Annealing Chaotic Pattern Search Learning Method for Multi-layer Neural Networks

Shangce Gao[†] , Hongwei Dai, Yunyi Zhu and Zheng Tang[††],

Faculty of Engineering, University of Toyama, Toyama-shi, Japan

Summary
As a novel optimization technique, chaos has gained much attention and some applications during the past decade. For a given energy or cost function, by following chaotic ergodic orbits, a chaotic dynamic system may eventually reach the global optimum or its good approximation with high probability. To enhance the performance of the pattern search method (PS), which is a derivative-free direct search algorithm, hybrid pattern search method is proposed by incorporating chaos. Furthermore, an annealing strategy is also utilized to eliminate the fluctuation of the chaos in the latter phrase of the process. We test this algorithm on several benchmark problems, such as exclusive-or (XOR) problem, parity problem and Arabic numerals recognition. Simulation results show that the systems can be trained efficiently by our method for all problems.

Key words:
multi-layer neural network, pattern search method, annealing, chaotic dynamic, direct search

1. Introduction

It is well known that the training of multi-layer feed-forward neural networks [1],[2] can be viewed as the optimization of a criterion function from a set of input-output pairs with respect to a set of parameters-the weights and thresholds [3]. In other words, it is a kind of multidimensional minimization of the error measure function. Minimizing the multi-layer neural network error measure function in realistic problems is a difficult task since many layers, the multitude of training patterns and the variety of categories cast a very complex landscape with wide plateaus and narrow valleys [4].

In Recent years several gradient descent based algorithms [2],[5],[6] and global optimization techniques (such as genetic algorithm [7] and simulated annealing [8],[9],[10],[11] have been present to train the multi-layer feed-forward neural networks. Although global optimization techniques provide an alternative method to problems that are different to solve with traditional optimization algorithms, they suffer from poor convergence properties and difficulties to reach high quality solutions when the structure of neural networks becomes complex and there are large training samples [12]. Furthermore, as a first-order gradient based non-linear optimization method, the back-propagation (BP) algorithm is most widely used and an effective algorithm for training multi-layer feed-forward neural networks [2]. It iteratively adjusts the network parameters (all weights and thresholds) to minimize the error measure function using a gradient descent technique. Most recently, to improve the performance of the original BP algorithm, researchers have concentrated on the following factors: selection of better energy function [13],[14] and selection of dynamic learning rate and momentum [15],[16],[17]. Moreover, some second-order gradient based methods such as the conjugate gradient algorithm [5] and the Levenberg-Marquardt based method [6] have also been proposed. Generally speaking, the methods that use derivatives are efficient. However some problems correspond to the error measure functions that by nature are non-differentiable or difficult to compute. Furthermore, they are usually different for hardware implementations for they need analog multipliers and other analog computations. Hence these methods will not work in hardware manner [18]. In addition, due to the highly nonlinear modeling power of such networks, the learned function may oscillate abruptly between carious training data. This is clearly undesirable for function approximation [19]. A metaheuristic [20], such as direct search [21] may provide a good solution to this problem. The authors have therefore proposed a learning method for multi-layer feed-forward neural networks using a pattern search method [22]. The pattern search based training algorithm, which does not require derivative information and indeed is one of the "derivative-free" direct search methods [23], can render the procedure efficient and robust and provided a very simple and effective means of searching the minima of objective function directly without any oscillation between training data.

However, due to their inherent local minimum problems, all these learning algorithms, either the BP based algorithms or the direct search based algorithms often converge to a local minimum solution that is far from the optimal solution. In this paper, we propose an annealing chaotic dynamic pattern search method (ACPS) for learning multi-layer feed-forward neural networks. The proposed algorithm maintains some trend of quick descent to local minimum, and at the same time has some chance

Manuscript received June 5, 2007
Manuscript revised June 20, 2007

of escaping from them by exploiting the landscape in an ergodic manner. We apply the algorithm to a diverse set of problems, including exclusive-or (XOR) problem, parity problems and Arabic numerals recognition. Simulation results show that our method can train multi-layer networks effectively and improve the pattern search method much.

2. Pattern Search Method for Multi-layer Feed-forward Neural Networks (MFNNs)

A multi-layer feed-forward neural network consists of three layers: an input layer, and output layer, and one or more hidden layers. Each layer is composed of a predefined number of neurons and each neuron has a threshold. It is usually assumed that each layer is fully connected with an adjacent layer without direct connections between layers that are not consecutive. Each connection has a weight. Fig.1 illustrates the structure of a typical three-layer feed-forward neural network.

The neurons in the input layer only act as buffers for distributing the input signals x_i $(i = 1,2,...,N)$ to neurons in the hidden layer. The input of each neuron in a layer (except input layer) is given by

$$net_{pj} = \sum_i w_{ji} o_{pi} + \theta_j \quad (1)$$

where net_{pj} is the net input to neuron j produced by the presentation of pattern p, w_{ji} is the weight from neuron i to neuron j, θ is a threshold of the neuron j and o_{pi} is the output value of neuron i for pattern p. The output of neuron j for pattern p is specified by

$$o_{pj} = f_j(net_{pj}) \quad (2)$$

where $f(x)$ is a semilinear activation function that is differentiable and nondecreasing and usually a sigmoidal or hyperbolic tangent function. Training a network can be performed by adjusting its weights and thresholds using a training algorithm. The training algorithms adopted in this paper optimize the weights and thresholds by attempting to minimize an over error measure E which is the total sum of squared differences between the desired and actual values of the output neurons for all patterns, namely:

$$E = \sum_p E_p(W, \Theta) \quad (3)$$

$$W = (w_{11}, w_{12}..., w_{ij},...) \quad (4)$$

$$\Theta = (\theta_1, \theta_2..., \theta_i,...) \quad (5)$$

where p indexes over all the patterns in the training set. E_p is defined by

$$E_p = \frac{1}{2}\sum_j (t_{pj} - o_{pj})^2 \quad (6)$$

where t_{pj} is the target value (desired output) of the j-th output neuron for pattern p, and o_{pj} is the actual output value of the j-th neuron produced by the presentation of input pattern p, and j indexes over the output neurons.

The pattern search method is a class of direct search methods for nonlinear optimization proposed by Hooke and Jeeves [21]. Since the introduction of the original pattern search method, it has remained popular with users due to its simplicity and the fact that it works well in practice on a variety of problems [24]. Recently, the analysis of convergence of the pattern search method has been present in [24],[25]. The basic idea of the pattern search method is that it first uses inexpensive surrogate objective and constraints to predict points that constitute improvements to the real problem and then make a local exploration near the current iteration whose properties enable the theory to guarantee convergence [26],[27],[28].

In a multi-layer feed-forward neural network, the variable space consists of the weights between all layers and thresholds of every neuron. So in order to minimize the sum of squared residuals, we consider adjusting a vector V whose elements include all the parameters in the variable space:

$$V = [w_{11}, w_{12},..., w_{ij},..., \theta_1, \theta_2,..., \theta_i,...]^T \quad (7)$$

Then the error function E in Eq.(3) can be expressed as:

$$E = E(V) \quad (8)$$

Then, we can iteratively adjust V to minimize the error function $E(V)$. First, the search starts at an initial point V_0 and moves along one of n directions. n denotes the number of elements of the vector V. Then the l-th direction vector e_k^l at iteration k can be defined as:

$$e_k^l = (0,...,0,\overset{l}{1},0,...,0)^T \quad (9)$$

A sequence of iterations $V_0, V_1,..., V_k,...$ can be produced as follows. For $k \geq 0$, iteration k is initiated with V_k, then find a new point along direction e_k^l, $l = 1,2,...,n$, such that

$$E(V_k + \Delta_k e_k^l) < E(V_k) \quad (10)$$

Or $$E(V_k - \Delta_k e_k^l) < E(V_k) \quad (11)$$

$l = 1,2,...,n$, where Δ_k is a positive step size parameter.

If such a point is found, then the iteration is declared successful, and the next iteration becomes

$$V_{k+1} = V_k + \Delta_k e_k^l \quad (12)$$

Or $V_{k+1} = V_k - \Delta_k e_k^l$ (13)
where $l = 1,2,...,n$.

If no such point is found, then the iteration is declared unsuccessful, and the algorithm will make a local exploration near the current iteration. That is to say, the next iteration is initiated at the same point $V_{k+1} = V_k$, and the step size parameter Δ_{k+1} is reduced to $\eta\Delta_k$, where $0 < \eta < 1$ is a constant over all iterations. And the initial vector V_0 and the step size Δ_0 are given in advance.
As can be seen, the pattern search method performs the local search iteratively and minimizes the error function along with the set of decent direction directly. It finally lead the network to the local minimum of E, and hence, to a solution of the problem. Namely the error function E is always decreased by any parameters change produced in the method.

3. Annealing Chaotic Pattern Search Method for MFNNs

Both the back-propagation-based learning algorithm (BP) and the pattern search-based learning algorithm (PS) have gained much attention and widespread applications in different fields. However, the performance of the BP algorithm and PS algorithm greatly depend on their initial values, and they often suffer the problem of being trapped in local optima so as to prematurely converge. In order to avoid these disadvantages, we propose a hybrid pattern search method of solving the local minimum problem by incorporating chaos and applied to MVL network learning.

As a kind of characteristic of nonlinear systems, chaos is a bounded unstable dynamic behavior that exhibits sensitive dependence on initial conditions and infinite unstable periodic motions. Although it appears to be stochastic, it occurs in a deterministic nonlinear system under deterministic conditions. In recently years, growing interests from physics, chemistry, biology and engineering have stimulated the studies of chaos for control [29], synchroniazation [30] and optimization [31].
One of the famous chaos system, Logistic equation, is introduced in the process of chaotic dynamic local search method defined by the following equation:

$$\lambda_i(k+1) = \alpha\lambda_i(k)(1-\lambda_i(k)) \quad (14)$$

where λ_i denotes the i-th chaotic variable and k represents the iteration number. When $\alpha = 4$, this system reveals chaotic phenomena. Obviously, $\lambda_i(k)$ is distributed in the interval $(0,1)$ under the conditions that the initial $\lambda_i(0) \in (0,1)$ and that $\lambda_i(0) \notin \{0.25,0.5,0.75\}$.

In the original pattern search method, a combination of exploratory move and pattern move is made iteratively to search out the optimum solution for the problem. As described above, an exploratory move is performed in the vicinity of the current point systematically to find the best point around the current point. Namely, a positive perturbation $(V_k + \Delta_k e_k^l)$ or a negative perturbation $(V_k - \Delta_k e_k^l)$. Then by incorporating the chaotic dynamics, we expand the exploratory move to be

$$V_k \pm \lambda(k)\Delta_k e_k^l \quad (15)$$

where the chaotic factor is given in Eq.(14). Furthermore, in order to eliminate the fluctuation of the chaos in the latter phrase of the process, an annealing strategy is also utilized. In this condition, $\lambda(k)$ in Eq.(15) can be expressed as:

$$\lambda'(k+1) = \alpha(k)\lambda'(k)(1-\lambda'(k)) \quad (16)$$

$$\alpha(k+1) = \alpha(k)(1-\beta) \quad (17)$$

$$\lambda(k) = 1 - 2\lambda'(k) \quad (18)$$

where β is a damping factor for the epoch-dependent $\alpha(k)$ $(0 \le \beta \le 1)$ and Fig.1 shows its chaotic dynamic where $\alpha(0) = 0.1$.

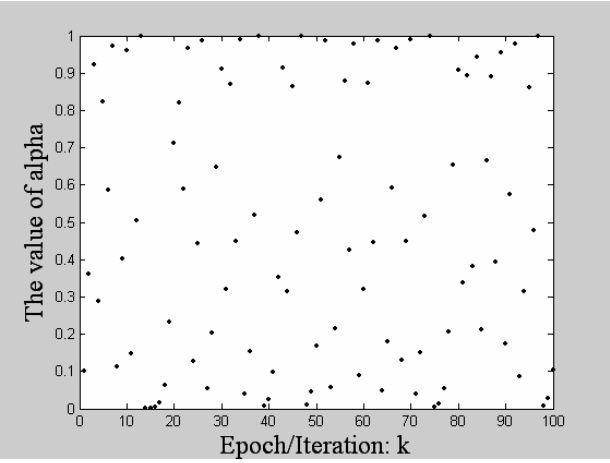


Fig.1 The dynamics of Logistic Map

By combining the chaotic dynamics into the pattern search method, the ACPS method has more elaborate dynamics. In the earlier phase of the approach the algorithm taking full advantages of the ergodic and stochastic properties of chaotic variables exploit the local solution space, and the motion of the chaotic variables in their ergodic space is used to explore the whole solution space. Moreover, with the iteration going, the fluctuation caused by chaotic system has less and less power on the algorithm and finally it comes to only have the trend of descent just the same with the original pattern search method. Obviously, the

convergence of the proposed algorithm can also be guaranteed [24],[25]. Furthermore, neither explicit estimate of the derivative nor anything like a Taylor's series appears in the proposed algorithm. This makes the algorithm useful in situations where derivatives are not available or difficult to get. Moreover, as the learning is performed by simply changing vector by a small positive or negative constant and it relies exclusively on values of the objective function, the algorithm is extremely simple to be specified and be implemented in hardware applications.

Generally, the proposed algorithm (ACPS) can be described as follows:

Step 1. Initialize the vector V and other parameters;
Generate a vector V_0 (the weight and threshold vector) randomly. Set the maximum epoch (Max_Epoch), the initial chaotic variable $\lambda(0)$ and the damping factor β, a step reduction factor η $(0<\eta<1)$, a error criterion ξ and a termination parameter which usually is a small positive constant ε.
Set $\delta_{L1},\delta_{L2},...,\delta_{Ln},\delta_{R1},\delta_{R2},...,\delta_{Rn}=\Delta_0>0$, where δ is the update step associated with each weight and threshold and here the subscripts L and R indicate the negative and positive move respectively.

Step 2. Set epoch counter $k=0$. Let $V_l^c=V_0$ and $l=1$.
Here V_l^c is a copy of vector V_0 and $V_l^c, V_2^c,..., V_n^c$ construct an epoch which searches a better solution in the variable space for all weights and thresholds (It should be noted that n iterations ($V_l^c, V_2^c,..., V_n^c$) construct an epoch in our algorithm.

Step 3. For $l=1$ to n (that is for all weights and thresholds), carry out:
If $E(V_l^c+\delta_{Rl}e_k^l)<E(V_l^c)$,
Then: $V_{l+1}^c=V_l^c+\lambda(k)\delta_{Rl}e_k^l$,
Else {
$\delta_{Rl}=\eta\delta_{Rl}$,
If $E(V_l^c-\delta_{Rl}e_k^l)<E(V_l^c)$
Then: $V_{l+1}^c=V_l^c-\lambda(k)\delta_{Rl}e_k^l$
Else: $\{V_{l+1}^c=V_l^c,\delta_{Ll}=\eta\delta_{Ll}.\}$
}

Step 4. Adapt the vector V (weights and thresholds);
If $E(V_n^c)<E(V_k)$ (That is to say, there is an improvement after an epoch),
Then {
Set the new vector $V_{k+1}=V_n^c$,
$V_l^c=V_{k+1}+\psi(V_{k+1}-V_k)$,
$k=k+1, l=1$.
}
Else {
Set the new vector $V_{k+1}=V_k$,
$V_l^c=V_k$,
$k=k+1, l=1$.
}
where the user-defined parameter ψ (usually, $\psi\geq 1$) is used to enlarge the search space of the current point and thus improve the search efficiency [22].

Step 5. Estimate stop conditions:
If $E(V_{k+1})<\xi$ (where ξ is a preselect error criterion),
Then {
Stop the pattern search at point V_{k+1} and the training is considered to be successful.
}
Else{
If all $\delta(\delta_{L1},...,\delta_{Rn})<\varepsilon$, or the number of iteration reach (Max_Epoch),
Then: stop search at point V_k and the training is considered to be unsuccessful.
Else: go to Step 2.
}

4. Simulation Results

In this section, we will demonstrate the effectiveness and robustness of ACPS method by applying it to several benchmark problems, such as exclusive-or (XOR) problem, parity problem and Arabic numerals recognition and then compare the simulation results with those of the traditional gradient descent method, backpropagation algorithm [16], the simulated annealing algorithm [9] and the original pattern search method [22].In our simulations, the weight update rule used in the backpropagation algorithm is given by:

$$\Delta_p w_{ji}(n+1)=-\nu\frac{\partial E_p}{\partial w_{ji}}+\mu\Delta_p w_{ji}(n) \qquad (19)$$

where ν is the learning rate, μ is the momentum term that determines the effect of past weights changes on the current weight changes. Moreover, ν was set to be 1.0 and μ was set to be 0.8 for all trials. The parameters used in the simulated annealing and original pattern search methods were the same with those in [9] and [22] respectively. In the ACPS method, some parameters were set as follows:

$\lambda(0) = 0.1, \beta = 0.009, \eta = 0.99, \varepsilon = 0.01, \Delta_0 = 0.1, \psi = 1.0$

It should be noted that although there may be more suitable values of these parameters for the problems, it is difficult to set them theoretically. Furthermore, in order to make the proposed algorithm be a general method, these parameters should not be affected by individual problem too much. Thus, we remained them the same values during all the simulations.

4.1 Exclusive-or (XOR) Problem

The exclusive-or problem which is a classic problem requiring hidden units and involved as a subproblem in many other different problems [2] is usually used as a basis for illustrating the limitations of the computational power of MFNNs. A simple architecture (2-2-1 network) with one hidden layer containing two hidden unites was employed to solve the exclusive-or problem, where the activation function was the sigmoid function.

In order to see the effect of the annealing chaotic dynamic to the training process, we compared the learning curves of the original pattern search method and the proposed one and illustrated them in Fig.2. As both of the methods were sensitive to the initial settings, especially the initial weights and thresholds V_0 of the network, we used the same randomly initialized vector V_0 in the training process to make a fair comparison. The maximum epoch was set as 5000 to give enough time for training and the error criteria was 0.01 in both cases. In Fig.2, the horizontal axis denoted the iterations of the algorithm (actually, 45253 and 2485 iterations for the original pattern search method and the ACPS method, respectively) and it was in a logarithmic scale. The vertical axis denoted the error (in Eq.(3)) and was in a linear scale. Form Fig.2, we can find that the pattern search method had a fast convergence speed at first and at about 700 iteration it fell into a local minimum and could not improve its performance further. However the proposed algorithm performed a fluctuant curve in the earlier phrase of learning. Then with the iteration going, it had a quick descent and finally led the network to a global minimum. Moreover, we illustrated the value of the annealing parameter $\lambda(k)$ versus the epoch of the process in Fig.3 whose horizontal axis denoted the epoch that is a cycle of iterations. If the value of $\lambda(k)$ is smaller than 0, the sequence of iterations will be the contrary direction of the original pattern search method and the vale of the error function will ascend; if the value of $\lambda(k)$ is equal to 0, the selected sequence will stay at the original value and the value of the error function will be unchanged; if the value of $\lambda(k)$ is larger than 0, the selected sequence will

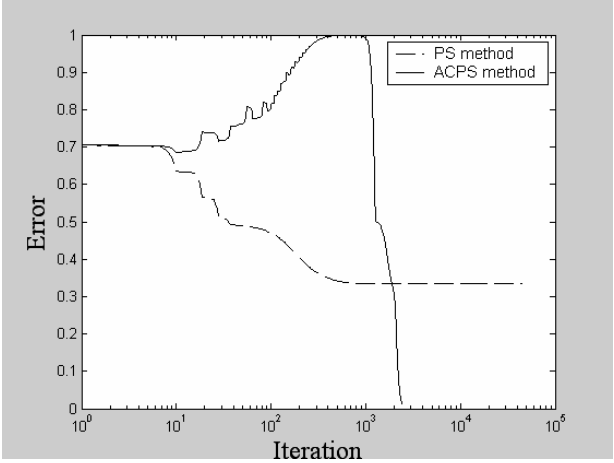


Fig. 2 The comparison of learning curve between the proposed algorithm and the original pattern search method for XOR problem

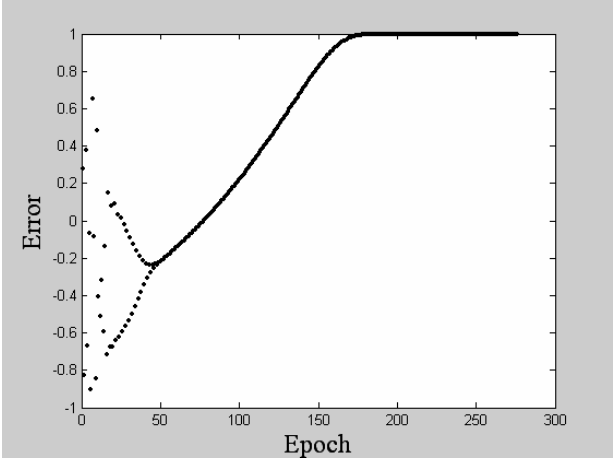


Fig. 3 The annealing parameter versus the epoch of the process.

search the landscape in the same direction with a different velocity, and in the case of $\lambda(k) = 1$, the algorithm is just the same as the original one. Then from Fig.3, we can find that in the early phrase the system reveals chaotic phenomena which caused the fluctuant of the learning, and in this period, the system exploited the landscape in an ergodic manner. Then the periodic value of $\lambda(k)$ which mostly is smaller than 0 caused the drastically increment and in this condition the system has powerful ability of escaping of the local minimum. Then when $\lambda(k)$ is equal to 1, the system has a direct descent and finally lead the system to the optimal solution. By comparing the learning performance between the original pattern search method and the proposed algorithm, we found that the proposed algorithm (ACPS) can search the landscape in an ergodic manner by introducing the annealing chaotic dynamic into the pattern search method and thus resulting in escape

from local minima and possible convergence to global minimum or a better solution.

Furthermore, for the purpose of evaluating the sensitivity of the proposed algorithm to the initial setting, we generated initial weights and thresholds vectors randomly for the XOR problem in (-1,+1), (-2,+2),...,(-10,+10) respectively and compared to the backpropagation algorithm and the original pattern search method. We implemented the simulation 100 times to make a statistical comparison. We illustrated the comparison results in terms of the success rate and the average computation time in Fig.4 and Fig.5 respectively. From Fig.4 we found that although the success rates of the BP algorithm were high at small initial settings, they became

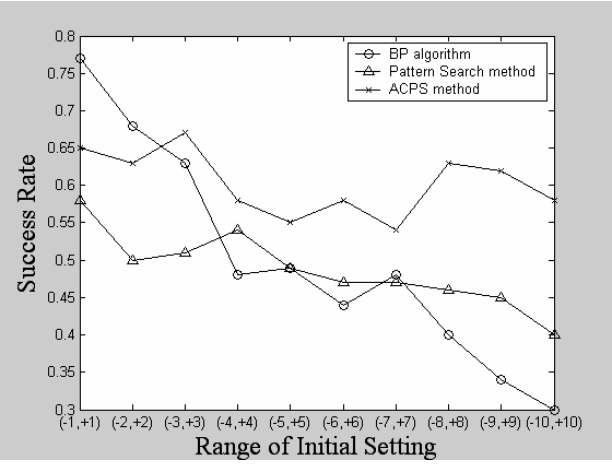


Fig.4 Comparison of networks with various initial settings: Success rate versus Range initial setting

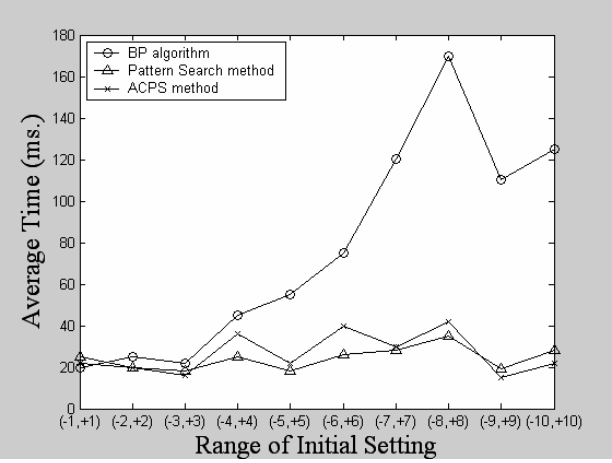


Fig.5 Comparison of networks with various initial settings: Average Time versus Range initial setting.

worse as the range of initial vectors grew larger. Compared to the backpropagation algorithm, both the original pattern search method and the proposed algorithm performed smoother success rate. This is because the pattern search method based algorithms do not require numerical function values which are affected by the initial settings significantly. The relative rank of objective values is sufficient and they can accept new iterates that produce simple decrease in the objective. Moreover, the proposed algorithm outperformed the original one inmost range of initial settings. Besides, Fig.5 revealed the average computation time (where the symbol ``ms." denoted ``millisecond") consumed for successful runs by the algorithms. It is obvious that the training time cost by the backpropagation algorithm increased drastically while that of the pattern search based algorithms did not change manifestly and remained low. Consequently, the proposed algorithm is more robust and efficient than both the backpropagation and patter search algorithms.

4.2 Parity Problem

The parity problem is one of the more widely used problems for testing neural network training algorithms. This problem is a mapping problem where the domain set consists of all distinct N-bit binary vectors and the result of the mapping is 0 if the number of ones in the vector if even, and 1 otherwise. The problem is considered to be very hard since the output changes whenever any single bit in the input changes. The case $N = 2$ is the well-known exclusive-or problem. An $N - M - 1$ (N-input, M-hidden neurons and 1-output) architecture was used for the parity problem and the number of hidden neurons were selected from.

Table 1: Simulation results for the parity problems

N -bit	Network	Algorithm	S.R.	A. Time
4-bit	4-6-1	BP	88%	7754
		SA	97%	293965
		PS	62%	676
		ACPS	78%	689
5-bit	5-10-1	BP	74%	53704
		SA	95%	2862956
		PS	83%	4909
		ACPS	95%	5123
6-bit	6-12-1	BP	10%	27555
		SA	100%	30105990
		PS	64%	14904
		ACPS	85%	15321
7-bit	7-14-1	BP	3%	1058061
		SA	0%	
		PS	42%	121779
		ACPS	73%	123520

Then we applied the proposed algorithm to learn a number of parity problems with input patterns ranging from size four to seven and compared the simulation results to those of the backpropagation algorithm (BP), simulated annealing (SA) and original pattern search (PS) methods. The maximum learning epoch was set as 30000 for all algorithms. Since the initial values of weights and thresholds affect the convergence of a learning algorithm, it is reasonable to judge each algorithm by the statistics obtained from multiple runs. In our experiments, for each simulation of the four algorithms, 100 sets of different initial weights and thresholds were randomly generated from -1 to 1 to be used for training. Table 1 showed the comparison results of these algorithms where "S.R." denotes the success rate of the algorithms and "A. Time" denotes the average CPU time (millisecond) consumed by the algorithms. It should be noted that the average times were obtained by averaging the computation times of the successful runs. Obviously, the backpropagation algorithm had a rapidly degressive success rate when the problem became larger. Although the simulated annealing method could learn the networks with very high success rate, it consumed too much computation time, and thus in the 7-bit parity problem the network could not convergence to a global minimum in reasonable times (30000 epoch). Compared to the backpropagation and simulated annealing methods, both the original pattern search method and the proposed method could lead the networks to a local or global minimum in a relative short time even for the large problems. Meanwhile, the proposed algorithm had a higher success rate than the original one because of its powerful ability of escaping from local minima.

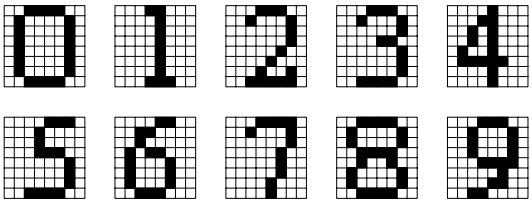


Fig. 6 Input patterns of Arabic numerals recognition

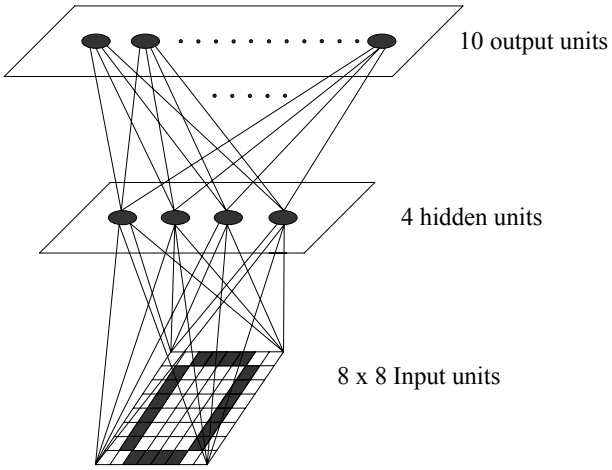


Fig. 7 The architecture of the network to solve the Arabic numerals recognition problem

Table 2: Simulation results for the Arabic numerals recognition problem

Algorithm	Success Rate	Average Time
BP	90%	28493
SA	0%	
PS	96%	17078
ACPS	99%	17536

4.3 Arabic numerals recognition

Furthermore, for the purpose of evaluating the effectiveness of our proposed algorithm for some high-dimensional and practical problems, we applied our algorithm to a larger network that is set up for a more artificial task, Arabic numerals recognition. This task is a classical pattern classification problem. Our simulation involved recognition Arabic numerals 0-9 in an 8 by 8 pixel input field (Fig.6). In order to solve this problem, a relatively complex architecture was employed. Fig.7 shows the basic structure of the network we employed. Input layer consisted of 64 units that were conceptualized as two-dimensional patterns corresponding to 8 by 8 pixel numeral array. Hidden layer had 4 units, each of which was fully connected to all output units. The number of the output units was set to 10. Therefore, each of the 10 output units corresponded to one of these characters.

We illustrated the simulation results in Table 2. The error criteria used for the algorithms was 0.01 and all weights and thresholds were also initialized from the range $(-1.0,+1.0)$. The statistics in this table were based on 100 trials of simulations. It can be seen form the table that the

proposed algorithm significantly outperformed the backpropagation algorithm, the simulated annealing algorithm and the original pattern search method in terms of both the global optimization and convergence speed.

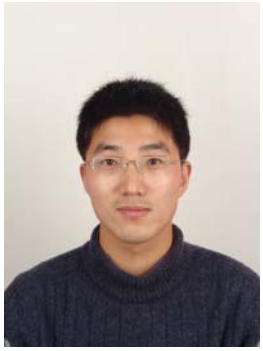
4. Conclusions

We proposed a pattern search method with an annealing chaotic dynamic for learning multi-layer artificial neural networks. The proposed approach was designed as a simple direct search method and thus it could be applied in hardware implementations easily. Furthermore, due to the introduction of chaotic dynamics, the proposed algorithm exploited the landscape in an ergodic manner so that it had ability to escape from local minima and eventually reached the global minimum state or its best approximation with very high probability. We tested this algorithm on several benchmark problems and the simulation results showed that the systems could be trained efficiently by our method for all problems.

References

[1] M.T.Hagan, H.B.Demuth, and M.Beale, Neural Network Design, PWS Pub.,1996.
[2] D.E.Rumelhart, G.E.Hinton, and R.J.Williams, "Learning internal representations by back-propagating errors," In Parallel Distributed Processing: Explorations in the Microstructure of Cognition, pp.318-362, MIT Press, Cambridge, MA, 1986.
[3] L.F.A.Wessels and E.Barnad, "Avoiding false local minima by properinitialization of connections," IEEE Trans. Neural Network, vol.3, no.6, pp.899-905, 1992.
[4] S.Haykin, Neural Networks: A Comprehensive Foundation, MacmillanPub., 1994.
[5] R.P.Brent, "Fast training algorithms for multilayer neural mets," IEEE Trans. Neural Network, vol.2, no.3, pp.346-354, 1991.
[6] M.T.Hagan and M.Menhaj, "Training feedforward networks withMarquardt algorithm," IEEE Trans. Neural Networks, vol.5, no.6, pp.989-993, 1994.
[7] P.J.Angenline, G.M.Sauders, and J.B.Pollack, "An evolutionaryalgorithm that constructs recurrent neural networks," IEEE Trans.Neural Networks, vol.5, no.1, pp.54-65, 1994.
[8] S.Kirkpatrick, C.D.Gelatt, and M.P.Vecchi, "Optimization bysimulated annealing," Science, vol.220, no.4598, pp.671-680, 1983.
[9] C.B.Owen and A.M.Abunawass, "Application of simulated annealing tothe backpropagation model improves convergence," SPIE Proc., vol.1966, pp.269-276, 1993.
[10] S.K.Chang, O.A.Mohammed, and S.Y.Hahn, "Detection of magnetic bodyusing article neural networks with modified simulated annealing," IEEE Trans. Magn., vol.30, pp.3644-3647, 1994.
[11] S.Shaw and W.Kinsner, "Chaotic simulated annealing in multilayerfeedforward networks," Proc. of Canadian Conf. on Electrical andComputer Engineering, vol.1, pp.265-269, 1996.
[12] C.Erick, Efficient and Accurate Parallel Genetic Algorithms, KluwerAcademic Pub., 2000.
[13] A.V.Ooyen and B.Nienhuis, "Improving the convergence of theback-propagation algorithm," Neural Networks, vol.5, no.4, pp.465-471, 1992.
[14] M.Ahmad and F.M.A.Salam, "supervised learning using the Cauchyenergy function," Proc. of International Conference on Fuzzy logicand Neural Networks, 1992.
[15] R.A.Jacobs, "Increased rates of convergence through learning rateadaption," Neural Networks, vol.1, pp.295-307, 1988.
[16] T.P.Vogl, J.K.Mangis, A.K.Zigler, W.T.Zink, and D.L.Alkon, "Accelerating the convergence of the backpropagation method," Biological Cybernetics., vol.59, pp.256-264, Sept. 1988.
[17] M.K.Weirs, "A method for self-determination of adaptive learningrates in back propagation," Neural Networks, vol.4, pp.371-379, 1991.
[18] A.Likas, D.A.Karras, and I.E.Lagaris, "Neural networks training andsimulation using a multidimensional optimization system," Int. J. ofComputer Mathematics, vol.67, pp.33-46, 1998.
[19] D.S.Chen and R.C.Jain, "A robust back propagation learningalgorithm for function approximation," IEEE Trans. Neural Networks, vol.5, no.3, pp.467-479, 1994.
[20] F.Glover and G.A.Kochenberger, eds., Handbook of Metaheuristics,International Series in Operations Research & Management Scientce(57), Kluwer, 2003.
[21] R.Hooke and T.A.Jeeves, "Direct search: Solution of numerical andstatistical problems," J.Association of Computing Machinery, pp.212-224, 1961.
[22] X.G.Wang, Z.Tang, H.Tamura, and M.Ishii, "Multi-layer NetworkLearning Algorithm Based on Pattern Search Method," IEICE Trans.Fundamentals, vol.E86-A, no.7, pp.1869-1875, 2003.
[23] R.M.Lewis, V.Torczon, and M.W.Trosset, "Direct search methods: thenand now," Journal of Computational and Applied Mathematics, vol.124, pp.191-207, 2000.
[24] R.M.Lewis, V.Torczon, and M.W.Trosset, "Why pattern search works," SIAM J. Optim., vol.59, pp.1-7, 1998.
[25] V.Torczon, "On the convergence of pattern search algorithms," SIAMJ. Optim. vol.7, pp.1-25, 1997.
[26] R.M.Lewis, and V.Torczom, "Pattern search methods for linearlyconstrained minimization," SIAM J. Optim. vol.10, pp.917-941, 2000.
[27] C.Audet, and J.E.Dennis Jr., "Analyis of generalized patternsearches," SIAM J. Optim. vol.13, pp.889-903, 2003.
[28] T.G.Kolda, R.W.Lewis, and V.Torczon, "Optimizaiton by directsearch: a new perspective on some classical and modern methods," SIAM Rev., vol.45, pp.385-482, 2003.
[29] T.Kapitaniak, "Continuous control and synchronization in chaotic system," Chaos, Solitons and Fractals, vol.6, pp.237-244, 1995.
[30] L.Pecora, T.Carroll, "Synchronization in chaostic system," Phys Rev Lett, vol.64, pp.821-824,1990.
[31] L.Zhao, L.S.Shieh, G.Chen, N.P.Coleman, "Simplex sliding mode control for nonlinear uncertain systems via chaos

optimization", Chaos,Solitions and Fractals, vol.23, pp.747-755,2005.



Shangce Gao received the B.S. degree from Southeast University, Nanjing, China in 2005. Now, he is working toward the M.S. degree at Toyama University, Toyama, Japan. His main research interests are multiple-valued logic and artificial neural networks.