Emergent On-Line Learning with a Gaussian Zero-Crossing Discriminant Function

Bao-Liang Lu and Michinori Ichikawa
Lab. for Brain-Operative Device, RIKEN Brain Science Institute
2-1 Hirosawa, Wako-shi, 351-0198, Japan
{lu;ichikawa}@brainway.riken.go.jp

Abstract - This paper presents a modified Gaussian zero-crossing (GZC) discriminant function with a restricted receptive field width for realizing emergent on-line learning. An important advantage of the GZC function over existing linear discriminant functions is its locally tuned response characteristics. By using the GZC discriminant function, both incorrect interpolation and incorrect extrapolation of trained networks can be significantly prevented by adjusting two threshold limits of networks. We demonstrate that the trained networks based on the GZC discriminant function have the proper capability for rejecting unknown inputs.

I. Introduction

On-line learning [8] is a popular method for training neural networks in which network parameters are updated after the presentation of each training example. In comparison with batch learning, on-line learning methods require less storage and computation time and also represent a more natural method for learning non-stationary tasks.

In our previous work [4], [6], we proposed an alternative on-line learning paradigm called *emergent on-line* learning. The basic idea behind emergent on-line learning is twofold. First, at each time step, an on-line learning task is decomposed into a reasonable number of linearly separable subproblems. These subproblems serve to discriminate the currently presented training example from previously learned training examples. Second, rather than directly solving the original on-line learning task, solutions to an on-line learning task are obtained by combining the solutions of linearly separable subproblems according to two emergent laws. Figure 1 shows a block diagram of the emergent on-line learning paradigm in a supervised fashion. The emergent on-line learning paradigm closely follows the divide-and-conquer strategy. In the emergent on-line learning paradigm, two simple emergent laws are used to guide both the problem decomposition and module combination processes.

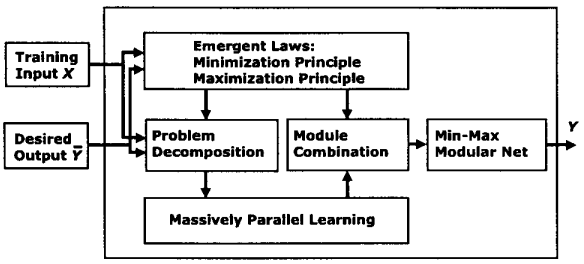


Fig. 1. Block diagram of the emergent on-line learning paradigm.

We show that the emergent on-line learning paradigm has the following three main advantages over existing on-line learning methods [4]. (1) Learning convergence can be guaranteed in polynomial time because the emergent on-line learning is performed by simply combining the solutions of a reasonable number of linearly separable subproblems instead of using gradient-based methods on a differentiable error measure. (2) During learning, min-max modular (M^3) networks [1], [2] used in emergent on-line learning grow gradually according to two emergent laws; therefore, the user is not required to design the networks before learning. (3) The M^3 networks produced by the emergent on-line learning paradigm are quick to response and facilitate hardware implementation because of their hierarchical, parallel, and modular structure.

To improve the generalization performance of trained networks, we have proposed a Gaussian zero-crossing (GZC) discriminant function [6]. The GZC function is designed for solving linearly separable problems, each of which contains only two different data. The GZC function has two different receptive field centers and the same

0-7803-7278-6/02/$10.00 ©2002 IEEE

receptive field width. Both the receptive field centers and the receptive field width are directly determined by two given training data belonging to two classes. The important advantage of the GZC function over conventional linear discriminant functions, such as Perceptrons, is its locally tuned response characteristics.

In this paper, we present a modified GZC discriminant function with a restricted receptive field width for realizing emergent on-line learning. In Section 2, the emergent on-line learning is described. In Section 3, the weakness of linear discriminant functions is analyzed, and a modified GZC discriminant function is presented. In Section 4, a simple example is presented to demonstrate the generalization performance of the networks using the modified GZC function. Conclusions are outlined in Section 5.

II. Emergent On-line Learning

In on-line learning, training examples from the environment are continually presented to the network at distinct time steps. At time step t, on-line learning might be regarded as an event of adding a new training example $(\boldsymbol{x}, \boldsymbol{d})$ to the current network **Net**, where $\boldsymbol{x}$ and $\boldsymbol{d}$ are the training input and desired output, respectively. The key to emergent on-line learning is to use two emergent laws to decompose an on-line learning problem into a reasonable number of linearly separable subproblems and to integrate the solutions of these linearly separable subproblems into solutions to the original on-line learning problem.

A. Emergent laws

The two emergent laws [2], [3] used in emergent on-line learning, namely the *minimization* principle and the *maximization* principle, are described as follows.

Minimization principle

Suppose a two-class problem $\mathcal{B}$ is divided into P relatively smaller two-class subproblems, $\mathcal{B}_i$ for $i = 1, \cdots, P$, and also suppose that all the subproblems have the same positive training data and different negative training data[1]. If the P subproblems are correctly learned by the corresponding P individual modules, M_i for $i = 1, \cdots, P$, then the combination of the P trained modules with a **MIN** unit produces the correct output for all the training inputs in $\mathcal{B}$, where the function of the **MIN** unit is to find a minimum value from its multiple inputs.

Maximization principle

Suppose a two-class problem $\mathcal{B}$ is divided into P relatively smaller two-class subproblems, $\mathcal{B}_i$ for $i = 1, \cdots, P$, and also suppose that all the subproblems have the same negative training data and different positive training data. If the P subproblems are correctly learned by the corresponding P individual modules, M_i for $i = 1, \cdots, P$, then the combination of the P trained modules with a **MAX** unit produces the correct output for all the training input in $\mathcal{B}$, where the function of the **MAX** unit is to find a maximum value from its multiple inputs.

Note that the **MIN** and **MAX** units are completely equivalent to logical **AND** and **OR** gates, respectively, when the values of the inputs to the units are binary.

[1] If the desired output of the training data is $1 - \epsilon$, then the training data is called *positive* training data. Otherwise, it is called *negative* training data.

B. Problem decomposition

Suppose that M training data belonging to K classes have been successfully learned by the current network **Net**, and also suppose that the currently presented training example is $(\boldsymbol{x}, \boldsymbol{d})$. The problem of adding $(\boldsymbol{x}, \boldsymbol{d})$ to **Net** can be decomposed into a reasonable number of linearly separable subproblems as follows [4].

i) If $\boldsymbol{x}$ belongs to a new class, the problem of adding $\boldsymbol{x}$ to **Net** can be divided into the following $\sum_{i=1}^{K} L_i$ linearly separable subproblems:

a) If $K > 1$, then the linearly separable subproblems are given by

$$\mathcal{T}_{i,K+1}^{(u,1)} = \left\{ (\boldsymbol{x}^{(iu)}, 1-\epsilon) \cup (\boldsymbol{x}^{(K+1,1)}, \epsilon) \right\} \qquad (1)$$

where $i = 1, \cdots, K$, $u = 1, \cdots, L_i$, L_i is the number of training data belonging to class C_i, and $\boldsymbol{x}^{(K+1,1)} \equiv \boldsymbol{x}$. Note that the new class C_{K+1} contains only one training data $\boldsymbol{x}$.

b) If $K = 1$, then the linearly separable subproblems are given by

$$\mathcal{T}_{K+1,1}^{(1,v)} = \left\{ (\boldsymbol{x}^{(K+1,1)}, 1-\epsilon) \cup (\boldsymbol{x}^{(1,v)}, \epsilon) \right\} \qquad (2)$$

where $v = 1, \cdots, L_1$,

ii) If $\boldsymbol{x}$ belongs to class C_s ($1 \le s \le K$ and $K > 1$), one of the classes that have been already learned, the task of adding $\boldsymbol{x}$ to **Net** can be divided into the following $\sum_{i=1}^{s-1} L_i + \sum_{j=s+1}^{K} L_i$ linearly separable subproblems:

$$\mathcal{T}_{is}^{(u,L_s+1)} = \left\{ (\boldsymbol{x}^{(iu)}, 1-\epsilon) \cup (\boldsymbol{x}^{(s,L_s+1)}, \epsilon) \right\} \qquad (3)$$

and

$$\mathcal{T}_{sj}^{(L_s+1,v)} = \left\{ (\boldsymbol{x}^{(s,L_s+1)}, 1-\epsilon) \cup (\boldsymbol{x}^{(jv)}, \epsilon) \right\} \qquad (4)$$

where $i = 1, \cdots, s-1$, $u = 1, \cdots, L_i$, $j = s+1, \cdots, K$, and $v = 1, \cdots, L_j$.

C. Parallel learning

A very attractive feature of the linearly separable subproblems defined by (1), (2), (3), and (4) is that each of them can be treated as a completely independent, noncommunicating problem in the learning phase. Therefore, all of the linearly separable subproblems can be learned in parallel.

D. Module combination

Suppose all of the linearly separable subproblems have been solved by associated network modules, these network modules can be easily added to the current network according to the two emergent laws as follows [4].

i) For $\sum_{i=1}^{K} L_i$ network modules trained for solving the linearly separable problems defined by (1) or (2), the following combination operations are performed.

a) If $K > 1$, then the L_i trained network modules are combined as follows:

$$\mathbf{M}_{i,K+1} = \mathrm{Max}\left(\mathbf{M}_{i,K+1}^{(1,1)}, \cdots, \mathbf{M}_{i,K+1}^{(L_i,1)}\right) \quad \text{for } i = 1, \cdots, K \tag{5}$$

where $\mathbf{M}_{i,K+1}^{(u,1)}$ denotes both the name of the network modules corresponding to the subproblem $T_{i,K+1}^{(u,1)}$ and its actual output.

The K new modules $\mathbf{M}_{i,K+1}$ produced by (5) are merged into a modular network with $\binom{K}{2}$ existing modules and their $\binom{K}{2}$ inversions as follows:

$$y_i = \mathrm{Min}\left(\mathbf{M}_{i,1}, \mathbf{M}_{i,2}, \cdots, \mathbf{M}_{i,j}, \cdots, \mathbf{M}_{i,K+1}\right) \tag{6}$$

where $i, j = 1, \cdots, K+1$, $i \neq j$, $\mathbf{M}_{ji} = \mathbf{INV}(\mathbf{M}_{ij})$ for $i < j$, and the function of the INV unit is to invert its single input.

b) If $K = 1$, then the L_1 modules are combined as follows:

$$\mathbf{M}_{K+1,1} = \mathrm{Min}(\mathbf{M}_{K+1,1}^{1,1}, \mathbf{M}_{K+1,1}^{1,2}, \cdots, \mathbf{M}_{K+1,1}^{1,L_1}) \tag{7}$$

ii) For $\sum_{i=1}^{s-1} L_i + \sum_{j=s+1}^{K} L_j$ new modules corresponding to (3) and (4), the following combination operations are performed.

a) The $\sum_{i=1}^{s-1} L_i$ new modules and existing modules are combined as follows:

$$\mathbf{M}_{is}^{(u)} = \mathrm{Min}\left(\mathbf{M}_{is}^{(u,1)}, \cdots, \mathbf{M}_{is}^{(u,L_s)}, \mathbf{M}_{is}^{(u,L_s+1)}\right) \tag{8}$$

and

$$\mathbf{M}_{is} = \mathrm{Max}\left(\mathbf{M}_{is}^{(1)}, \cdots, \mathbf{M}_{is}^{(L_i)}\right) \tag{9}$$

where $i = 1, \cdots, s-1$ and $u = 1, \cdots, L_i$.

TABLE I
Number of elements of the M^3 network for a two-class problem

Modules	$\sum_{i=1}^{K}\sum_{j=i+1}^{K} L_i \times L_j$
MIN	$\sum_{i=1}^{K}\sum_{j=i+1}^{K} L_i\lceil\frac{L_j-1}{L_j}\rceil$
MAX	$\sum_{i=1}^{K}(K-i)\lceil\frac{L_i-1}{L_i}\rceil$
INV	0

TABLE II
Number of elements of the M^3 network for a K-class problem ($K > 2$)

Modules	$2\sum_{i=1}^{K}\sum_{j=i+1}^{K} L_i \times L_j$
MIN	$K + 2\sum_{i=1}^{K}\sum_{j=i+1}^{K} L_i\lceil\frac{L_j-1}{L_j}\rceil$
MAX	$2\sum_{i=1}^{K}(K-i)\lceil\frac{L_i-1}{L_i}\rceil$
INV	$\binom{K}{2}$

b) The $\sum_{j=s+1}^{K} L_j$ new modules and existing modules are combined as follows:

$$\mathbf{M}_{sj}^{(L_s+1)} = \mathrm{Min}\left(\mathbf{M}_{sj}^{(L_s+1,1)}, \cdots, \mathbf{M}_{sj}^{(L_s+1,L_j)}\right) \tag{10}$$

$$\mathbf{M}_{sj} = \mathrm{Max}\left(\mathbf{M}_{sj}^{(1)}, \cdots, \mathbf{M}_{sj}^{(L_s)}, \mathbf{M}_{sj}^{(L_s+1)}\right) \tag{11}$$

where $j = s+1, \cdots, K$.

E. Complex Analysis

From the discussion of the preceding subsection, we can see that once a K-class problem has been learned successfully by an M^3 network at time step t, the size of the M^3 network is uniquely determined. The numbers of the MIN, MAX, and INV units, and modules are shown in Tables I and II, where $\lceil z \rceil$ denotes the smallest integer greater then or equal to z.

III. A Modified GZC Discriminant Function

A. Linear discriminant functions

It is well known that a linearly separable problem can be solved by using a linear discriminant function that

divides the feature space with a hyperplane decision surface. If a linear separable problem has only two different data, a hyperplane to separate these training data can be easily created. A useful hyperplane is the perpendicular bisector (see Fig. 2) of the line joining two training inputs c_i and c_j [7]. This hyperplane can be written as

$$f_{ij}(x) = (c_j - c_i)^t x + \frac{1}{2}(\|c_i\|^2 - \|c_j\|^2) = 0 \qquad (12)$$

where $\|z\|^2$ is the squared magnitude of the vector z.

In terms of generalization, the hyperplane defined by (12) is an *optimal* hyperplane because the margin of separation between the hyperplane and the training input is maximum. A fatal weakness of the hyperplane, however, is that it lacks locally tuned response characteristics. This deficiency may lead classifiers to mistakenly produce proper output even when an unknown input is presented.

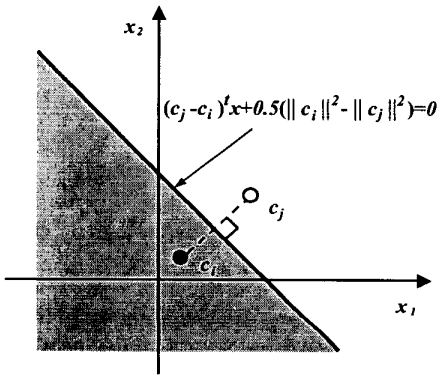


Fig. 2. The decision boundary for a hyperplane.

B. The GZC discriminant function

To overcome the weakness of existing linear discriminant functions, we have proposed a Gaussian zero-crossing function [6] for solving linearly separable problems. The definition of the GZC discriminant function is given by

$$f_{ij}(x) = \exp\left[-\left(\frac{\|x - c_i\|}{\sigma}\right)^2\right] - \exp\left[-\left(\frac{\|x - c_j\|}{\sigma}\right)^2\right] \qquad (13)$$

where $x \in \mathbf{R}^n$ is the input vector, $c_i \in \mathbf{R}^n$ and $c_j \in \mathbf{R}^n$ are the given training inputs belonging to class $\mathcal{C}_i$ and class $\mathcal{C}_j$ $(i \neq j)$, respectively, and are used as two different receptive field centers, $\sigma = \lambda\|c_j - c_i\|$ is the receptive field width, λ is a user-defined constant $(0 < \lambda)$, and the norm $\|z\|$ is the Euclidean norm of vector z. An important advantage of the GZC function over existing linear discriminant functions is its locally tuned response characteristics.

C. Restricted receptive field width

From the definition of the GZC function, we see that the receptive field width σ increases in direct ratio with the distance between the two training inputs c_i and c_j. That is, the greater the distance between c_i and c_j, the wider the receptive field width. From the viewpoint of both experimental data of neurophysiology and theoretical results of artificial neural networks, the receptive field width should be restricted within a limited value. Here, we modify the receptive field width of the GZC function as follows:

$$\sigma = \text{Min}\,(\lambda\|c_j - c_i\|,\ \gamma_{\max}) \qquad (14)$$

where $\gamma_{\max}$ is a user-defined maximum receptive field width.

The modified GZC discriminant function with a restricted receptive field width has two advantages over the original GZC discriminant function of (13). 1) It can keep locally tuned response characteristics even for very sparse training inputs. 2) It might lead to a fewer number of units required for updating during on-line learning. Figure 3 illustrates the GZC discriminant functions.

D. Upper and lower thresholds

After all modules were integrated into an M^3 network using the MIN, MAX, and INV units, the output of the network is controlled using two parameters as follows:

$$g_i(x) = \begin{cases} 1 & \text{if } y_i(x) > \theta^+ \\ \text{Unknown} & \text{if } \theta^- \leq y_i(x) \leq \theta^+ \\ -1 & \text{if } y_i(x) < \theta^- \end{cases} \qquad (15)$$

where θ^+ and θ^- are the upper and lower threshold limits of the network, respectively, and $y_i(x)$ denotes the transfer function of the M^3 network for class $\mathcal{C}_i$, which discriminates the pattern of class $\mathcal{C}_i$ from those of the rest of the classes.

The solutions to the original K-class classification problem is given by

$$\mathcal{C} = \arg\max_i \{\text{MIN}_i\} \text{ for } i = 1, \cdots, K \qquad (16)$$

where $\mathcal{C}$ is the class that the M^3 network has assigned to the input.

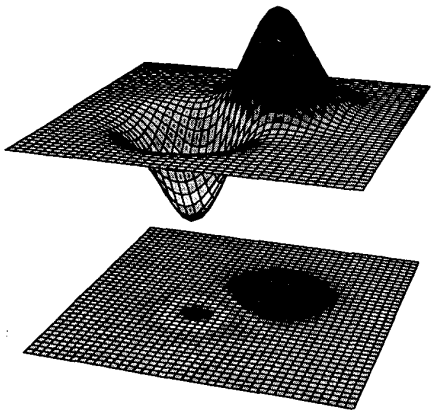
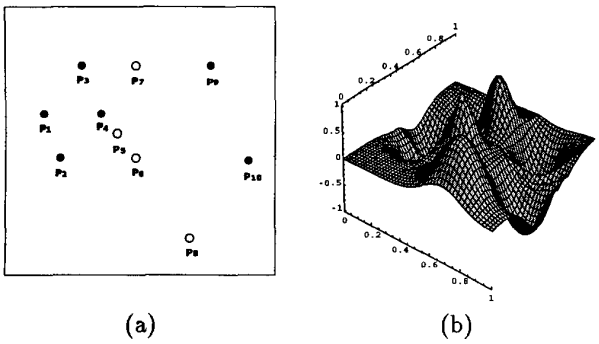


Fig. 3. The GZC discriminant function for two-dimensional input and its decision regions.

IV. An Illustrative Example

An illustrative example is presented in this section to demonstrate the generalization performance of the M^3 networks using the GZC discriminant function. Consider the two-class problem shown in Fig. 4 (a), where the point and small open circle represent the inputs whose desired outputs are '0' (class $\mathcal{C}_2$) and '1' (class $\mathcal{C}_1$), respectively. The number under the points and circles denotes the sequence of the training inputs to be presented to the network during on-line learning.

The process of learning the two-class problem using the GZC discriminant function is described as follows.



(a) (b)

Fig. 4. A two-class problem (a) and the corresponding input-output mapping produced by the M^3 network using the GZC discriminant function (b).

a) Since the first four training examples P_1 through P_4 belong to the same class $\mathcal{C}_2$, the network needs only to store these training data.

b) When the 5th training example P_5 is presented to the network, four linearly separable subproblems are generated to learn P_5 according to (2) since P_5 belongs to a new class $\mathcal{C}_1$. Four modules for solving these four linearly separable subproblems are combined using a **MIN** unit because the four linearly separable subproblems have the same positive training data P_5 and different negative training data P_1, P_2, P_3, and P_4.

c) When the 6th training example P_6 is presented to the network, four linearly separable subproblems are produced according to (4), and the corresponding four modules are added to the current M^3 network according to (10) and (11).

d) Following the same procedure mentioned at c), the training examples P_7 and P_8 are learned.

e) When the 9th training example P_9 is presented to the network, four linearly separable subproblems are produced according to (3) since P_9 belongs to the class $\mathcal{C}_2$ and the associated four modules are added to current network according to (8) and (9).

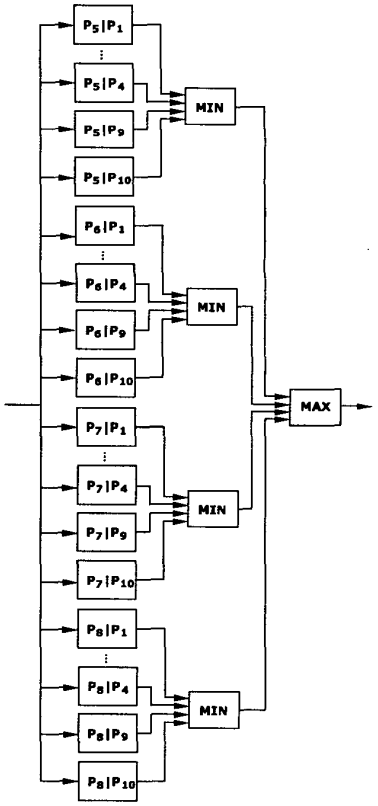


Fig. 5. The M^3 network for solving a two-class problem.

f) Following the same procedure as described in e), the last training example P_{10} is learned. The whole M^3 network and its input-output mapping are shown in Figs. 5 and 4(b), respectively.

By selecting different values of upper and lower threshold limits θ^+ and θ^-, we can obtain various decision regions as shown in Figs. 6(a), 6(b), and 6(c). From these figures, we can see that the interpolation and extrapolation capabilities of the M^3 network can be easily controlled by adjusting the upper and lower threshold limits. For example, if θ^+ and θ^- are set to 0.7 and -0.7, respectively, then the M^3 network will reject almost all of the novel inputs that are far from the training inputs (See Fig. 6(c)).

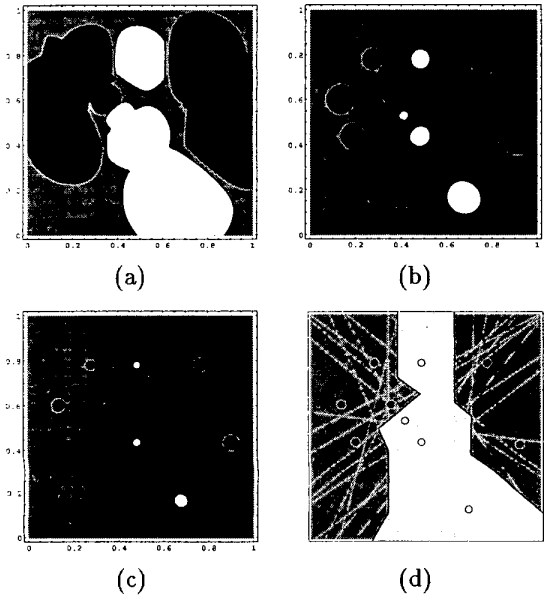


Fig. 6. Various decision regions formed by using the GZC discriminant functions under different upper and lower threshold limits (a), (b), and (c), and linear discriminant functions (d), where (a) $\theta^+ = 0.1$ and $\theta^- = -0.1$; (b) $\theta^+ = 0.5$ and $\theta^- = -0.5$; and (c) $\theta^+ = 0.7$ and $\theta^- = -0.7$. In (a), (b) and (c), the gray denotes unknown decision regions. In (d) the gray denotes the decision region of class C_2 and the gray line denotes hyperplanes used.

To compare the performance of the GZC discriminant function with that of conventional linear discriminant functions, Fig. 6(d) illustrates the decision regions formed by the M^3 network using the linear discriminant functions for solving the two-class problem. Looking at Fig. 6(d), we see that the M^3 network based on the linear discriminant functions will produce the same response to a novel input regardless of the distance between this novel input and the training inputs that have been already learned. That is, the network lacks locally tuned response characteristics. From Figs. 6(a), 6(b), and 6(c), we see that this deficiency can be dealt with by using the GZC discriminant function.

V. Conclusions

We have presented a modified Gaussian zero-crossing discriminant function with a restricted receptive field width for emergent on-line learning. We have demonstrated that the networks using this discriminant function have locally tuned response characteristics and their interpolation and extrapolation capabilities can be easily controlled by adjusting the upper and lower threshold limits of networks.

References

[1] B. L. Lu and M. Ito, "Task decomposition based on class relations: a modular neural network architecture for pattern classification", *Biological and Artificial Computation: From Neuroscience to Technology, Lecture Notes in Computer Science*, J. Mira, R. Moreno-Diaz and J. Cabestany, Eds., vol. 1240, pp. 330-339, Springer, 1997.

[2] B. L. Lu and M. Ito, "Task decomposition and module combination based on class relations: a modular neural network for pattern classification", *IEEE Trans. Neural Networks*, vol. 10, no. 5, pp. 1244-1256, 1999.

[3] B. L. Lu and M. Ichikawa, "Emergence of learning: an approach to coping with NP-complete problems in learning", *Proc. of IJCNN'2000*, vol. IV, pp. 159-164, Como, Italy, 24-27 July, 2000.

[4] B. L. Lu and M. Ichikawa, "Emergent on-line learning in min-max modular neural networks", *Proc. of IJCNN'01*, pp. 2650-2655, Washington, DC, USA, 14-19 July, 2001.

[5] B. L. Lu and M. Ichikawa, "Emergent on-line learning for pattern classification", Application for **Japan Patent** (Serial number 2001-212947), July 2001.

[6] B. L. Lu and M. Ichikawa, "A Gaussian zero-crossing discriminant function for min-max modular neural networks", *Proc. of 5th International Conference on Knowledge-Based Intelligent Information Engineering Systems & Allied Technologies*, pp. 298-302, N. Baba et al. Eds., IOS Press, 2001.

[7] N. J. Nilsson, *The mathematical Foundations of Learning Machines*, Morgan Kaufmann, San Mateo, Calif., 1990.

[8] D. Saad (Ed.), *On-line Learning in Neural Networks*, Cambridge, Cambridge University Press, 1998.