A New Error Function at Hidden Layers for Fast Training of Multilayer Perceptrons

Sang-Hoon Oh and Soo-Young Lee *

Abstract

This letter proposes a new error function at hidden layers to speed up the training of multilayer perceptrons(MLPs). With this new hidden error function, the layer-by-layer(LBL) algorithm approximately converges to the error back-propagation algorithm with optimum learning rates. Especially, the optimum learning rate for a hidden weight vector appears approximately as a multiplication of two optimum factors, one for minimizing the new hidden error function and the other for assigning hidden targets. Effectiveness of the proposed error function was demonstrated for handwritten digit recognition and isolated-word recognition tasks. Very fast learning convergence was obtained for MLPs without the stalling problem experienced in conventional LBL algorithms.

Keywords - Multilayer perceptrons, error function, fast learning, layer-by-layer algorithm, hidden nodes.

1. Introduction

Multilayer perceptrons(MLPs) have been widely applied to pattern recognition, time series prediction, nonlinear control, and telecommunications. A popular method of training MLPs is the error back-propagation(EBP) algorithm, which is a gradient descent with a fixed learning rate[1] and has a drawback with slow convergence. To accelerate the EBP algorithm, several heuristic rules have been proposed for adapting the learning rates[2, 3]. Also, modified error functions, which are different from popular mean-squared errors(MSEs), show a faster convergence through decreasing the possibility of a premature saturation[4, 5]. In spite of this improved convergence, these methods are still based on the gradient descent with non-optimal learning rates.

One may use second-order nonlinear optimizing methods to accelerate the MLP learning. The critical drawbacks of these methods, however, are the ill conditioning of the Hessian matrix in many applications and the computational complexity related to the Hessian[6].

Recently, the layer-by-layer(LBL) optimizing algorithm was proposed in which each layer of MLPs is decomposed into both a linear part and a nonlinear part[6, 7, 8]. The linear part of each layer is solved through formulation of the well-known least squares problem. Although this algorithm shows fast convergence with much less computational complexity than those of the conjugate gradient or Newton methods, it results in an inevitable problem due to target assignments at hidden nodes. When the targets for a hidden

*The authors are with the Department of Electrical Engineering & Brain Science Research Center, KAIST, 373-1 Kusong-dong, Yusong-gu, Taejon, Korea

layer becomes linearly inseparable, it is impossible to reduce the MSE sufficiently at both the hidden layer as well as the output layer.

Yam and Chow proposed a merged method of the LBL and the EBP algorithms to resolve the stalling problem due to the hidden targets[9]. This merged method does not assign hidden targets for updating hidden weights. However, it adapts the learning rate of the EBP portion heuristically. Ergezinger and Thomsen proposed an optimization layer by layer(OLL) learning algorithm, in which the optimization of hidden weights was reduced to a linear problem by the linearization of the sigmoidal activation function at hidden nodes[10]. A special penalty term was also added to the cost function for limiting the introduced linearization error. Although the OLL algorithm is free from hidden targets, it uses a heuristic rule to control the effect of the penalty term.

This letter proposes a new error function for the hidden layer in order to accelerate the training of MLPs. Especially, it intends to overcome the stalling problem of the LBL algorithm without heuristics or non-optimal learning rates. Section II briefly reviews the LBL algorithm. Section III introduces the new error function of a hidden layer, as an MSE multiplied by the squares of sigmoid slopes for hidden targets, which makes the LBL algorithm approximately equal to the EBP algorithm with optimum learning rates. In section IV, the effectiveness of the proposed error function is demonstrated for handwritten digit recognition and isolated-word recognition tasks. Finally, section V concludes this letter.

2. Layer-By-Layer Optimizing Algorithm

Although it is applicable to general MLPs, let's consider a single hidden-layer perceptron with linear output nodes for the sake of simplicity. When P training patterns $\mathbf{x}^{(p)} = [x_1^{(p)}, x_2^{(p)}, \ldots, x_N^{(p)}]^T$ $(p = 1, 2, \ldots, P)$ with associated target vectors of output layer $\mathbf{t}^{(p)} = [t_1^{(p)}, t_2^{(p)}, \ldots, t_M^{(p)}]^T$ are presented to this MLP, the activations of hidden and output nodes for each training pattern are given as

$$\hat{h_j}^{(p)} = \sum_{i=0}^{N} w_{ji} x_i^{(p)}, \quad h_j^{(p)} = f(\hat{h_j}^{(p)}) = \tanh(\hat{h_j}^{(p)}/2), \quad \text{and} \quad y_k^{(p)} = \hat{y_k}^{(p)} = \sum_{j=0}^{H} v_{kj} h_j^{(p)}, \tag{1}$$

with $x_0^{(p)} = 1$ and $h_0^{(p)} = 1$, respectively, and $t_k^{(p)} = \hat{t_k}^{(p)}$ because of linear output nodes. Here $\mathbf{W} = \{w_{ji}\}$ and $\mathbf{V} = \{v_{kj}\}$ are synaptic matrices for the first and second layers, respectively. The weights should be optimized in order to minimize the MSE at the output layer defined as

$$E^{out} = \frac{1}{2} \sum_{p=1}^{P} \sum_{k=1}^{M} (t_k^{(p)} - y_k^{(p)})^2 = \frac{1}{2} \sum_{p=1}^{P} \sum_{k=1}^{M} (\hat{t_k}^{(p)} - \sum_{j=0}^{H} v_{kj} h_j^{(p)})^2. \tag{2}$$

In the LBL algorithm, each learning epoch is divided into three independent least squares problems which may be solved by a matrix inversion or iterative gradient descent algorithms. Since the least squares problem has a quadratic form of error functional, the gradient descent has an optimum learning rate for efficient convergence. The following steps summarize the three gradient descent algorithms.

- Step 1: With fixed $\mathbf{W}$ and given targets $\mathbf{t}^{(p)}$'s, optimize $\mathbf{V}$ for minimum of E^{out} using

$$\Delta v_{kj} = \eta_k^{out}\alpha_{kj}, \quad \alpha_{kj} \equiv -\frac{\partial E^{out}}{\partial v_{kj}} = \sum_{p=1}^{P} \hat{d_k}^{(p)} h_j^{(p)}, \quad \eta_k^{out} = \frac{\sum_{j=0}^{H} {\alpha_{kj}}^2}{\sum_{p=1}^{P}(\sum_{j=0}^{H}\alpha_{kj}h_j^{(p)})^2} = \frac{\vec{\alpha_k}^T\vec{\alpha_k}}{\vec{\alpha_k}^T\mathbf{C}_h\vec{\alpha_k}}, \quad (3)$$

where $\hat{d_k}^{(p)} = -\partial E^{out}/\partial \hat{y_k}^{(p)} = \hat{t_k}^{(p)} - \hat{y_k}^{(p)}$, $\vec{\alpha_k} = [\alpha_{k0}, \ldots, \alpha_{kH}]^T$, and $\mathbf{C}_h = \{\sum_{p=1}^{P} h_j^{(p)} h_{j'}^{(p)}\}_{(H+1)\times(H+1)}$.

- Step 2: With the new $\mathbf{V}$, assign hidden targets denoted by $z_j^{(p)}$'s using

$$z_j^{(p)} = h_j^{(p)} + \zeta_p\beta_j^{(p)}, \quad \beta_j^{(p)} \equiv -\frac{\partial E^{out}}{\partial h_j^{(p)}} = \sum_{k=1}^{M}\hat{d_k}^{(p)} v_{kj}, \quad \zeta_p = \frac{\sum_{j=1}^{H}{\beta_j^{(p)}}^2}{\sum_{k=1}^{M}(\sum_{j=1}^{H} v_{kj}\beta_j^{(p)})^2} = \frac{\vec{\beta^{(p)}}^T\vec{\beta^{(p)}}}{\vec{\beta^{(p)}}^T\mathbf{C}_v\vec{\beta^{(p)}}}, \quad (4)$$

where $\vec{\beta^{(p)}} = [\beta_1^{(p)}, \ldots, \beta_H^{(p)}]^T$ and $\mathbf{C}_v = \{\sum_{k=1}^{M} v_{kj}v_{kj'}\}_{H\times H}$. And assign $\hat{z_j}^{(p)} = f^{-1}(z_j^{(p)})$ after truncating $z_j^{(p)}$ to satisfy $-1 < z_j^{(p)} < 1$.

- Step 3: With the $\hat{z_j}^{(p)}$'s and given training patterns $x_i^{(p)}$'s, optimize $\mathbf{W}$ for minimizing E^{hid} given as

$$E^{hid} = \frac{1}{2}\sum_{p=1}^{P}\sum_{j=1}^{H}(\hat{z_j}^{(p)} - \hat{h_j}^{(p)})^2 = \frac{1}{2}\sum_{p=1}^{P}\sum_{j=1}^{H}(\hat{z_j}^{(p)} - \sum_{i=0}^{N} w_{ji}x_i^{(p)})^2. \quad (5)$$

That is, hidden weights are updated according to

$$\Delta w_{ji} = \eta_j^{hid}\gamma_{ji}, \quad \gamma_{ji} \equiv -\frac{\partial E^{hid}}{\partial w_{ji}} = \sum_{p=1}^{P}\hat{e_j}^{(p)}x_i^{(p)}, \quad \eta_j^{hid} = \frac{\sum_{i=0}^{N}{\gamma_{ji}}^2}{\sum_{p=1}^{P}(\sum_{i=0}^{N}\gamma_{ji}x_i^{(p)})^2} = \frac{\vec{\gamma_j}^T\vec{\gamma_j}}{\vec{\gamma_j}^T\mathbf{C}_x\vec{\gamma_j}}, \quad (6)$$

where $\hat{e_j}^{(p)} = -\partial E^{hid}/\partial \hat{h_j}^{(p)} = \hat{z_j}^{(p)} - \hat{h_j}^{(p)}$, $\vec{\gamma_j} = [\gamma_{j0}, \ldots, \gamma_{jN}]^T$, and $\mathbf{C}_x = \{\sum_{p=1}^{P} x_i^{(p)}x_{i'}^{(p)}\}_{(N+1)\times(N+1)}$.

Due to its linear characteristics at each step, the LBL algorithm shows fast convergence. However, if the targets of a hidden layer become linearly inseparable, it is impossible to reduce the MSE of a hidden layer sufficiently. This results in the stalling problem, i.e., the LBL algorithm can not further reduce the output error[9].

3. New Error Function for Hidden Layer

The quadratic error function of net-input to hidden node given by Eq. (5) is essential to derive the optimum learning rates of hidden weights. The net-input to hidden node, however, is forward propagated to the output layer through a sigmoid transformation, and the final goal of learning is to minimize the error function at the output layer. Therefore, the hidden-layer error function needs to have a term related to the sigmoid derivatives. In this sense, a new error function for a hidden layer is proposed as

$$E_n^{hid} = \frac{1}{2}\sum_{p=1}^{P}\sum_{j=1}^{H}(\hat{z_j}^{(p)} - \hat{h_j}^{(p)})^2[f'(\hat{z_j}^{(p)})]^2, \quad (7)$$

which is the MSE multiplied by a term related to the sigmoidal slope at $z_j^{(p)}$. If $\hat{z_j}^{(p)}$ and $\hat{h_j}^{(p)}$ are in the saturation region, the difference between $z_j^{(p)}$ and $h_j^{(p)}$ becomes much smaller than the difference between $\hat{z_j}^{(p)}$ and $\hat{h_j}^{(p)}$. In this case, it is inadequate to change the associated weights according to the large difference between $\hat{z_j}^{(p)}$ and $\hat{h_j}^{(p)}$, and the slope term reduces the updating amount.

For minimizing E_n^{hid}, the hidden weight w_{ji} is updated by

$$\Delta w_{ji} = -\eta_j^{hid} \frac{\partial E_n^{hid}}{\partial w_{ji}} = \eta_j^{hid} \sum_{p=1}^{P} \hat{e_j}^{(p)} [f'(\hat{z_j}^{(p)})]^2 x_i^{(p)} \tag{8}$$

without any approximation since the slope of hidden target $f'(\hat{z_j}^{(p)})$ does not depend on w_{ji}. By substituting $w_{ji} + \Delta w_{ji}$ into Eq. (7) with the definition of $\gamma_{ji}^{new} = -\partial E_n^{hid} / \partial w_{ji}$, one obtains the new optimum η_j^{hid} as

$$\eta_j^{hid} = \frac{\sum_{i=0}^{N} (\gamma_{ji}^{new})^2}{\sum_{p=1}^{P} (\sum_{i=0}^{N} \gamma_{ji}^{new} x_i^{(p)})^2 [f'(\hat{z_j}^{(p)})]^2} = \frac{\vec{\gamma_j^{new}}^T \vec{\gamma_j^{new}}}{\vec{\gamma_j^{new}}^T \mathbf{C}_x^{new} \vec{\gamma_j^{new}}}, j = 1, 2, \ldots, H, \tag{9}$$

where $\vec{\gamma_j^{new}} = [\gamma_{j0}^{new}, \ldots, \gamma_{jN}^{new}]^T$ and $\mathbf{C}_x^{new} = \{\sum_{p=1}^{P} x_i^{(p)} x_{i'}^{(p)} [f'(\hat{z_j}^{(p)})]^2\}_{(N+1)\times(N+1)}$. Thus, using the new hidden error function, the hidden weights are updated by Eq. (8) with the optimum learning rate given by Eq. (9). This proposed LBL takes the same formulations with the LBL algorithm for updating output weights and assigning hidden targets.

Now let's compare the proposed LBL and the standard EBP algorithms. For small values of $\zeta_p \beta_j^{(p)}$ in Eq. (4), $\hat{z_j}^{(p)}$ can be approximated by a first order Taylor series and

$$\hat{e_j}^{(p)} = \hat{z_j}^{(p)} - \hat{h_j}^{(p)} \approx \zeta_p \beta_j^{(p)} \frac{2}{1 - {h_j^{(p)}}^2}. \tag{10}$$

Also under the same assumption that there is no need of abrupt change on $z_j^{(p)}$, it can be approximated as

$$f'(\hat{z_j}^{(p)}) \approx f'(\hat{h_j}^{(p)}) = \frac{1 - {h_j^{(p)}}^2}{2}. \tag{11}$$

From Eqs. (10) and (11), an approximation may be made as

$$\hat{e_j}^{(p)} [f'(\hat{z_j}^{(p)})]^2 \approx \zeta_p \delta_j^{hid}(\mathbf{x}^{(p)}), \tag{12}$$

where

$$\delta_j^{hid}(\mathbf{x}^{(p)}) = -\frac{\partial E^{out}}{\partial \hat{h_j}^{(p)}} = f'(\hat{h_j}^{(p)}) \beta_j^{(p)}. \tag{13}$$

Thus, Δw_{ji} using E_n^{hid} given by Eq. (8) is approximated as

$$\Delta w_{ji} \approx \eta_j^{hid} \sum_{p=1}^{P} \zeta_p \delta_j^{hid}(\mathbf{x}^{(p)}) x_i^{(p)}. \tag{14}$$

The learning rate η_j^{hid} given by Eq. (9) also can be approximated using Eqs. (11) and (12) without hidden targets. The approximated Δw_{ji} using the new E_n^{hid} is similar to the one in the EBP algorithm given as

$$\Delta w_{ji} = \frac{\eta}{P} \sum_{p=1}^{P} \delta_j^{hid}(\mathbf{x}^{(p)}) x_i^{(p)}. \tag{15}$$

The only difference is that the learning rate η/P in the EBP algorithm is decomposed into the optimum learning rates η_j^{hid} and ζ_p in the proposed LBL algorithm. It is worth noting that the optimum learning rates are defined as functions of the hidden-node index j and pattern index p. After some estimation, it can be easily shown that the proposed LBL algorithm takes about $(4N + 6M)HP$ multiplications per learning epoch, which is approximately two times that in the standard EBP algorithm. Thus, the proposed LBL algorithm can resolve the stalling problem without heavily increasing computational costs.

In the conventional LBL algorithm, Δw_{ji} in Eq. (6) may be approximated as

$$\Delta w_{ji} \approx \eta_j^{hid} \sum_{p=1}^{P} \frac{\zeta_p}{f'(\hat{h}_j^{(p)})^2} \delta_j^{hid}(\mathbf{x}^{(p)}) x_i^{(p)}. \tag{16}$$

The difference between Eqs. (14) and (16) is the slope term $f'(\hat{h}_j^{(p)})^2$. In the initial stage of learning, hidden nodes are in the linear region of the sigmoid function and the slope term does not make much difference. After some learning epoches, some hidden nodes approach the saturation region and the slope term makes the updating amount of hidden weights very large. This makes the hidden nodes more saturated toward the direction of hidden targets. If the targets of the hidden nodes are not linearly separable, the hidden nodes are then heavily saturated without a sufficient reduction of the MSE, and additional training can not reduce the MSE of the output layer.

4. Simulation

A handwritten digit recognition problem was used to verify the effectiveness of the new hidden error function. 5,000 handwritten digitized images from the CEDAR database were used for training after size normalization[11]. A digit consisted of 12×12 pixels and each pixel took an integer value from zero to 15. The MLP consisted of 144 inputs, 30 hidden nodes, and ten output nodes.

We simulated the EBP algorithm with the bold driver(BD) technique[3], one also with the delta-bar-delta(DBD) rule, a merged method of LBL and EBP algorithms[9], a conventional LBL algorithm, the proposed LBL algorithm using the new hidden error function, and finally the proposed one with linear approximation given by Eq. (14). The BD technique of the EBP algorithm took the initial learning rate of 0.05, which was also used for the EBP algorithm of the merged method. The DBD parameters were $\theta = 0.9, \phi = 0.01$, and $\kappa = 1 \times 10^{-7}$. In all training algorithms using the LBL for updating output weights, the learning rate η_k^{out} was restricted to be less than one at the first and the second epoches. This was to prevent the output weights from becoming very large in the initial stage of learning. The approximation of Eq. (11) was used for the proposed LBL. Especially in the proposed algorithm with the approximation

Eq. (14), hidden weights were updated in the case that $\sqrt{E[(\zeta_p \delta_j^{hid})^2]} < 10$ to decrease the approximation errors. The expectation condition excluded only the first epoch from updating the hidden weights in the simulations. Four simulations were conducted using each method with initial weights drawn at random from a uniform distribution on $[-10^{-4}, 10^{-4}]$, and the results were averaged in Figure 1, in which data points began at one learning epoch because of the log scale in x-axis.

Fig.1(a) shows the MSE for the training patterns in each training method. The EBP algorithms with the BD technique and the DBD rule show a slow convergence since their learning rates are not optimum. Although the conventional LBL algorithm decreases MSE drastically in the initial stage of learning, it reveals the stalling problem. While the merged method decreases MSE below the stalling level, it is still slow due to the non-optimal learning rate of the EBP portion.

Contrary to the four methods, the proposed LBL algorithms display very fast decrease of MSE without the stalling problem. Although the linear approximation slows down the convergence, at the early stage both proposed algorithms converge much faster than the others. The improved convergence of the proposed LBL algorithm, whether it is approximated or not, supports the previous argument that the new error function of the hidden layer makes the LBL algorithm approximate the EBP algorithm with optimal learning rates and resolves the stalling problem.

In Fig.1(b), misclassification rates for 2,213 test patterns are shown. Consistently with the MSEs for the training patterns, the misclassification curves of the EBP-based algorithms decrease slowly and steadily. On the contrary, the proposed LBL algorithms, with and without the linear approximation, hit the minimum misclassification rates at about 400 and 200 epoches, respectively. Although there are overfittings after the minimum hit of misclassification rates, they are common problems in training of neural networks. Fig.1 shows that the proposed algorithms greatly speed up the learning convergence without sacrificing the generalization capability.

For more verification of the proposed algorithms, an isolated-word recognition problem was used in which the vocabulary consisted of 30 words and each word was spoken two times by 9 speakers. The 540 patterns were used for training the MLP with 50 hidden nodes after extracting the ZCPA feature of 1,024 dimensions[12]. Fig. 2 shows the MSE for the training patterns, which is consistent with Fig. 1.

If all hidden nodes and their targets are saturated before successful training, the proposed error function does not work due to the slope term. Hidden nodes, however, are not saturated in the initial stage of learning, since MLPs are generally initialized with small random weights to prevent their premature saturation[13]. Also, hidden nodes are trained not to be saturated but to extract near-orthogonal features from input patterns[14]. Therefore, there will be no real problems in which the proposed error function flaws.

5. Conclusion

To accelerate the training speed of MLPs, this letter proposed a new error function at the hidden layer. The new hidden error function made the LBL algorithm update hidden weights proportional to the derivative of the output error function with respect to the associated hidden weights. Especially, it was proven that the LBL using the new hidden error approximated the standard EBP algorithm with optimal learning rates. The proposed LBL algorithm removed the stalling problem in the conventional LBL algorithms, and thereby greatly speeded up the learning convergence.

Acknowledgment : This research was supported by the Korean Ministry of Science and Technology.

References

[1] D. E. Rumelhart and J. L. McClelland, *Parallel Distributed Processing.* MIT Press, Cambridge, MA, 1986.

[2] R. A. Jacobs, "Increased rates of convergence through learning rate adaptation," *Neural Networks*, vol. 1, pp. 295–307, 1988.

[3] 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.

[4] S.-H. Oh, "Improving the error backpropagation algorithm with a modified error function," *IEEE Trans. Neural Networks*, vol. 8, no. 3, pp. 799–803, 1997.

[5] A. van Ooyen and B. Nienhuis, "Improving the convergence of the back-propagation algorithm," *Neural Networks*, vol. 5, pp. 465–471, 1992.

[6] R. Parisi, 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.

[7] G.-J. Wang and C.-C. Chen, "A fast multilayer neural networks training algorithm based on the layer-by-layer optimizing procedures," *IEEE Trans. Neural Networks*, vol. 7, pp. 768–775, 1996.

[8] F. Biegler-Konig and F. Marmann, "A learning algorithm for multilayered neural networks based on linear least squares problems," *Neural Networks*, vol. 6, pp. 127–131, 1993.

[9] J. Y. F. Yam and W. S. Chow, "Extended least squares based algorithm for training feedforward networks," *IEEE Trans. Neural Networks*, vol. 8, pp. 806–810, 1997.

[10] S. Ergezinger and E. Thomsen, "An accelerated learning algorithm for multilayer pereceptrons: Optimization layer by layer," *IEEE Trans. Neural Networks*, vol. 6, pp. 31–42, 1995.

[11] J. J. Hull, "A database for handwritten text recognition research," *IEEE Trans. Pat. Ana. Mach. Int.*, vol. 16, no. 5, pp. 550–554, May 1994.

[12] 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 Audio Processing*, vol. 7, pp. 55–69, 1999.

[13] 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.

[14] S.-H. Oh and Y. Lee, "Effect of nonlinear functions on correlation between weighted sums in multilayer perceptrons," *IEEE Trans. Neural Networks*, vol. 5, pp. 508–510, 1994.

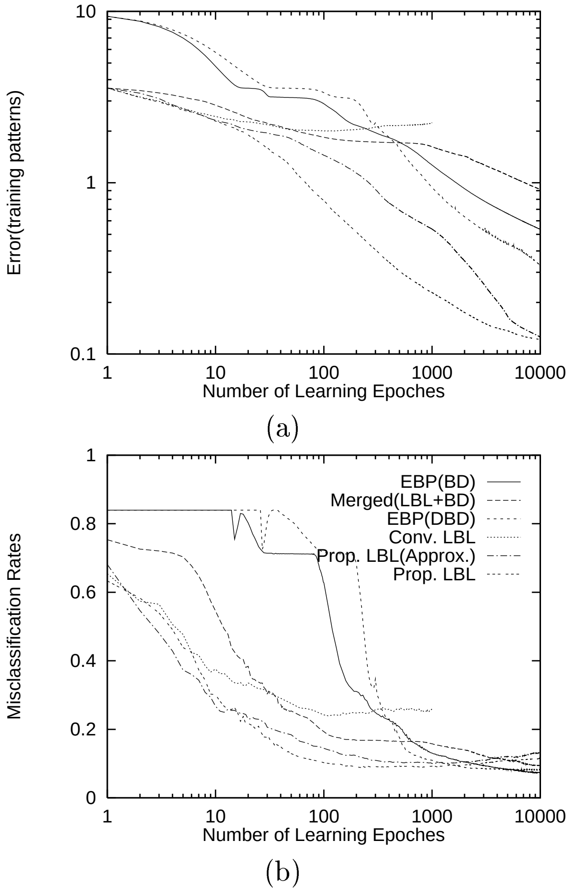

Figure 1: The simulation results in a handwritten digit recognition task. (a) A mean-squared error for training patterns. (b) Misclassification rates for test patterns.

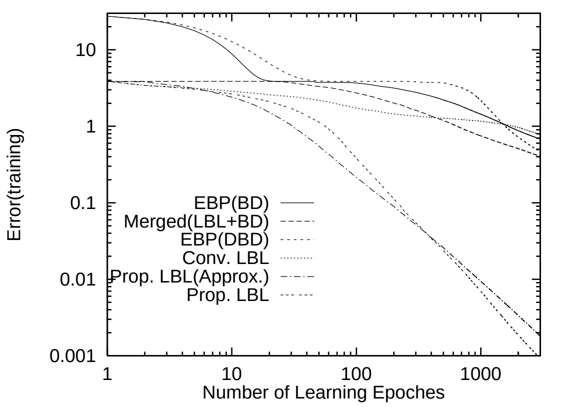

Figure 2: A mean-squared error in the simulation of an isolated-word recognition task.