classifier, and we used the structure shown in Fig. 4 for 10,000 testing samples, then the recognition time was about 94.6 s. This time includes the reading and saving of data from the disk without considering the time for feature extraction. The classification speed for our system is 94.6 s/10,000digits=9.46ms/digit.

For the feature extraction and random feature selection, approximately 10,000 digits took about 282 s, namely: The feature extraction speed: 282,000/10, 000digits=28.2ms/digit. So the recognition time for feature extraction and MLP (BP) classification is about 41ms/digit. Our proposed system can recognize 24 digits per second. All of the experiments were conducted on a Pentium 4 personal computer, CPU 2.80GHz, 1.00GB of RAM.

VI. CONCLUSION

Handwritten character recognition, an almost fifty years old research problem is still very much pertinent due to the enormous variations in writing styles among different writers. This paper investigated the design of a high efficient system for recognition of handwritten digits. In this paper we focused on the improvement of the classical ANFIS model by means of the integration of IBA and ANFIS. Based on the experimental results, this paper recommends the use of proposed system (IBA-ANFIS) for handwritten digits recognition. The complexity of the recognition system is very low in comparison with other works. The highest level of accuracy obtained by ANFIS using unprocessed data was 96.34%. The proposed method improves the accuracy up to 97.74% by using effective features as the classifier inputs. Furthermore, optimizing the structure of the ANFIS and using effective features as the input of optimized classifier (IBA-ANFIS) significantly, improves the accuracy of the proposed system up to 99.52%. The highest recognition accuracy (99.52%) is achieved with only 56 fuzzy rules.

REFERENCES

[1] J. pradeep, E. Strinivasan, and S. Himavathi, Neural network based handwritten character recognition system with feature extraction. International conference on computer, communication and electrical technology- ICCCET 2011, 18th & 19th Mar. 2011.
[2] W. Wu and Y. Bao, Online handwritten magnolia words recognition based on multiple classifiers, 2009.
[3] C. L. Liu, K. Nakashima, H. Sako, and H. Fujisawa, Handwritten digit recognition: investigation of normalization and feature extraction techniques, Pattern Recognition, vol. 37, iss. 2, pp. 265-279, 2004.
[4] C. Y. Suen, C. Nadal, R. Legault, T. A. Mai, and L. Lam, Computer recognition of unconstrained handwritten numerals, Proc. IEEE, vol. 80, iss. 7, pp. 1162-1180, 1992.
[5] R. M. Brown, T. H. Fay, and C. L. Walker, Handprinted Symbol Recognition System, Pattern Recognition, vol. 21, iss. 2, pp. 91-118, 1988.
[6] L. Stringa, A New Set of Constraint-Free Character Recognition Grammars, IEEE Transactions on PAMI, vol. 12, iss.12, pp. 1210-1217, 1990.
[7] C. Y. Suen, C. Nadal, R. Legault, T. A. Mai, and L. Lam, Computer Recognition of Unconstrained Handwritten Numerals, Proceedings of the IEEE, vol. 80, iss. 70, pp. 1162-1180, Jul. 1992.
[8] B. T. Mittchell and A. M. Gillies, A Model-Based Computer Vision System for Recognizing Handwritten ZIP Codes, Machine Vision and Applications, vol. 21, iss.4, pp.231-243, 1989.
[9] L. Cun, L. Bottou, Y. Bengio, and P. Haffner, Gradient-Based Learning Applied to Document Recognition, Proceedings of the IEEE, vol. 86, iss. 11, pp. 2278-2324, Nov. 1998.
[10] Y. Tao, R. C. M. Lam, and Y. Y. Tang, Feature Extraction Using Wavelet and Fractal, Pattern Recognition Letters vol. 22, iss. 1, pp. 271-287, 2001.
[11] M. Shi, Y. Fujisawa, T. Wakbayashi, and F. Kimura, Handwritten Numeral Recognition Using Gradient and Curvature of Gray Scale Image, Patter Recognition, vol. 35, iss. 10, pp. 2051-2059, 2002.
[12] L. N. Teow and K. F. Loe, Robust Vision-Based Feature and Classification Schemes for Off-Line Handwritten Digit Recognition, Pattern Recognition, vol. 35, iss. 1, pp. 2355-2364, 2002.
[13] D. Decoste and B. Scholkopf, Training Invariant Support Vector Machines, Machine Learning, vol. 46, iss. 1-3, pp. 160-190, 2002.
[14] C. L. Liu, K. Nakashima, H. Sako, and H. Fujisawa, Handwritten Digit Recognition Using State-of-the-art Techniques, Proceedings of the 8th International Workshop on Frontiers in Handwritten Recognition, Ontario, Canada, pp. 320-325, Aug. 2002.
[15] E. Kussul and T. Baidyk, Improved Method of Handwritten Digit Recognition Tested on MNIST Database, Image and Vision Computing, vol. 22, iss. 12, pp. 971-981, 2004.
[16] F. Lauer, G. Bloch, and C. Y. Suen, Increasing the Recognition Rate of Handwritten Digit Classifiers, Technical Report, CENPARMI, Concordia University, 2005.
[17] E. Kussul, T. Baidyk, and D. C. Wunsch II, Image Recognition Systems with Permutative Coding, Proceedings of International Joint Conference on Neural Networks, Montreal, Canada, pp. 1788-1793, Aug. 2005.
[18] J. X. Dong, Speed and Accuracy: Large-Scale Machine Learning Algorithms and Their Applications, Doctoral thesis, Computer Science Department, Concordia University, Montreal, Oct. 2003.
[19] J. X. Dong, A. Krzyzak, and C. Y. Suen, Fast SVM Training Algorithm with Decomposition on Very Large Datasets, IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 27, iss. 4, pp. 603-618, Apr. 2005.
[20] C. L. Liu, K. Nakashima, H. Sako, and H. Fujisawa, Handwritten Digit Recognition: Benchmarking of State-of-the-Art Techniques, Pattern Recognition, vol. 36, iss.10, pp. 2271-2285, 2003.
[21] E. Avci, D. Hanbay, and A. Varol, An expert Discrete Wavelet Adaptive Network Based Fuzzy Inference System for digital modulation recognition. Expert Systems with Applications, vol.33, pp. 582-589, 2007.
[22] M. Hosoz, H. M. Ertunc, and H. Bulgurcu, An adaptive neuro-fuzzy inference system model for predicting the performance of a refrigeration system with a cooling tower. Expert Systems with Applications, vol. 38, pp. 14148-14155, 2011.
[23] A. Keles, A. Keles, and U. Yavuz, Expert system based on neuro-fuzzy rules for diagnosis breast cancer. Expert Systems with Applications, vol.38, pp. 5719-5726, 2011.

[24] D. T. Pham, A. Ghanbarzadeh, E. Koc, S. Otri, S. Rahim, and M. Zaidi, The bees algorithm – a novel tool for complex optimisation problems, in: Intelligent Production Machines and Systems, pp. 454-459, 2006.
[25] A. Ebrahimzadeh, J. Addeh, and V. Ranaee, Recognition of control chart patterns using an intelligent technique, Applied Soft Computing, vol. 13, pp. 2970-2980, 2013.
[26] P. Zhang, T. D. Bui, and C. Y. Suen, Nonlinear feature dimensionality reduction for handwritten numeral verification, Pattern Anal. vol. 7, iss.3, pp. 296-307, 2004.
[27] N. G. Kingsbury, Image processing with complex wavelets, Philos. Trans. R. Soc. London, Ser. A, pp. 2543-2560, 1999.
[28] P. Zhang, T. D. Bui, and C. Y. Suen, Extraction of hybrid complex wavelet features for the verification of handwritten numerals, in: Proceedings of the 9th International Workshop on Frontiers of Handwriting Recognition (IWFHR'9), Tokyo, Japan, pp. 347-352, 2004.
[29] W. K. Pratt, Digital Image Processing, Wiley, New York, 2001.
[30] P. Zhang and L. H. Chen, A novel feature extraction method and hybrid tree classification for handwritten numeral recognition, Pattern Recognition Lett. vol. 23, isss.1, pp. 45-56, 2002.
[31] M. Sugeno and G. T. Kang, "Structure identification of fuzzy model," Fuzzy Sets and System, vol. 28, pp. 15-33, 1988.
[32] T. Takagi and M. Sugeno, "Fuzzy identification of systems and its applications to modeling and control," IEEE Systems, Man, and Cybernetics Society, vol. 15, pp. 116-132, 1985.
[33] T. Takagi and M. Sugeno, "Derivation of fuzzy control rules from human operator's control actions," Proceedings of IFAC Symposium on Fuzzy Information, Knowledge Representation and Decision Analysis, pp. 55-60, 1985.
[34] E. H. Mamdani and S. Assilian, "An experiment in linguistic synthesis with a fuzzy logic controller," International Journal of Man-Machine Studu, vol. 7, pp. 1-13, 1975.
[35] J. S. Jang, "ANFIS: Adaptive-network-based fuzzy inference systems," IEEE Transactions on Systems, Man, and Cybernetics, vol. 23, pp. 665-685, 1993.
[36] S. Haykin, "Neural Networks-A Comprehensive Foundation," New Delhi, India: Prentice-Hall of India Pvt. Ltd, pp. 236-340, 2003.
[37] J. M. Zurada, "Introduction to Artificial Neural Systems," West Publishing Company. 1992.
[38] M. T. Hagan, H. B. Demuth, and M. H. Beale, "Neural Network Design," Boston, MA: PWS Publishing, 1996.
[39] S. Chiu, "Fuzzy model identification based on cluster estimation," Journal of Intelligent & Fuzzy Systems, vol. 2, pp. 267-278, 1994.
[40] S. Chiu, "Selecting input variables for fuzzy models," Journal of Intelligent & Fuzzy Systems, vol. 4, pp. 243-256, 1996.
[41] M. Buragohain and C. Mahanta, "A novel approach for ANFIS modelling based on full factorial design," Applied Soft Computing, vol. 8, pp. 609-625, 2008.
[42] S. L. Salzberg. On comparing classifiers: pitfalls to avoid and a recommended approach. Data mining and knowledge discovery, vol. 1, pp. 317-328, 1997.
[43] T. Takagi and M. Sugeno, Derivation of fuzzy control rules from human operator's control actions, in: Proc. IFAC Symp. Fuzzy Inform., Knowledge Representation and Decision Analysis, pp. 55-60, Jul. 1985.
[44] E. Atashpaz-Gargari and C. Lucas. Imperialist competitive algorithm: an algorithm for optimization inspired by imperialistic competition. In: Proceedings of the IEEE Congress on Evolutionary Computation, Singapore, pp. 4661-4667, 2007.
[45] J. Kennedy and R. Eberhart, Particle swarm optimization, in: Proceedings of IEEE International Conference on Neural Networks, vol. 4, pp. 1942-1948, 1995.
[46] D. F. Specht, Probabilistic neural networks, Neural Networks, pp. 109-118, 1990.
[47] T. Poggio and F. Girosi, Networks for approximation and learning, Proceedings of the IEEE, vol.78, pp. 1481-1497, 1990.
[48] S. Haykin, Neural Networks: A Comprehensive Foundation, MacMillan, New York, 1999.
[49] M. Riedmiller and H. Braun, A direct adaptive method for faster back propagation learning: the RPROP algorithm, in: Proceedings of the IEEE Int. Conf. On Neural Networks, San Francisco, CA, Mar. 28, 1993.