Available online at www.sciencedirect.com
ScienceDirect
Neurocomputing 70 (2007) 896–903
NEUROCOMPUTING
www.elsevier.com/locate/neucom

Shape recognition based on neural networks trained by differential evolution algorithm

Ji-Xiang Du[a,b,c,*], De-Shuang Huang[a], Xiao-Feng Wang[a,b], Xiao Gu[a]

[a]Intelligent Computing Lab, Hefei Institute of Intelligent Machines, Chinese Academy of Sciences, Hefei, Anhui 230031, P.R. China
[b]Department of Automation, University of Science and Technology of China, P.R. China
[c]Department of Computer Science and Technology, Huaqiao University, P.R. China

Available online 18 October 2006

Abstract

In this paper a new method for recognition of 2D occluded shapes based on neural networks using generalized differential evolution training algorithm is proposed. Firstly, a generalization strategy of differential evolution algorithm is introduced. And this global optimization algorithm is applied to train the multilayer perceptron neural networks. The proposed algorithms are evaluated through a plant species identification task involving 25 plant species. For this practical problem, a multiscale Fourier descriptors (MFDs) method is applied to the plant images to extract shape features. Finally, the experimental results show that our proposed GDE training method is feasible and efficient for large-scale shape recognition problem. Moreover, the experimental results illustrated that the GDE training algorithm combined with gradient-based training algorithms will achieve better convergence performance.

Keywords: Shape recognition; Generalization strategy; Differential evolution algorithm; Multiscale Fourier descriptors; Leaf image database

1. Introduction

The shape feature is one of the most important features for characterizing an object, which is commonly used in object recognition, matching and registration. In addition, the shape recognition is also an important part of machine intelligence that is useful for both decision-making and data processing. More importantly, the recognition-based on shape feature is also a central problem in those fields such as pattern recognition, image technology and computer vision, etc., which have received considerable attention recent years. Face recognition, image preprocessing, computer vision, fingerprint identification, handwriting analysis, and medical diagnosis, etc., are some of the common application areas of shape recognition. In particular, shape recognition has mutual effects with other research areas such as signal processing, neural networks, optimization theory, structural modeling and formal languages, etc. For shape recognition, there have been a wide range of methods proposed [1,2]: structural methods organizing local features into graphs, trees, or strings; Fuzzy methods; Statistical methods; Transform methods, such as Fourier transform [3] or Hough transforms; Neural networks methods [4,5], and so on. But most of approaches are confined to specific image types and require that all shapes must be preprocessed before recognition. However, an ever-increasing amount of image data in many application domains has generated additional requirements for real-time management and retrieval of images. Therefore, the emphasis on image recognition is not only on the accuracy, but also on the efficiency.

On the other hand, the neural network is widely used in pattern recognition kingdom as an effective classifier. Since the development of the back-propagation method, many algorithms have been proposed and used to train neural networks, such as modified back-propagation [6], back-propagation using the conjugate-gradient approach [7], scaled conjugate-gradient [8], the Levenberg–Marquadt algorithm [9]. The simulated annealing (SA) method and genetic algorithm (GA) method also have been proposed

*Corresponding author. Intelligent Computing Lab, Hefei Institute of Intelligent Machines, Chinese Academy of Sciences, Hefei, Anhui 230031, China.
E-mail addresses: du_jx@iim.ac.cn (J.-X. Du), dshuang@iim.ac.cn (D.-S. Huang), xfwang@iim.ac.cn (X.-F. Wang), xgu@iim.ac.cn (X. Gu).

0925-2312/$ - see front matter © 2006 Elsevier B.V. All rights reserved.
doi:10.1016/j.neucom.2006.10.026

for network with non-differentiable transfer functions where the gradient information is not available [10–12]. Many of the existing training algorithms are suitable to small or middle scale networks structures and have a rather fast convergence speed. For the small-scale problem or small network structure, the gradient information usually is available and the training methods based on gradient information are rather fast and can convergence to global minima by repeated training and using randomly initialized weight values. However, for some large-scale real world problems, such as the above-mentioned shape recognition problem, many of them have worse performances than small or middle scale problems on convergence accuracy and speed. There are seldom suitable and reasonable network training algorithms when the neural network structures or the number of network parameters grow rapidly. For such large-scale neural networks structure, many of the training methods need an unacceptable computation cost in time and space. And the local minima problem also must be considered. The global optimization algorithms, such as GA and SA, may be useful to avoid such a local minima problem. In fact, there is no single training algorithm can have the best performance compared with all other methods on all problem domains. One feasible solution method is that use the global optimization algorithms combined with gradient information methods to train the neural network to achieve acceptable solution.

For the global optimization methods, GAs have been studied and found to be promising stochastic optimization methods. A survey and overview of GAs in evolving neural networks can be found in [13,14]. Differential Evolution (DE) is one of the recent population-based global optimization techniques [15,16]. Some works have applied DE to train neural networks [11,12], in which the experimental data are small-scale problems. In this paper, a new generalization strategy of DE is applied to train feed-forward multi-layer perceptron neural networks (MLPNN) and compared with different type of training algorithms. Furthermore, this paper focuses on using the novel neural network-based method to perform shape recognition task through multiscale Fourier descriptors (MFDs) of shapes.

This paper is organized as follows: In Section 2, a generalization strategy of DE algorithm is described and discussed. In Section 3, a novel training method for neural network based on generalized DE algorithm is presented. In Section 4, the MFDs feature extraction method for shapes is presented. The experimental results are reported in Section 5, and Section 6 concludes the whole paper and gives related conclusions.

2. Generalization strategy of DE algorithm

DE is one of the recent population-based global optimization techniques [15,16], which is a heuristic method for minimizing nonlinear and non-differentiable continuous space functions. The DE scheme entirely corresponds to a typical GA. But the principle difference consists in the mutation operation. In GA mutation is caused by small alterations of genes, whereas in DE Mutation is provided by combinations of individuals. The core of this operation is the formation of a difference vector, which makes mutate an individual.

As all GAs, DE deals with a population of solutions. The population p^g of a generation g has NP vectors, i.e., individuals X^g. Each individual represents a potential optimal solution:

$$P^g = \{X_i^g\}, \quad i = 1, 2, \ldots, NP.$$

In turn, the individual X^g contains D variables (chromosomes):

$$X_i^g = x_{i,j}^g, \quad j = 1, 2, \ldots, D.$$

For each generation, the individuals are updated by means of a reproduction scheme. Thereto for each individual X_i^g, a set of other individuals is randomly extracted. To produce a new one, the operations of differentiation and recombination are applied to this set.

A set of individuals is firstly and randomly extracted for differentiation operation. Then a difference vector δ and a base vector β are designed based on these extracted individuals. Thus, the result of differentiation operation, i.e., trial individual, is

$$\omega = \beta + CF\,\delta, \tag{1}$$

where $CF > 0$ is the constant of differentiation.

Next recombination operation represents a typical case of exchange of chromosomes. The trial individual ω inherits chromosomes with some probability. Thus,

$$\omega_j = \begin{cases} \omega_j & \text{if} \quad \text{rand}() CR, \\ x_{i,j}^g & \text{otherwise}, \end{cases} \tag{2}$$

where $j = 1, 2, \ldots, D$ and $CR \in [0, 1]$ is the constant of recombination.

In general, CF and CR affect the convergence speed and robustness of the search process. Their optimal values depend both on objective function characteristics and the population size NP, and thus, the selection of optimal parameter values is an application dependent task.

Selection is used to choose the best:

$$X_i^{g+1} = \begin{cases} \omega & \text{if} \quad f(\omega) < f(X_i^g), \\ X_i^g & \text{otherwise}. \end{cases} \tag{3}$$

Differentiation operation plays a quite important role in the reproduction cycle. Geometrically, it consists in two simultaneous operations: the first one is the choice of a differentiation's direction and the second one is the step length. The principle of differentiation operation is based on a random extraction of individuals from the population. Possible directions entirely depend on the disposition of extracted individuals. Also, their disposition influences the step length. Furthermore, by increasing the number of extracted individuals, the diversity of possible directions and the variety of step lengths will be augmented. Thereby

the exploration of the search space will be intensified. But on the other hand, the probability to find the best combination of extracted individuals goes considerably down. Based on such an analysis, a generalization strategy of DE (GDE) is proposed as followed [17].

Let randomly extracted individuals X_i be divided into two classes C^+ and C^- with n^+ and n^- elements so that for each element from the class C^+ its cost function value would be less than the cost function value of any element from class C^-:

$$(\forall X_i \in C^+) \wedge (\forall X_j \in C^-) : f(X_i) \leqslant f(X_j)$$
$$i = 1, 2, \ldots, n^+, \quad j = 1, 2, \ldots, n^-. \tag{4}$$

Then find the maximal and minimal elements, $X_{\pm}^{\mathrm{Max}}$, $X_{\pm}^{\mathrm{Min}}$ of each of the classes. So we can calculate the shift inside of the classes.

$$V_{\mathrm{S}} = 0.5\,\lambda(X_{+}^{\mathrm{Min}} - X_{+}^{\mathrm{Max}} + X_{-}^{\mathrm{Min}} - X_{-}^{\mathrm{Max}}). \tag{5}$$

Hence, taking into account the information about the best individual, the differentiation formula (1) is modified as:

$$\omega = V_{\mathrm{b}} + CF \cdot (V_{C+} - V_{C-} + V_{\mathrm{S}}), \tag{6}$$

where V_{C+} and V_{C-} are barycenters of the C^+ and C^- classes accordingly, V_{b} is the best individual.

3. GDE training algorithm

From the above, it is shown that DE is a heuristic method for minimizing nonlinear and non-differentiable continuous space functions, so it can be applied to global searches within the weight space of a typical neural network.

The most popular neural network model is the so-called MLPNN. Training an MLPNN to recognize objectors is typically realized by adopting an error correction strategy that adjusts the network weights through minimization of learning error:

$$E = E(Y_0, Y), \tag{7}$$

where, Y is the real output vector of a MLPNN, Y_0 is the target output vector. And Y is a function of synaptic weights W and input values X:

$$Y = f(W, X). \tag{8}$$

For MLPNN, both the input vector x and the target output vector Y_0 are known, and the synaptic weights in W are adapted to obtain appropriate functional mappings from the input x to the output Y_0. Generally, the adaptation can be carried out by minimizing the network error function E, i.e., network training procedure. The optimization goal is to minimize the objective function E by optimizing the values of the network weights: $W = (w_1, w_2, \ldots, w_D)$.

Similar to other evolutionary algorithms, DE maintains a population of constant size that consists of NP, real value vectors W_i^g, where $i (i = 1, 2, \ldots, NP)$ is the index to the population and $g (g = 1, 2, \ldots, g_{\max})$ is the generation to which the population belongs: $P^g = \{W_1^g, W_2^g, \ldots, W_{NP}^g\}$.

For each generation, a trial individual ω of the population is generated using the generalized strategy, which has been discussed in Section 2:

$$\omega = W_{\mathrm{b}} + CF(W_{C+} - W_{C-} + W_S), \tag{9}$$

$$\omega_j = w_{\mathrm{b},j}^g + CF(w_{C+,j}^g - w_{C-,j}^g + w_{S,j}^g) \tag{10}$$

$$\omega_j = \begin{cases} \omega_j & \text{if} \quad \mathrm{Rand}() < CR, \\ W_{i,j}^g & \text{otherwise.} \end{cases} \tag{11}$$

The population for the next generation is selected from the current population or the child population according to the following rule:

$$W_i^{g+1} = \begin{cases} \omega & \text{if } E(Y_0, f(X, \omega)) < E(Y_0, f(X, W_i^g)), \\ W_i^g & \text{otherwise.} \end{cases} \tag{12}$$

Thus, each individual of the population is compared to its counterpart in the current population; the vector with the lower objective function value wins a place in the next generation's population. As a result, all the individuals of the next generation are as good as or better than their counterparts in the current generation.

4. MFDs of shape

In this section, the shape feature extraction method is described. Let Γ a planar curve defined by

$$\Gamma = \{(x(u), y(u)) | u \in [0, 1]\}, \tag{13}$$

where u is the normalized arc length parameter. The curvature κ of a planar curve is defined as the derivative of the tangent angle with respect to the arc length s. Then the curvature function of can be expressed as follows:

$$\kappa(u) = \frac{\dot{x}(u)\ddot{y}(u) - \ddot{x}(u)\dot{y}(u)}{((\dot{x}(u))^2 + (\dot{y}(u))^2)^{3/2}}, \tag{14}$$

where

$$\dot{x}(u) = \frac{\mathrm{d}x}{\mathrm{d}u}, \ddot{x}(u) = \frac{\mathrm{d}^2x}{\mathrm{d}u^2},$$
$$\dot{y}(u) = \frac{\mathrm{d}y}{\mathrm{d}u}, \ddot{y}(u) = \frac{\mathrm{d}^2y}{\mathrm{d}u^2}. \tag{15}$$

An evolved version of the curve is defined by

$$\Gamma_\sigma = \{(X(u, \sigma), Y(u, \sigma)) | u \in [0, 1]\}, \tag{16}$$

where

$$X(u, \sigma) = x(u) * g(u, \sigma),$$
$$Y(u, \sigma) = y(u) * g(u, \sigma), \tag{17}$$

$$g(u, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(\frac{-u^2}{2\sigma^2}\right), \tag{18}$$

$g(u, \sigma)$ denotes a one dimensional Gaussian kernel.

Functions $X(u, \sigma)$ and $Y(u, \sigma)$ are, respectively, given explicitly by

$$X(u,\sigma) = \int_{-\infty}^{\infty} x(v)\frac{1}{\sigma\sqrt{2\pi}}\exp\left(\frac{-(u-v)^2}{2\sigma^2}\right)\mathrm{d}v,$$
$$Y(u,\sigma) = \int_{-\infty}^{\infty} y(v)\frac{1}{\sigma\sqrt{2\pi}}\exp\left(\frac{-(u-v)^2}{2\sigma^2}\right)\mathrm{d}v. \quad (19)$$

The curvature of Γ_σ can be computed as follows:

$$\kappa(u,\sigma) = \frac{X_u(u,\sigma)Y_{uu}(u,\sigma) - Y_u(u,\sigma)X_{uu}(u,\sigma)}{(X_u(u,\sigma)^2 + Y_u(u,\sigma)^2)^{3/2}}, \quad (20)$$

where

$$X_u(u,\sigma) = x(u)*\dot{g}(u,\sigma),$$
$$X_{uu}(u,\sigma) = x(u)*\ddot{g}(u,\sigma),$$
$$Y_u(u,\sigma) = y(u)*\dot{g}(u,\sigma),$$
$$Y_{uu}(u,\sigma) = y(u)*\ddot{g}(u,\sigma), \quad (21)$$

where $*$ means the convolution operation

$$\dot{g}(u,\sigma) = \frac{\partial}{\partial u}g(u,\sigma),$$
$$\ddot{g}(u,\sigma) = \frac{\partial^2}{\partial u^2}g(u,\sigma). \quad (22)$$

At each scale σ, a curvature curve of shape can be obtained. By this mean, a planar shape can be regarded as a series of one-dimensional local feature curves.

It is well known that Fourier transform is a right candidate to describe smooth change and sudden changes in its frequency domain. Based on our practical issue, we propose a modified way to apply Fourier transform to the shape analysis. The modification is as follows:

$$d(u,\sigma) = \log(\kappa(u,\sigma)). \quad (23)$$

$d(u, \sigma)$ is the logarithm transform of curvature curve. The Fourier Descriptors (FD) can be obtained by the Fourier transformation of $d(u, \sigma)$ [3] and be called MFDs. The MFDs are not sensitive to scaling, rotation and translation. So the MFDs should be much more suitable to our shape analysis. And an example is shown in Fig. 1.

5. Experimental results

In our work, a leaf image database is used in the following experiment, which was collected and built by ourselves in our lab. This database includes 25 species of different plants. Each species includes at least 40 leaves images, 20 of which are used as training samples. There are totally 2000 images with the database. A subset of the images (16 images from 4 different plant species) is shown in Fig. 2.

For each test image, 5 different scales were selected $\sigma = 20, 40, 60, 80, 100$. And for each scale, the lower order 10 coefficient (excluding the 0th coefficient) of the Fourier transform are used. So there are 50 features for each test image, which is also the input vector of MLPNN. The number of training samples is 500, and the rest samples are test data. The algorithm is programming with Matlab 6.5, and run on Pentium 4 with the clock of 2.6 GHz and the RAM of 256 M under Microsoft Windows XP environment.

Firstly, the efficiency of GDE training algorithm for MLPNN is tested. In this experiment, we set $CF = 0.9, CR = 0.5$. And four commonly used training algorithms (gdx, scg, lm and bfg) are applied to the training of the networks with a different number n of hidden layer units:

(1) *gdx*: Gradient descent with momentum and adaptive learning rate backpropagation;
(2) *scg*: Scaled conjugate gradient backpropagation;
(3) *lm*: Levenberg–Marquardt backpropagation;
(4) *bfg*: BFGS quasi-Newton backpropagation.

In this experiment, the mean CPU time of one iteration is calculated for the different training algorithms. As shown in Fig. 3, the lm and bfg training algorithms converge fast for a small number of hidden units but gave a worse performance for a large one. The gdx and scg training algorithms have a better performance and converge fast for both a small and a large number of hidden units. And the performance of GDE training algorithm with a small population size NP is as same as the gdx and scg training algorithms. However, with a

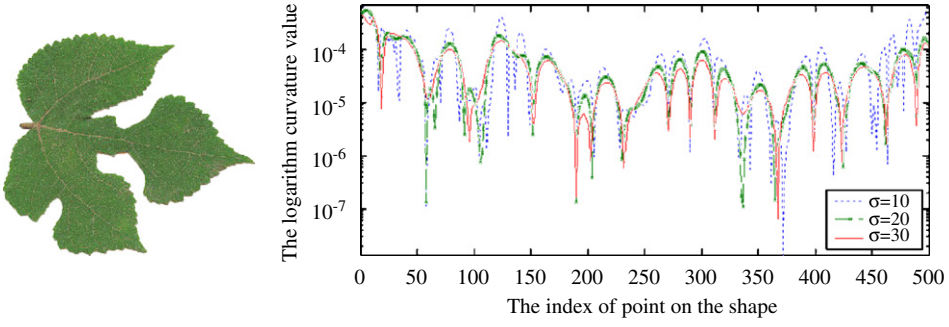

Fig. 1. A shape and its multiscale curvature curves.

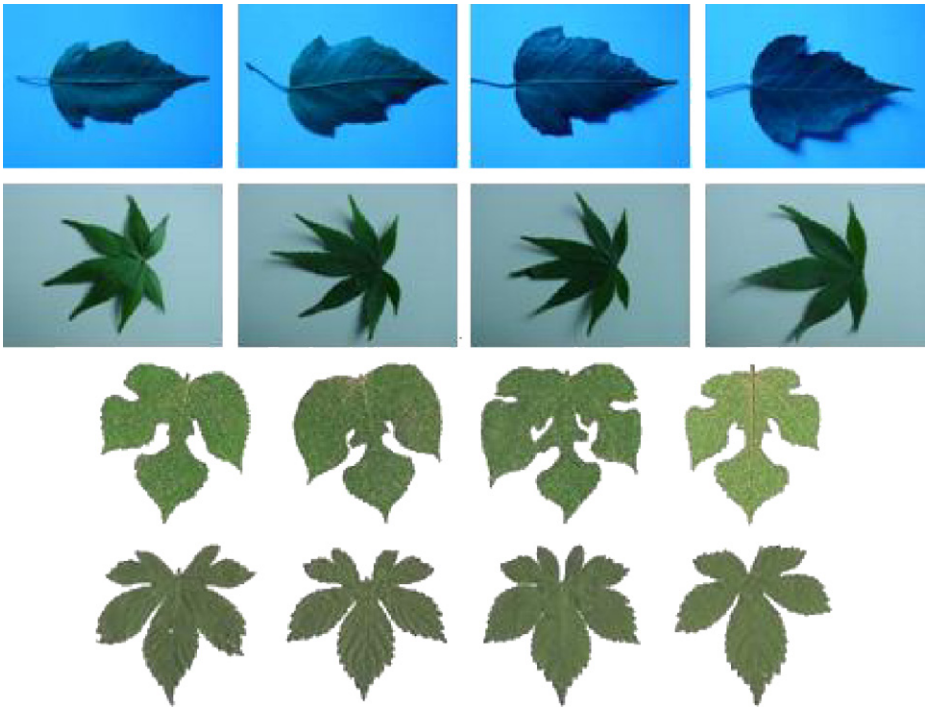

Fig. 2. A subset of the leaf images.

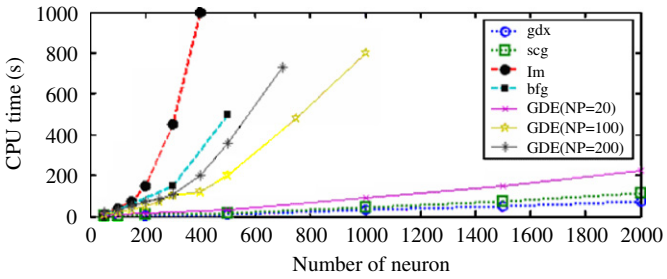

Fig. 3. Mean CPU time of one iteration for training algorithms.

rather big *NP*, the computational time is unacceptable. Therefore, it is will be suitable for training large scale MLPNN for GDE algorithm with an appropriate population size.

To test the robustness of the training algorithms, two different initializations methods for training algorithms are used to initialize the weight values. One is the random initialization method and the other is the commonly used Nguyen–Widrow weight initialization method. The number of hidden layer units is set to 30; the max number of iteration is 1500. For each training method, the experiment is repeated 10 times, and the mean correct recognition rate with different initializations is shown in Table 1. From Table 1, it can be seen that the training methods based on gradient is sensitive to initialization of the network weights, they will have a good results if the proper initialization is used. The GDE training method is a robust method for it is not sensitive to initialization of the network weights and find the most optimum at most case though it is a stochastic search optimization method.

Furthermore, we also compared the performances of different neural networks training algorithms. For all the tested algorithms, the experimental conditions are same. In this experiment, the network weights were initialized by the Nguyen–Widrow method to provide good initial weights. The performance was estimated by measuring the mean-squared error at a restricted training iteration number. The number of hidden layer units is set to 30; the max number of iteration is set to 3000.Three global optimization algorithms, GA, SA and the original strategy of DE are selected. And the resilient back-propagation (r.p.) training method [18] is also used. For the GDE, we also trained the networks combined with the scg, gdx and r.p. methods. For each training method, the experiment is repeated 10 times, and the results were shown in Table 1 and Figs. 4–7. From all the experimental results, we can see that, although the GDE training method is not the best one among all the single training algorithms, it is a better alternative and can be comparable to that of the gdx and scg methods. For the GDE method, the bigger size of population may converge to a local minimum which is more near to the global minimum, however, which will lead much more computational cost. If combined with gradient information, the networks can achieve the best convergence performances. Therefore, if the gradient information is available, using both the GDE optimization and gradient information could be the best training method. Moreover, if the error surface is very rough, the gradient information frequently leads to local optimums and is difficult to be utilized, the

Table 1
The mean correct recognition rate (%) with different initialization methods

Training methods	Initialization methods	
	Random	Nguyen–Widrow
gdx	86.5	87.3
scg	86.7	87.6
lm	86.1	87.1
bfg	86.7	87.2
GDE(NP = 20)	87.1	87.3
GDE(NP = 30)	87.1	87.3
GDE(NP = 50)	87.3	87.8

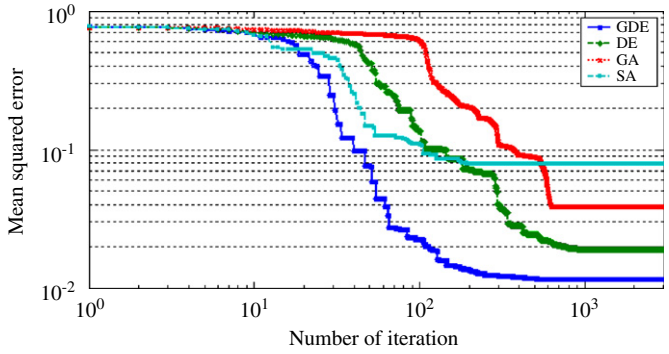

Fig. 4. The performance comparison for different training algorithms based on global optimization algorithm.

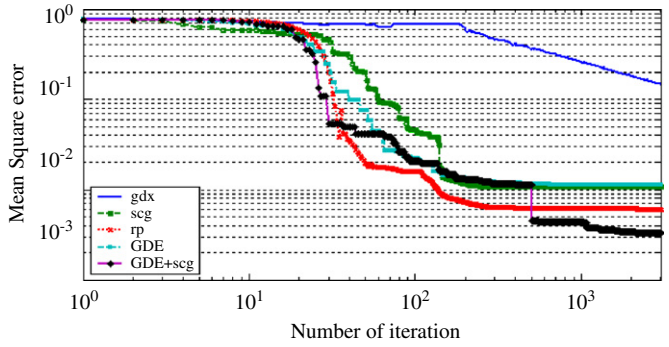

Fig. 5. The performance comparison for different training algorithms.

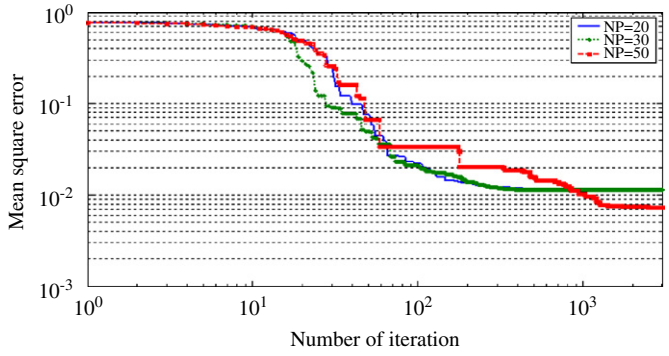

Fig. 6. The performance comparison for the GDE training algorithm with different population size.

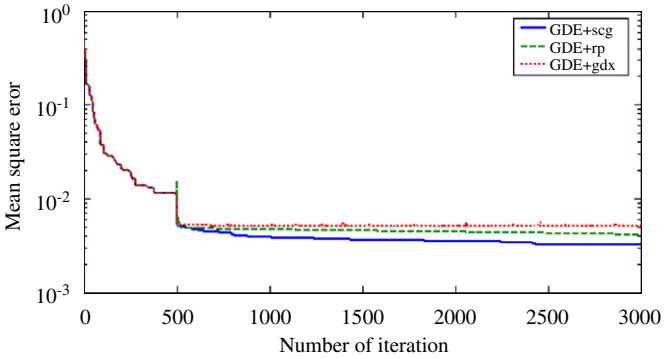

Fig. 7. Performance comparison for the GDE training algorithm combined with gradient information.

global optimization methods, such as DE and GDE, may be the only practical methods (Table 2).

Finally, in order to demonstrate the efficiency of MFD features for shape recognition, three different commonly used feature extraction methods are used as the competitors: the FD [3], Hu invariant moments (HM) and contour moments (CM) [19]. All the MLPNNs are trained by the GDE algorithm combined with the gradient-based scg method. As shown in Table 3, our proposed MFD method can achieve the best recognition result.

6. Conclusions

In this paper, a generalized differential evolution (GDE) training algorithm was used to train multilayer perceptron neural networks and the trained networks are used to shape recognition. Furthermore, a multiscale Fourier descriptors (MFDs) feature extraction method for shape recognition is proposed. For using the multiscale local features of shapes, it is much more robust and effective than other method using global features. The superiority of GDE training method to network training and MFD method for shape recognition has been demonstrated by experiments. The main advantages of GDE training algorithm are that there are no major restrictions on the error function; it can be expected to converge to a global minimum if there is enough time; and there are only three parameters needed to be modified. With a reasonable population size, it is suitable for training large-scale MLPNN and it is more robust than other methods based on gradient. It is a good alternative to train networks compared other methods. If the gradient information is available, it had better be used combining with GDE method and can achieve the best convergence performances. In the future work, the selection of optimal parameter values (CF, CR) for GDE training algorithm should be studied and the global features of shapes should be combined with our MDF local features.

Table 2
The performance comparison results for different training methods

Train methods	Mean square error (MSE)			
	Mean	Max	Min	Variance
gdx	0.14537	0.15131	0.14023	0.0112
scg	0.01065	0.00902	0.01687	0.0098
Rp	0.00707	0.10189	0.00571	0.0137
GA	0.03840	0.04210	0.03523	0.0568
SA	0.07899	0.09103	0.05992	0.0683
DE(NP = 20)	0.01879	0.03061	0.01487	0.0413
GDE(NP = 20)	0.01143	0.01518	0.00980	0.0529
GDE(NP = 30)	0.01133	0.01446	0.00884	0.0610
GDE(NP = 50)	0.00733	0.01003	0.00521	0.0313
GDE(NP = 20) + scg	0.003189	0.006341	0.001798	0.0071
GDE(NP = 20) + rp	0.004095	0.007432	0.002887	0.0086
GDE(NP = 20) + gdx	0.005179	0.010401	0.003369	0.0081

Table 3
The performance comparison results for different feature extraction methods

Methods	Recognition rate (%)
MFD	88.2
FD	86.3
HM	84.1
CM	76.4

Acknowledgement

This work was supported by the National Science Foundation of China (No. 60472111 and 60405002), the Postdoctoral Science Foundation of China (No. 20060390180) and the Scientific Research Foundation of Huaqiao University (No. 06BS217).

References

[1] S. Loncaric, A survey of shape analysis techniques, Pattern Recogn. 31 (8) (1998) 983–1001.
[2] D. Zhang, G. Lu, Review of shape representation and description techniques, Pattern Recogn. 37 (1) (2004) 1–19.
[3] Y. Rui, A.C. She, T.S. Huang, Modified Fourier descriptors for shape representation-a practical approach, In: First International Workshop on Image Databases and Multi Media Search, Amsterdam, The Netherlands, August 1996.
[4] G. Papadourakis, G. Bebis, S. Orphanoudakis, Curvature scale space driven object recognition with an indexing scheme based on artificial neural networks, Pattern Recogn. 32 (7) (1999) 1175–1201.
[5] D.S. Huang, Systematic Theory of Neural Networks for Pattern Recognition, Publishing House of Electronic Industry of China, Beijing, 1996.
[6] L. Min, W. Shi-Xi, L. Yong-Hong, Fast learning algorithms for multi-layered feedforward neural network, In: Proceedings of the IEEE National Conference on Aerospace and Electonics, NAECON, vol. 2, 1994, pp. 787–790.
[7] C. Charalambous, Conjugate gradient algorithm for efficient training of artificial neural networks, IEE G (Circuits, Dev. Syst.) 139 (3) (1992) 301–310.
[8] S. Doyle, D. Corcoran, J. Connell, Automated mirror design using an evolution strategy, Opt. Eng. 38 (2) (1999) 323–333.
[9] M.T. Hagan, M.B. Menhaj, Training feedforward networks with the Marquadt algorithm, IEEE Trans. Neural Networks 5 (6) (1994).
[10] S.P. Day, D.S. Camporese, A stochastic training technique for feed-forward neural networks, Int. Joint Conf. Neural Networks 3 (1990) 607–612.
[11] I. Zelinka, J. Lampinen, An evolutionary learning algorithms for neural networks, Fifth International Conference on Soft Computing, MENDEL, 1999, pp. 410–414.
[12] J. Ilonen, J.K. Kamarainen, J. Lampinen, Differential evolution training algorithm for feed forward neural networks, Neural Proc. Lett. 17 (2003) 93–105.
[13] X. Yao, Evolving artificial neural networks, Proc. IEEE 87 (9) (1999) 1423–1447.
[14] G.P. Schmitz, C. Aldrich, Combinatorial evolution of regression nodes in feed forward neural networks, Neural Networks 12 (1) (1999) 175–189.
[15] R. Storn, K. Price, Differential Evolution—A Simple and Efficient Adaptive Scheme for Global Optimization Over Continuous Spaces, Technical Report TR-95-012, International Computer Science Institute, Berkeley, CA, USA, 1995.
[16] K. Price, New Ideas in Optimization, Part 2: Differential Evolution, McGraw-Hill, London, UK, 2003.
[17] F. Vitaliy, J. Stefan, Generalization of the strategies in differential evolution, in: Proceedings of the 18th International Parallel and Distributed Processing Symposium (IPDPS'04), 2004.
[18] M. Riedmiller, H. Braun, A direct adaptive method for faster backpropagation learning: the RPROP algorithm, in: Proceedings of the IEEE International Conference on Neural Networks, San Francisco, 1993.
[19] L. Gurta, M.D. Srinath, Contour sequence moments for the classification of closed planar shapes, Pattern Recogn. 20 (3) (1987) 267–271.

Ji-Xiang Du took courses as B.Sc. degree candidate in Vehicle Engineering, Hefei University of Technology, from September 1995 to July 1999, and obtained B.Sc. degree in July 1999. From September 1999 to July 2002, took courses as M.Sc. degree candidate in Vehicle Engineering, Hefei University of Technology, and obtained M.Sc. degree in July 2002. From February 2003 on, in pursuit for Ph.D. degree in Pattern Recognition and Intelligent System in University

of Science and Technology of China (USTC), Hefei, China, and in December 2005, he received Ph.D. degree. Now, he is engaged in the postdoctoral study in Department of Automation of USTC.

De-Shuang Huang (SM'98) is a Professor and Ph.D. Advisor in the University of Science and Technology of China (USTC), Hefei, China, and Professor of Graduate School of the Institute of Intelligent Machines, Chinese Academy of Sciences (CAS). From September 2000 to March 2001, he worked as Research Associate in Hong Kong Polytechnic University. From April 2002 to June 2003, he worked as Research Fellow in City University of Hong Kong. From October to December 2003, he worked as Research Fellow in Hong Kong Polytechnic University. From July to December 2004, he worked as the University Fellow in Hong Kong Baptist University. Dr. Huang is a senior member of the IEEE.

Xiao-Feng Wang took courses as a bachelor's degree candidate in Computer Science Technology, Anhui University from September 1995 to July 1999, and obtained bachelor's degree in July 1999. From 1999 to 2002, he worked in Institute of Intelligent Machines, Chinese Academy of Sciences (CAS). From September 2002 to July 2005, took courses as a master's degree candidate in Pattern Recognition and Intelligent System, Institute of Intelligent Machines, and obtained master's degree in July 2005. Now, he is working in Department of Computer and Science Technology of Hefei University.

Xiao Gu obtained bachelor degree in Computer Science from Nanjing Normal University in July 2003. From September 2003 on, in pursuit for M.Sc. degree in Pattern Recognition and Intelligent System in Institute of Intelligent Machines, Chinese Academy of Sciences, Hefei. Her research interests: include Digital Image Processing, Software Exploitation, Neural Networks, etc.