**神经网络概述**

**Summary of Artificial Neural Networks**

**Abstract:** This paper made a brief summary on artificial neural networks(ANN): the definition; the development by way of introducing the historic celebrities (McCullock, Pitts, Hebbs, John Von Neumann, Franr Rosenblatt, Bernard Widrow, Marcian Hoff, Marvin Minsky, Seymour Pappert and John Hopfield); basic components (neurons and weights); training (supervised training and unsupervised training); threshold functions; characteristics; typical neural network models (perceptron, adaptive linear element neural network(adaline), back-propagation neural network(BP), radial basis function neural network(RBF), competitive learning neural network, learning vector quantization neural network(LVQ), Elman neural network, Hopfield neural network, Boltzmann neural network); and the applications (complex system modeling, data compression, character recognition, target classification, noise filtering, servo-control systems, text-to-speech conversion).

**Key words:** artificial neural network; training; threshold function

关键词：神经网络 训练 泛化 阈函数

**1. Introduction**

Artificial Neural Networks (ANN) are computing systems made up of a number of simple, highly interconnected processing elements, which processes information by their dynamic state response to external inputs[1]. Garrett[2] has given an interesting engineering definition of the ANN as: “A computational mechanism able to acquire, represent and compute mapping from one multivariate space of information to another, given a set of data representing that mapping”.

The study of artificial neural network originally grew out of a desire to understand the function of the biological brain[3]. The human brain is a complex computing system capable of thinking, remembering, and solving problems. There have been a number of attempts to emulate the brain functions with a computer model, and generally these have involved the simulation of a network of neurons, commonly called neural

|  |
| --- |
| **Fig.1 Sketch of a Biological Neuron** |

|  |
| --- |
| Dendrite |

|  |
| --- |
| Soma |

|  |
| --- |
| Axon |

|  |
| --- |
| Synapse |

networks. The brain contains approximately 100 billion neurons that are densely interconnected with one thousand to ten thousand connections per neuron.

A neuron is the fundamental cellular unit of the Brain’s nervous system. It is a simple processing unit (soma) that receives and combines signals from other neurons through input paths called dendrites which contain synaptic junctions. The basic components of a neuron are shown in Fig.1. If the combined signal from all the dendrites is strong enough, the neuron “fires”, producing an output signal along a path called the axon. The axon splits up and connects to thousands of dendrites (input paths) of other neurons through synapses (junctions containing a neurotransmitter fluid that controls the flow of electrical signals) located in the dendrites. Transmission of the signals across the synapses are electro-chemical in nature, and the magnitudes of the signals depend upon the synaptic strengths of the synaptic junctions. The strength or conductance (the inverse of resistance) of a synaptic junction is modified as the brain “learns”. In other words, the synapses are the basic “memory units” of the brain.

A general introduction to the subject of artificial neural networks is given and the tenuous relationship of neural networks to the biological neuron structure of the brain is also briefly outlined. The development of artificial neural networks has been marked by periods of considerable optimism and others of disillusionment.

**2. Historical summary**

Artificial Neural Networks now have a relatively long history and a correspondingly large amount of literature exists on their properties and development. It would be unrealistic to attempt to provide an extensive coverage of the literature. The study is accordingly restricted to some of the most significant historical markers in the development of ANN’s and these are given below. The references and events outlined in this section cover only what are believed to be major developments and turning points in ANN research[4].

**2.1. McCulock And Pitts**

The first significant paper on artificial neural networks is considered to be that of McCullock and Pitts in 1943. This paper generally outlined some concepts concerning how biological neurons could be expected to operate. The neuron models proposed were modeled by simp1e arrangements of hardware which attempted to mimic the performance of the single neural cell.

**2.2. Hebbs**

The book *The Organisation of Behaviour* written by Hebb in 1949 formed the basis of “Hebbian Learning” which forms an important part of ANN theory today. The basic concept under lying “Hebbian Learning” is the principle that every time a neural circuit is used, the pathway is strengthened.

**2.3. John Von Neumann**

In 1958 Neumann wrote a book *The Computer and the Brain* in which he proposes modeling the brain performance by items of hardware available at that time. About this time of neural network development the digital computer became more widely available and its availability proved to be of great practical value in the further investigation of ANN performance.

**2.4. Franr Rosenblatt**

Rosenblatt constructed neuron models in hardware during 1957. These models ultimately resulted in the concept of the Perceptron. This was an important development and the underlying concept is still in wide use today.

**2.5. Bernard Widrow And Marcian Hoff**

The researchers Barnard Widrow and Marcian Hoff were responsible for the development of first the ADALINE and then the MADALINE networks. The name ADALINE comes from “ADAptive LINEar Complier” and the name MADALINE comes from “Multiple ADALZNE” respectively. Much was made of the potential scope of the technique but this was succeeded by a period of disillusionment.

**2.6. Marvin Minsky And Seymour Pappert**

In 1969 Marvin Minsky and Seymour Pappert published an influential book *Perceptions* which showed that the perception as developed by Rosenblatt had serious limitations. This very thorough work was well researched and showed that the Perception in the form it had at the time suffered from severe limitations. The essence of the book *Perceptions* was the assumption that the inability of the perception to be able to handle the “exclusive or” function was a common feature shared by all neural networks. As a result of this assumption interest in neural networks was greatly reduced. The overall effect of the book was to reduce the amount of research work on neural networks for the next 10 years. The book served to dampen the unrealistically high expectations previously held for ANN’s. Despite the reduction in ANN research, a number of people still persisted in ANN research work.

**2.7. John Hopfield**

After 10 years in the doldrums John Hopfield produced a paper in 1982 which showed that the ANN had potential for successful operation and showed how this could be developed. This paper was timely as it marked a second beginning for the ANN. While Hopfield is the name frequently associated with the resurgence in interest in ANN it probably represented the culmination of many peoples work in the field. From this time onwards the field of neural computing began to expand and now there is worldwide enthusiasm as we11 as a growing number of important practical applications.

**2.8. Resurgence**

The resurgence of interest in artificial neural networks over the last few years is due to a number of factors. One of the continuing and significant driving forces in neural network research has been the desire of many diverse groups which include neuro physicians, engineers, psychiatrists, psychologists and biologists to gain an understanding of the workings and behaviour of the brain.

Some of the valuable features of artificial neural networks which distinguish this method of computation from other algorithm based methods of computation are now considered.

One of the most important if not the most important attribute of neural networks is the ability to generalize. By this is meant the ability of a neural network to successfully interpret data which it has not previously encountered and to provide a sensible result. It is this property which sets the neural network in a different category to a system such as a look up table where it is necessary to store all of the information likely to be required for reference on future occasions.

**3. Basic components of neural networks**

|  |
| --- |
| **Fig.2 Schematic Representation of an Artificial Neuron** |

The computer simulation of this brain function usually takes the form of artificial neural systems which consists of many artificial neurons, usually called processing elements or neurodes. The schematic equivalents is shown in Fig.2. These processing elements are analogous to the neuron in that they have many inputs (dendrites) and combine (sum up) the values of the inputs. This sum is then subjected to a nonlinear filter usually called a transfer function, which is usually a threshold function or a bias in which output signals are generated only if the output exceeds the threshold value. Alternately, the output can be a continuous function (typically a sigmoid function limited to the range 0 to +1 or an arctangent or hyperbolic tangent function limited to -1 to +1) of the combined input. Sometimes the outputs are “competitive” in which only one processing element has an output. The output of a processing element (axon) branches out and becomes the input to many other processing elements. These signals pass through connection weights (synaptic junctions) that correspond to the synaptic strength of the neural connections. The input signals to a processing element are modified by the connection weights prior to being summed by the processing element. There is an analogy between a processing element and an operational amplifier in an analog computer in which many inputs are summed. The potentiometer settings on the amplifier inputs correspond to the connection weights and the output of the operational amplifier goes through some sort of nonlinear function generator.

The range of types of neural networks which have been developed is large and it is desirable to attempt to establish some kind of relationship between the various kinds of networks[5]. There are a number of ways in which neural networks may be categorized based on characteristics such as,

-The method of training adopted, directed or non-directed

-Whether after training feedback or non feedback operation is involved

-The type of training algorithm employed

In order to consider the operation of artificial neural networks it is first necessary to introduce some of the terms used. This will be done in the following section.

**3.1. Neurons**

The neuron forms the node at which connections with other neurons in the network occur. Like the biological network, the neuron in the artificial network is also central to network operation as much of the activity on the system occurs at the neuron. Although the infinitely more successful biological neural network neurons are not arranged in any geometric pattern those in the electronic network are generally arranged in one or more layers which contain neurons performing a similar function. Depending on the type of neural network being considered, connections may or may not exist between neurons within the layer in which they are located. For example in the Back Propagation Network there are no connections between the neurons in the same layer but in the case of the Hopfield Network every neuron is connected to all neurons in the layer.

**3.2. Weights**

In the trained artificial neural network the intelligence of the network is stored in the values of the connections existing between the neurons. In artificial neural network terminology the values of the connections between the neurons are generally referred to as weights.

Hidden layers also take part in producing output when training is complete and the network is being interrogated. The number of hidden layers provided is problem dependant. There may be advantages in providing several hidden layers but additional layers may mean a marked increase in the training time taken.

**4. Training (Learning)**

In contrast to expert systems which incorporate a knowledge base, neural networks do not have such a collection of information. They need to be trained for a given problem or situation so that the weights will then contain the required information. One of the ways of classifying training procedures into two categories is whether directed training or non directed training is employed and these methods are now considered[6].

**4.1. Supervised Training or Directed training**

When employing directed training it is necessary to include among the set of data presented to the neural network the result or answer corresponding to each particular set of data .The data set is repeatedly presented to the neural network until the network output corresponds closely enough to the result in the data set. Should the difference between the network output and the “Target Value” exceed the permitted tolerance the network training process is repeated. The algorithm being used for the network training causes further adjustment to occur and the process is repeated until the tolerance is acceptable.

In supervised training, training data set contains a collection of information representing the input pattern vector together with the target value or desired output. This set of training data is presented to the network repeatedly until the difference between the target output and the actual output of the network reaches a certain predetermined value.

**4.2. Unsupervised training or non-directed training**

In the case of non-directed training the target value or answer, is not provided and the information in the training data set is continuously presented until some convergence criteria is satisfied.

In unsupervised training the output pattern for a given input pattern is not provided. The neural network constructs internal models that capture regularities in input pattern.

For any problem it is necessary to provide the neural network with training data which covers the extent of the problem. This data will need to include sufficient information so that the problem is unambiguous. The training set is repeatedly applied to the neural network until some specified training criteria is met.

**5. Threshold functions**

A threshold function provides a means of further processing the output a neuron after the initial processing has taken place. It enables signals of greatly differing amplitudes to be satisfactorily dealt with in the neuron. Many different functions such as the sigmoid and the tanh have been utilized for this purpose. Both of these functions provide the following desirable characteristics, (1)small signals are dealt with in a linear manner; (2)large signals are limited to a maximum value.

**Table 1 MOST COMMONLY USED THRESHOLD FUNCTIONS**

|  |  |
| --- | --- |
| LINEAR FUNCTION  (Ramp Function)  This function  — has a linear zone  — is easily implemented | f (x)=αx  where α is a constant |
| NON-LINEAR(RAMP) FUNCTION  This function is used to represent simplified non-linear operation |  |
| STEP FUNCTION  (HARD LIMITING)  This function is  — fast and easy to implement  — has no linear range  —cannot smoothly imitate functions |  |
| SIGMOID FUNCTION  (S-SHAPED FUNCTION)  This function is  — continuously differentiable  — can make fuzzy decisions  — not easy to implement  — the output is limited to positive values |  |
| HYPERBOLIC TANGENT FUNCTION |  |

The sigmoid function is favoured as it has a simple mathematical form which in certain circumstances enables the analysis to be simplified. Threshold functions[7] are also called activation functions, signal functions, or squashing functions are used to map the input pattern of a neuron to the specified output range .The most commonly used threshold functions are listed in Table 1.

**6. Characteristics of Neural Networks**

The characteristics that make neural network systems different from traditional computing and artificial intelligence are: 1) learning by example, 2) distributed associative memory, 3) fault tolerance and 4) pattern recognition.

The memory of a neural network is both distributive and associative. Distributed means that the storage of a unit of knowledge is distributed across all memory units (connection weights) in the network. A unit of knowledge shares these memory units with all other items of knowledge stored in the network. Associative means that when the trained network is presented with a partial input, the network will choose the closest match to that input in its memory and generate an output that corresponds to the full output.

Traditional computer systems are rendered useless by any damage to its memory. However neural-computing systems are fault- tolerant in that if some processing elements are destroyed or disabled or have their connections altered incorrectly, the behavior of the network is changed only slightly. As more processing elements are destroyed, performance degrades gradually, i.e., the network performance suffers but the system does not fail catastrophically. This is because the information is not contained in any single memory unit, but rather is distributed among all the connection weights of the network. Such arrangements are well-suited for systems where failure may be unacceptable or introduce difficult problems (e.g., in nuclear power plants, missile guidance, and high performance aircraft).

Pattern recognition is the ability to match large amounts of input information simultaneously and generate a categorical or generalized output. It requires that the network provide a reasonable response to noisy or incomplete inputs. Experience shows that neural networks are very good pattern recognizers which also have the ability to learn and build unique structures for a particular problem.

**7. Typical Neural Network Models**

**The taxonomy of some typical neural network architectures is shown in Fig.3.**

|  |
| --- |
| Neural Network |

|  |
| --- |
| Feed-forward Network |

|  |
| --- |
| Recurrent/feedback Network |

|  |
| --- |
| Single-layer  Perceptron |

|  |
| --- |
| Multilayer  Perceptron |

|  |
| --- |
| Competitive Learning Network |

|  |
| --- |
| Kohonen’s  SOM |

|  |
| --- |
| ART  Model |

|  |
| --- |
| Radial Basis  Function Network |

|  |
| --- |
| Hopfield Network |

|  |
| --- |
| **Fig.3 Taxonomy of neural network architectures** |

**7.1. Perceptron**

The multilayer perceptron[8] consists of a system of simple interconnected neurons, or nodes, which is a model representing a nonlinear mapping between an input vector and an output vector. The nodes are connected by weights and output signals which are a function of the sum of the inputs to the node modified by a simple nonlinear transfer, or activation, function. It is the superposition of many simple nonlinear transfer functions that enables the multilayer perception to approximate extremely non-linear functions. If the transfer function was linear then the multilayer perception would only be able to model linear functions. Due to its easily computed derivative a commonly used transfer function is the logistic function. The output of a node is scaled by the connecting weight and fed forward to be an input to the nodes in the next layer of the network. This implies a direction of information processing, hence the multilayer perception is known as a feed-forward neural network. The architecture of a multilayer perception is variable but in general will consist of several layers of neurons. The input layer plays no computational role but merely serves to pass the input vector to the network. The terms input and output vectors refer to the inputs and outputs of the multilayer perception and can be represented as single vectors. A multilayer perception may have one or more hidden layers and finally an output layer. Multilayer perceptions are described as being fully connected, with each node connected to every node in the next and previous layer.

By selecting a suitable set of connecting weights and transfer functions, it has been shown that a multilayer perception can approximate any smooth, measurable function between the input and output vectors. Multilayer perceptions have the ability to learn through training. Training requires a set of training data, which consists of a series of input and associated output vectors. During training the multilayer perception is repeatedly presented with the training data and the weights in the network are adjusted until the desired input-output mapping occurs. Multilayer perceptions learn in a supervised manner. During training the output from the multilayer perception, for a given input vector, may not equal the desired output. An error signal is defined as the difference between the desired and actual output Training uses the magnitude of this error signal to determine to what degree the weights in the network should be adjusted so that the overall error of the multilayer perception is reduced. There are many algorithms that can be used to train a multilayer perception. Once trained with suitably representative training data the multilayer perception can generalize to new, unseen input data.

|  |
| --- |
| Initialize all weights and node bias to random values |

|  |
| --- |
| Choose a suitable value of learning rate *η* |

|  |
| --- |
| Calculate the error e = djp-Ojp  [djp =desired output unit k]  [Ojp =actual output unit k]  n卜0  …J…  .do  f..毛﹃...︸ |

|  |
| --- |
| Present to the neural network the input and target value  (the training set) |

|  |
| --- |
| adjust weights only if xi<1  where xi=wij+*η*exi |

|  |
| --- |
| Specified accuracy? |

|  |
| --- |
| S |

|  |
| --- |
| N |

|  |
| --- |
| Y |

|  |
| --- |
| Fig.4 Flow diagram of perceptron training program |

The perceptron training flow diagram given in Figure 4 uses directed or supervised training and the procedure for obtaining the difference value may be seen in the figure.

**7.2. Adaptive linear element neural network**

|  |
| --- |
| **Fig. 5 Adaline diagram** |

Adaline was firstly proposed by Widrow and Hoff[9, 10] from Standford University. An adaline is a multi-input, single-output, single layer linear neural element, and its characteristics are:

(1) Train on-line based on the changing inputs and the target response;

(2) Self adaptive algorithm can be applied to the weights training;

(3) Simple structure makes it easily implemented on hardware.

Graphically, an adaline is represented by the construction[11] shown in Fig.5. Where, *k* is the time index; *Xk* = [*x1k*, *x2k*, …, *xnk*] the input vector; *Wk* = [*w1k*, *w2k*, …, *wnk*] the weight vector; the network output; the error; is the target output.

Training algorithm is the main characteristic of the artificial neural network, and the training process of adaline is also the process of modifying the weights of the network. Through this, the error between target output and real output yk can be minimized. Widrow–Hoff learning rules are adopted here. Firstly, an output error function of the linear network is defined as:

(1)

Because *E* is dependent on the weights and the target output, we can regulate the weights to minimize *E*. Widrow–Hoff learning rules are based on an approximate steepest descent procedure. Widrow and Hoff had the insight that they could estimate the mean square error by using the squared error at each iteration. If we take the partial derivative of the squared error with respect to the weights and biases at the *k*th iteration, we have

(2)

where *η* is the learning rate, generally 0 < *η* < 1.

If *η* is large, learning occurs quickly, but if it is too large it may lead to instability and errors may even increase. To ensure stable learning, the learning rate must be less than the reciprocal of the largest eigenvalue of the correlation matrix *X*T*X* of the input vectors. Thus, weight increase is

(3)

where 0 < *η* < 1.

To produce a faster convergence in the presence of random noise, a non-linear weight adaptation algorithm is desirable. Rewritten the weight adjustment algorithm as:

(4)

where

With

Widrow–Hoff learning rules make the change of the net weights have a direct proportion to the output error and the inputs of the adaline. This algorithm does not need to calculate the derivatives, so it can be computed simply and make the adaline converge fast. From Fig. 1 and Eq. (4), each iteration needs only 2N times multiplication and N + 5 times addition calculation (N, the number of the inputs). For example, if N = 10, then each iteration needs only 20 times multiplication and 15 times addition calculation. So it can be easily implemented through hardware and good at on-line using. But FFT approaches and wavelets need a great number of calculations. And FFT might not give the accurate time of occurrence without special amelioration.

**7.3. Back-propagation neural network**

The primary characteristics of ANNs are ability to learn, distributed memory and parallel operation, eventually leading to fault tolerance[12]. Thanks to these advantages, ANNs has been widely applied in natural science. Since the principle of ANNs has been well documented in the literature (Fausett, 1994; Bishop, 1995), only a brief outlook is given in this section.

A typical three-layered network with an input layer (I), a hidden layer (H) and an output layer (O), is adopted in this study. Each layer consists of several neurons and the layers are interconnected by sets of correlation weights. The neurons receive inputs from the initial inputs or the interconnections and produce outputs by transformation using an adequate nonlinear transfer function. A common transfer function is the sigmoid function expressed by *f*(*x*)=(1+*e*−*x*)−1, which has a characteristic of d*f*/d*x*=*f*(*x*)[1−*f*(*x*)].

The training processing of neural network is essentially executed through a series of patterns. In the learning process, the interconnection weights are adjusted within input and output value.

The BPN is the most representative learning model for the ANN. The procedure of the BPN is the error at the output layer that propagates backward to the input layer through the hidden layer in the network to obtain the final desired outputs. The gradient descent method is utilized to calculate the weight of the network and adjusts the weight of interconnections to minimize the output error. The error function at the output neuron is defined

(5)

where *Tk* and *Ok* are separately the values of target and output.

The gradient decent algorithm adapts the weights according to the gradient error, which is given by

(6)

where *η* is the learning rate, and the general form of the ∂*E*/∂*Wij* term is expressed by the following form (Rumelhart et al., 1986):

(7)

Substituting (7) into (6), we obtain the gradient error:

(8)

in which is the output value of sublayer related to the connective weight (*Wij*)**.***δjn* is the error signal, which is computed based on whether or not neuron *j* is in the output layer. If neuron *j* is one of the output neurons, then

*δj*=(*Tj*-*Yj*)·*Yj*·(1-*Yj*) (9)

If neuron *j* is the neuron of the hidden layer

(10)

where *Hh* is the value of hidden layer. Finally, the value of weight of interconnective neuron can be expressed as

(11)

To accelerate the convergence of the error in the learning procedure, the momentum term with the momentum gain, *α*, was included into the Eq. (9) (Jacobs, 1988).

(12)

in which the value for *α* is within 0 and 1. The main steps of this procedure are as follows:

Step 1: Initialize all weights to small random values within the range;

Step 2: Given the input vectors and output vectors;

Step 3: Compute the output values in a feedforward direction for each unit of each layer;

Step 4: Use the values computed by the final layer units and the corresponding target value to compute the delta quantities;

Step 5: Compute the deltas for each of the preceding layers by back propagating the errors;

Step 6: Update all weights;

Step 7: Return to step 2 and repeat for each pattern until the iteration has reached;

Step 8: Stop the procedure of training as the iteration has reached.

**7.4. Radial Basis Function Neural Network**

The general architecture of an RBF[13] neural network as illustrated in Fig.6, is a two-tiered mapping:

|  |
| --- |
| **Fig. 6 Structure of a typical RBF neural network** |

(13)

where *φ*(·) is a non-linear mapping from , the d-dimensional input space onto , an (h+1)-dimensional basis space, while the second, linear mapping from the basis space onto , the q-dimensional output space, is represented by the weight matrix *W*.

The non-linear mapping, *φ*(·), in Eq. (13) consists of a set of *h* radial basis functions, and a constant φ0=1 for the bias

*φ*(·)={*φ*0=1,*φ*1(·),…,*φ*h(·)}. (14)

When each of the radial basis functions in Eq. (14) represents a disjoint hyper-region of input space it can be shown that the response of the *j*th basis function is the posterior probability of an input originating from the region represented by the basis function, whereas the post basis weight wjk is the posterior probability of the *j*th basis function being associated with the class label of the *k*th output node [29]. The basis functions are commonly chosen to be Gaussians, where the *j*th basis function has the form

(15)

where *μj* is the prototype (mean) vector at the center of the basis function and Σ*j* is the covariance matrix characterizing the spread of data in the local neighborhood of *μj*. In practice, the means and covariances are unknown and need to be estimated from training data. For this, the input training data can be clustered, and each cluster can be represented by a basis function with covariance matrix and mean estimated from samples in the cluster. When only estimated covariance matrices are available, it has been found expedient to introduce a scale parameter *ωj* to control the spread of the basis function in input space. It is usually determined heuristically during training to avoid overlapping with other basis functions in its neighborhood. The basis functions with estimated parameters can then be written as

(16)

When the input dimensionality is larger than the number of samples in a cluster, the covariance matrix Σj will be singular. In such cases is approximated by a diagonal matrix containing estimated variances of individual features or is taken as the pseudo-inverse obtained by singular value decomposition (SVD)[14]. The simplest basis functions are hyper-spherical and have the form

(17)

**7.5. Competitive Learning Neural Network**

Competitive learning[15] is employed to cluster a set of input patterns without supervision. The neurons in the competitive layer learn to recognize different regions of the input space where input vectors occur. The ANN has a single layer of output neurons which are fully connected to the input nodes. All the neurons compete with each other to win the competition for a given input vector. A neuron wins if it has the largest internal activity, *μk*, for a specific input pattern among all the other neurons. The winning neurons’ output is set to 1 while all the neuron outputs are set to 0. The winning neuron learns by moving its weight vector in the direction of the input vector, while the losing neurons do not learn on this input pattern. The standard competitive learning rule is given by

(18)

where *η* is a learning rate parameter.

**7.6. Learning Vector Quantization Neural Network**

LVQ network is a supervised version of vector qua

|  |
| --- |
| **Fig.7 Topology of the LVQ network** |

ntization[16]. As shown in Fig. 7, the LVQ network has three layers: the input layer, the competitive layer of the hidden layer, and the output layer. The neurons in the competitive layer are divided into *n* groups. Each group has the same number of neurons and corresponds to an output layer neuron. Classification information is stored in the weight matrix connecting between the input layer neurons and the competitive layer neurons.

LVQ forms a quantized approximation of the distribution of an input data set using a finite number of reference vectors. The LVQ algorithm belongs to a class of signal approximation methods that model the probability density function *f*(*x*), of some stochastic variable , using a finite set of codebook vectors, , i=1,2,…,*k*, where the subscript *i* represents the hypothesis index in detection techniques. Once a set of codebook vectors are determined, the approximation of *X* implies finding the codebook *mc* closest to *X* in the input space for a given distance metric (typically the *Lp* space, with *p*=1,2,…,∞). The determination of *mc* is achieved by the following decision process:

(19)

That is

(20)

An optimal selection of *mi* minimizes the average expected square of the quantization error, defined below:

(21)

The codebook vectors are recursively updated by minimizing Eq. (21) as follows:

(22)

(23)

where, *α*, 0≤*α*(*t*) ≤1, is a learning rate, is the complement set of Sc.

The adjustment of the learning rate follows the Kohonen rule,

(24)

Here, *ξ*=1 if the classification is correct and *ξ*=−1 if the pattern is misclassified.

**7.7. Elman Neural Network**

The learning of the Elman network[17] has been realized using backpropagation (BP) algorithm, which is a widely used algorithm and can map nonlinear processes. This algorithm is a systematic method for training multilayer neural networks. It has a strong mathematical foundation based on gradient descent learning. Furthermore, it is a most widely used algorithm in the training of traditional neural networks. However, the BP algorithm is also used in the new generation neural network models, e.g. the Elman network can be trained using the BP learning algorithm.

**7.8. Hopfield Neural Network**

|  |
| --- |
| **Fig.8 Architecture of Hopfield neural network for optimization** |

Hopfield Neural Network (HNN)[18] is a type of recurrent network that operates in an unsupervised manner. The HNNs have three major forms of parallel organisations found in neural systems, namely (i) parallel input channels, (ii) parallel output channels, and (iii) a large amount of interconnectivity between the neural processing elements. The architecture of HNN is as shown in Fig. 8.

The processing elements are modeled as amplifiers in conjunction with feedback circuits[19]. The amplifiers have sigmoidal monotonic input–output relations. For these symmetrically connected neurons, an energy function can be defined which is specific to a particular connection. The action of HNN is to minimize this energy function. If the cost function or performance index of an optimization problem can be mapped on to this energy function, then the network will converge to an optimal solution. Basically there are two types of HNN—(i) the discrete type, and (ii) the continuous type.

**7.9. Boltzmann Neural Network**

The Boltzmann neural network is a discrete-time model having the same structure as the binary Hopfield model[20]. The global state of the Boltzmann neural network fluctuates randomly, and its probability distribution can be made to approximate any desired probability distribution by a simple learning algorithm. Owing to this property, the Boltzmann neural network can be used as a variable source of random fluctuation. The simplicity and elegance of the learning algorithm make the Boltzmann neural network a favorable model from both theoretical and applicational viewpoint. However, the Boltzmann neural network has the same disadvantage as the serial Hopfield model. That is, neurons are assumed to operate one-by-one in the Boltzmann neural network. We appreciate the simplicity of the learning algorithm but consequently must accept its serial dynamics. In fact, the discrete-time parallel Boltzmann neural network has a different probabilistic character, and is accompanied by a different learning algorithm.

**8. Application**

A few of the recent applications of neural networks are given below to illustrate the wide spectrum of applications to which neural networks have been applied.

Complex system modeling; a system with multiple inputs and outputs can be modeled using a neural network by applying the system inputs to the network and using the system outputs as the desired outputs of the neural network.

Image (data) compression[21] involves the transforming of image data to a different representation that requires less memory.

Character recognition, a special case of pattern recognition, is the process of visually interpreting and classifying symbols.

Target classification; neural networks have been used to classify sonar targets by distinguishing between large metal cylinders and rocks of a similar size.

Noise filtering; neural networks are able to filter noisy data and preserve a greater depth of structure and detail than any of the traditional filters while still removing the noise.

Servo-control systems; complex mechanical servo-systems, such as those used in robots, must compensate for physical variations in the system introduced by misalignments in the axes, or deviation in members due to bending and stretching induced by loads. These quantities are extremely difficult to describe analytically. A neural network can be trained to predict and respond to these errors in the final position of a robot member. This information is then combined with the desired position to provide an adaptive position correction and improve the accuracy of the member’s position.

Text-to-speech conversion; in this application the printed symbols or letters in a text were converted into the spoken language using a neural network that taught itself to translate written text into speech in the same way that a human child learns to read.

**Reference**

[1] Artificial Neural Networks. Understanding neural networks and fuzzy logic: basic concepts and applications [M], Washington, DC, USA: Wiley-IEEE Press, September 1995: 46–52.

[2] J.H. Garrett. Where and why artificial neural networks are applicable in civil engineering [J], J Comp Civ Engng [special issue], 1994, 82, 129–130.

[3] R.E. Uhrig, Introduction to artificial neural networks [A], Proceedings of the 1995 IEEE IECON 21st International Conference [C], Orlando, USA: ISS, 1995. pp: 33-37.

[4] Samuel H. Huang, Hong-Chao Zhang. Artificial neural networks in manufacturing: concepts, applications, and perspectives [J], IEEE transactions on components, packaging, and manufacturing technology, 1994, 17(2): 212-228.

[5] Vemuri, V. Artificial Neural Networks[M]. Washington, DC, USA: IEEE computer society, 1990.

[6] N. B. Karayiannis, A. N. Venetsanopoulos. Artificial Neural Networks [M], Dordrecht, Netherlands: Kluwer Academic Publisher, 1993.

[7] L.C. Jain, Recent advances in artificial neural networks [J]. Boca Raton, USA: CRC Press, 2000.

[8] M.W.Gardner, S.R.Dorling. Artificial neural networks (the multilayer perceptron)-a review of applications in the atmospheric sciences [J], Atmospheric Environment, 1998,32(14):2627-2636.

[9] B. Widrow, M.E. Hoff, Adaptive switching circuits, WESTCON Convention, Record Part IV, 1960, pp. 96–104.

[10] B. Widrow and M.A. Lehr, 30 years of adaptive neural networks: perceptrons, adaline and back propagation [J], IEEE Proc. 1990, 78(9), pp. 1415–1442.

[11] Qian Ai，Yuguang Zhou, Weihua Xu. Adaline and its application in power quality disturbances detection and frequency tracking[J], Electric Power Systems Research, 2007, 77:462-469.

[12] Tsung-Lin Lee. Back-propagation neural network for the prediction of the short-term storm surge in Taichung harbor, Taiwan [J], Engineering Applications of Artificial Intelligence, 2008, 21(1):63-72.

[13] Chathura R. De Silva, Surendra Ranganath，Liyanage C. De Silva. Cloud basis function neural network: A modified RBF network architecture for holistic facial expression recognition [J]. Pattern Recognition, 2008, 41:1241-1253.

[14] Ralf Wieland, Wilfried Mirschel, Adaptive fuzzy modeling versus artificial neural networks [J]. Environmenral modelling & software, 2008, 23: 215-244.

[15] Imran Khan, Howard C.Card. Adaptive information agents using competitive learning [J]. Journal of Network and Computer Applications, 1998, 21:69-89.

[16] A.M.Bassiuny, Xiaoli Li, R.Du. Fault diagnosis of stamping process based on empirical mode decomposition and learning vector quantization [J