Performance Improvement of Multilayer Perceptrons with Increased Output Nodes per Class

Sang-Hoon Oh*
**Mokwon University, Korea and also Georgia Institute of Technology, USA*
E-mail:shoh@mokwon.ac.kr

Abstract

Generally, we allocate one output node per class in pattern recognition applications of MLPs(multilayer perceptrons). In this paper, we propose a method to improve generalization capability of MLPs through increasing the number of output nodes per class. We verify that the proposed method decreases misclassification ratios of MLPs through a short mathematical aspect. And then, simulations of isolated-word recognition show the effectiveness of our method.

1. Introduction

MLPs(multilayer perceptrons) have been widely applied to many areas including pattern recognition with a theoretical background[1][2]. Usually, EBP(error back-propagation) algorithm is used for training of MLPs. However, the EBP algorithm based on MSE(mean-squared error) has a drawback with incorrect saturation phenomenon, in which some output nodes in wrong saturation region of sigmoidal activation function cannot escape[3]. This phenomenon degrades generalization capability as well as training convergence of MLPs.

In order to resolve this problem, CE(cross-entropy)[4] and nCE(*n*th-order extension of cross-entropy)[5][6] error functions were proposed. Also, for better generalization performance, CFM(classification figure-of-merit) cost function was proposed[7]. This seeks to maximize the difference between the output value of node representing the correct classification, so called “target node”, and all other nodes.

Besides the above cost function approaches, there have been approaches of boundary pattern selection[8], selective attention[9], and incorporating additional layers[10]. Regarding the boundary pattern selection, *k*-neighbors are found for each datum. If a datum is near the decision boundary, then all of these *k*-neighbors would not come from the same class. Training is performed using the boundary patterns. In order to implement the selective attention ability of human, early filtering model was incorporated in the input layer of MLPs[9]. On the other hand, MOLP(multi-output-layer perceptron) increases the linearly separable ability of networks with incorporating additional output layers[10]. However, increasing the number of layers generally degrades convergence speed.

On the other hand, some tried to accelerate convergence of MLPs through adaptive learning rates[6][11][12]. Also, one may use second order nonlinear optimizing method for accelerated convergence. However, it has a drawback with ill-conditioning of Hessian matrix in many applications and the computational complexity related to the Hessian[13]. Also, LBL(layer-by-layer) optimizing

methods was proposed in which each layer of MLPs is decomposed into both a linear part and a nonlinear part[13][14][15].

In this paper, we propose a new approach to improve classification capability of MLPs, which increases output nodes per class. In pattern recognition applications, we usually assign one output node per class and the index of the maximum output node denotes a classified result. In this structure, the output node value can be interpreted as an estimation of posteriori probability that an input pattern originates from a certain class [5]. In order to improve the performance of classifier, some proposed to implement many classifiers for the same classification problem and to judge the classification result based on all outputs of the classifiers[16]. However, this strategy needs to train the additional judge network for final decision. Contrary to this method, we propose to increase output nodes without additional training strategy.

This paper is organized as follows. In section 2, we briefly introduce MLPs. We propose to increase output nodes per class for better generalization in section 3 and simulations are described in section 4. Finally, section 5 concludes this paper.

2. MLP(Multilayer Perceptrons)

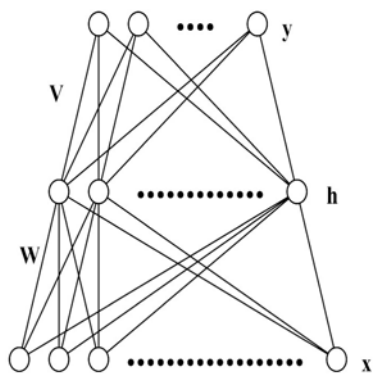

Fig. 1. Multilayer Perceptron

Consider an MLP consisting of *N* inputs, *H* hidden, and *M* output nodes. Here, each node has a value between -1 and 1. Also, let the desired output vector corresponding to an training input vector $\mathbf{x}=[x_1,x_2,\cdots,x_N]^T$ be $\mathbf{t}=[t_1,t_2,\cdots,t_M]^T$. When $\mathbf{x}$ is presented to the network, the state of *j*th hidden node is

$$h_j = f(\hat{h}_j) = \tanh(\hat{h}_j / 2) \tag{1}$$

where

$$\hat{h}_j = \sum_{i=1}^{N} w_{ji} x_i + w_{j0} . \tag{2}$$

Also, $f(.)$ is the sigmoidal activation function, w_{ji} denotes the weight connecting x_i to h_j, and w_{j0} is the bias to h_j. Now, the *k*th output node is

$$y_k = f(\hat{y}_k) = \tanh(\hat{y}_k / 2) \tag{3}$$

where

$$\hat{y}_k = \sum_{j=1}^{H} v_{kj} h_j + v_{k0} . \tag{4}$$

Here, v_{kj} denotes the weight connecting h_j to y_k and v_{k0} is the bias to y_k.

The conventional MSE function is

$$E_m(\mathbf{x}) = \sum_{k=1}^{M} (t_k - y_k)^2 / 2 . \tag{5}$$

To minimize $E_m(\mathbf{x})$ [17], output weights are updated as

$$\Delta v_{kj} = -\eta \frac{\partial E_m(\mathbf{x})}{\partial v_{kj}} = \eta \delta_k^{(out)} h_j \tag{6}$$

where,

$$\delta_k^{(out)} = -\frac{\partial E_m(\mathbf{x})}{\partial \hat{y}_k} = (t_k - y_k) f'(\hat{y}_k) . \tag{7}$$

The hidden weights are updated as

$$\Delta w_{ji} = -\eta \frac{\partial E_m(\mathbf{x})}{\partial w_{kj}} = \eta \delta_j^{(hid)} x_i \tag{8}$$

where,

$$\delta_j^{(hid)} = -\frac{\partial E_m(\mathbf{x})}{\partial \hat{h}_j} = f'(\hat{h}_j) \sum_{k=1}^{M} v_{kj} \delta_k^{(out)} . \tag{9}$$

3. Improving Performance of MLPs

In the above EBP algorithm, $\delta_k^{(out)}$ in Eq. (7) is the difference $(t_k - y_k)$ multiplied by the gradient of the sigmoid function. If y_k approaches one of the two extreme values of sigmoid function, the gradient factor in Eq. (7) makes the delta signal very small. Thus, the output node y_k which has an extreme value opposite to t_k can not make a strong delta signal for adjusting the weights significantly. This incorrect saturation retards the search for a minimum in the error surface. In order to resolve this problem, one can use the nCE error function[5]

$$E_{nCE}(\mathbf{x}) = -\sum_{k=1}^{M} \int \frac{t_k^{n+1} (t_k - y_k)^n}{2^{n-2} (1 - y_k^2)} dy_k . \tag{10}$$

Using the above error function, the delta signal of output layer is

$$\delta_k^{(out)} = -\frac{\partial E_{nCE}(\mathbf{x})}{\partial \hat{y}_k} = \frac{t_k^{n+1} (t_k - y_k)^n}{2^{n-1}} . \tag{11}$$

The other equations in the EBP algorithm are the same. The nCE error signal with $n \geq 2$ can generate a strong error signal for an incorrectly saturated output node and a weak error signal for a correctly saturated output node. This nCE error function shows better generalization performance than the conventional EBP. If we code the target node as

$$t_k = \begin{cases} +1, & \text{if } \mathbf{x} \text{ originates from class } k \\ -1, & \text{otherwise,} \end{cases} \tag{12}$$

with enough training patterns, classifiers based on the nCE error function can be Bayesian[5][18][19].

Usually, we assign one output node per class, which is coded as (12). Some proposed to incorporate many MLPs which are trained separately to resolve a same problem[16]. Even though this strategy can improve the performance, we need to train another network that decides to adopt which network's outputs or merges the outputs of many networks as a final classification result. On the contrary, we propose to increase the number of outputs per class without any incorporation of other networks for final decisions. In this method, we train the MLP based on the nCE error function and use Max. rule for decision, which decides that the index of maximum output node is the classified result.

In order to prove the effectiveness of our method which increases output nodes, we model a two-class problem c_1 and c_2 in which their prior probabilities $p(c_1) = p(c_2)$. Also, we assume that one-output MLP trained to resolve this problem has uniform output distribution for each class. That is, $p(\mathbf{y} \mid c_1)$ is uniform in $[-3\alpha, \alpha]$ and $p(\mathbf{y} \mid c_2)$ is uniform in $[-\alpha, 3\alpha]$. After some derivations, we can estimate that the misclassification ratio based on Bayes' rule is 1/4. If we model two-output MLP with same idea, we could estimate that the misclassification ratio is 1/8. This estimation proves that our method is effective.

4. Simulations

In order to verify the effectiveness of our method, an isolated-word recognition problem was used in which the vocabulary consisted of 50 words and each word was spoken two times by nine speakers. The 900 patterns were used for training after extracting the ZCPA(zero-crossing peak amplitude) feature of 1,024 dimensions[20]. The MLP consisted of 1,024 inputs and 50 hidden nodes. Nine simulations were conducted with initial weights randomly drawn from a uniform distribution on $\left[-1\times10^{-4}, 1\times10^{-4}\right]$ and the results were averaged. Generalization performance for this task was evaluated using untrained 1050 test patterns, which were the 50 words spoken three times by seven speakers. In each simulation, learning rate was 0.05 and we varied the number of outputs per class from 1 to 5.

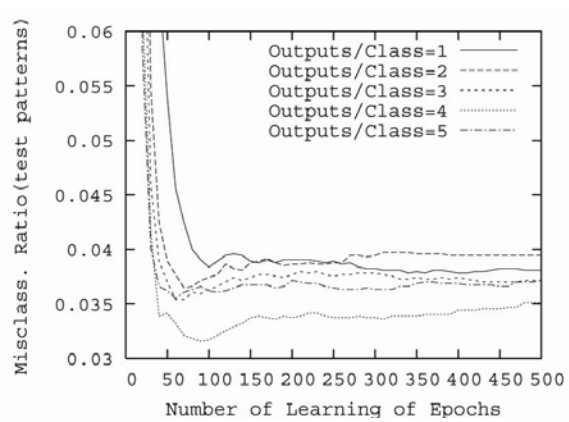

Fig. 2. Misclassification ratio for test patterns with various outputs per class

Fig. 2 shows the misclassification ratio of test patterns with various outputs per class. When the output per class is 1, we took about 3.78% misclassification at 340th epoch. The ratio decreases to 3.65% at 70th epoch with 2 outputs/class, 3.54% at 60th epoch with 3 outputs/class, and 3.16% at 90th epoch with 4 outputs/class. However, the misclassification ratio increases to 3.55% at 60th epoch when the number of outputs per class is 5. From these results, we can verify that the generalization performance is improved with increasing the number of outputs per class. The proposed method has the effect that many classifiers are incorporated into one for better performance. Without any intentional post-processing for incorporating many networks, we can attain improved performance only by EBP training of MLPs with multi-outputs per class.

5. Conclusion

This paper proposed to increase the number of outputs per class for improved generalization performance of MLPs in pattern recognition applications. We verified the effectiveness of proposed method through probabilistic derivation and simulations of isolated-word recognition problems. The proposed method can be interpreted as an incorporation of many classifiers without any additional post-processing.

References

[1] K. Hornik, M. Stincombe, and H. White, “Multilayer Feedforward Networks are Universal Approximators,” *Neural Networks*, Vol. 2, pp. 359-366, 1989.

[2] R. P. Lippmann, “Pattern Classification Using Neural Networks,” *IEEE Communications magazine*, pp. 47-64, Nov. 1989.

[3] Y. Lee, S.-H. Oh, and M. W. Kim, "An analysis of premature saturation in back-propagation learning," Neural networks, Vol. 6, pp. 719-728, 1993.

[4] A. van Ooyen and B. Nienhuis, "Improving the convgence of the back-propagation algorithm," *Neural Networks*, Vol. 5, pp. 465-471, 9992.

[5] S.-H. Oh, "Improving the error back-propagation algorithm with a modified error function," *IEEE Trans. Neural Networks*, Vol. 8, pp. 799-803, 1997.

[6] Sang-Hoon Oh and Soo-Young Lee, ``An adaptive learning rate with limited error signals for training of multilayer perceptrons," *ETRI Journal*, Vol. 22, No. 3, pp. 10-18, Sept. 2000.

[7] J. B. Hampshire II and A. H. Waibel, "A Novel Objective Function for Improved Phoneme Recognirtion Using
Time-Delay Neural Networks," *IEEE Trans. Neural Networks*, Vol. 1, pp. 216-228, June 1990.

[8] B. B. Chaudhuri and U. Bhattacharya, “Efficient Training and Improved Performance of Multilayer Perceptron in Pattern Classification,” *Neurocomputing*, Vol. 34, pp. 11-27, 2000.

[9] K.-Y. Park and S.-Y. Lee, “Out-of Vocabulary rejection Based on Selective Attention Model,” *Neural Processing Letters*, Vol. 12, pp. 41-48, 2000.

[10] F. J. Owens, G. H. Zheng, and D. A. Irvine, “A Multi-Output-Layer Perceptron,” *Neural Computation & Applications*, Vol. 4, pp. 10-20, 1996.

[11] T. P. Vogl, J. K. Mangis, A. K. Rigler, W. T. Zink, and D. L. Alkon, "Accelerating the Convergence of the Back-Propagation Method," *Biol. Cybern.*, Vol. 59, pp. 257-263, 1988.

[12] J. Y. F. Yam and W. S. Chow, "Extended Least Squares Based Algorithm for Training Feedfoward Networks,"
IEEE Trans. Neural Networks, Vol. 8, pp. 806-810, 1997.

[13] R. Paris, E. D. Di Claudio, G. Orlandi, and B. D. Rao, "A generalized Learning paradigm Exploiting the Structure of Feedforward Neural Networks," *IEEE Trans. Neural Networks*, Vol. 7, pp. 1450-1459, 1996.

[14] S.-H. Oh and S.-Y. Lee, ``A New Error Function at Hidden Layers for Fast Training of Multilayer Perceptrons," *IEEE Trans. Neural Networks*, Vol. 10, pp. 960-964, 1999.

[15] C. Yu, M. T. Manry, J. Li, and P. L. Narasimha, "An Efficient Hidden Layer Training Method for the Multilayer Perceptron," *Neurocomputing*, Vol. 70, pp. 525-535, 2006.

[16] J.-H. Jeong, H. Kim, D.-S. Kim, and S.-Y. Lee, "Speaker Adaptation Based on Judge Neural Networks for Real
World Implementations of Voice-Command Systems," *Information Sciences*, Vol 123, pp. 13-24, 2000.

[17] D. E. Rumelhart and J. L. McClelland, *Parallel Distributed Processing*, MIT Press, Cambridge, MA, 1986, pp.
318-362.

[18] H. White, "Learning in Artificial Neural Networks: a Statistical Perspective," *Neural Networks*, Vol. 1, pp. 425-464, 1989.

[19] M. D. Richard and R. P. Lippmann, "Neural Network classifier Estimate Bayesian a Posteriori Probabilities," *Neural Computation*, Vol. 3, pp. 461-483, 1991.

[20]D.-S. Kim, S.-Y. Lee, and R. M. Kil, "Auditory Processing of Speech Signals for Robust Speech Recognition in
Real-world Noisy Environments," *IEEE Trans. Speech and Autio Processing*, Vol. 7, pp. 55-69, 1999.