2004 RISP International Workshop on Nonlinear
Circuit and Signal Processing (NCSP'04)
Hawaii, USA, Mar. 5-7, 2004.

An Improvement of Learning Efficiency by Noise Adding to a Feed-Forward Neural Network

Koki Nishizono†, Yoko Uwate† and Yoshifumi Nishio†

Tokushima University
2-1 Minami-Josanjima, Tokushima 770-8506, Japan
Tel: +81-88-656-7470
Fax: +81-88-656-7471
Email:{koki, uwate, nishio}@ee.tokushima-u.ac.jp

1. Introduction

As a learning algorithm of feed-forward neural networks, the error reverse propagation learning (BP, Back Propagation) method is used widely. However, the BP method requires a large amount of time because it needs repeated learning.

In this study, in order to improve learning efficiency of a feed-forward neural network, a technique adding various sizes and patterns of noise to the connection weights between the input layer and the middle layer or between the middle layer and the output layer is proposed. At first, a noise with various sizes is added to learn exclusive logical sum (EXOR), which is an example of easy problems. Next, a noise divided to various patterns is added to learn some grayscale patterns, which is a somewhat complicated problem.

2. Noise to be Added

The standard noise used in this research is a uniform noise spread over an interval [-0.5, 0.5] as shown in Fig. 1. This noise is added to the connection weights between the input layer and the middle layer or between the middle layer and the output layer of a feed-forward neural networks. As varying the amplitude of the noise by multiplying an integer between 0 and 8, the learning time of the networks is investigated.

3. EXOR Problem

Figure 2 shows the feed-forward neural network used for the EXOR problem. Number of the middle layer is one. Accordingly, this network consists of 3 layers. The input layer is composed of two elements and the output layer is composed of one. The number of the middle layer's elements N_m is changed from 3 to 15. A teacher signal is given to the input layer and the weights are updated according to the BP method. The learning is repeated until an error signal becomes smaller than a certain value (in this research, this value is set to 10^{-4}).

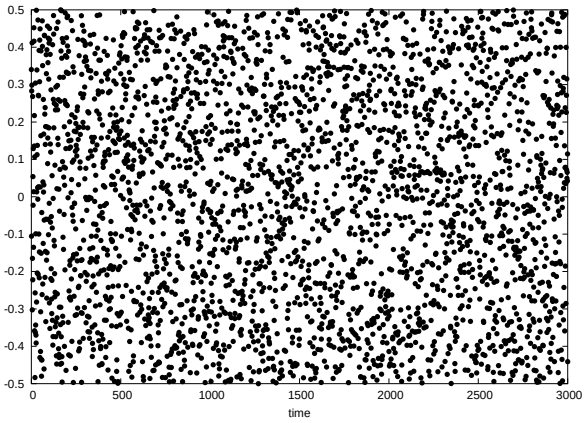

Figure 1: Uniform noise.

Figure 3 shows some of the simulation results for the EXOR problem. The figure shows the relation between the amplitude of the uniform noise and the average of 100 simulation results of the learning time.

When the number of the middle layer's elements N_m is small, the learning time is almost constant regardless of the amplitude of the noise. While when N_m is large, the learning time varies depending on the amplitude of the noise and it is not good for a wide range of the amplitude of the noise. But, we should mention that the best learning time is obtained from the case of $N_m = 15$.

4. Pattern Recognition

Next, we consider a pattern recognition problem. Figure 4 shows four gray scale patterns used for the pattern recognition problem. Figure 5 shows the feed-forward neural network used for this problem. The input and the middle layers have four elements and the output layer has two elements.

At first, the uniform noise with various sizes of the ampli-

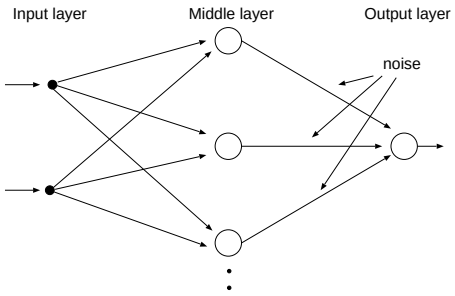

Figure 2: Feed-forward neural network for EXOR problem.

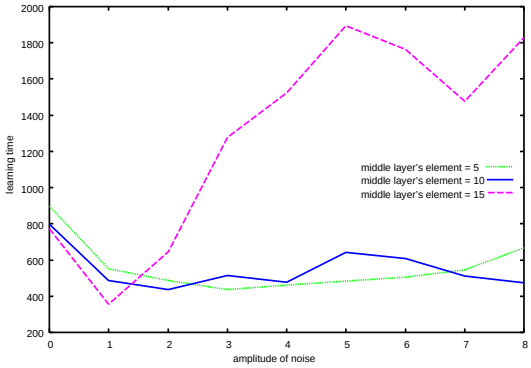

Figure 3: Simulated results for EXOR problem.

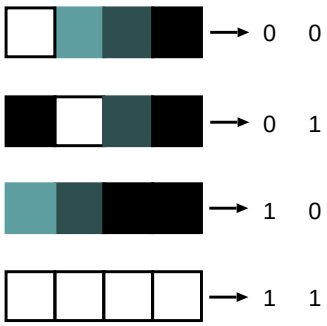

Figure 4: Four grayscale patterns.

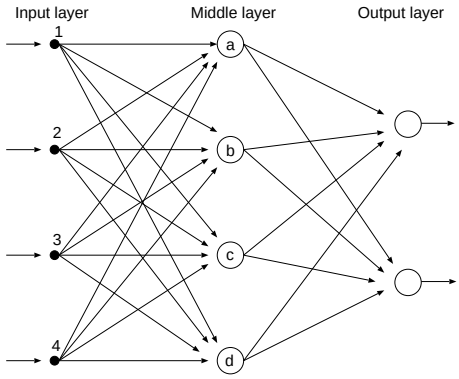

Figure 5: Feed-forward neural network for pattern recognition problem.

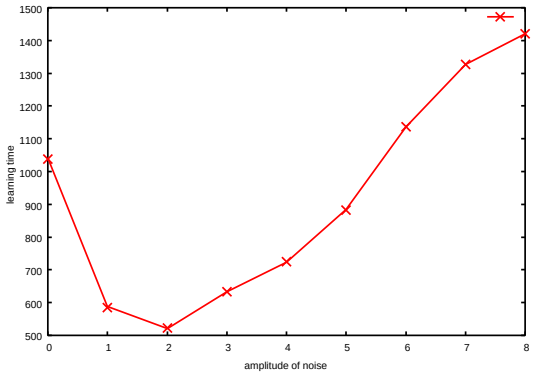

Figure 6: Simulated results for pattern recognition problem.

tude is added to all the connection weights between the input layer and the middle layer.

Figure 6 shows the simulation results for the pattern recognition problem. From the figure, we can say the best result is obtained from the case of $N_m = 2$ and the best learning time is 527.08.

Next, we consider changing the input patterns of the noise. Namely, we choose the weights to which the noise is added. Because there are 16 weights between the input layer and the middle layer, the number of the combinations of the choosing becomes $2^{16} = 65536$. In order to reduce the computation cost, we carry out computer simulations for the four different cases. Namely, the first case is adding the noise to only the weights between the element 1 and the middle layer and the second case is adding the noise to only the weights between the element 2 and the middle layer. Similarly, the third and four cases are adding the noise to only the weights from the elements 3 and 4, respectively. Tables 1, 2, 3, and 4 summarize the learning times for the four cases. In the tables, "1" means that the noise with $N_m = 2$ is added to the corresponding weight, while "0" means that no noise is added to the weight. From Table 1, the two best learning times with noise to w_{1i} are obtained for the cases of $\{0,1,1,1\}$ and $\{1,1,1,1\}$. From Table 2, the two best learning times with noise to w_{2i}

are obtained for the cases of $\{1,1,1,0\}$ and $\{1,1,1,1\}$. From Table 3, the two best learning times with noise to w_{3i} are obtained for the cases of $\{0,1,1,1\}$ and $\{1,1,1,1\}$. From Table 4, the two best learning times with noise to w_{4i} are obtained for the cases of $\{0,1,1,1\}$ and $\{1,1,1,1\}$. Hence, we consider combining the cases. For the sixteen combinations using the two best cases in the four tables, we carried out computer simulations. The results are summarized in Table 5. The best learning time is 438.63 and this is smaller than the learning time 527.08 obtained when the noise is added to all of the weights.

Therefore, we can say that adding noise to the connection weights can improve the learning time of the pattern recognition problem. However, selecting the weights to which noise is added and the amplitude of the noise is important to obtain good results.

5. Conclusions

In this study, in order to improve learning efficiency of a feed-forward neural network, a technique adding the various sizes and patterns of noise to the connection weights between the input layer and the middle layer or between the middle layer and the output layer has been proposed.

We confirmed the effectiveness of the method for the two examples.

References

[1] Itsuo Kumazawa, *Learning and Neural Networks*, Morikita Shuppan, 1998.

[2] Norio Baba, Fumio Kojima and Seiichi Ozawa, *Bases and Applications of Neural Networks*, Kyoritsu Shuppan, 1994.

[3] Yoshikazu Nishikawa and Shinzou Kitamura, *Neural Networks as Applied to Measurement and Control*, Asakura Syoten, 1995.

Table 1: Learning time with noise to w_{1i}

w_{1a}	w_{1b}	w_{1c}	w_{1d}	learning time
0	0	0	1	1000.27
0	0	1	0	984.67
0	0	1	1	955.52
0	1	0	0	979.32
0	1	0	1	961.84
0	1	1	0	943.91
0	1	1	1	**903.25**
1	0	0	0	986.47
1	0	0	1	961.53
1	0	1	0	954.15
1	0	1	1	933.72
1	1	0	0	951.47
1	1	0	1	931.75
1	1	1	0	943.52
1	1	1	1	**912.63**

Table 2: Learning time with noise to w_{2i}

w_{2a}	w_{2b}	w_{2c}	w_{2d}	learning time
0	0	0	1	946.12
0	0	1	0	930.53
0	0	1	1	866.78
0	1	0	0	931.40
0	1	0	1	868.04
0	1	1	0	868.08
0	1	1	1	810.42
1	0	0	0	942.03
1	0	0	1	886.41
1	0	1	0	871.04
1	0	1	1	811.55
1	1	0	0	866.78
1	1	0	1	805.79
1	1	1	0	**796.65**
1	1	1	1	**767.40**

Table 3: Learning time with noise to w_{3i}

w_{3a}	w_{3b}	w_{3c}	w_{3d}	learning time
0	0	0	1	872.64
0	0	1	0	863.81
0	0	1	1	749.74
0	1	0	0	867.16
0	1	0	1	755.38
0	1	1	0	757.11
0	1	1	1	**657.65**
1	0	0	0	878.35
1	0	0	1	773.94
1	0	1	0	773.04
1	0	1	1	675.94
1	1	0	0	766.43
1	1	0	1	660.42
1	1	1	0	683.44
1	1	1	1	**600.28**

Table 4: Learning time with noise to w_{4i}

w_{4a}	w_{4b}	w_{4c}	w_{4d}	learning time
0	0	0	1	871.05
0	0	1	0	862.72
0	0	1	1	761.00
0	1	0	0	861.33
0	1	0	1	774.26
0	1	1	0	753.80
0	1	1	1	**633.50**
1	0	0	0	876.25
1	0	0	1	763.61
1	0	1	0	773.82
1	0	1	1	668.59
1	1	0	0	754.12
1	1	0	1	656.06
1	1	1	0	638.63
1	1	1	1	**562.56**

Table 5: Learning time with noise to all weight connections

w_{1a}	w_{1b}	w_{1c}	w_{1d}	w_{2a}	w_{2b}	w_{2c}	w_{2d}	w_{3a}	w_{3b}	w_{3c}	w_{3d}	w_{4a}	w_{4b}	w_{4c}	w_{4d}	learning time
0	1	1	1	1	1	1	0	0	1	1	1	0	1	1	1	583.52
0	1	1	1	1	1	1	0	0	1	1	1	1	1	1	1	539.26
0	1	1	1	1	1	1	0	1	1	1	1	0	1	1	1	542.52
0	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1	563.61
0	1	1	1	1	1	1	1	0	1	1	1	0	1	1	1	589.46
0	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1	550.21
0	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1	538.69
0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	**438.63**
1	1	1	1	1	1	1	0	0	1	1	1	0	1	1	1	595.90
1	1	1	1	1	1	1	0	0	1	1	1	1	1	1	1	566.18
1	1	1	1	1	1	1	0	1	1	1	1	0	1	1	1	578.53
1	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1	501.95
1	1	1	1	1	1	1	1	0	1	1	1	0	1	1	1	590.74
1	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1	538.48
1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1	546.92