ELSEVIER

Neurocomputing 11 (1996) 101–106

NEUROCOMPUTING

Letters

Fast learning method for back-propagation neural network by evolutionary adaptation of learning rates

Heung Bum Kim [*], Sung Hoon Jung, Tag Gon Kim, Kyu Ho Park

Department of Electrical Engineering, Korea Advanced Institute of Science and Technology, 373-1, Kusong-dong, Yusong-gu, Taejon 305-701, South Korea

Received 31 October 1995; accepted 8 February 1996

Abstract

In training a back-propagation neural network, the learning speed of the network is greatly affected by its learning rate. None, however, has offered a deterministic method for selecting the optimal learning rate. Some researchers have tried to find the sub-optimal learning rates using various techniques at each training step. This paper proposes a new method for selecting the sub-optimal learning rates by an evolutionary adaptation of learning rates for each layer at every training step. Simulation results show that the learning speed achieved by our method is superior to that of other adaptive selection methods.

Keywords: Back-propagation neural network; Adaptive learning rates; Evolutionary programming

1. Introduction

One of the most important things in training a neural network is its learning speed because the training process generally takes much time. The learning speed of the network greatly depends on the value of the learning rate, η. However, it is very difficult or even impossible to find the optimal learning rate for a neural network.

A back-propagation algorithm with a gradient method has a tendency to get stuck in the local minima of the error surface and becomes unable to find the global minimum. Some researchers have tried to find a sub-optimal learning rate using various techniques at every training step [2–4]. In this paper, we describe an evolutionary programming technique for finding the sub-optimal learning rates by an evolutionary adaptation of η.

[*] Corresponding author. Email: hbkim@fast.kaist.ac.kr

SSDI 0925-2312(96)00009-4

In this approach, we used different learning rates for every layer at each training step. It was shown through simulation that the convergence rate and the learning speed of our method were superior to those of several adaptive selection methods [2–4].

2. Evolutionary programming

Evolutionary programming [1] is an algorithm that simulates the natural evolution process. This algorithm consists of three main processes: (1) evaluating fitness, (2) selecting parents through competition, and (3) generating offspring. Algorithm 1 shows the evolutionary programming procedure used in our experiments.

Algorithm 1 Evolutionary-Programming()

```
begin
    // $2m$: the number of population ($m$ parents + $m$ offspring) //
    // $X_i$: the $i$th vector of population $P$ //
    // $F(X_i)$: the objective function of vector $X_i$ //
    // $J_i$: the fitness score of vector $X_i$, $J_i \leftarrow F(X_i)$ //
    // $W_i$: the number of winning in competition //
    // $\sigma_i$: the perturbation factor of vector $X_i$, $\sigma_i \propto 1/J_i$ //
1   initialize population $P = [X_0, X_1, \ldots, X_{2m-1}]$
2   while (not termination-condition)
3       assign a score $J_i$ to each solution vector $X_i$
4       compete and count the winning number of $X_i$ with the others $X_j(\neq X_i)$
5       select $m$ parents solutions according to the number order of $W_i$
6       generate offspring by adding a Gaussian perturbation N(0, $\sigma_i$) to parents
7   end while
```

3. Evolutionary adaptation of learning rates

To apply the evolutionary programming to our problem, the objective function $F(X_i)$ and the standard deviation σ_i of Algorithm 1 must be first defined. The objective function evaluates the fitness of the solutions in a population. In our application, the inverse of the Total Sum Square Error (TSSE) of a back-propagation network can be used as the fitness.

Definition 1 (Objective function). Let X_i be a vector (learning rates) of a population P. Then an *objective function* is defined as

$$F(X_i) = \frac{1}{TSSE_{X_i}} = \frac{1}{\sum_{p=1}^{R} \sum_{p=1}^{K} \left(d_{pk} - o_{pk}^{X_i}\right)^2},$$

where d_{pk} and $o_{pk}^{X_i}$ are the desired output and the actual output of a pattern p in an output neuron k on a vector X_i, respectively. R and K are the number of patterns and output neurons, respectively.

The perturbation factor, σ_i, is closely related with the fitness and the number of patterns and output neurons. The perturbation must be inversely proportional to the fitness and the two numbers, R and K. This is because if a vector has small fitness, its offspring must be largely mutated to broadly search for better solutions; the searching of a new solution should be more scrutinized when the number of patterns and output neurons is large.

Definition 2. (Standard deviation (perturbation factor)). Let X_i be a vector of population. Then the *standard deviation* is defined as

$$\sigma_i = \frac{1}{\sqrt{F(X_i)} * R * K},$$

where R and K are the same as the above definition.

Fig. 1 shows the overall block diagram for training a back-propagation neural network. Evolutionary programming iteratively applies the three operations – assigning costs to population, selecting parents through competition, and generating offspring – to the vectors at every training step. The best learning rates selected from the evolutionary programming are offered to the back-propagation algorithm at every iteration step.

4. Simulation

A classification problem with 1024 sample patterns was employed to measure the performance of our method. The problem is to classify a 10-bit binary input to a decimal

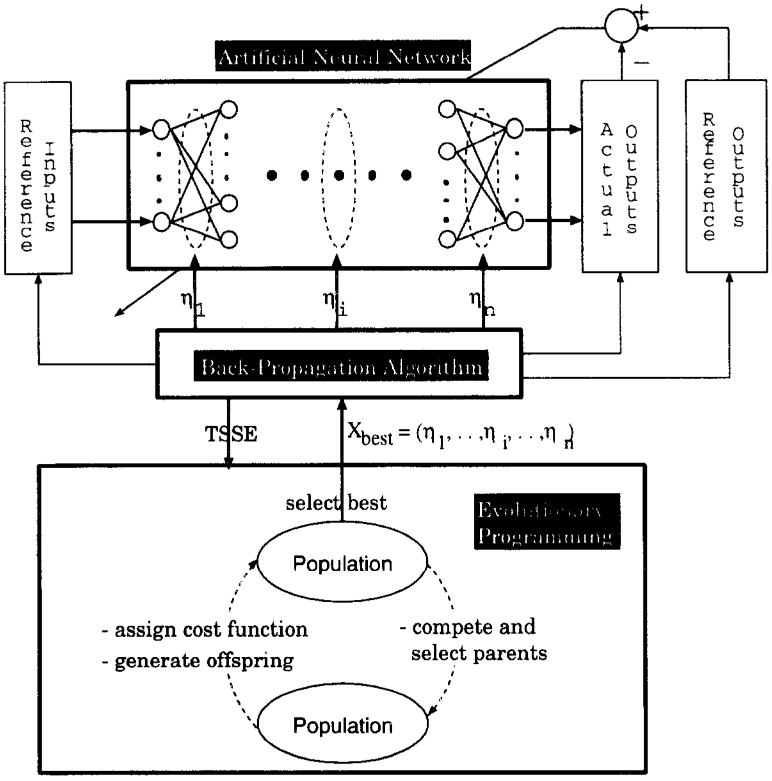


Fig. 1. Overall block diagram of our method.

modulus output. Thus, this problem takes 10-bit binary inputs and generates 10 modulus outputs. For example, if a 10-bit binary 0000001011 is provided, then only the second output becomes one and all other outputs become zeros, 0100000000. This is because the binary number corresponds to modulus 1. We used a three-layered (10-10-10) neural network structure. Our method and three other adaptive selection methods [2–4] are extensively simulated. Figs. 2 and 3 show the experimental results in terms of convergence rate and training time, respectively. In all of the results, the values of TSSE are the average values of 10 runs under different initial learning rates. "Our Methods(#1, #2)" indicate the two results for (50, 20) population and (5, 3) competition, respectively. The learning rate of Weir method is given as

$$\frac{E}{extreme(\delta E/\delta w)(\delta E/\delta w)},$$

where E is error, $\delta E/\delta w$ is the gradient of E with respect to weight w, and $extreme(\delta E/\delta w)$ is the extreme value of $(\delta E/\delta w)$. In our simulation, the $E/extreme(\delta E/\delta w)$ is set to 0.5 which is recommended in his paper. Thus the resulting equation of learning rate is $0.5/(\delta E/\delta w)$. Hsin [2] suggested a computing method for learning rate. The learning rate is given by

$$\eta(k) = \alpha_0 \frac{\Delta w(k) \cdot \Delta w(k-1)}{\| \Delta w(k) \| \cdot \| \Delta w(k-1) \|} + \cdots + \alpha_L \frac{\Delta w(k-L) \cdot \Delta w(k-L-1)}{\| \Delta w(k-L) \| \cdot \| \Delta w(k-L-1) \|}$$

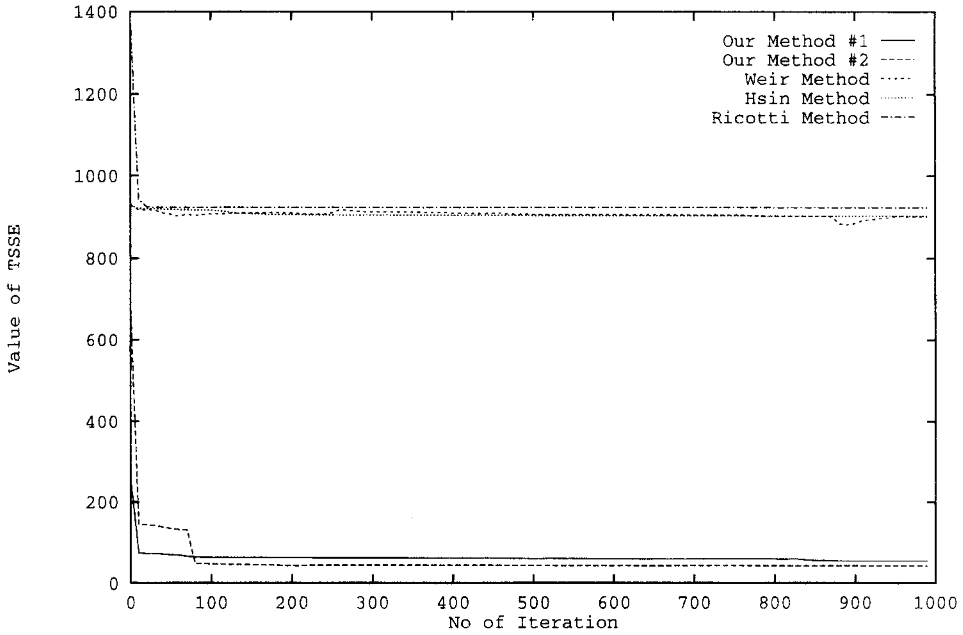


Fig. 2. TSSE vs. no of iteration in 10-10-10 network.

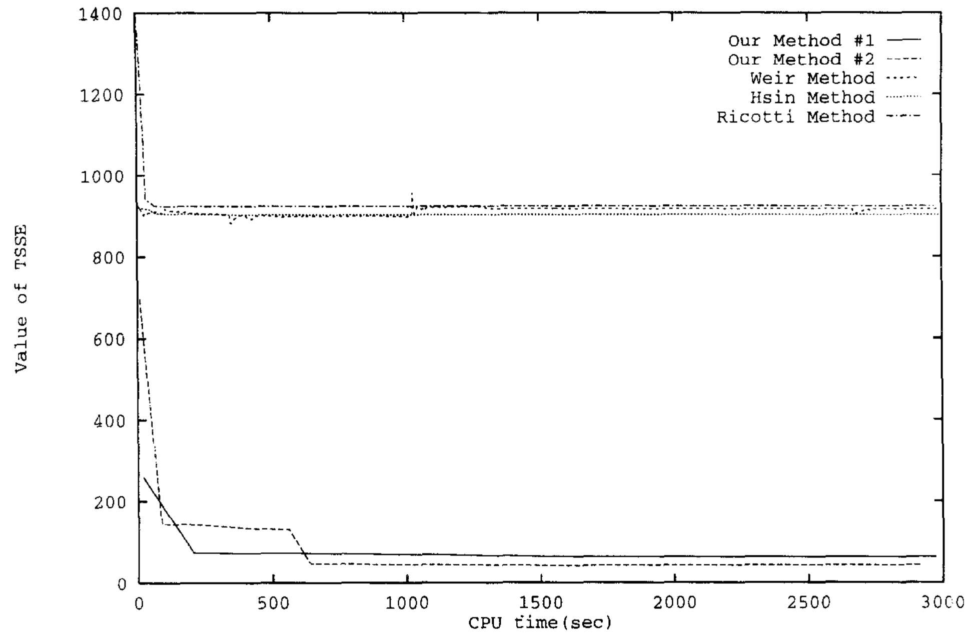


Fig. 3. TSSE vs. CPU time on an ALPHA workstation in 10-10-10 network.

with constraint $\alpha_0 + \alpha_1 + \cdots + \alpha_L = 1$, and $\alpha_0 \geqslant \alpha_1 \geqslant \cdots \geqslant \alpha_L$. The coefficients α_i's of "Hsin Method" were assigned by the following relationships: $\alpha_i = Ae^{-i}$ $(i = 0, 1)$, $A - 1/\Sigma_{k=0}^{1} e^{-k}$. In Fig. 3, the CPU time of our methods includes the network training time and the run-time of the evolutionary programming algorithm. Results show that our method outperforms several adaptive selection methods considerably in the terms of the convergence rate and training speed.

5. Conclusion

A new method for fast training of back-propagation neural networks was proposed. This method is based on the evolutionary adaptation of learning rates. Using a classification problem of 10bit-to-decimal_modulus, we simulated a neural network learning with 10-10-10 structure. Simulation results show that our method was faster and more reliable than some adaptive selection methods [2–4].

Acknowledgement

We would like to thank the three anonymous reviewers, whose insightful suggestions contributed to improve this article.

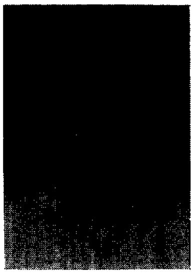
References

[1] D.B. Fogel, An introduction to simulated evolutionary optimization, *IEEE Trans. Neural Networks* 5 (1994) 3–14.

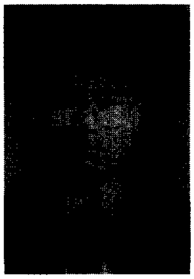
[2] H. Hsin, C. Li, M. Sun and R.J. Sclabassi, An adaptive training algorithm for back-propagation neural networks, *IEEE Trans. SMC* 25 (1995) 512–514.

[3] L.P. Ricotti, S. Ragazzini and G. Martinelli, Learning of word stress in a sub-optimal second order back-propagation neural network, *Proc. IEEE Internat. Conf. on Neural Networks* (1988) 355–363.

[4] M.K. Weir, A method for self-determination of adaptive learning rates in back propagation, *Neural Networks* 4 (1991) 371–379.

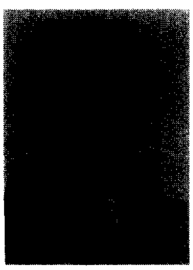


Heung Bum Kim is a candidate of Ph.D. in the Department of Electrical Engineering at Korea Advanced Institute of Science and Technology. He received his B.S. degree from Hankook Aviation College, Korea, in 1979 and M.S. from KAIST, in 1989. His research interests are neural networks, genetic algorithms, discrete-event-system, and artificial intelligence.



Sung Hoon Jung holds a postdoctoral position in the Department of Electrical Engineering at Korea Advanced Institute of Science and Technology. He received his B.S.E.E. degree from Hanyang University, Korea, in 1988 and M.S. and Ph.D. degrees from KAIST, in 1991 and 1995, respectively. His research interests are in the field of intelligent control, neural networks, fuzzy control, genetic algorithms, and event-based control.

Tag Gon Kim is an Associative Professor in the Department of Electrical Engineering at Korea Advanced Institute of Science and Technology. He received his B.S.E.E. and M.S.E.E. degrees from Pusan National University, Korea, in 1975 and 1980, respectively. He received the Ph.D. degree in electrical engineering from the University of Arizona, Tucson, AZ, in 1988. His research interests include neural network system, artificial intelligent for advanced simulation methodology and computer system modeling.



Kyu Ho Park is a Professor in the Department of Electrical Engineering at Korea Advanced Institute of Science and Technology. He received his B.S. degree from Seoul National University, Korea, in 1973 and the M.S. degree from KAIST, in 1975. He received the Dr. Ing. degree from the Université de Paris, France, in 1983 in electrical engineering. His major interests include computer vision, computer architecture and parallel processing.