IEEE Int. Conference Neural Networks & Signal Processing
Zhenjiang, China, June 8~10, 2008

Back-propagation with Chaos

Farideh Fazayeli, Lipo Wang, and Wen Liu
School of Electrical and Electronic Engineering
Nanyang Technological University
Singapore 639798
elpwang@ntu.edu.sg

***Abstract*—Multilayer feed-forward neural networks are widely used based on minimization of an error function. Back-propagation is a famous training method used in the multilayer networks but it often suffers from a local minima problem. To avoid this problem, we propose a new back-propagation training based on chaos. We investigate whether randomicity and ergodicity property of chaos can enable the learning algorithm to escape from local minima. Validity of the proposed method is examined by performing simulations on three real classification tasks, namely, the Ionosphere, the Wincson Breast Cancer (WBC), and the credit-screening datasets. The algorithm is shown to work better than the original back-propagation and is comparable with the Levenberg-Marquardt algorithm, but simpler and easier to implement comparing to Levenberg-Marquardt algorithm.**

I. Introduction

Multilayer feed-forward neural networks are widely used and are based on minimization of an error function. The basic learning method for the feed-forward networks is the back-propagation (BP) algorithm [2]. BP learning uses the gradient descent procedure to modify the connection weights such that the network can approximate a subjective function. Although BP works well for many problems, such as classification and function approximation, it often suffers from the local minima problem. Assorted methods have been proposed for the BP to avoid being entrapped into the local minima, such as Adaptive Learning Rate, Levenberg-Marquardt algorithms [4], etc. In this paper we propose a new method for BP learning with chaos. Chaos has some important properties e.g., randomicity and ergodicity, which make chaos useful for a wide range of fields, e.g., mathematics, physics, engineering, and economics.

Some methods have been proposed which embed chaotic dynamics into the neural network. Nozawa [3] showed the existence of chaos in Euler approximation of the Hopfield network [1] by adding a negative self-feedback connection. Chaotic simulated annealing (CSA) is proposed by Chen and Aihara [5] and uses a sufficiently large negative self-feedback to a Hopfield neural network and gradually reduces the self-feedback. Wang and Smith [6] suggested reducing the time step rather than the self-feedback in CSA. In [8], Wang et al. proposed stochastic chaotic simulated annealing, by adding a stochastic noise into CSA.

In this paper we study the effects of chaos in improving the performance of BP learning. We study whether randomicity and ergodicity property of chaos can enable the BP networks to escape from the local minima and settle down in global optimal points.

The paper is organized as follows. Section 2 summarizes the basics of the BP theory. In Section 3, the proposed method of back-propagation learning based on chaos is outlined. Section 4 shows the potential of the proposed method on classifying three real datasets, namely, the Ionosphere [9], the Wincson Breast Cancer (WBC) [9], and the credit-screening [11] datasets. We discuss our results and draw some conclusions in the last section.

II. Basic theory of BP

Multilayer neural networks (MLN) [2] consist of one input layer, one output layer, and one or more hidden layers. Each layer contains a set of neurons and is fully connected with an adjacent layer. Each connection link is represented by a weight. The goal of MLN learning is approximation of a subjective function. It adjusts the weights such that the discrepancies between the network outputs and the target values are minimized. This process is called supervised learning. The MLN Learning includes two phases. The first phase is the feed-forward stage which calculates the network outputs in response to the corresponding inputs. In this phase the connection weights are fixed. The input signals propagate through the network's layer during the feed-forward phase. Each neuron performs a transfer function which is differentiable and non-decreasing. The most commonly used transfer function is the sigmoid function as follows,

$$f(x) = \frac{1}{1+e^{-x}} \tag{1}$$

The second phase, BP, modifies the connection weights. In this phase an error signal will be obtained by comparing the network outputs with the target values. Then, the error signal spreads backward in the network layers and the weights are adjusted. The most popularly used error function is the mean squared error (MSE) that is given by:

$$E = \frac{1}{n}\sum_{p=1}^{n}\sum_{i=1}^{m}(\hat{y}_{pi} - t_{pi})^2 \tag{2}$$

where the notations are:

n	number of training samples
m	output vector dimension
$\hat{y}_{pi}$	network output of i-th neuron for pattern p
t_{pi}	target value of i-th component for pattern p

BP applies a gradient descent procedure to minimize the error function E as follows,

$$\Delta W_{ij} = -\eta \frac{\partial E}{\partial W_{ij}} \tag{3}$$

where the notations are:

W_{ij}	connection weight between neurons i and j
η	learning rate

Two phases perform alternatively till the subjective function can be approximated with the smallest error. Sometimes BP be entrapped in the local minima which stops the learning in a suboptimal point. We propose a new method for BP learning based on chaos to avoid the local minima problem. The details of the proposed method are discribed in the next section.

III. Back-propagation with Chaos

Chaos describes the aperiodic behavior of nonlinear dynamic systems. It is highly sensitive to the initial conditions. Chaotic dynamics may appear random, but are deterministic, meaning that the dynamic behaviors can be predicted by their initial conditions. In this paper we investigate how chaos can enable the back-propagation learning to escape from the local minima. We propose the following formula for updating the connection weights instead of (3),

$$\Delta W_{ij}(t+1) = -\eta(t) \frac{\partial E}{\partial W_{ij}(t)} \tag{4}$$

$$\eta(t+1) = (1-\beta)\eta(t) \tag{5}$$

where β $(0 \leq \beta \leq 1)$ is the damping factor of the time-dependent $\eta(t)$.

In order to observe the chaos behavior in the network, $\eta(0)$ should be chosen sufficiently large and β should be sufficiently small. Both of them are problem dependent and should be chosen empirically. With choosing a large learning rate, the network jumps through the weight space. Hence the network can trace the larger space of the weights and reach the global minimum. We then gradually reduce the learning rate to stabilize the network.

However, with choosing a very large η, the network may fall in the saturation region and the learning process becomes too slow to converge to any local minima. In order to avoid it, we need to restrict the weights to a special range such as $(-\alpha, \alpha)$ i.e., if the weight becomes greater than α, we map it to α and if the weight becomes less than $-\alpha$, it is mapped to $-\alpha$, where α is problem dependent. In the next section we describe some experimental simulation on some real datasets.

IV. Simulation results

In order to verify the effectiveness of the BP with chaos we compared it with some famous algorithms, such as the standard BP, the Adaptive Learning Rate (ALR), and the Levenberg-Marquardt algorithms [4] (LM). The BP, ALR, and LM algorithms are those included in the Matlab. We have applied the proposed method to three real classification problems, namely, the Ionosphere, the Wincson Breast Cancer (WBC), and the credit-screening datasets.

All data were standardized into mean value of zero and variance of one. We randomly divided data into 3 parts i.e., 60% as the training, 20% as the test, and 20% as the validation set. The training and validation sets are used for finding the structure of network and tuning the initial parameters. The input attributes were rescaled to the range $(-1.0, 1.0)$ by a linear function. The target outputs were encoded using a 1-of-m output representation for the m classes. The tangent hyperbolic function was used as the transformation function in all layers as follows,

$$tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{6}$$

Hence the target outputs were mapped to the range $(-0.9, 0.9)$ using a linear function. The parameter β was set to 0.001 for all three datasets. In the following, we explain the simulation results for the three datasets.

A. Ionosphere dataset

The problem is classification of radar returns from the ionosphere. The dataset was created by Johns Hopkins University. The radar data was collected by a system in Goose Bay, Labrador. There were 351 examples in total and 34 attributes used to represent the pattern. All attributes were continuous. Each pattern belonged to two classes: 'good' or 'bad', which were free electrons in the ionosphere. This dataset was obtained from the UCI repository [9].

Finding the optimal structure of a network is a challenging job. We applied the LM algorithm to 10 different network topologies to find the best structure of the network with the same initial parameters [7]. Then, we chose the smallest topology which had the least generalization error. The following structure were examined, 33-2-2, 33-4-2, 33-8-2, 33-16-2, 33-2-2-2, 33-6-3-2, 33-4-4-2, 33-8-4-2, 33-8-8-2, and 33-16-8. We chose 33-6-3-2 (33 inputs, 6 hidden neurons in the first layer, 3 hidden neurons in the second layer, and 2 outputs) as the structure of the network which had the smallest test error.

Each algorithm had some initial parameters that needed to be tuned before the learning stage. We tuned the parameters for each algorithm using a 10-fold cross validation procedure and selected those parameters which had the smallest test error. Table I represents the tuned parameters for each algorithm. For comparing the algorithms, we called each algorithm on 100 trials with different initial weights and the same network topology, and the tuned parameters. All initial weights were randomly chosen from a uniform distribution in the range $(-2.0, 2.0)$.

In order to avoid the overfitting problem, we let the network learning complete 3000 epochs for each trial. Then, the adjusted weights with the minimal validation error were returned. In Figure 1, the chaos behavior of the BP is observed. As shown in the figure, it is obtained that the proposed method enables the BP to escape from the local minima.

TABLE I
TUNED PARAMETERS FOR THREE DATASETS

Algorithm	Parameters	Ionosphere	WBC	Credit-screening
BP	Lr[a]	0.7	0.1	0.9
ALR	Lr[a]	0.05	0.05	0.9
	Lr dec[b]	0.1	0.1	0.5
	Lr inc[c]	1.4	1.2	1.05
LM	Mu[d]	10	20	20
	Mu dec[e]	0.5	0.1	0.9
	Mu inc[f]	5	20	20
BP with chaos	$\eta(0)$[g]	5	5	5
	Alpha[h]	0.7	0.9	1.2

[a] Lr : Learning rate
[b] Lr dec : Lr decreasing factor
[c] Lr inc : Lr increasing factor
[d] Mu: The initial value of μ
[e] Mu dec : Mu decreasing factor
[f] Mu inc : Mu increasing factor
[g] $\eta(0)$: The initial value of learning rate
[h] Alpha: Weight range

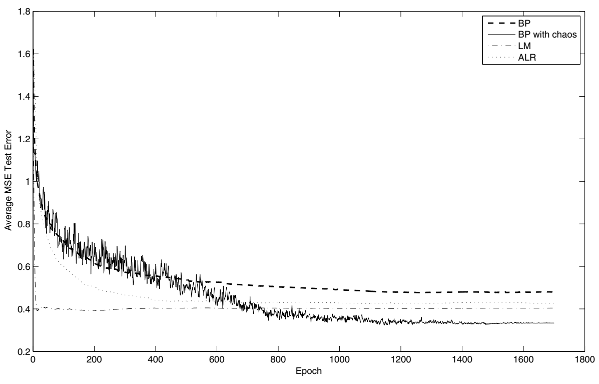

Fig. 1. Average Learning Curve for the Ionosophere dataset

TABLE II
AVERAGE MSE ERROR FOR THE IONOSOPHERE DATASET

Algorithm	Training set	Validation set	Test set
BP	0.1671 ± 0.19	0.3759 ± 0.20	0.4698 ± 0.19
ALR	0.0928 ± 0.09	0.3058 ± 0.11	0.4137 ± 0.12
LM	0.0669 ± 0.08	0.264 ± 0.09	0.3849 ± 0.09
BP with chaos	0.0094 ± 0.01	0.2259 ± 0.04	0.3272 ± 0.04

Average learning error for the training, the test, and the validation sets are presented in Table II. The results demonstrate that the proposed method has a smaller learning error for all training, test, and validation sets than the original back-propagation. Chaos can noticeably improve the training error compared to all other algorithms. The test and validation errors are a little better than LM algorithm and dramatically less than the BP errors.

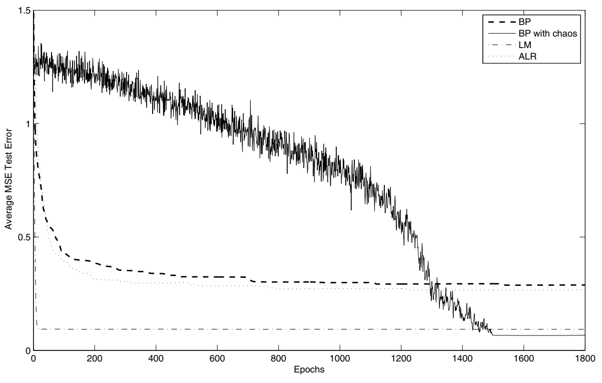

Fig. 2. Average Learning Curve for the WBC dataset

B. Wincson Breast Cancer(WBC) dataset

It was originally obtained by W. H. Wolberg at the University Wisconsin Hospitals, Madison [10]. The dataset contained 699 examples i.e., 458 patterns belonged to the benign and 241 patterns belonged to the malignant class. There were 10 integer attributes in this set including the class attribute. This dataset was obtained from the UCI repository [9].

In order to find the best network structure, the same procedure as the Ionosphere dataset was applied. Here we examined the following topologies, 9-2-2, 9-4-2, 9-8-2, 9-16-2, 9-24-2, 9-32-2, 9-2-2-2, 9-4-2-2, 9-4-4-2, 9-8-4-2, 9-8-8-2, and 9-16-8-2. Among them the 9-2-2-2 (9 inputs, 2 hidden neurons in the first layer, 2 hidden neurons in the second layer, and 2 outputs) structure is chosen for the following procedures. Then, the initial parameters were tuned for each algorithm as we had done for the Ionosphere dataset. Table I shows the tuned parameters for each algorithm.

For comparing the algorithms, each algorithm was called on 100 trials with different initial weights but the same network structure and tuned parameters. All initial weights were randomly chosen from a uniform distribution in the range $(-2.0, 2.0)$. We let the network learning complete 3000 epochs and then returned the adjusted weights of network with minimal validation error to avoid the overfitting problem. The chaos behavior of the BP is represented in Figure 2. It is obtained that the BP with chaos has the least average MSE test error.

Table III shows the average learning error for the training, the test, and the validation sets. The results demonstrate that the proposed method has a smaller learning error for all three sets. Chaos can noticeably improve the test and validation errors in contrast with all other algorithms. The training error is as good as the LM and the ALR errors and less than the BP training error.

C. credit-screening dataset

The problem was to classify the approval or non-approval of a credit card to a customer. It had a good mix of attributes:

TABLE III
AVERAGE MSE ERROR FOR THE WBC DATASET

Algorithm	Training set	Validation set	Test set
BP	0.3103 ± 0.41	0.2974 ± 0.41	0.2884 ± 0.41
ALR	0.1923 ± 0.32	0.1909 ± 0.32	0.1738 ± 0.32
LM	0.1017 ± 0.18	0.1039 ± 0.18	0.0978 ± 0.19
BP with chaos	0.094 ± 0.005	0.064 ± 0.004	0.043 ± 0.004

TABLE IV
AVERAGE MSE ERROR FOR THE CREDIT-SCREENING DATASET

Algorithm	Training set	Validation set	Test set
BP	0.2984 ± 0.04	0.3642 ± 0.02	0.3037 ± 0.03
ALR	0.3094 ± 0.07	0.3685 ± 0.05	0.3054 ± 0.06
LM	0.296 ± 0.04	0.3745 ± 0.03	0.3076 ± 0.03
BP with chaos	0.2237 ± 0.02	0.4081 ± 0.04	0.3487 ± 0.03

continuous, nominal with small numbers of values, and nominal with larger numbers of values. There were totally 690 examples and 52 attributes. Each pattern belonged to either the approval or the non-approval class. 44% of samples belonged to the positive class. This dataset was obtained from Proben1 benchmark datasets [11].

The same procedure as the Ionosphere dataset was used for finding the best network structure. Here 51-2-2-2 structure (51 inputs, 2 hidden neurons in the first layer, 2 hidden neurons in the second layer, and 2 outputs) with the smallest test error was selected among the following structures, 51-2-2, 51-4-2, 51-8-2, 51-16-2, 51-2-2-2, 51-4-2-2, 51-4-4-2, 51-8-4-2, 51-8-8-2, and 51-16-8-2. Then, the initial parameters were tuned for each algorithm as we had done for the Ionosphere dataset. The tuned parameters for each algorithm are presented in Table I.

For comparing the algorithms, we called the algorithms on 100 trials with different initial weights but the same network structure and tuned parameters. All initial weights were randomly chosen from a uniform distribution in the range $(-2.0, 2.0)$. We let network learning complete 3000 epochs. Then we returned the adjusted weights of network with minimal validation error to avoid the over fitting problem. Table IV shows the average learning errors. The result demonstrates that all algorithms have the same behavior. The data may have had a flat local minimum that all algorithms were not able to avoid.

V. CONCLUSION

MLN [2] uses BP as a learning algorithm, but the BP often suffers from the local minima problem. In this paper, we proposed a new method for the BP learning with chaos. We investigated whether randomicity and ergodicity of chaos can enable the BP algorithm to escape from the local minima. We started the learning with a significantly large learning rate. Then it was gradually reduced to stablize the network at a global minimum. This method does not require any additional computing and does not change the network topology compared to the original BP. The proposed method also requires as much memory as the original BP. We applied the proposed method to three real classification datasets. For evaluating the proposed method, we compared it with three famous learning algorithms i.e., the BP, the ALR, and the LM [4] algorithms. The proposed method had a smaller learning error than all other three algorithms for the Ionosphere and the WBC datasets. The proposed method was not able to handle the flat local minima problem such as the minima in the credit-screening dataset. The algorithm is shown to work better than the original BP and is comparable with the LM, but simpler and easier to implement comparing to the LM algorithm.

REFERENCES

[1] J. J. Hopfield, *Neurons with graded response have collective computational properties like those of two-state neurons*, in Proc. Nat. Acad. Sci. USA, vol. 81, pp. 3088-3092, May 1984.

[2] D. E. Rumeihart, G. E. Hinton, and R. J. Williams, *Learning Internal Representations by Error Propagation*, In Parallel Distributed Processing, Explorations in the Microstructure of Cognition, Volume 1: Foundations. MIT Press, Cambridge, Massachusetts, pp. 318-362, 1986.

[3] H. Nozawa, *A neural-network model as a globally coupled map and applications based on chaos*, Chaos vol. 2, no.3, pp. 377-386, 1992.

[4] M. T. Hagan, and M. Menhaj, *Training feed-forward networks with the Marquardt algorithm*, IEEE Transactions on Neural Networks, vol. 5, no. 6, pp. 989-993, 1994.

[5] L. N. Chen and K. Aihara, *Chaotic simulated annealing by a neural network model with transient chaos*, Neural Netw., vol. 8, no. 6, pp. 915-930, 1995.

[6] L. P. Wang and K. Smith, *On chaotic simulated annealing*, IEEE Trans. Neural Netw., vol. 9, no. 4, pp. 716-718, Jul. 1998. 1

[7] S. Haykin, *Neural Networks, A Comprehensive Foundation.*, second ed., Englewood Cliffs, N.J.: Prentice Hall, 1999.

[8] L. P. Wang, S. Li, F. Y. Tian, and X. J. Fu, *A noisy chaotic neural network for solving combinatorial optimization problems: Stochastic chaotic simulated annealing*, IEEE Trans. Syst., Man, Cybern. B, Cybern., vol. 34, no. 5, pp. 2119-2125, May 2004.

[9] A. Asuncion, and D. J. Newman, (2007). UCI Machine Learning Repository [http://www.ics.uci.edu/ mlearn/MLRepository.html]. Irvine, CA: University of California, School of Information and Computer Science.

[10] W. H. Wolberg, and O. L. Mangasarian, *Multisurface method of pattern separation for medical diagnosis applied to breast cytology*, Proceedings of the National Academy of Sciences, U.S.A., vol. 87, December 1990, pp 9193-9196

[11] L. Prechelt, *PROBEN1A set of neural network benchmark problems and benchmarking rules*, Tech. Rep. 21/94, Univ. Karlsruhe, Karlsruhe, Germany, 1994.