

# A selective learning algorithm for nonlinear synapses in multilayer neural networks

メタデータ	言語: English 出版者: 公開日: 2017-10-03 キーワード (Ja): キーワード (En): 作成者: Nakayama, Kenji, Hirano, Akihiro, Fusakawa, M. メールアドレス: 所属:
URL	<a href="http://hdl.handle.net/2297/6829">http://hdl.handle.net/2297/6829</a>

# A Selective Learning Algorithm for Nonlinear Synapses in Multilayer Neural Networks

Kenji NAKAYAMA      Akihiro HIRANO      Minoru FUSAKAWA

Dept of Information and Systems Eng., Faculty of Eng., Kanazawa Univ.  
2-40-20 Kodatsuno, Kanazawa, 920-8667, Japan  
e-mail: nakayama@t.kanazawa-u.ac.jp

## Abstract

*In multilayer neural networks, network size reduction and fast convergence are important. For this purpose, trainable activation functions and nonlinear synapses have been proposed. When high-order polynomials are used for nonlinearity, the number of terms in the polynomial becomes a very large for a high-dimensional input. It causes very complicated networks and slow convergence. In this paper, a method to select the useful terms in the polynomial in a learning process is proposed. This method is based on the genetic algorithm (GA), and combines the internal information, magnitude of connection weights, to select the gene in the next generation. A mechanism of pruning the terms is inherently included. Many examples demonstrate usefulness of the proposed method compared with the ordinary GA method. Convergence is stable and the number of the selected terms is well reduced.*

## 1 Introduction

A main function of multilayer neural networks is pattern mapping, which includes function approximation, pattern classification, prediction, diagnosis and the other many applications [1]. In designing multilayer neural networks, there are several parameters concerning network structure, for instance the number of layers and units, activation functions, connection density and so on. By optimizing them, fast convergence and small size networks are available. Reduction of the hidden units has been well studied. Pruning methods using only sigmoid functions [2],[3] and several kinds of activation functions [4],[5] have been proposed. Furthermore, activation functions are optimized through a learning process [6],[7],[8],[9]. Properties of the activation functions, which can be optimized, are extended [10]. Furthermore, the input potential of the unit is extended from a linear combination of the inputs or the lower layer outputs to the nonlinear function [11]. The activation functions are

formed on the hypersurface rather than the hyperplane. As a result, more complicated relation between the inputs and the output can be realized.

One approach to realize nonlinearity is to use polynomial [11]. However, in actual applications, the input data are usually high-dimensional vectors, so the number of terms in the polynomial becomes a very large. It causes large networks and slow convergence. Furthermore, all terms are not required in many applications.

In this paper, a selective learning algorithm is proposed, which combine the genetic algorithm (GA) and the internal information, which is magnitude of the connection weights. Useful terms are selected in a learning process. This method can cooperate with the simultaneous learning method for connection weights and activation functions [10]. In computer simulation, function approximation and pattern classification are dealt with. Usefulness of the proposed methods will be confirmed based on comparison with the ordinary GA method.

## 2 Network Structure and Activation Functions

### 2.1 Learning Activation Functions

An activation function is shown in Eq.(2). As shown in Fig1, an uni-pole function and a periodic function are formed as a linear combination of sigmoid functions in Eq.(1). The sigmoid function is a squashing function and stable. The parameters  $a_m, b_m, c_m, d_m$  are trained together with the connections weights [10].

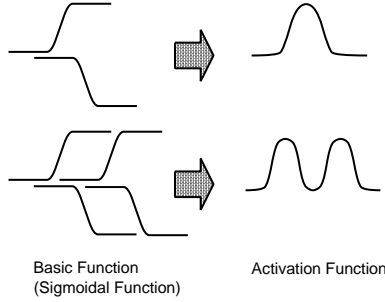
$$f(u) = \frac{1}{1 + e^{-u}} \quad (1)$$

$$f(u) = \sum_{m=1}^L \left\{ \frac{a_m}{1 + \exp(-(b_m u + c_m))} + d_m \right\} \quad (2)$$

## 3 Nonlinear Input Potential

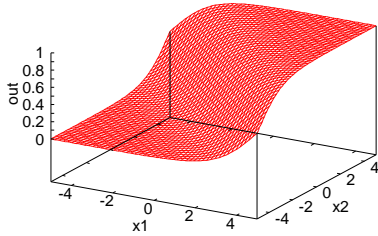
### 3.1 Linear Combiner and Nonlinear Inputs

When the input potential  $u$  of the unit is given by a linear combination of the inputs or the lower layer out-



**Figure 1:** Function synthesis by a linear combination of sigmoid functions

puts, a hyperplane is formed by setting  $u$  as a constant, on which the activation function defined by Eq.(2) is formed. In this case, even though the activation function holds a degree of freedom, it forms only arbitrary zonal shape as shown in Fig.2. By expanding the input part to a nonlinear form, the activation functions are formed on the hypersurface. This results more flexible relation between the inputs and the output. One example is shown in Fig.3, where the sigmoid activation functions used as in Figs.2. Like this, the nonlinear input potential holds a high degree of freedom for pattern mapping.



**Figure 2:** Relation of input and unit output using linear combination and sigmoid activation function.

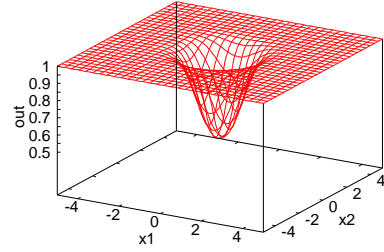
### 3.2 Nonlinear Function Expression

The nonlinear input can be expressed by using a polynomial of the inputs or the lower layer outputs [11]. Let the unit inputs be

$$\mathbf{x} = [x_1, x_2, \dots, x_N]^T \quad (3)$$

The 2-dimensional and 2nd-order polynomial is expressed as,

$$\begin{aligned} u &= w_0 + w_{11}x_1 + w_{12}x_2 \\ &+ w_{21}x_1^2 + w_{22}x_2^2 + w_{31}x_1x_2 \end{aligned} \quad (4)$$



**Figure 3:** Relation of input and unit output using nonlinear input and sigmoid activation function.

$w_{ij}$  are connection weights. By setting  $u$  to be some constant, Eq.(4) becomes a general 2nd-order function on the  $x_1$ - $x_2$  coordinate. On this curves, the activation function is formed.

$$y = f(u) \quad (5)$$

In other words,  $y$  becomes a collection of the contour lines on the curves determined by Eq.(4).

### 3.3 Necessity of Limiting Terms

When  $w_{ij}$  are trained using all terms in Eq.(4), if some of  $w_{ij}$  take a small value, then it can be judged that the corresponding terms are not necessary, and the unnecessary terms are removed. However, computations for them are not omitted in a learning process. Furthermore, if the problems are time varying and the necessary terms are changed, then all terms must be trained. For instance, the number of the terms of the polynomial is  $N(N+3)/2$  for the 2nd-order polynomial of  $N$ -dimensional input. Learning the network using all terms is not good from the view points of network size and convergence speed. For this reason, it is desirable to use only useful terms in a learning process.

## 4 A Selecting Method for Useful Terms in Polynomial

### 4.1 Genetic Algorithm

Since the proposed method is based on the genetic algorithm (GA), which to find an optimum discrete combination through an evolutionary process including cross-fertilization, natural selection and mutation.

Property of the network to be discretely optimized is mapped onto a chromosome, including genes. A population consists of many individuals, which hold their own chromosome. From one generation, new individuals are generated through cross-fertilization, natural selection and mutation. In the new individuals, that is the next generation, the dominants having good fitness are selected. This process is repeated until the fitness reaches

the desired level.

The genes arranged on a line, and are expressed by

$$c = [g_1, g_2, \dots, g_L] \quad (6)$$

where  $c$  and  $g_i$  indicate the chromosome and the genes. The individual corresponds to the neural network with the nonlinear inputs. The genes are assigned to the terms of the polynomials, and express usefulness of each term. A learning process based on GA is as follows:

1. Generation of initial population of individuals.
2. Learning individuals.
3. Evaluation of fitness of the trained individuals. If the highest fitness satisfies the threshold, then stop the learning, otherwise go to 4.
4. Selection of dominants having high fitness.
5. Generation of new individuals through cross-fertilization, natural selection and mutation using the dominants above. Go to 2.

#### 4.2 Gene Expression

An  $M$ th-order polynomial of  $N$ -dimensional input Eq.(3) is expressed by  $x_1^{l_1} x_2^{l_2} \dots x_N^{l_N}$ ,  $0 \leq l_1 + l_2 + \dots + l_N \leq M$ . These terms include from 0th-order to  $M$ th-order terms. The useful terms are selected by multiplying the genes having  $g_\alpha = 1$  or 0. Thus, the terms with  $g_\alpha = 1$  are selected.

$$\hat{x}_i = g_\alpha x_1^{l_1} x_2^{l_2} \dots x_N^{l_N}, \quad g_\alpha = 1 \quad (7)$$

The input potential and the output of the hidden unit are expressed as

$$u_j = \sum_{i=0}^{K-1} w_{ji} \hat{x}_i \quad (8)$$

$$y_j = f_j(u_j) \quad (9)$$

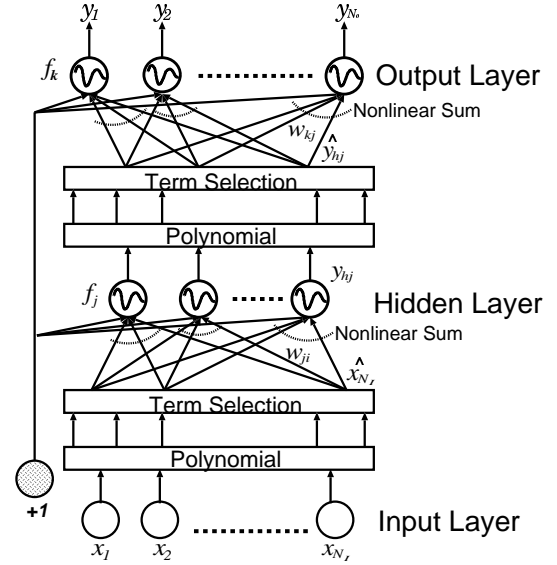
Here,  $K$  terms are selected as the useful terms. The same process is done from the hidden layer to the output layer.

#### 4.3 Network Synthesis

The proposed multilayer neural network is shown in Fig.4, in which the polynomials are selectively trained. A block "Polynomial" generate all terms. They are selected through a "Term Selection" block. Since  $\hat{x}_i$  is multiplied by  $g_i = 1$ , the gene  $g_i = 1$  is related to the connection weights  $w_{ji}$ ,  $j = 1, 2, \dots, N_H$ . Magnitude of  $w_{ji}$  are used to evaluate  $g_i$  to be selected or not to be selected.

#### 4.4 Selection Algorithm for Useful Terms

The individual in the GA process corresponds to the multilayer neural network. Let the number of the individuals be  $P$ , and the genes having  $g_\alpha=1$  be  $K$ . The



**Figure 4:** Proposed multilayer neural network with polynomial nonlinear inputs and term selection blocks.

latter means the number of the useful terms is limited to  $K$ . As a result, letting the total number of the genes in the chromosome be  $L$ , among them  $K$  genes have  $g_\alpha = 1$ , and  $L - K$  genes  $g_\alpha = 0$ .

##### Step 1

The initial  $P$  individuals are generated as follows:  $K$  genes having  $g_\alpha = 1$  and the connection weights are randomly generated. The former means position of the genes having  $g_\alpha = 1$  are randomly determined.

##### Step 2

The individuals generated in Step 1 or Step 3 are trained, that is their connection weights and activation functions are simultaneously trained [10], [11]. If the output error is less than the threshold and the number of the selected terms is well reduced, then the learning is stopped. Otherwise, go to Step 3.

##### Step 3

$P' (< P)$  individuals having the small output error are selected from the  $P$  individuals, trained in Step 2, as the dominants. A pair of two individuals  $G_1$  and  $G_2$  is selected from the  $P'$  dominants. They are combined through the evolutionary process to generate new individuals. The order of selecting the genes is determined taking the corresponding connection weights into account. Suppose  $G_1$  is dominant to  $G_2$ . First, the gene of  $G_1$ , whose connection weight has the maximum magnitude in  $G_1$ , is selected. The magnitude of the connection

weights is defined by

$$gw_i = \sum_{j=1}^{N_H} |w_{ji}| \quad (10)$$

Next, the gene of  $G_2$ , whose weight has the maximum value in  $G_2$  is selected. Furthermore, the gene of  $G_1$ , having the second largest weight is selected. This process is repeated  $K$  times.

In this Step,  $P$  individuals are generated, and return to Step 2.

**Example**

An example of  $L = 6$  and  $K = 4$  is shown in Fig.5. Here, let also  $G_1$  be dominant to  $G_2$ . The numbers attached to the genes indicate the order of magnitude of the corresponding weight. In **Step 3**, the genes having  $g_\alpha = 1$  in the next generation are selected as follows: The 2nd gene in  $G_1 \rightarrow$  the 1st gene in  $G_2 \rightarrow$  the 6th gene in  $G_1 \rightarrow$  the 2nd gene in  $G_2$ . Since the 2nd gene is selected twice, the number of the genes in a new individual is reduced from 4 to 3.

$$\begin{array}{l} \begin{array}{cc} 1 & 2 \\ G_1 = \{0,1,1,1,0,1\} \\ 1 & 2 \end{array} \rightarrow G_{1,2} = \{1,1,0,0,0,1\} \\ G_2 = \{1,1,0,1,1,0\} \end{array}$$

**Figure 5:** Selection process of genes in next generation.

**Pruning Selected Terms**

By using  $K$  as the bound of selecting times, not the number of the selected terms, a pruning process is inherently included in the proposed method. The genes are selected  $K$  times, not  $K$  genes. Thus, if the same gene is selected more than one time, then the number of the selected genes is reduced. Usually, the necessary number of the terms is not known in advance. Therefore,  $K$  is determined to be a little large. It should be minimized through the learning process. This requirement is satisfied by the proposed method.

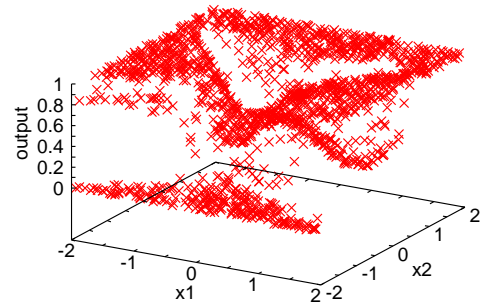
**5 Simulation and Discussions**

First, the activation functions are fixed as a sigmoid function. In Sec.5.5, the activation functions are trained together with the connection weights and the term selection.

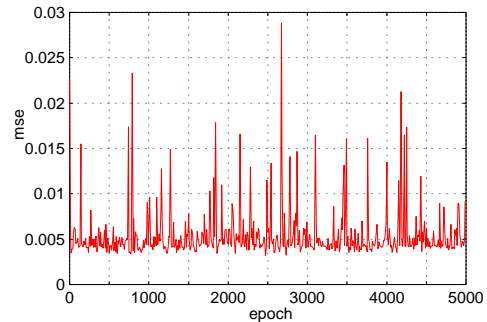
**5.1 Function Approximation**

In order to evaluate selection of useful terms by the proposed method, the training data are generated using the network shown in Fig.4, whose connection weights and

the terms used in the polynomial are randomly determined.  $x^2y$ ,  $x^4$  and  $y^5$  are selected in the polynomial. A set of the inputs and the corresponding outputs of this network is used as the training data. The number of the input units, the hidden units and the output units are 2, 2 and 1, respectively. The target function is shown in Fig.6. A 5th-order polynomial, the selection times  $K = 6$  and the number of the individuals in each generation  $P = 10$  are used. The learning curves are shown in Fig.7. The learning converges at the 3rd-generation. The approximation error is very small, and the selected terms are the same as those used in the target network.



**Figure 6:** Target function generated using network shown in Fig.4



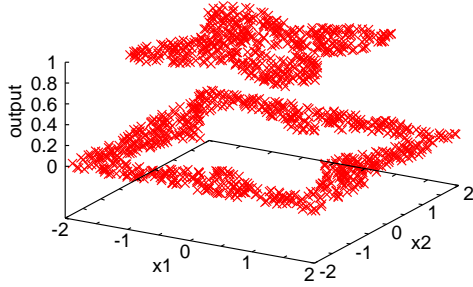
**Figure 7:** Learning curve for function approximation by proposed method.

**5.2 2-Dimensional Pattern Classification**

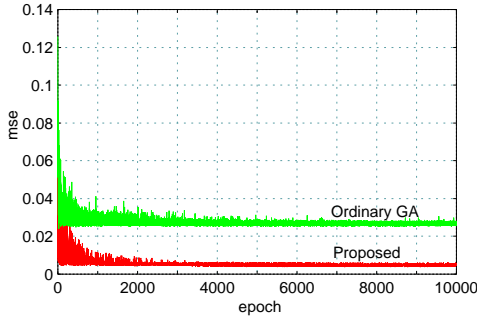
The training data are shown in Fig.8. A network with 2 hidden units, a 5th-order polynomial,  $P = 10$  and  $K = 8$  are used. In the proposed method,  $x^2$ ,  $y^2$ ,  $x^4$ ,  $x^2y^2$ ,  $y^4$  are selected, and the learning converges at the 2nd generation. For comparison, the following ordinary GA method is considered. In **Step 3**, the individuals are generated through simple cross-fertilization, natural selection and mutation without any internal information.

So, we can evaluate effects of the internal information. In this method,  $x, x^3, x^2y, y^3, x^4, x^2y^2, y^4$  are selected. The best solution is obtained in the 1st generation.

From simulation results shown in Fig.9, it can be confirmed that the proposed learning method can minimize both the error and the number of the selected terms.



**Figure 8:** Training data for 2-dimensional pattern classification.



**Figure 9:** Learning curves by proposed method and ordinary GA method.

### 5.3 Statistical Evaluation for Selecting Terms

Using the 2-dimensional pattern classification in Sec.5.2, convergence property is statistically evaluated. The learning is carried out 10 times by changing the initial guesses, including the position of the genes with  $g_i = 1$  and the connection weights.

The simulation results are shown in Table1 and Table2. In these tables,  $\circ$  and  $\times$  indicate the trial converges and not converges, respectively. In the proposed method, 6 trials converge, and the same set of the terms  $x^2, y^2, x^4, x^2, y^2, y^5$  is selected in these trials. On the contrary, in the ordinary GA method, only 3 trials converge, and the selected terms are increased from that of the proposed. They are different for each trial. The successful trials by both methods always include the terms  $x^2, y^2, x^4, x^2, y^2, y^5$ . Thus, these terms are very

important for this pattern classification. The proposed method selected only these terms.

**Table 1:** Learning results for 10 trials by proposed method.

Gener-Indi	Selected Terms	Error
2 - 2	$x^2, y^2, x^4, x^2y^2, y^4$	$\circ$
1 - 1	$x^2, y^2, x^4, x^2y^2, y^4$	$\circ$
2 - 10	$x^2, y^2, x^4, x^2y^2, x^2y^3, y^5$	$\times$
2 - 4	$x^2, y^2, x^4, x^2y^2, y^4$	$\circ$
2 - 1	$x^2, y^2, x^4, x^2y^2, y^4$	$\circ$
3 - 6	$x^2, y^2, x^2y, x^4, y^4$	$\times$
2 - 6	$x^2, y^2, x^4, xy^3, y^4$	$\times$
2 - 3	$x^2, y^2, x^3, x^4, y^4, x^5$	$\times$
2 - 4	$x^2, y^2, x^4, x^2y^2, y^4$	$\circ$
2 - 1	$x^2, y^2, x^4, x^2y^2, y^4$	$\circ$

**Table 2:** Learning results for 10 trials by ordinary GA method.

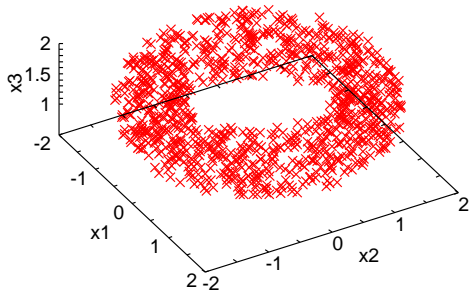
Gener-Indi	Selected Terms	Error
1 - 2	$x, x^3, x^2y, y^3, x^4, x^2y^2, y^4$	$\times$
1 - 1	$x^2, y^2, x^4, x^2y^2, y^4, x^4y$	$\circ$
1 - 6	$y, x^2, xy, y^2, x^2y, x^3y, x^2y^2, x^4y, x^2y^3, y^5$	$\times$
2 - 1	$x^2, y^2, x^3, xy^2, x^4, x^2y^2, y^4, x^3y^2, x^2y^3, xy^4$	$\circ$
1 - 10	$y, x^2, y^2, x^3, x^2y, xy^2, x^4, x^3y, x^2y^2, y^4, x^5, x^3y^2, xy^4$	$\circ$
2 - 9	$x, xy, y^2, x^2y, x^4, x^3y, x^2y^2, y^4, x^4y, x^3y^2, x^2y^3, xy^4, y^5$	$\times$
1 - 1	$y, x^2, x^3, x^4, x^3y, y^4, x^5, x^3y^2, y^5$	$\times$
2 - 9	$y^2, x^3, xy^2, x^4, x^2y^2, xy^3, y^4, x^5, x^4y, x^3y^2, x^2y^3, xy^4, y^5$	$\times$
1 - 1	$xy, y^2, x^3, y^3, x^4, x^2y^2, y^4, x^3y^2, xy^4, y^5$	$\times$
1 - 9	$y, y^2, x^3, x^2y, x^4, x^2y^2, xy^3$	$\times$

### 5.4 3-Dimensional Pattern Classification

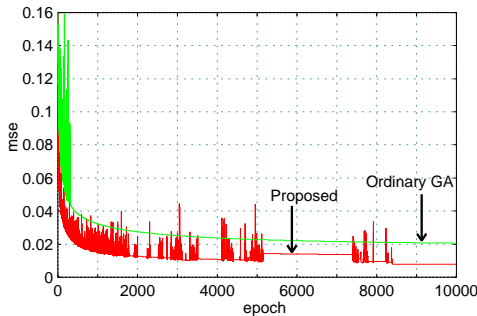
Figure 10 shows the training data, where the data locate inside and outside the doughnut are classified into two classes. Two hidden units, the individuals  $P = 10$ , and 3rd-order polynomials are used.

In the proposed method, the following 8 terms are selected while  $K = 12$ ,  $z, x^2, y^2, z^2, zx, x^3, xy^2, y^2z$ . The learning converges at the 4th generation. In the ordinary GA method, the best solution is found at the 1st generation and the 4th child, which has the terms  $x, z, x^2, y^2, xy, z^3, xy^2, y^2z, z^2x, zx^2, xyz$ . This result shows that

the simple cross-fertilization without the internal information cannot generate useful individuals. The learning curves by both methods are shown in Fig.11.



**Figure 10:** Training data for 3-dimensional pattern classification.



**Figure 11:** Learning curves by proposed and ordinary GA method.

### 5.5 Learning Activation Functions

In the previous 2- and 3-dimensional pattern classification problems, the simultaneous learning of the connection weights and the activation functions [10] and the term selection are carried out at the same time. The hidden units are reduced from 2 to 1 and from 5 to 3 in the 2- and 3-dimensional pattern classification problems, respectively. Furthermore, the best solution can be found in the 2nd generation instead the 4th generation. From these results, fast learning and small networks are possible by combining the simultaneous learning and the term selection.

## 6 Conclusions

By using nonlinear inputs to the units, activation functions are formed on a hypersurface, then more flexible relations between the input and the output are realized. The polynomials are employed to realize nonlinearity.

In this paper, a selection method based on GA for the useful terms has been proposed. The internal information, that is magnitude of the corresponding connection weight, manages order of selecting the useful terms. Through simulation for many problems, it has been confirmed that the proposed method can provide fast learning, small error and small networks in comparison with the ordinary GA method.

## References

- [1] S.Haykin, *Neural Networks-A Comprehensive Foundation*, MacmillanCollege Publishing Co. Inc., 1994.
- [2] J.Sietsma and R.J.F.Dow, "Neural net pruning-Why and how," *Proc. IEEE ICNN'88*, pp.325-333, 1988.
- [3] J.Sietsma and R.J.F.Dow, "Creating artificial neural networks that generalize," *INNS Neural Networks*, vol.4, pp.67-79, 1991.
- [4] K.Nakayama and Y.Kimura, "Optimization of activation functions in multilayer neural network," *Proc. IEEE ICNN'94*, Orlando, pp.431-436, June 1994.
- [5] K.Hara and K.Nakayama, "Comparison of activation functions in multilayer neural network for pattern classification," *Proc. ICANN'94*, Sorrento, pp.819-822, May 1994.
- [6] J.C.Zhang, M.Zhang and J.Fulcher, "Financial prediction using higher order trigonometric polynomial neural network group model," *Proc. IEEE ICNN'97*, Houston, pp.2231-2234, June 1997.
- [7] C.T.Chen and W.D.Chabg, "A feedforward neural network with function shape autotuning," *INNS Neural Networks*, vol.9, no.4, pp.627-641, 1996.
- [8] Y.Wu, M.Zhao and X.Ding, "Beyond weights adaptation: A new neural model with trainable activation function and its supervised learning," *Proc. IEEE ICNN'97*, Houston, pp.1152-1157, June 1997.
- [9] T.Burg and N.T-Gurman, "An extended neuron model for efficient time-series generation and prediction," *Proc ICANN'97*, Lausanne, pp.1005-1010, Oct. 1997.
- [10] K.Nakayama and M.Ohsugi, "A simultaneous learning method for both activation functions and connection weights of multilayer neural networks", *Proc. IEEE and INNS, IJCNN'98*, Anchorage, pp.2253-2257, May 1998.
- [11] K.Nakayama, A.Hirano and .Ido, "A multilayer neural network with nonlinear inputs and trainable activation functions: Structure and simultaneous learning algorithm", *Proc. IEEE and INNS, IJCNN'99*, Washington DC., pp.1657-1661, July 1999.