International Journal of Pattern Recognition and Artificial Intelligence
Vol. 15, No. 6 (2001) 937–947
© World Scientific Publishing Company

A NEURAL NETWORK APPROACH TO REAL-TIME PATTERN RECOGNITION*

JINWEN MA
Department of Information Science, School of Mathematical Sciences,
Peking University, Beijing 100871, People's Republic of China
E-mail: jwma@mailserv.stu.edu.cn

This paper presents a new neural network approach to real-time pattern recognition on a given set of binary (or bipolar) sample patterns. The perceptive neuron of a binary pattern is defined and constructed as a binary neuron with a neighborhood perceptive field. Letting its hidden units be the respective perceptive neurons of the patterns, a three-layer forward neural network is constructed to recognize these patterns with minimum error probability in a noisy environment. The theoretical and simulation analyses show that the network is effective for pattern recognition and can be easily implemented under strict real-time constraints.

Keywords: Forward neural network; binary neuron; pattern recognition; real-time system; radius of attraction.

1. Introduction

The major application of artificial neural network may be pattern recognition. Given a set of binary (or bipolar) sample patterns, the problem of pattern recognition via a neural network is how to design a neural network which is able to learn and store these patterns, and then recognize each of these patterns from any noisy pattern within its possibly large neighborhood. Obviously, we meet the same problem when we apply a neural network to an associative memory on the same sample pattern set. As an important model of associative memory, Hopfield network has been extensively studied and applied to such kind of pattern recognition with the sum-of-outer product scheme.[4] Since the memory capacity of Hopfield network with the sum-of-outer product scheme is very low,[9] further researches have been made on the asymmetric or generalized Hopfield model with the other learning algorithms.[1,5,6,12] Recently, the author proposed the object perceptron learning algorithm on the generalized Hopfield network which makes each sample pattern have a larger basin of attraction in comparison with the sum-of-outer product scheme on Hopfield network.[7] Unfortunately, this kind of recurrent neural networks cannot

*This work was supported by the NSFC(19701022) and the Natural Science Foundation of Guangdong Province (970377).

be applied to pattern recognition under the real-time constraints simply because the time for one recognition process, i.e. the number of iterations from one initial pattern to the final stable pattern (one of the sample patterns), cannot be known or limited.

For the purpose of real-time pattern recognition, multilayer forward neural networks (e.g. the multilayer Back-Propagation network,[11] the radial basis function network[2]) may be more suitable since the processing time for an input pattern is clear and limited. However, in attempting to apply a multilayer forward neural network to real-time pattern recognition, two major difficulties will be encountered:

(1) The existing learning algorithms are capable of making the network recognize (or retrieve) each sample pattern from itself, but it is rather difficult to make the network recognize each sample pattern from any noisy pattern within a large neighborhood of patterns.
(2) For real-time pattern recognition, the network should be implemented by the electronical devices and circuits. But the weights of the network obtained by any existing learning algorithm are real numbers and thus require high precision. Therefore, it is difficult to design these weights on the implementation of the network.

In order to overcome these two difficulties, we will propose a three-layer forward neural network based on so-called perceptive neurons for real-time pattern recognition in this paper.

In the sequel, the definition and constructivity theorem of the perceptive neuron of a binary pattern is given in Sec. 2. Section 3 proposes the neural network architecture for real-time pattern recognition. The implementation of the network is then discussed in Sec. 4. Section 5 further presents some simulation results and comparisons. A brief conclusion appears in Sec. 6.

2. The Perceptive Neuron

We now introduce the perceptive neuron and begin with a brief description of a binary neuron which is also referred to as a M-P neuron[8] or perceptron.[10]

As sketched in Fig. 1, a binary neuron is a processing element with n input signals $x_1, x_2, \ldots, x_n$ and an output signal y. There is a weight w_i on the connection from each input signal x_i to the neuron. And there is a threshold value θ for the neuron. For an input signal pattern $X = [x_1, x_2, \ldots, x_n]^T$, the output signal y of the neuron is computed by

$$y = \mathrm{Sgn}(H(x)) = \begin{cases} 1 & \text{if } H(x) > 0 \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

where

$$H(x) = \sum_{i=1}^{n} w_i x_i - \theta \,.$$

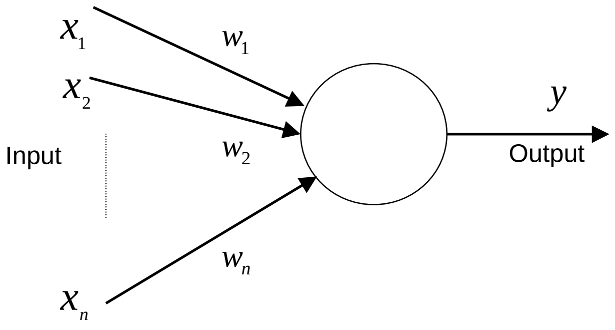

Fig. 1. The sketch of a neuron.

Supposing that $X = [x_1, x_2, \ldots, x_n]^T$ is a n-dim binary input pattern, i.e. $X \in \{0,1\}^n$, and that $C = [c_1, c_2, \ldots, c_n]^T$ is a fixed n-dim binary pattern, we define

$$d_H(X, C) = \sum_{i=1}^{n} |x_i - c_i| \tag{2}$$

as the Hamming distance between X and C. We then define the t-neighborhood of C over the n-dim binary space $\{0,1\}^n$ as follows:

$$R_t(C) = \{X : d_H(X, C) \le t\}\,. \tag{3}$$

Then, we give the definition of the perceptive neuron as follows.

Definition 1. If a binary neuron with a fixed weight vector $W = [w_1, w_2, \ldots, w_n]^T$ and a fixed threshold value θ, satisfies the following input–output relation:

$$y(X) = \mathrm{Sgn}\left(\sum_{i=1}^{n} w_i x_i - \theta\right) = \begin{cases} 1 & \text{if } X \in R_t(C) \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

it is called a t-neighborhood perceptive neuron of (*pattern*) C.

For a binary neuron, we can consider that the neuron perceives the pattern when its output is one, and it does not perceive the pattern when its output is zero. From Definition 1, the perceptive neuron perceives or recognizes a unique pattern under a noisy environment. The perceptive field of a t-neighborhood perceptive neuron of C is just the t-neighborhood of C.

We further give the constructivity theorem of the perceptive neuron as follows.

Theorem 1. *Suppose that $C = [c_1, c_2, \ldots, c_n]^T$ is a constant binary pattern, and that $d_H(C) = \sum_{i=1}^{n} c_i$ is the Hamming weight of C. If a neuron is constructed by*

$$w_i = (-1)^{1+c_i}, \quad i = 1, \ldots, n \tag{5}$$

$$\theta = d_H(C) - (t+1) \tag{6}$$

then it is a t-neighborhood perceptive neuron of C.

Proof. We first introduce the variables on input pattern X with C as follows:

$$N^{ij}(X) = |\{k : x_k = i, c_k = j\}| \quad i, j = 0, 1 \tag{7}$$

where $|A|$ is the number of elements of a set A. According to above notations, we certainly have the following two equalities:

$$d_H(C) = N^{11}(X) + N^{01}(X) \tag{8}$$

$$d_H(X, C) = N^{10}(X) + N^{01}(X)\,. \tag{9}$$

We then have

$$\begin{aligned}\sum_{k=1}^{n} w_k x_k &= \sum_{k=1}^{n} (-1)^{1+c_k} x_k = \sum_{x_k=1} (-1)^{1+c_k} \\ &= \sum_{x_k=1, c_k=1} 1 + \sum_{x_k=1, c_k=0} (-1) \\ &= N^{11}(X) - N^{10}(X)\,.\end{aligned}$$

From Eqs. (8) and (9), we further have

$$\begin{aligned}\sum_{k=1}^{n} w_k x_k &= (d_H(C) - N^{01}(X)) - N^{10}(X) \\ &= d_H(C) - (N^{01}(X) + N^{10}(X)) \\ &= d_H(C) - D_H(X, C)\,.\end{aligned}$$

Thus we have

$$\begin{aligned}y(X) &= \mathrm{Sgn}\left(\sum_{i=1}^{n} w_i x_i - \theta\right) \\ &= \mathrm{Sgn}(d_H(C) - d_H(X, C) - d_H(C) + (t+1)) \\ &= \mathrm{Sgn}(t + 1 - d_H(X, C)) \\ &= \begin{cases} 1 & \text{if } X \in R_t(C) \\ 0 & \text{otherwise.} \end{cases}\end{aligned}$$

Therefore the constructed neuron is a t-neighborhood perceptive neuron of C. The proof is completed. □

By Theorem 1, we find that the perceptive neuron can be easily constructed from the components of the pattern. Moreover, t is dominated by the threshold value of the neuron, which is very useful to design our neural network architecture for real-time pattern recognition.

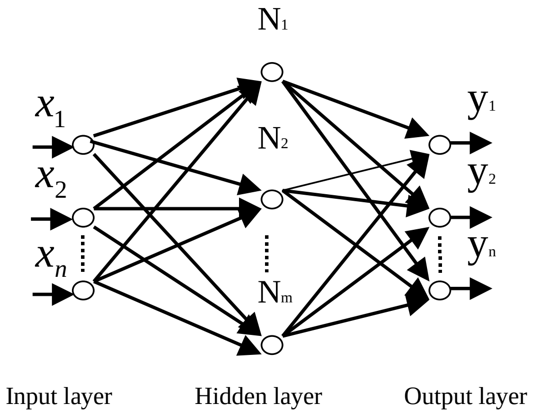

Fig. 2. The sketch of the proposed neural network for real-time pattern recognition.

3. The Neural Network Architecture for Real-Time Pattern Recognition

Given a binary sample pattern set $\mathbf{S} = \{S^1, S^2, \dots, S^m\}$, where $S^i = [s^i_1, s^i_2, \dots, s^i_n]^T \in \{0,1\}^n$ for $i = 1, 2, \dots, m$. We define the minimum Hamming distance of each sample pattern S^i to the other ones by

$$d^*_i = \min_{j \neq i} \; d_H(S^i, S^j) = \min\{d_H(S^i, S^j) : j = 1, \dots, i-1, i+1, \dots, m\}\,. \tag{10}$$

As is well-known in coding theory, $d^*_1, d^*_2, \dots, d^*_n$ really give the bounds of radii of error-correcting hyperspheres of the sample patterns (codes) in n-dim binary space. In fact, the reasonable radius of error-correcting hypersphere of each S^i should be no more than $t^*_i = [\frac{d^*_i - 1}{2}]$ by any learning algorithm or scheme. (Here $[x]$ denotes the integer part of the real number x.) For a recognition system, only when the radius of the error-correcting hypersphere of each S^i is just t^*_i, the error probability of recognition reaches the minimum in a noisy environment.

We now propose our neural network for real-time pattern recognition. As sketched in Fig. 2, it is a three-layer forward neural network which has n neurons on both the input and output layers, and m neurons on the hidden layer. We let the hidden neuron i with its weights $w^1_{i1}, w^1_{i2}, \dots, w^1_{in}$ for n input signals and its threshold value θ^1_i, be a t^*_i-neighborhood perceptive neuron of S^i. According to Theorem 1, we can construct it by

$$w^1_{ij} = (-1)^{1+s^i_j}\,, \quad j = 1, \dots, n \tag{11}$$

$$\theta^1_i = d^*_i - (t^*_i + 1)\,. \tag{12}$$

From m hidden neurons to n output neurons, the weights are given by

$$w^2_{ij} = s^i_j\,, \quad i = 1, 2, \dots, m, j = 1, 2, \dots, n\,. \tag{13}$$

And the threshold values of all output neurons are assumed to be zero, i.e.

$$\theta_j^2 = 0\,, \quad j = 1, 2, \ldots, n\,. \tag{14}$$

We further analyze the function of the neural network in a noisy environment. Suppose that $X = [x_1, x_2, \ldots, x_n]^T$ ($\in \{0,1\}^n$) is an input pattern and $Y = [y_1, y_2, \ldots, y_n]^T$ is the corresponding output pattern. Denote the output of each hidden neuron i to be u_i. When X is input to the neural network, the hidden neurons process perceptively at the same time. The results are then transmitted to the output layer and we finally get the output Y from the output neurons. Obviously, this process can be done in a very short time. (The real-time processing properties will be more discussed in the following section.) For pattern recognition, we suppose that X is a noisy pattern of S^i in $R_{t_i^*}(S^i)$, that is, $d_H(X, S^i) \le t_i^*$. Because $d_i^* \ge 2t_i^* + 1$, we have $X \notin R_{t_k^*}(S^k)$, where $k \neq i$. Therefore we have

$$u_i = \mathrm{Sgn}\left(\sum_{j=1}^{n} w_{ij}^1 x_j - \theta_i^1\right) = 1$$

$$u_k = \mathrm{Sgn}\left(\sum_{j=1}^{n} w_{kj}^1 x_j - \theta_k^1\right) = 0\,, (k \neq i)\,.$$

We further have

$$Y(X) = (y_1(X), y_2(X), \ldots, y_n(X))^T = S^i.$$

If X is not in any $R_{t_i^*}(S^i)$, that is, $d(X, S^i) > t_i^*$ for $i = 1, 2, \ldots, m$, the output of each hidden neuron becomes zero. Then the output of the network is zero pattern (assuming that zero pattern is not a sample pattern). This shows that X has too many error bits to be recognized by our neural network. Moreover, the network makes mistakes in the case that one sample pattern polluted by too many error bits turns to be in the neighborhood of the other sample pattern. However, since the radius of the error-correcting hypersphere of each S^i is t_i^* by our neural network, the error probability of recognition certainly reaches the minimum in a noisy environment. In fact, this minimum error probability is generally very small and we can neglect the two cases.

As a whole, our proposed neural network is able to store any set of m binary sample patterns and recognize each of them from any noisy pattern within its possibly large neighborhood. That is, the network recognizes the sample patterns with the minimum error probability in a noisy environment. Furthermore, the process of pattern recognition for any input pattern can be done in a very short time.

When the sample patterns are bipolar, we can use bipolar neurons in our proposed neural network and design the weights and threshold values by the corresponding bipolar patterns. It is easy to prove that the neural network in the bipolar case has the same real-time pattern recognition function as the above one.

4. The Implementation of the Neural Network

We further discuss the implementation of our proposed neural network. As long as the sample pattern set is given, the weights and threshold values of our neural network can be computed directly from the binary sample patterns just as those of Hopfield network with the sum-of-outer product learning scheme. Moreover, every weight of the neural network is either 1 or -1 if it is not zero, which makes it very easy to realize our neural network by electronic devices and circuits.

The complexity of architecture of a three-layer forward neural network depends on the number of processing neurons (having the computing capability) and the distribution of the fan-in connection density of processing neurons in the network. Clearly, the input neurons are not processing neurons, but the hidden neurons are certainly processing neurons. Generally, the output neurons are processing ones. But in our neural network, each output neuron works actually as a signal-inspecting device since there is always at most one connection to it carrying the positive signal, that is, $u_i = 1$. Therefore the output neurons of the neural network do not need the computing capability and we will not consider them as processing neurons. Thus there are only m processing neurons in our neural network. It is clear that the fan-in connection density of each processing neurons is n. By summing up two results, we find that the complexity of architecture of our proposed neural network is much less than that of a general three-layer forward neural network. Therefore it can be implemented much easily than a general three-layer forward neural network.

With the development of technique of VLSI and artificial neural network, we can implement the neural network with eight thousand processing neurons.[3] So the proposed neural network can be constructed with enough processing neurons to fulfill a practical and complex pattern recognition task. On the other hand, one computation of a neuron can be fulfilled in a nanosecond. Then the time for processing an input pattern by our neural network is at most a few nanoseconds. Therefore our neural network can be used for pattern recognition under strict real-time constraints.

5. Simulation Analyses and Comparisons

In this section, we make some simulations to compare our proposed neural network with the other typical neural networks or learning algorithms for real-time pattern recognition. Since the real-time processing characteristic of each of these neural networks is very clear, we do not consider the executing time of every simulation. So we here compare our neural network with the other typical neural networks or learning algorithms only on the performance of pattern recognition or error-correcting.

Our simulation experiments are undertaken on the sample set of ten Arabic numerals $\{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$ which are expressed by 7×7 pixies in Fig. 3. For the sake of convenience, here we use the same set of sample patterns as that of Ref. 7 in which the author made some simulations to check the performance of

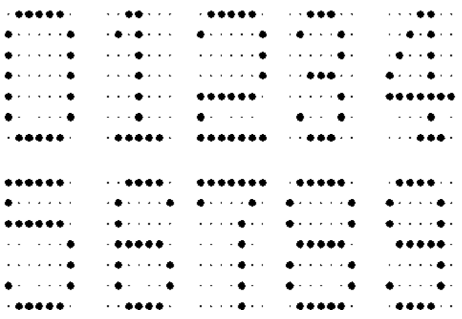

Fig. 3. The sample patterns of ten Arabic numerals $\{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$.

the generalized Hopfield network on associative memory of this sample set via the object pecerptron learning algorithm.

Based on the Hamming distances between these sample patterns, we have

$$(t_1^*, t_2^*, t_3^*, t_4^*, t_5^*, t_6^*, t_7^*, t_8^*, t_9^*, t_{10}^*) = (3, 6, 5, 4, 9, 4, 5, 8, 3, 4)\,.$$

With these sample patterns and above data, we can easily construct our neural network with 49 input neurons, 10 hidden neurons and 49 output neurons respectively by Eqs. (11)–(14). Then the radius of attraction of a sample pattern $S^i (= i - 1)$, i.e. the radius of the largest error-correcting hypersphere of S^i, is just t_i^* for $i = 1, 2, \ldots, 10$.

Before we make the following discussions, we introduce how to estimate the radius of attraction of a sample pattern (as a stable state) under a neural network if it cannot be computed directly. For a sample pattern $S^i (i \in \{1, 2, \ldots, 10\})$ and the integer number $j (j \geq 0)$, we randomly select 1000 initial (or input) patterns with a Hamming distance being j from S^i for a recurrent (or forward) neural network. These initial (or input) patterns can be considered as S^i polluted by j errors in some j components. Then the neural network operates with each initial (or input) pattern. We check whether the network finally evolves to (or produce) S^i or not. If the network evolves to (or produces) S^i for all 1000 initial patterns, we are sure that j is a possible radius of attraction of S^k. In this way for j from $0, 1, 2, \ldots,$ we can estimate the radius of attraction of S^k — the largest possible radius of attraction of S^k.

We first compare the performance of our neural network with those of Hopfield network and the generalized Hopfield network. According to the simulation results given in Ref. 7, the ten sample patterns cannot be all stable on the Hopfield network constructed by the sum-of-outer product scheme. Therefore we cannot use Hopfield network with the sum-of-outer product scheme to fulfill the pattern recognition or associative memory task on these sample patterns. By the object perceptron learning algorithm on the generalized Hopfield network, the ten sample patterns can be all stable with their radii of attraction, $R_{GHN}(S^i)$, listed in the third row of Table 1. Comparing the radii of attraction under our neural network, i.e. the data

Table 1. The radii of attraction of the ten Arabic numerals under the four neural networks for pattern recognition.

S^i	0	1	2	3	4	5	6	7	8	9
t_i^*	3	6	5	4	9	4	5	8	3	4
$R_{GHN}(S^i)$	2	6	3	2	8	4	3	9	2	3
$R_{BPN}(S^i)$	1	2	2	5	4	1	2	3	1	1
$R_{RBFN}(S^i)$	2	3	4	2	5	6	2	3	2	2

of t_i^* in the second row of Table 1, with these data, we find that the performance of our neural network on pattern recognition is better than that of the generalized Hopfield with the object perceptron learning algorithm.

We further compare the performance of our neural network with that of the three-layer Back-Propagation network. Obviously, the three-layer Back-Propagation network has the same architecture as our neural network. We apply the Back-Propagation algorithm to training the network, i.e. the weights and threshold values of the network, to fit the following object function:

$$F(S^i) = e_i\,, \quad i = 1, 2, \ldots, 10 \tag{15}$$

where e_i is the ten-dim vector of which the ith component is one and the other components are all zero. After 2306 iterations, we obtained a three-layer Back-Propagation network which can fit the object function with the total error being less than 0.1. When we estimate the radii of attraction of the sample pattern with the trained network, the output of an output neuron is considered to be zero if its actual output for an input pattern is less than 0.5, otherwise, it is considered to be one. By the simulation on the Back-Propagation network, we obtain the radii of attraction of the ten sample patterns, $R_{BPN}(S^i)$, listed in the fourth row of Table 1. Comparing the data of t_i^* with these data, we find that the performance of our neural network is much better than that of the Back-Propagation network with the same architecture on pattern recognition.

We finally compare the performance of our neural network with that of the radial basis function network for the same architecture. We select the Gaussian functions of $\sigma = 1$ as the radial basis functions and use the ten sample patterns as the centers of the ten Gaussian functions respectively. The object function is now changed to

$$F(S^i) = E^i\,, \quad i = 1, 2, \ldots, 10 \tag{16}$$

where E_i is the ten-dim vector of which the ith component is one and the other components are all -1. We use the Widrow learning algorithm to train the coefficients from the Gaussian functions to each output linear element in the network. After about 400 iterations of the coefficients of the radial basis functions for each output linear elements, we finally obtained a radial basis function network which can fit the object function with the total error being less than 0.1. When we estimate the radii of attraction of the sample pattern with the trained radial basis function network in this case, the output of an output neuron is considered to be -1 if its actual output for an input pattern is less than zero, otherwise, it is considered to be one. By the simulation, we obtain the radii of attraction of the ten sample patterns, $R_{RBFN}(S^i)$, listed in the fifth row of Table 1. Comparing the data of t_i^* with these data, we find that the performance of our neural network is still much better than that of the radial basis function network with the same architecture on pattern recognition. However, the performance of the radial basis function network is improved in comparison with that of the Back-Propagation network.

As a result of the simulation analyses and comparisons, our proposed neural network is more effective for pattern recognition than the other typical neural networks or learning algorithms.

6. Conclusion

Based on the perceptive neurons, we have constructed a three-layer forward neural network for real-time pattern recognition on a sample set of binary (or bipolar) patterns. The neural network is able to store these sample patterns and to recognize each of them from any noisy pattern within its possibly large neighborhood in a few nanoseconds. Moreover, the neural network can be easily implemented by electronical devices and circuits. The simulation results also show this neural network is more effective for pattern recognition than the other existing models or learning algorithms.

References

1. L. F. Abbott and T.B. Kepler, “Optimal learning in neural network memories,” *J. Phys. A: Math. General* **22** (1989) L711–L717.
2. D. S. Broomhead and D. Lowe, “Multivariable functional interpolation and adaptive networks,” *Compl. Syst.* **2** (1988) 321–355.
3. T. Clarkson, “Neural networks in VLSI hardware,” *Neural Networks and Their Applications*, ed. J. G. Taylor, John Wiley, Chichester, 1996, pp. 245–253.
4. J. J. Hopfield, “Neural networks and physical system with emergent collective computational abilities,” *Proc. Nat. Acad. Sci. USA* **79** (1982) 2554–2558.
5. J. Ma, “The asymmetric Hopfield model for associative memory,” *Proc. Int. Joint Conf. Neural Networks* (*IJCNN’93*), Nagoya, Japan, October 25–29, 1993, pp. 2611–2614.
6. J. Ma, “The stability of the generalized Hopfield networks in randomly asynchronous mode,” *Neural Networks* **10** (1997) 1109–1116.
7. J. Ma, “The object perceptron learning algorithm on the generalised Hopfield networks for associative memory,” *Neural Comput. Appl.* **8** (1999) 25–32.
8. W. S. McCulloch and W. Pitts, “A logical calculus of the ideas imminent in nervous activity,” *Bull. Math. Biophys.* **5** (1942) 115–133.

9. R. E. McEliece, E. C. Posner, E. R. Rodemich and S. S. Venkatesh, "The capacity of the Hopfield associative memory," *IEEE Trans. Inform. Th.* **IT-33** (1987) 461–483.
10. F. Rosenblatt, *Principles of Neurodynamics*, Spartan Books, NY, 1962.
11. D. E. Rumelhart, G. E. Hinton and R. J. Williams, "Learning internal representations by error propagation," *Parallel Distributed Processing: Exploration in the Microstructure of Cognition*, eds. D. E. Rumelhart and J. L. McClelland, MIT Press, Cambridge, MA, 1986, pp. 318–362.
12. S. S. Venkatesh and D. Pitts, "Linear and logarithmic capacities in associative memory," *IEEE Trans. Inform. Th.* **IT-35** (1989) 558–568.

Jinwen Ma received the MSc. degree in applied mathematics from Xi'an Jiaotong University in 1988 and the Ph.D. in probability theory and statistics from Nankai University in 1992. From July 1992 to November 1999, he worked in the Department of Mathematics, Shantou University. From December 1999, he worked as a full professor at the Institute of Mathematics, Shantou University. In September 2001, he was transferred to the Department of Information Science at the School of Mathematical Sciences, Peking University. During 1995 and 2000, he also visited and studied at the Department of Computer Science & Engineering, the Chinese University of Hong Kong. He has published about 40 academic papers on neural networks, pattern recognition, artificial intelligence, and information theory.