http://dx.doi.org/10.5392/IJoC.2014.10.1.023

# Comparison of Objective Functions for Feed-forward Neural Network Classifiers Using Receiver Operating Characteristics Graph

**Sang-Hoon Oh**
Department of Information Communication Engineering
Mokwon University, Daejon, 302-729, Korea

**Hiroshi Wakuya**
Graduate School of Science and Engineering
Saga University, Saga, 840-8502, Japan

***ABSTRACT***

*When developing a classifier using various objective functions, it is important to compare the performances of the classifiers. Although there are statistical analyses of objective functions for classifiers, simulation results can provide us with direct comparison results and in this case, a comparison criterion is considerably critical. A Receiver Operating Characteristics (ROC) graph is a simulation technique for comparing classifiers and selecting a better one based on a performance. In this paper, we adopt the ROC graph to compare classifiers trained by mean-squared error, cross-entropy error, classification figure of merit, and the n-th order extension of cross-entropy error functions. After the training of feed-forward neural networks using the CEDAR database, the ROC graphs are plotted to help us identify which objective function is better.*

***Key words****: Receiver Operating Characteristic Graph, Feed-Forward Neural Networks, Objective Function, Performance Comparison.*

## 1. INTRODUCTION

Based on the theoretical results that feed-forward neural networks (FNNs) can be universal approximators of any function with enough number of hidden nodes [1]-[4], FNNs are widely applied to many fields such as pattern recognition, time series prediction, nonlinear control, telecommunications, credit assessment, gene ontology, remote sensing and bio-medical diagnoses. For real applications of FNNs, how to train FNNs is still a challenging problem and various objective functions have been proposed to improve the performance of FNNs[5]-[8]. Since these objective functions have peculiar properties, they can be compared by diverse ways [9]-[13].

There have been many researches of statistically analyzing objective functions [10]-[13]. Firstly, it was proved that FNNs are Bayesian optimal classifiers if FNNs are trained with infinite number of training samples [10], [11]. Based on the result, optimal outputs of FNNs were derived as a function of a posteriori probability that an input sample belongs to a certain class [7]. There was an analytical comparison of the optimal outputs for various objective functions [12]. And the relationship between two optimal outputs in each objective function was analyzed [13]. These results show the reason why some objective functions have problems of over-fitting and slow convergence of learning. Also, the contours of objective functions were derived in a two-dimensional space[9], which can give us more informative comparison results for training of FNNs.

Contrary to the analytical comparisons, we can directly compare performances of FNNs which are trained with objective functions for some application problems. In the simulation comparisons, performance criteria are very important [14]. Usually we use error, accuracy (recognition ratio), convergence speed, and computational cost as performance criteria. Among them, we usually focus on classification accuracy because it is the final goal of classifier. However, simple classification accuracy is often a poor metric for measuring performance of classifiers in many cases. Therefore, recent years have seen an increase in the use of ROC(receiver operating characteristics) graphs in machine learning community [15]. The ROC graphs are insensitive to class distribution and this property is very important in the case that class distribution is not equal. Also, the ROC graphs can be used to depict the tradeoff between detection probability and false alarm rate.

In this paper, we adopt the ROC graphs to compare various objective functions which are proposed to improve the classification performance of FNNs. The comparison results

* *Corresponding author, Email: shoh@mokwon.ac.kr*
*Manuscript received Oct. 28, 2013; revised Feb. 07, 2014; accepted Feb. 17, 2014*

can firmly provide which classifier is better. In section 2, we briefly introduce the ROC graphs and their important properties. And, in section 3, some objective functions such as the conventional mean-squared error(MSE) [5], cross-entropy(CE) error [6], *n*-th order extension of CE(*n*CE) [7], and classification figure of merit(CFM) [8] are introduced with their peculiar characteristics for improvement of classifiers' performance. Section 4 simulates the objective functions to train FNNs on hand-written digit recognition task and compare their performances based on the ROC graphs. Finally, section 5 concludes this paper.

## 2. ROC(RECEIVER OPERATING CHARACTERISTICS) GRAPH

A ROC graph has long been used in the signal detection theory to depict the tradeoff between detection probability and false alarm rate of signal detectors [16]. This is adopted as a technique for visualizing, organizing and selecting classifiers based on their performances [15], [17]. Let us begin by considering classification problems using only two classes. When an input sample is presented to a classifier, it is mapped to one element of the set $\{p,n\}$ of positive and negative classes. For simplicity of explanation, we assume that the classifier has continuous outputs and different thresholds may be applied to predict class membership. The labels $\{P,N\}$ denote the predicted classes and $\{p,n\}$ denote the actual classes.

Given a classifier and instances, there are four possible outcomes of prediction or decision. If a positive instance is classified as positive, it is counted as a true positive; if it is classified as negative, it is counted as a false negative. If a negative instance is classified as negative, it is counted as a true negative; if it is classified as a positive, it is counted as a false positive. Thus, a two-by-two confusion matrix can be constructed as shown in Fig. 1 [15]. Then, the detection probability or true positive rate is defined by

$$tp\ rate = \frac{\text{positives correctly classified}}{\text{total positives}}. \quad (1)$$

Also, the false alarm rate or false positive rate is defined by

$$fp\ rate = \frac{\text{negatives incorrectly classified}}{\text{total negatives}}. \quad (2)$$

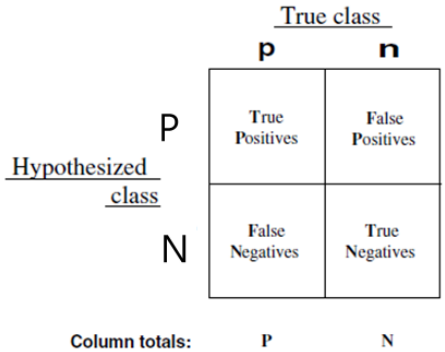



Fig. 1. Confusion matrix of a two-class classifier

ROC graphs are two-dimensional graphs in which *tp rate* is plotted on the vertical axis and *fp rate* is plotted on the horizontal axis. Fig. 2 shows a basic ROC space. There are several important points in the ROC space. The point (0,1) corresponds to a perfect classifier, that is, all positive and negative instances are correctly classified. On the contrary, the point (1,0) represents a never-guessing classifier; all positives are classified as negatives and all negatives are classified as positives. The point (0,0) corresponds to a classifier of never issuing a positive class. While the point (1,1) represents a classifier of unconditionally issuing positive class.

The threshold of a classifier is the difference of outputs for decision. When the threshold is zero, the index (or label) of maximum output node corresponds to a decision of class. If the threshold has a certain value, we decide a class of input instance as the maximum output index when the difference between the maximum and the other outputs is greater than the threshold. By varying the threshold of a classifier, the points (*tp rate, fp rate*) have a shape of graph and the ROC graph depicts relative tradeoffs between benefits (true positives) and costs (false positives) [15]. Let's assume that we have a random classifier which guesses a positive class with probability *y*. Then, it can be expected to get the positives correct with probability *y* but its false positive rate is also *y*. Thus, the diagonal line ***tp rate=fp rate*** in the ROC space represents classifiers of random guessing. Classifiers in the lower right triangle perform worse than random. Contrary, classifiers in the upper left triangle are better ones. Among them, we select the upper left-most graph as the best one.

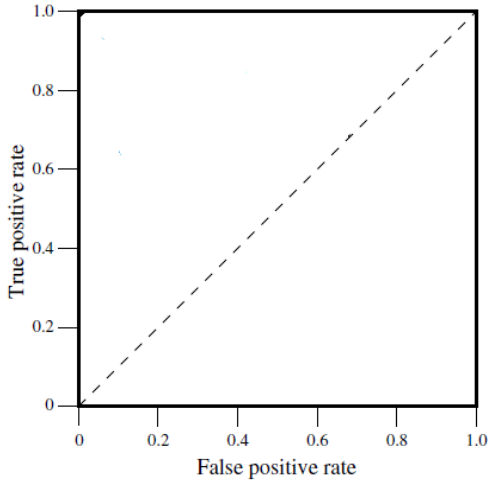



Fig. 2. ROC space whose horizontal axis is the false positive rate and vertical axis is the true positive rate

An attractive property of ROC graphs is that they are insensitive to class distribution. If class priors are equal, the total accuracy or classification ratio can be a performance measure of classifier. However, in many real problems, there is a severe imbalance of class distributions [18]-[21]. In this case, the total accuracy is not adequate as a performance measure since it heavily depends on the accuracy of majority class [21]. Therefore, we need a performance measure which is insensitive to class distribution. ROC graphs are plots of (*tp rate, fp rate*) points, in which each dimension is a strict column ratio. That is,

variation of the column totals in the Fig. 1 is irrelevant of *tp rate* and *fp rate*. As a result, the point (*tp rate, fp rate*) does not depend on the variation of class distribution. Therefore, the ROC graphs are especially useful for domains with skewed class distribution and unequal classification error costs [15].

When we simulate classifiers many times, we have many ROC graphs and they should be averaged. However, the ROC graphs are two-dimensional and averaging is a one-dimensional process. So, we need averaging process which preserves characteristics of interest. There are two averaging methods: vertical and threshold averaging [15]. Vertical averaging takes vertical samples of the ROC graphs for fixed *fp rates* and averages the corresponding *tp rates*. If *fp rate* is not controllable, we prefer to average ROC points using an independent variable which can be controlled directly. Since we can directly control the threshold of a classifier, we find the corresponding point of each ROC graph for each threshold and average them. This is the threshold averaging.

In the case of *n*-class problems more than two, the confusion matrix becomes an $n \times n$ matrix containing $n$ correct classifications on the major diagonal entries and $n^2 - n$ possible misclassifications on the off-diagonal entries. Instead of managing trade-offs between *tp rate* and *fp rate,* we have $n$ benefits of correct classification and $n^2 - n$ errors of misclassification. Thus, in the multi-class case of ROC graphs, we plot ROC points with the strategy that the *tp rate* corresponds to the classification rate and the *fp rate* corresponds to the misclassification rate.

## 3. OBJECTIVE FUNCTIONS FOR CLASSIFICATION

For simplicity of explanation, we consider a FNN consisting of $N$ inputs, $H$ hidden nodes, and $M$ output nodes. When a sample $\mathbf{x}^{(p)} = [x_1^{(p)}, x_2^{(p)}, \ldots, x_N^{(p)}]$ $(p = 1,2,\ldots,P)$ is presented to the FNN, the $j$-th hidden node is given by

$$h_j^{(p)} = h_j(\mathbf{x}^{(p)}) = \tanh((w_{j0} + \sum_{i=1}^{N} w_{ji} x_i^{(p)})/2), \ j = 1,2,\ldots,H. \quad (3)$$

Here, tanh(.) is the activation function, $w_{ji}$ denotes the weight connecting $x_i$ to $h_j$, and $w_{j0}$ is a bias. The $k$-th output node is

$$y_k^{(p)} = y_k(\mathbf{x}^{(p)}) = \tanh(\hat{y}_k^{(p)}/2), \ k = 1,2,\ldots,M, \quad (4)$$

where

$$\hat{y}_k^{(p)} = v_{k0} + \sum_{j=1}^{H} v_{kj} h_j^{(p)} . \quad (5)$$

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

Let the desired output vector corresponding to the training sample $\mathbf{x}^{(p)}$ be $\mathbf{t}^{(p)} = [t_1^{(p)}, t_2^{(p)}, \ldots, t_M^{(p)}]$, which is coded as follows:

$$t_k^{(p)} = \begin{cases} +1 & \text{if } \mathbf{x}^{(p)} \text{ originates from class } k \\ -1 & \text{otherwise .} \end{cases} \quad (6)$$

As a distance measure between the actual and desired outputs, we usually use the MSE function for $P$ training samples defined by

$$E_{MSE} = \sum_{p=1}^{P} \sum_{k=1}^{M} \left(t_k^{(p)} - y_k^{(p)}\right)^2 . \quad (7)$$

To minimize $E_{MSE}$, weights $v_{kj}$'s are iteratively updated by

$$\Delta v_{kj} = -\eta \frac{\partial E_{MSE}}{\partial v_{kj}} = \eta \delta_k^{(p)} h_j^{(p)}, \quad (8)$$

where

$$\delta_k^{(p)} = -\frac{\partial E_{MSE}}{\partial \hat{y}_k^{(p)}} = \left(t_k^{(p)} - y_k^{(p)}\right) \frac{\left(1 - y_k^{(p)}\right)\left(1 + y_k^{(p)}\right)}{2} \quad (9)$$

is the error signal and $\eta$ is the learning rate. The error signal consists of the difference between desired and actual outputs and the gradient of activation function. Also, by the backward propagation of the error signal, weights $w_{ji}$'s are updated by

$$\Delta w_{ji} = -\eta \frac{\partial E_{MSE}}{\partial w_{ji}} = \eta x_i^{(p)} \sum_{k=1}^{M} v_{kj} \delta_k^{(p)} . \quad (10)$$

The above weight-updating procedure is the EBP(error back-propagation) algorithm [5].

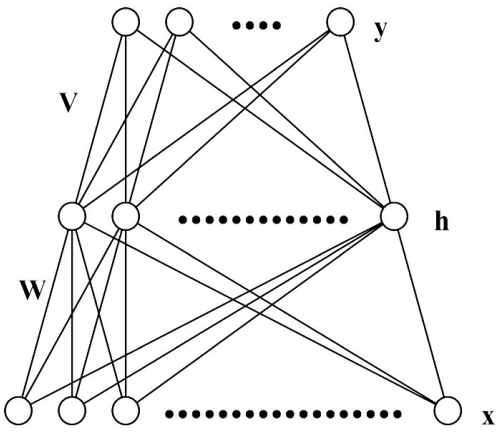



Fig. 3. The architecture of a feed-forward neural network

The EBP algorithm using MSE has a drawback with slow convergence. Since the activation function has two extreme saturation regions, we say that an output node is incorrectly saturated if the output node is in the opposite extreme region of its desired value. When an output node is incorrectly saturated, the amount of weight change is small due to the small gradient of activation function in the error signal given by Eq. (9). Therefore, the error remains nearly unchanged[7][22]. In order to resolve this problem, the CE error function [6] was proposed by

$$E_{CE} = -\sum_{k=1}^{M} \left[\left(1 + t_k^{(p)}\right) \ln\left(1 + y_k^{(p)}\right) + \left(1 - t_k^{(p)}\right) \ln\left(1 - y_k^{(p)}\right)\right]. \quad (11)$$

Then, the error signal associated with the output layer is given by

$$\delta_k^{(p)} = -\frac{\partial E_{CE}}{\partial \hat{y}_k^{(p)}} = \left(t_k^{(p)} - y_k^{(p)}\right). \quad (12)$$

The above error signal is proportional only to the difference between desired and actual output values and this can prevent the incorrect saturation of output nodes. As a result, the learning convergence is accelerated. However, the CE error function method suffers from overspecialization for training samples since the error signal for a correctly saturated output node is too strong [7].

During the learning process, a strong error signal is necessary for the incorrectly saturated output nodes like the CE error function [6]. For correctly saturated output nodes, the weak error signal is necessary to prevent overspecialization of learning to training samples. In this sense, the *n*CE error function [7] was proposed by

$$E_{nCE} = -\sum_{k=1}^{M} \int \frac{t_k^{(p)^{n+1}} \left(t_k^{(p)} - y_k^{(p)}\right)^n}{2^{n-2}\left(1 - y_k^{(p)^2}\right)} dy_k^{(p)} . \quad (13)$$

Then , the error signal is given by

$$\delta_k^{(p)} = -\frac{\partial E_{nCE}}{\partial \hat{y}_k^{(p)}} = \frac{t_k^{(p)^{n+1}} \left(t_k^{(p)} - y_k^{(p)}\right)^n}{2^{n-1}} . \quad (14)$$

Contrary to MSE, CE, and nCE error functions, CFM has three essential features that distinguish it from the other error functions [8]. Firstly, it has not the desired value but the index of output node representing the correct classification outcome (so called "target node"). Secondly, CFM yields decreasing marginal "rewards" for increasingly ideal output sample. Thirdly, CFM yields decreasing marginal "penalties" for increasingly bad misclassifications. This is to discourage the FNN from attempting to outliers heavily.

The resulting CFM objective function compares the output node value that should be at high state with values of all output nodes that should be at low state. Using CFM, learning focuses most heavily on the reduction of misclassifications rather than reducing the difference between the output node value and its target output. Thus, CFM is defined by

$$O_{CFM} = \sum_{k \neq c} \frac{1}{1 + \exp\left(-\beta\left(y_c^{(p)} - y_k^{(p)}\right)\right)}, \quad (15)$$

where $y_c^{(p)}$ denotes the correct node and $y_k^{(p)}(k \neq c)$ denotes the incorrect node[8]. And training of FNN is done to increase $O_{CFM}$ .

## 4. SIMULATIONS

Among many objective functions for training of FNNs, we selected the above four objective functions for ROC comparison. Firstly, MSE is the most popular error function and originally EBP algorithm of FNNs was developed based on MSE [5]. Secondly, we select CE error function since it resolved the convergence speed problem of MSE [6]. Furthermore, *n*CE resolved the overspecialization to training samples which is a main weak point of CE error and MSE [7]. Lastly, CFM is selected for ROC comparison because it was developed not to decrease the distance between actual and desired output values but to increase classification ratio [8].

A handwritten digit recognition task is used to compare the ROC graphs of objective functions. A total of 18468 handwritten digitized images from the CEDAR database [23] are used for training after size normalization. A digit image consists of $12 \times 12$ pixels and each pixel takes on integer values from zero to 15. The FNN consists of 144 inputs, 30 hidden nodes, and ten output nodes.

Since no fair comparison is possible if the learning rate is kept the same for all training methods [6], we derive the learning rates so that $E\left\{\eta \delta_k^{(p)}\right\}$ of the target node has the same value in each method. Here, we assume that $y_k^{(p)}$ has uniform distribution on $[-1, +1]$. As a result, the learning rates of 0.006, 0.002, 0.015, 0.003, and 0.005 are used for the conventional MSE, CE, CFM, and *n*CE with *n*=2 and 4, respectively. Nine simulations are conducted using each method with same initializations. The initial weights were drawn at random from a uniform distribution on $\left[-1 \times 10^{-4}, 1 \times 10^{-4}\right]$.

Since *n*CE must have a specific parameter value of *n*, we simulated various values of *n* for training of FNNs with 18468 training patterns. FNNs were trained for 500 sweeps with learning rate $\eta = 0.001 \times (n+1)$ and their performances were evaluated with 2213 validation patterns. As shown in Fig. 4, which shows the misclassification ratio for the validation patterns during learning process, *n*=4 is better than other values in viewpoints of misclassification ratio and learning speed. Also, *n*=2 is the primitive value which has the peculiar characteristics of *n*CE. So, we select *n*CE with n=2 and 4 for ROC comparison.

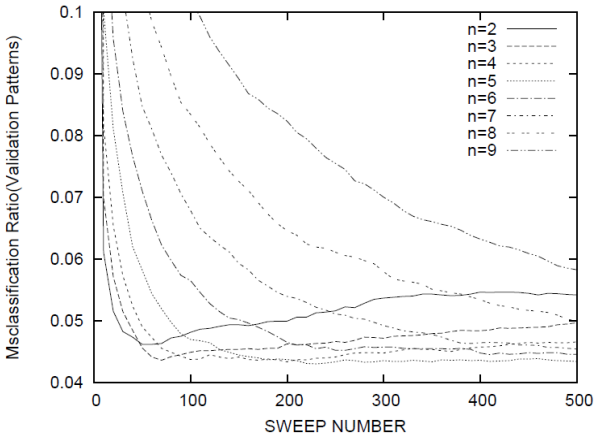
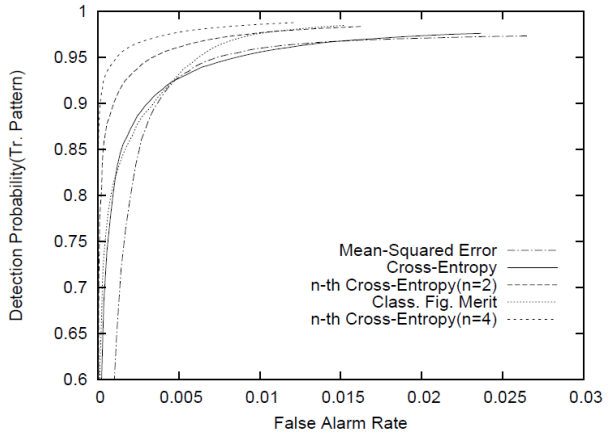


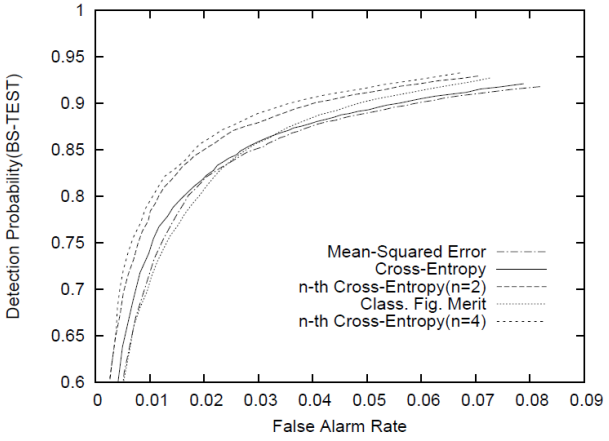

Fig. 4. The misclassification ratio for the validation patterns when FNNs are trained based on nCE

During the learning process in each method, we evaluate the classification rate with 2213 validation images. If there is no improvement of classification ratio for the validation samples with additional learning of 50 epochs, we stop the learning process and evaluate the ROC graph for 18468 training samples and 2711 test samples. We conducted nine times simulation and average the ROC graphs for comparison. There are two methods to average the ROC graphs: vertical

averaging and threshold averaging. Since we cannot directly control *fp rate*, we adopt the threshold averaging [15].




(a)




(b)

Fig. 5. ROC graphs for handwritten digit recognition simulation. (a) for 18468 training patterns, (b) for 2711 test patterns.

Fig. 5(a) is the ROC graph for training samples and Fig. 5(b) is for test samples. As shown in Fig. 5(a), MSE is the worst and CE is better than MSE. CFM improves the performance. *n*CE with *n*=2 is better and *n*CE with *n*=4 shows the best performance. In Fig. 5(b), ROC graphs for MSE, CE, and CFM are similar. *n*CE with *n*=2 is better and nCE with *n*=4 is the best. Thus, we can say that *n*CE is better than MSE, CE, and CFM in the comparison based on ROC graphs. One point of ROC space corresponds to a performance of classifier with a certain threshold value and the right-most points of the graphs correspond to classifiers with zero threshold. (The simulation results in [7] corresponds to classifiers with zero threshold.) With the zero threshold, *n*CE with *n*=2 shows similar performance with CFM in Figs. 5(a) and (b). However, in a viewpoint of ROC, *n*CE with *n*=2 is much better than CFM since *n*CE with *n*=2 is more left-upper side of ROC space than CFM. Thus, ROC graphs can give us more concrete comparison results than simple classification accuracy.

As explained in section 3, CE tried to accelerate the learning convergence of FNNs by reducing incorrect saturation of output nodes. However, CE has a weakness of over-fitting to training patterns. Although CFM was proposed to resolve the over-fitting problem, CFM did not consider the incorrect saturation of output nodes. Since *n*CE adopted the strategy of reducing the incorrect saturation and preventing the over-specialization, *n*CE was better than CE and CFM.

Nowadays, there are heavy increasing of interest in the deep architecture of neural networks [24], which has more powerful than neural networks with single hidden layer. The poor performance of FNNs is due to the specialization to training samples, or in some cases, FNNs cannot fit the true-function described by the training samples. When adopting the deep architecture, we can increase the performance of neural networks for pattern classifications.

## 5. CONCLUSIONS

When comparing FNN classifiers, performance criterion is important but simple classification accuracy is often a poor matric in many real application problems. So, there have been increasing of interest for the ROC graphs, which are insensitive to class distributions and provide tradeoffs between benefits and costs of classifiers.

In this paper, we adopted the ROC graphs as a direct comparison method of FNN classifiers. Firstly, the ROC graphs were introduced with their important properties. Then, we selected representative objective functions to train FNN classifiers. After training of FNNs for handwritten digit recognition task, we stopped the training and estimated the ROC graph in each FNN classifier. Contrary to comparisons of classification accuracies, the ROC graphs firmly informed us which classifier is better than the others.

## REFERENCES

[1] K. Hornik, M. Stincombe, and H. White, “Multilayer feed-forward networks are universal approximators,” Neural Networks, vol. 2, 1989, pp. 359-366.

[2] K. Hornik, “Approxiamtion Capabilitis of Multilayer Feedforward Networks,” Neural Networks, vol. 4, 1991, pp. 251-257.

[3] S. Suzuki, “Constructive Function Approximation by Three-layer Artificial Neural Networks,” Neural Networks, vol. 11, 1998, pp. 1049-1058.

[4] Y. Liao, S. C. Fang, and H. L. W. Nuttle, “Relaxed Conditions for Radial-Basis Function Networks to be Universal Approximators,” Neural Networks, vol. 16, 2003, pp. 1019-1028.

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

[6] A. van Ooyen and B. Nienhuis, “Improving the Convergence of the Backpropagation Algorithm,” Neural Networks, vol. 5, 1992, pp. 465-471.

[7] S. H. Oh, “Improving the Error Back-Propagation Algorithm with a Modified Error Function,” IEEE Trans. Neural Networks, vol. 8, 1997, pp. 799-803.

[8] J. B. Hampshare and A. H. Waibel, “A Novel Objective Function for Improved Phoneme Recognition Using Time-

Delay Neural Networks," IEEE Trans. Neural Networks, vol. 1, 1990, pp. 216-228.
[9] S. H. Oh, "Contour Plots of Objective Functions for Feed-Forward Neural Networks," Int. Journal of Contents, vol. 8, no. 4, 2012, pp. 30-35.
[10] H. White, "Learning in Artificial Neural Networks: A Statistical Perspective," Neural Computation, vol. 1, no. 4, 1989, pp. 425-464.
[11] M. D. Richard and R. P. Lippmann, "Neural Network Classifier Estimate Bayesian a posteriori probabilities," Neural Computation, vol. 3, 1991, pp. 461-483.
[12] S. H. Oh, "Statistical Analyses of Various Error functions for Pattern Classifiers," Proc. Convergence on Hybrid Information Technology, CCIS vol. 206, 2011, pp. 129-133.
[13] S. H. Oh, "A Statistical Perspective of Neural Networks for Imbalanced Data Problems," Int. Journal of Contents, vol. 7, 2011, pp. 1-5.
[14] I. Kononenko and I. Bratko, "Information-Based Evaluation criterion for Classifier's Performance," Machine Learning, vol. 6, 1991, pp. 67-80.
[15] T. Fawcett, "An Introduction to ROC Analysis," Pattern Recognition Letters, vol. 27, 2006, pp. 861-874.
[16] R. N. McDonough and A. D. Whalen, *Detection of Signals in Noise*, Academic Press, 1995.
[17] Z. H. Michalopoulou, L. W. Nolte, and D. Alexandrou, "Performance Evaluation of Multilayer Perceptrons in Signal Detection and Classification," IEEE Trans. Neural Networks, vol. 6, 1995, pp. 381-386.
[18] R. Bi, Y. Zhou, F. Lu, and W. Wang, "Predicting gene ontology functions based on support vector machines and statistical significance estimation," Neurocomputing, vol. 70, 2007, pp. 718-725.
[19] L. Bruzzone and S. B. Serpico, "Classification of Remote-Sensing Data by Neural Networks," Pattern Recognition Letters, vol. 18, 1997, pp. 1323-1328.
[20] Y. M. Huang, C. M. Hung, and H. C. Jiau, "Evaluation of Neural Networks and Data Mining Methods on a Credit Assessment Task for Class Imbalance Problem," Nonlinear Analysis, vol. 7, 2006, pp. 720-747.
[21] S. H. Oh, "Error back-propagation algorithm for classification of imbalanced data", Neurocomputing, vol. 74, 2011, pp. 1058-1061.
[22] Y. Lee, S. H. Oh, and M. W. Kim, "An Analysis of Premature Saturation in Back-Propagation Learning," Neural Networks, vol. 6, 1993, pp. 719-728.
[23] J. J. Hull, "A Database for Handwritten Text recognition research," IEEE Trans. Pattern Anal. Machine Intell., vol. 16, 1994, pp. 550-554.
[24] Y. Bengio, "Learning Deep Architecture for AI," Foundations and Trends in Machine Learning, vol.2, 2009, pp. 1-127.

**Sang-Hoon Oh**
received his B.S. and M.S degrees in Electronics Engineering from Pusan National University in 1986 and 1988, respectively. He received his Ph.D. degree in Electrical Engineering from Korea Advanced Institute of Science and Technology in 1999. From 1988 to 1989, he worked for LG semiconductor, Ltd., Korea. From 1990 to 1998, he was a senior research staff in Electronics and Telecommunication Research Institute (ETRI), Korea. From 1999 to 2000, he was with Brain Science Research Center, KAIST. In 2000, he was with Brain Science Institute, RIKEN, Japan, as a research scientist. In 2001, he was an R&D manager of Extell Technology Corporation, Korea. Since 2002, he has been with the Department of Information Communication Engineering, Mokwon University, Daejon, Korea, and is now a full professor. Also, he was with the Division of Computational Science and Engineering, Georgia Institute of Technology, Atlanta, USA, as a visiting scholar from August 2008 to August 2009. His research interests are machine learning, speech signal processing, pattern recognition, and bioinformatics.

**Hiroshi Wakuya**
He received the B.E. degree in electronic engineering from Kyushu Institute of Technology, Kitakyushu, Japan, in 1989, and the M.E. and Ph.D. degrees in electrical and communication engineering from Tohoku University, Sendai, Japan, in 1991 and 1994, respectively. In 1994, he joined the staff of Saga University as a Research Associate. In 1995, he became a Lecturer. From 1999 to 2000, he was a Visiting Scientist in University of Louisville, Louisville, Kentucky, USA. Since 2004, he has been an Associate Professor. His research interests include neural networks, intelligent instrumentation, and biological engineering.