IT-802 (SOFT COMPUTING)
BE VIII SEMESTER

Q.1 (a) State the various learning rules in network?

Ans: A neuron is considered to be an adaptive element. Its weights are modifiable depending on the input signal it receives, its output value, and the associated teacher response. In some cases the teacher signal is not available and no error information can be used, thus a neuron will modify its weights, based only on the input and/or output. There are many types of Neural Network Learning Rules

Hebbian Learning Rule: Hebb's rule is a postulate proposed by Donald Hebb in 1949. It is a learning rule that describes how the neuronal activities influence the connection between neurons, i.e., the synaptic plasticity. It provides an algorithm to update weight of neuronal connection within neural network. Hebb's rule provides a simplistic physiology-based model to mimic the activity dependent features of synaptic plasticity and has been widely used in the area of artificial neural network. Different versions of the rule have been proposed to make the updating rule more realistic.

The weight of connection between neurons is a function of the neuronal activity. The classical Hebb's rule indicates "neurons that fire together, wire together". In the simplest form of Hebb's rule, Eq. (1), *wij* stands for the weight of the connection from neuron j to neuron i

$$wij = xi\ xj \quad \text{-----(1)}$$

It was invented in 1943 by neurophysiologist Warren McCulloch and logician Walter Pitts. Now networks of the McCulloch-Pitts type tend to be overlooked in favour of "gradient descent" type neural networks and this is a shame.

Delta Rule: Developed by Widrow and Hoff, the delta rule, also called the Least Mean Square (LMS) method, is one of the most commonly used learning rules. For a given input vector, the output vector is compared to the correct answer. If the difference is zero, no learning takes place; otherwise, the weights are adjusted to reduce this difference. The change in weight from ui to uj is given by: dwij = r* ai * ej, where r is the learning rate, ai represents the activation of ui and ej is the difference between the expected output and the actual output of uj. If the set of input patterns form a linearly independent set then arbitrary associations can be learned using the delta rule. It has been shown that for networks with linear activation functions and with no hidden units hidden units are found in networks with more than two layers, the error squared vs. the weight graph is a paraboloid in n-space. Since the proportionality constant is negative, the graph of such a function is concave upward and has a minimum value. The vertex of this paraboloid represents the point where the error is minimized. The weight vector corresponding to this point is then the ideal weight vector. This learning rule not only moves the weight vector nearer to the ideal weight vector, it does so in the most efficient way. The delta rule implements a gradient descent by moving the weight vector from the point on the surface of the paraboloid down toward the lowest point, the vertex. There is no such powerful rule as the delat rule for networks with hidden units. There have been a number of theories in response to this problem. These include the generalized delta rule and the unsupervised competitive learning model.

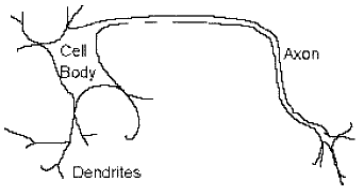
The Generalized Delta Rule: A generalized form of the delta rule, developed by D.E. Rumelhart, G.E. Hinton, and R.J. Williams, is needed for networks with hidden layers. They showed that this method works for the class of semi-linear activation functions non-decreasing and differentiable.

Generalizing the ideas of the delta rule, consider a hierarchical network with an input layer, an output layer and a number of hidden layers. We will consider only the case where there is one hidden layer. The network is presented with input signals which produce output signals that act as input to the middle layer. Output signals from the middle layer in turn act as input to the output layer to produce the final output vector. This vector is compared to the desired output vector. Since both the output and the desired output vectors are known, the delta rule can be used to adjust the weights in the output layer. Both the input signal to each unit of the middle layer and the output signal are known. What is not known is the error generated from the output of the middle layer since we do not know the desired output. To get this error, back propagate through the middle layer to the units that are responsible for generating that output. The error generated from the middle layer could be used with the delta rule to adjust the weights.

(b) Explain the difference between a mathematical simulation of biological neural system and an artificial neural network?

Ans: An artificial neural network (ANN), usually called neural network (NN), is a mathematical model or computational model that is inspired by the structure and/or functional aspects of biological neural networks. A neural network consists of an interconnected group of artificial neurons, and it processes information using a connectionist approach to computation. In most cases an ANN is an adaptive system that changes its structure based on external or internal information that flows through the network during the learning phase. Modern neural networks are non-linear statistical data modeling tools. They are usually used to model complex relationships between inputs and outputs or to find patterns in data.

One type of network sees the nodes as 'artificial neurons'. These are called artificial neural networks (ANNs). An artificial neuron is a computational model inspired in the natural neurons. Natural neurons receive signals through *synapses* located on the dendrites or membrane of the neuron. When the signals received are strong enough (surpass a certain *threshold*), the neuron is *activated* and emits a signal though the *axon*. This signal might be sent to another synapse, and might activate other neurons.



The biological neuron has four main regions to its structure. The cell body, or soma, has two offshoots from it. The dendrites and the axon end in pre-synaptic terminals. The cell body is the heart of the cell. It contains the nucleolus and maintains protein synthesis. A neuron has many dendrites, which look like a tree structure, receives signals from other neurons.

A single neuron usually has one axon, which expands off from a part of the cell body. This I called the axon hillock. The axon main purpose is to conduct electrical signals generated at the axon hillock down its length. These signals are called action potentials.The other end of the axon may split into several branches, which end in a pre-synaptic terminal. The electrical signals (action potential) that the neurons use to convey the information of the brain are all identical. The brain can determine which type of information is being received based on the path of the signal.

The brain analyzes all patterns of signals sent, and from that information it interprets the type of information received. .The synapse is the area of contact between two neurons. They do not physically touch because they are separated by a cleft. The electric signals are sent through chemical interaction. The neuron sending the signal is called pre-synaptic cell and the neuron receiving the electrical signal is called postsynaptic cell. The electrical signals are generated by the membrane potential which is based on differences in concentration of sodium and potassium ions and outside the cell membrane. Once modeling an artificial functional model from the biological neuron, we must take into account three basic components. First off, the synapses of the biological neuron are modeled as weights. Let's remember that the synapse of the biological neuron is the one which interconnects the neural network and gives the strength of the connection. For an artificial neuron, the weight is a number, and represents the synapse. A negative weight reflects an inhibitory connection, while positive values designate excitatory connections. The following components of the model represent the actual activity of the neuron cell. All inputs are summed altogether and modified by the weights. This activity is referred as a linear combination. Finally, an activation function controls the amplitude of the output. For example, an acceptable range of output is usually between 0 and 1, or it could be -1 and 1.Mathematically, this process is described in the figure

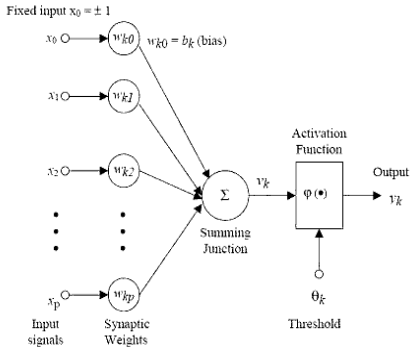


Fig. Mathematical Model Of ANN

From this model the interval activity of the neuron can be shown to be:

$$v_k = \sum_{j=1}^{p} w_{kj} x_j$$

The output of the neuron, yk, would therefore be the outcome of some activation function on the value of vk.

Q.2 (a) Explain the following:

(i) Supervised learning

(ii) Incremental learning

(iii) Unsupervised leaning

Ans: (i) Supervised Learning: Supervised learning is a machine learning technique that sets parameters of an artificial neural network from training data. The task of the learning artificial neural network is to set the value of its parameters for any valid input value after having seen

output value. The training data consist of pairs of input and desired output values that are traditionally represented in data vectors. Supervised learning can also be referred as classification, where we have a wide range of classifiers, each with its strengths and weaknesses. In order to solve a given problem of supervised learning various steps has to be considered. In the first step we have to determine the type of training examples. In the second step we need to gather a training data set that satisfactory describe a given problem. In the third step we need to describe gathered training data set in form understandable to a chosen artificial neural network. In the fourth step we do the learning and after the learning we can test the performance of learned artificial neural network with the test validation data set. Test data set consist of data that has not been introduced to artificial neural network while learning.

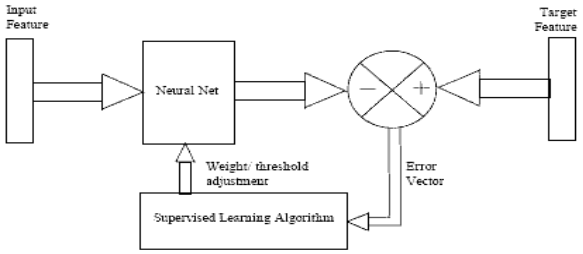


Fig. Supervised Learning

(ii) Incremental Learning:Incremental learning is the fastest and the most comprehensive way of learning available to students at the moment of writing. Incremental learning is a consolidation of computer-based techniques that accelerate and optimize the process of learning from all conceivable material available in electronic form, and not only. Currently, Super Memo is the only software that implements incremental learning. In Super Memo, the student feeds the program with all forms of learning material and/or data (texts, pictures, videos, sounds, etc.). Those learning materials are then gradually converted into durable knowledge that can last a lifetime. In incremental learning, the student usually remembers 95% of his or her top priority material. That knowledge is relatively stable and lasts in student's memory as long as the process continues, and well beyond. Incremental learning tools differ substantially for various forms of learning material, media, and goals. Here are the main components of incremental learning:

(iii)Unsupervised learning: Unsupervised learning or Self-organisation in which an (output) unit is trained to respond to clusters of pattern within the input. In this paradigm the system is supposed to discover statistically salient features of the input population. Unlike the supervised learning paradigm, there is no a priori set of categories into which the patterns are to be classified; rather the system must develop its own representation of the input stimuli. Unsupervised learning seems much harder: the goal is to have the computer learn how to do something that we don't tell it how to do! There are actually two approaches to unsupervised learning. The first approach is to teach the agent not by giving explicit categorizations, but by using some sort of reward system to indicate success. Note that this type of training will generally fit into the decision problem framework because the goal is not to produce a classification but to make decisions that maximize rewards. This approach nicely generalizes to the real world, where agents might be rewarded for doing certain actions and punished for doing others. Unfortunately, even unsupervised learning suffers from the problem of over fitting the training data. There's no silver bullet to avoiding the problem because any algorithm that can learn from its inputs needs to be quite powerful.

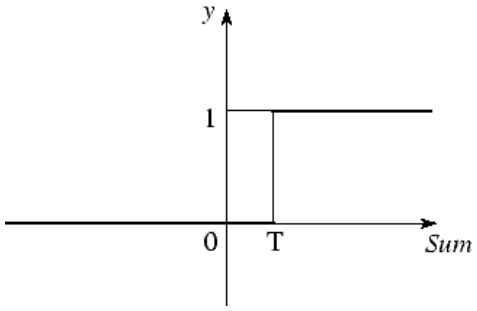
(b) Explain Mcculloch, Pitt's neuron model with example.

Ans: The early model of an artificial neuron is introduced by Warren McCulloch and Walter Pitts in 1943. The McCulloch-Pitts neural model is also known as linear threshold gate. It is a neuron of a set of inputs $I_1, I_2, \ldots I_n$ and one output y . The linear threshold gate simply classifies the set of inputs into two different classes. Thus the output y is binary. Such a function can be described mathematically using these equations:

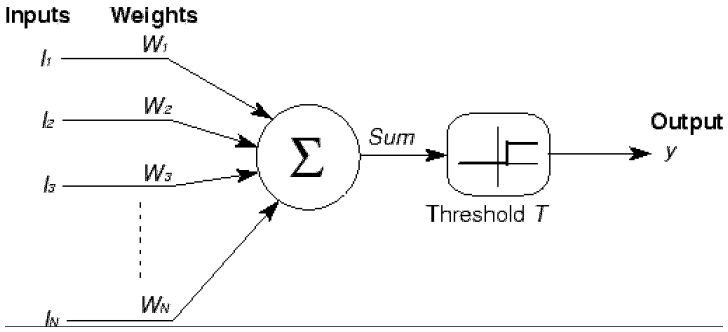
$$Sum = \sum_{i=1}^{N} I_i W_i, \tag{2.1}$$

$$Y = f(Sum) \tag{2.2}$$

$W_1, W_2, \ldots W_n$ are weight values normalized in the range of either (0,1) or (-1,1) and associated with each input line, Sum is the weighted sum, and T is a threshold constant. The function f is a linear step function at threshold T as shown in fig. The symbolic representation of the linear threshold gate is shown in fig.



Linear Threshold Function



Symbolic Illustration of Linear Threshold Gate

The McCulloch-Pitts model of a neuron is simple yet has substantial computing potential. It also has a precise mathematical definition. However, this model is so simplistic that it only generates a binary output and also the weight and threshold values are fixed. The neural computing algorithm has diverse features for various applications. Thus, we need to obtain the neural model with more flexible computational features.

The interesting thing about McCulloch-Pitts model of a neural network is that it can be used as the components of computer-like systems. The basic idea of a McCulloch-Pitts model is to use components which have some of the characteristics of real neurons. A real neuron has a number of inputs which are "excitatory" and some which are "inhibitory". What the neuron does depends on the sum of inputs. The excitatory inputs tend to make the cell fire and the inhibitory inputs make is not fire – i.e. pass on the signal.

Q.3 (a) Explain the back propagation algorithm and derive the expression for weight update relations.

Ans : Back-propagation: Backpropagation is a common method of training artificial neural networks so as to minimize the objective function. Arthur E. Bryson and Yu-Chi Ho described it as a multi-stage dynamic system optimization method in 1969. It wasn't until 1974 and later, when applied in the context of neural networks and through the work of Paul Werbos,[3] David E. Rumelhart, Geoffrey E. Hinton and Ronald J. Williams,[4][5] that it gained recognition, and it led to a "renaissance" in the field of artificial neural network research.

It is a supervised learning method, and is a generalization of the delta rule. It requires a dataset of the desired output for many inputs, making up the training set. It is most useful for feed-forward networks (networks that have no feedback, or simply, that have no connections that loop). The term is an abbreviation for "backward propagation of errors". Backpropagation requires that the activation function used by the artificial neurons (or "nodes") be differentiable.

For better understanding, the backpropagation learning algorithm can be divided into two phases: propagation and weight update.

Phase 1: Propagation

Each propagation involves the following steps:

1. Forward propagation of a training pattern's input through the neural network in order to generate the propagation's output activations.

2. Backward propagation of the propagation's output activations through the neural network using the training pattern's target in order to generate the deltas of all output and hidden neurons.

Phase 2: Weight update

For each weight-synapse follow the following steps:

1. Multiply its output delta and input activation to get the gradient of the weight.

2. Bring the weight in the opposite direction of the gradient by subtracting a ratio of it from the weight.

This ratio influences the speed and quality of learning; it is called the *learning rate*. The sign of the gradient of a weight indicates where the error is increasing, this is why the weight must be updated in the opposite direction.

Repeat phase 1 and 2 until the performance of the network is satisfactory.

First we can write total error as a sum of the errors at each node k:

$E = \Sigma_k \; E_k$

where $E_k = 1/2 \; (y_k - O_k)^2$

Now note that y_k, x_k and w_{jk} each only affect the error at one particular output node k (they only affect E_k). So from the point of view of these 3 variables, total error:

E = (a constant) + (error at node k)

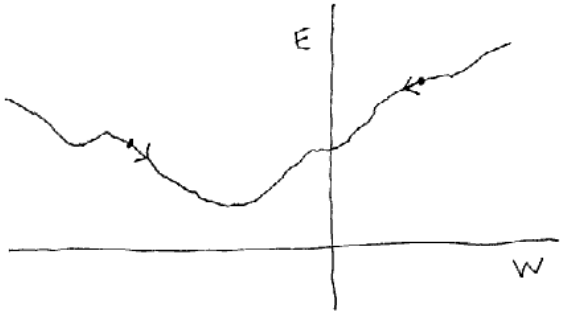
hence: (derivative of total error E with respect to any of these 3 variables) = 0 + (derivative of error at node k)

e.g.

$\partial E/\partial y_k = 0 + \partial E_k/\partial y_k$

We can see how the error changes as y_k changes, or as x_k changes. But note we *can't* change y_k or x_k - at least not directly. They follow in a predetermined way from the previous inputs and weights.
But we *can* change w_{jk} as we work backwards, the situation changes. y_j feeds forward into *all* of the output nodes. Since: E = (sum of errors at k)
we get: (derivative of E) = (sum of derivatives of error at k) x_j and w_{ij} then *only* affect y_j (though y_j affects many things). We *can't* (directly) change y_j or x_j But we *can* change w_{ij}

Changing the weights to reduce the error: Now we have an equation for each $\frac{\delta E}{\delta w}$ - how error changes as you change the weight.



Now, to reduce error:

On RHS, slope is positive: $\frac{\delta E}{\delta w} > 0$. Move left to reduce error:

$W := W - C \frac{\delta E}{\delta w}$

where C is a positive constant.

On LHS, slope is negative: $\frac{\delta E}{\delta w} < 0$. Move right to reduce error:

$W := W - C \frac{\delta E}{\delta w}$

= W - C (negative quantity)
= W + (positive quantity)

Hence the same update rule works for both positive and negative slopes:

$W := W - C \frac{\delta E}{\delta w}$

The constant C is the LEARNING RATE $0 < C < 1$.

Limitations

- The convergence obtained from backpropagation learning is very slow.
- The convergence in backpropagation learning is not guaranteed.

Q.3 (b) Give the comparison between the radial basis function networks and multi layer perception.

Ans: A radial basis function network is an artificial neural network that uses radial basis functions as activation functions. The output of the network is a linear combination of radial basis functions of the inputs and neuron parameters. Radial basis function networks have many uses, including approximation, time, classification, and system control. Radial basis function (RBF) networks typically have three layers: an input layer, a hidden layer with a non-linear RBF activation function and a linear output layer. The input can be modeled as a vector of real numbers $\mathbf{x} \in \mathbb{R}^n$. The output of the network is then a scalar function of the input vector, $\varphi : \mathbb{R}^n \to \mathbb{R}$.

A multilayer perceptron (MLP) is a feed forward network model that maps sets of input data onto a set of appropriate outputs. A MLP consists of multiple layers of nodes in a directed graph, with each layer fully connected to the next one. Except for the input nodes, each node is a neuron (or processing element) with a nonlinear activation function. MLP utilizes a supervised technique called back propagation for training the network. MLP is a modification of the standard linear perceptron and can distinguish data that are not linearly separable.

Q.4 (a) Discuss the application of neural in forecasting.

Ans: Artificial Neural Networks have become objects of everyday use. Superior performance in optical character recognition, speech recognition, signal filtering in computer modems etc. have established NN as an accepted model & method. However, neural networks have not yet been established as a valid and reliable method in the business forecasting domain, either on a strategic, tactical or operational level. Following we present selected applications of NN which have demonstrated their applicability in specific scenarios. Considering forecasting objectives, we must differentiate between predictive classification tasks where the forecasted values are class memberships or the probabilities of belonging to certain class, i.e. binary predictors, and regression tasks, where the forecasted value represent a single number of metric scale, e.g. sales, temperature, rainfall etc., as in regression problems. Following we will refer to this as forecasting as opposed to classification. In forecasting applications, many classification problems also encompass the "prediction" of a categorical or nominal scaled variable. In order to distinguish the distinct modelling approaches and pre-processing required, we consider forecasting applications where the predictor is of metric scale being regression or point prediction based (in the demand planning area simply denoted as sales or demand forecasting). Consequently, a rise / fall-predictions as in financial modelling of buy-hold-strategies would receive consideration as under classification tasks due to their nominal predictors.

(b) Explain Adeline and Madeline briefly.

Ans: ADALINE (Adaptive Linear Neuron or later Adaptive Linear Element) is a single layer neural network. It was developed by Professor Bernard Widrow and his graduate student Ted Hoff at Stanford University in 1960. It is based on the McCulloch–Pitts neuron. It consists of a weight, a bias and a summation function.

The difference between Adaline and the standard (McCulloch–Pitts) perceptron is that in the learning phase the weights are adjusted according to the weighted sum of the inputs (the net). In the standard perceptron, the net is passed to the activation (transfer) function and the function's output is used for adjusting the weights.

There also exists an extension known as Madaline. Adaline is a single layer neural network with multiple nodes where each node accepts multiple inputs and generates one output. Given the following variables:

- x is the input vector
- w is the weight vector
- n is the number of inputs
- θ some constant
- y is the output

then we find that the output is $y = \sum_{j=1}^{n} x_j w_j + \theta$. If we further assume that

- $x_{n+1} = 1$
- $w_{n+1} = \theta$

then the output reduces to the dot product of x and w $y = x_j.w_j$.

Madaline (Multiple Adaline) is a two layer neural network with a set of ADALINEs in parallel as its input layer and a single PE (processing element) in its output layer. For problems with multiple input variables and one output, each input is applied to one Adaline. For similar problems with multiple outputs, madalines in parallel can be used. The madaline network is useful for problems which involve prediction based on multiple inputs, such as weather forecasting (Input variables: barometric pressure, difference in pressure. Output variables: rain, cloudy, sunny).

A MADALINE (many Adaline) network is created by combining a number of Adalines. The network of ADALINES can span many layers. The use of Multiple Adalines help counter the problem of non linear separability. For example, the MADALINE network with two units exhibits the capability to solve the XOR problem**.**

Q.5 (a) what are the advantages of ART network? Discuss about the gain control in ART network.

Ans: It solves the stability- plasticity problem. It is Capable of truly autonomous learning (unsupervised). It continues to learn new information until memory is exhausted and continually refines existing knowledge. Addition of new patterns does not affect already learned information or the speed of the classification process. It does not suffer from local minima problems. A major drawback is the ability to handle only binary inputs which restricts the number of distinct input patterns (2^n) and requires preprocessing of real-world signals.

The ART Algorithm

1. **Initialize**

$t_{ij}(0) = 1$

$b_{ij}(0) = 1/(1 + N)$

$0 \le i \le N - 1$

$0 \le j \le M - 1$

set ρ, $0 \le \rho \le N - 1$

Where $b_{ij}(t)$ is the bottom-up and $t_{ij}(t)$ is the top-down connection weights between input node i and output node j at time t. The fraction ρ is the vigilance threshold which indicates how close an input must be to a stored exemplar to match.

2. **Apply new input**
3. **Compute matching scores**

$$\mu_j = \sum_{i=0}^{N-1} b_{ij}(t)x_i \qquad 0 \le j \le M-1$$

where μ_j is the output of output node j and x_i is element i of the input which can be 0 or 1.

4. **Select best matching exemplar**

$$\mu_j^* = max_j \{\mu_j\}$$

5. **Test**
6. **Disable best matching exemplar**

The output of the best matching node selected in step 4 is temporarily set to zero and no longer takes part in the maximization of step 4.

7. **Adapt best matching exemplar**

$$t_{ij}^*(t+1) = t_{ij}^*(t)x_i$$

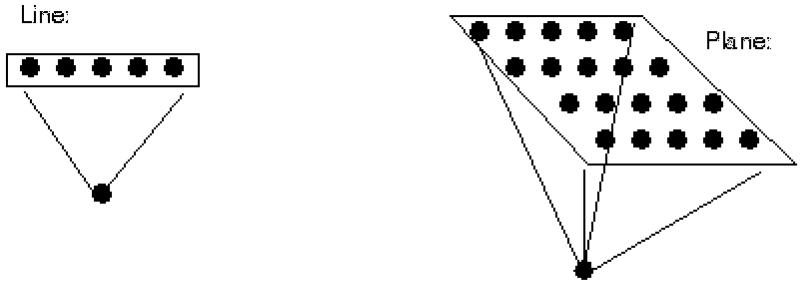
$$b_{ij}^*(t+1) = t_{ij}^*(t)x_i/(0.5 + \Sigma i^{N-1} t_{ij}^*(t)\, x_i)$$

8. Repeat

Enable any disabled node, then go to step 2

Q.5 (b) Explain the architecture and training of Kohonen self organizing network.

Kohonon's SOMs are a type of unsupervised learning. The goal is to discover some underlying structure of the data. However, the kind of structure we are looking for is very different than, say, PCA or vector quantization. Kohonen's SOM is called a topology-preserving map because there is a topological structure imposed on the nodes in the network. A topological map is simply a mapping that preserves neighborhood relations. Each node in a given layer has been identical in that each is connected with all of the nodes in the upper and/or lower layer. We are now going to take into consideration that physical arrangement of these nodes. Nodes that are "close" together are going to interact differently than nodes that are "far" apart.



The goal is to train the net so that nearby outputs correspond to nearby inputs. In the brain, neurons tend to cluster in groups. The connections within the group are much greater than the connections with the neurons outside of the group. Assume output nodes are connected in an array (usually 1 or 2 dimensional)

- Assume that the network is fully connected - all nodes in input layer are connected to all nodes in output layer.
- Use the competitive learning algorithm as follows:

- Randomly choose an input vector x
- Determine the "winning" output node i, where w_i is the weight vector connecting the inputs to output node i.
- Note: the above equation is equivalent to $w_i x >= w_k x$ only if the weights are normalized.

$$\left|\omega_i - x\right| \leq \left|\omega_k - x\right| \qquad \forall k$$

- Given the winning node i, the weight update is

$$w_k(new) = w_k(old) + \mu \aleph(i, k)(x - w_k)$$

Q.6 (a) Discuss the application of neural networks in Robotic vision?

Ans: In machine learning, artificial neural networks (ANNs) are a family of statistical learning algorithms inspired by biological neural networks (the central nervous systems of animals, in particular the brain) and are used to estimate or approximate functions that can depend on a large number of inputs and are generally unknown. Artificial neural networks are generally presented as systems of interconnected "neurons" which can compute values from inputs, and are capable of machine learning as well as recognition thanks to their adaptive nature. Like other machine learning methods - systems that learn from data - neural networks have been used to solve a wide variety of tasks that are hard to solve using ordinary rule-based programming, including computer vision and speech recognition. The utility of artificial neural network models lies in the fact that they can be used to infer a function from observations. This is particularly useful in applications where the complexity of the data or task makes the design of such a function by hand impractical.

Real-life applications

The tasks artificial neural networks are applied to tend to fall within the following broad categories:

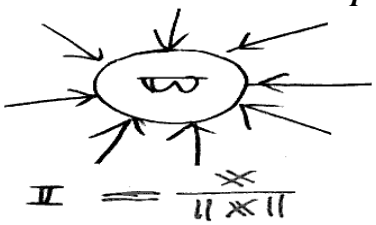
- Function approximation, or regression analysis, including time series prediction, fitness approximation and modeling.
- Classification, including pattern and sequence recognition, novelty detection and sequential decision making.
- Data processing, including filtering, clustering, blind source separation and compression.
- Robotics, including directing manipulators, prosthesis.
- Control, including Computer numerical control.

Ques 6 b) Draw the architecture of Full CPN and explain how CPN nets are used for function approximation.

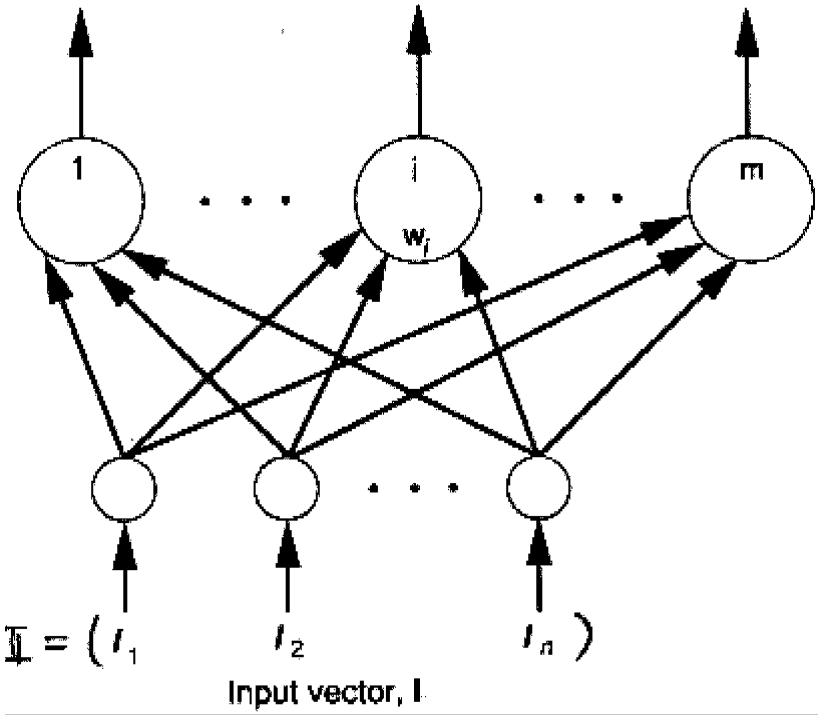
Ans: The counter propagation network is a hybrid network. It is guaranteed to find the correct weights, unlike regular back propagation networks that can become trapped in local minimums during training. The input layer neurode connect to each neurode in the hidden layer. The hidden layer is a Kohonen network which categorizes the pattern that was input. The output layer is an outstar array which reproduces the correct output pattern for the category. Three major components of CPN

1. Instar: hidden node with input weights
2. Competitive layer: hidden layer composed of instars
3. Outstar: a structure

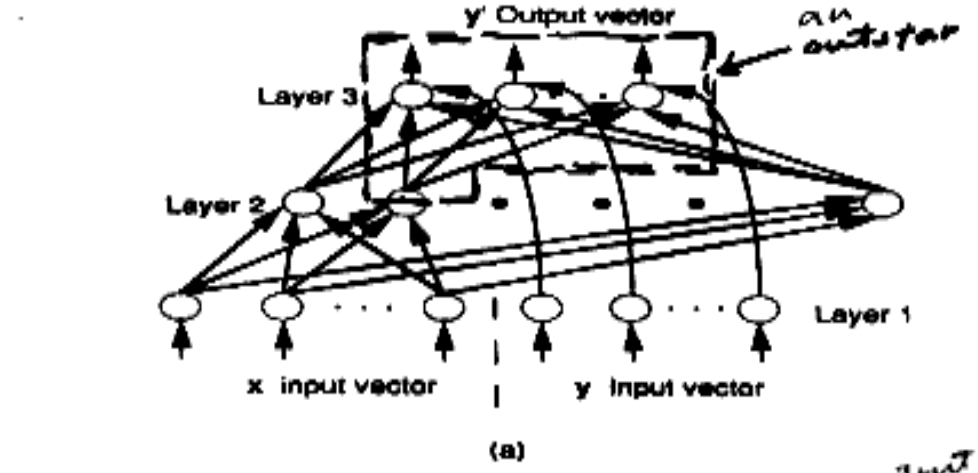
Instar: hidden node with input weights



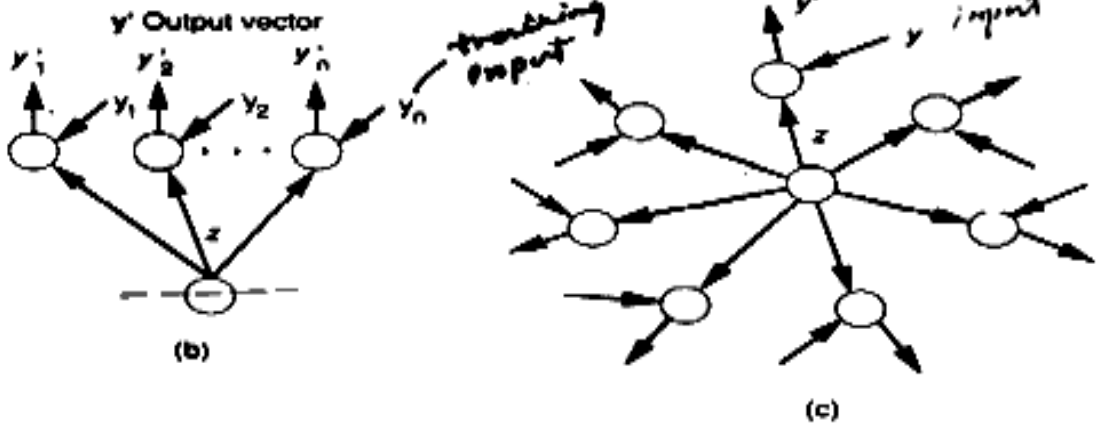
The hidden layer is a competitive layer formed by instars



Outstar is composed of a single hidden unit and all output units



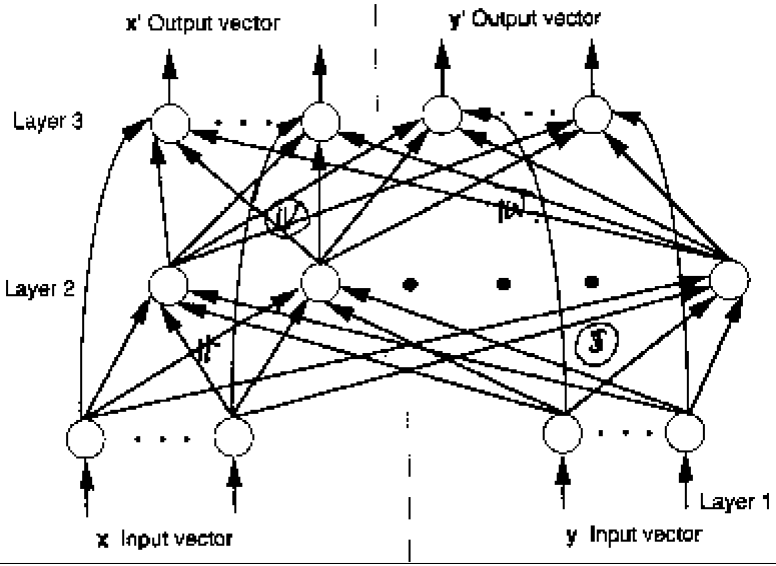
(a)



(b)

(c)

The Full CPN: $\begin{cases} \text{Forward mapping} & x \rightarrow y' \\ \text{Reverse mapping} & y \rightarrow x' \end{cases}$



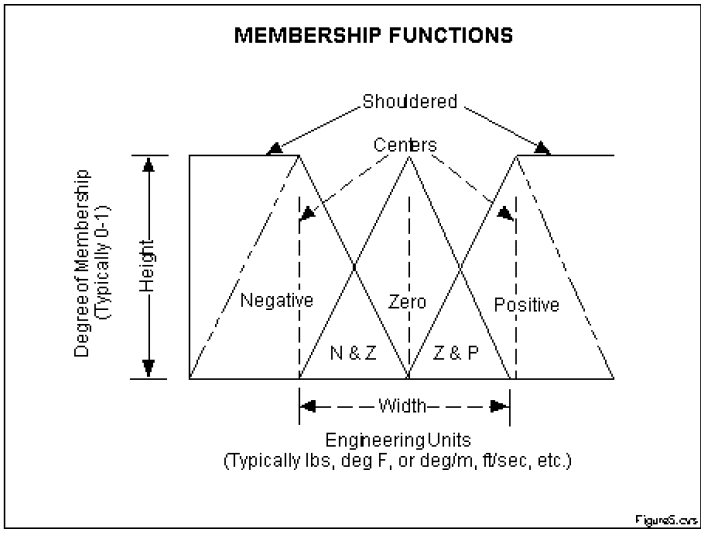
A counter propagation network architecture for continuous function approximation is done. The paradigm consists of a splitting Kohonen layer architecture, functional-link network, continuous activation functions, and a modified training procedure. The network mapping capabilities are

analyzed. Improved function approximation accuracy compared to standard counter propagation networks.

Ques 7 a) Define Membership function and discuss its importance in fuzzy logic.

Fuzzy Logic (FL) is a problem-solving control system methodology that lends itself to implementation in systems ranging from simple, small, embedded micro-controllers to large, networked, multi-channel PC or workstation-based data acquisition and control systems. It can be implemented in hardware, software, or a combination of both. It requires some numerical parameters in order to operate such as what is considered significant error and significant rate-of-change-of-error, but exact values of these numbers are usually not critical unless very responsive performance is required in which case empirical tuning would determine them. The membership function is a graphical representation of the magnitude of participation of each input. It associates a weighting with each of the inputs that are processed, define functional overlap between inputs, and ultimately determines an output response.

The rules use the input membership values as weighting factors to determine their influence on the fuzzy output sets of the final output conclusion. Once the functions are inferred, scaled, and combined, they are defuzzified into a crisp output which drives the system. There are different membership functions associated with each input and output response. The degree of membership (DOM) is determined by plugging the selected input parameter (error or error-dot) into the horizontal axis and projecting vertically to the upper boundary of the membership function(s). There is a unique membership function associated with each input parameter. The membership functions associate a weighting factor with values of each input and the effective rules. These weighting factors determine the degree of influence or degree of membership (DOM) each active rule has. By computing the logical product of the membership weights for each active rule, a set of fuzzy output response magnitudes are produced. All that remains is to combine and defuzzify these output responses. A firing strength for each output membership function is computed. All that remains is to combine these logical sums in a defuzzification process to produce the crisp output.



Ques 7 b) Explain the sugeno fuzzy inference model with a suitable example.

Ans: Fuzzy inference is the process of formulating the mapping from a given input to an output using fuzzy logic. The mapping then provides a basis from which decisions can be made, or

patterns discerned. The process of fuzzy inference involves all of the pieces that are described in Membership Functions, Logical Operations, and If-Then Rules.
The **Sugeno Fuzzy model** (also known as the *TSK fuzzy model*) was proposed by Takagi, Sugeno, and Kang in an effort to develop a systematic approach to generating fuzzy rules from a given input-output dataset. A typical fuzzy rule in a Sugeno fuzzy model has the form:

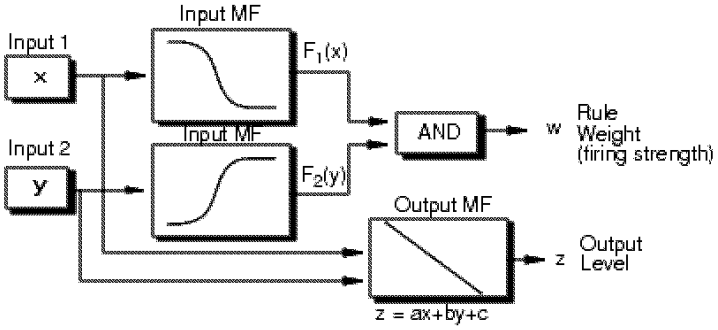
$$if\ x\ is\ A\ and\ y\ is\ B\ then\ z = f(x, y)$$

Where A and B are fuzzy sets in the antecedent, while z=f(x, y) is a crisp function in the consequent. Usually f(x, y) is a polynomial in the input variables x and y, but it can be any function as long as it can appropriately describe the output of the model within the fuzzy region specified by the antecedent of the rule. A Sugeno rule operates as shown in the following diagram. Following are Advantages of the Sugeno Method

- It is computationally efficient.
- It works well with linear techniques (e.g., PID control).
- It works well with optimization and adaptive techniques.
- It has guaranteed continuity of the output surface.
- It is well suited to mathematical analysis.

An example of a single-input Sugeno fuzzy model can be expressed as:

$$\begin{cases} If\ X\ is\ small\ then\ Y = 0.1X + 6.4 \\ If\ X\ is\ medium\ then\ Y = -0.5X + 4 \\ If\ X\ is\ large\ then\ Y = X - 2 \end{cases}$$



Ques 8 a) Discuss the advantages of fuzzy logic controller over that of conventional controller.

Ans. Fuzzy control has emerged one of the most active and fruitful areas of research especially in industrial processes which do not rely upon the conventional methods because of lack of quantitative data regarding the input and output relations. Fuzzy control is based on fuzzy logic, a logical system which is much closer to human thinking and natural language than traditional logical systems. Fuzzy logic controller (FLC) based on fuzzy logic provides a means of converting a linguistic control strategy based on expert knowledge into an automatic control strategy. Fuzzification, defuzzification strategies and fuzzy control rules are used in fuzzy reasoning mechanism.

Fuzzification: Converting crisp facts into fuzzy sets described by linguistic expressions. Membership functions can be flat on the top, piece-wise linear and triangle shaped, rectangular, or ramps with horizontal shoulders.
Inference: The fuzzy IF-THEN rule expresses a fuzzy implication relation between the fuzzy sets of the premise and the fuzzy sets of the conclusion. The following steps describe this process:
1. Matching of the facts with the rule premises (determination of the degree of firing DOF of the facts to the rule premises).
2. If the rule contains logic connections between several premises by fuzzy AND or fuzzy OR the evaluation is performed by t-norms or t-conorms (the result gives then the DOF of the facts to the complete premise).
3. Implication: The next step is the determination of the individual rule output. The DOF of a rule interacts with its consequent to provide the output of the rule. It will be a fuzzy subset over the output universe.
Aggregation: This process aggregates the individual rule outputs to obtain the overall system output. It will be also a fuzzy subset over the output universe (a union operation yields a global fuzzy set of the output).
Defuzzification to obtain crisp output (various defuzzification methods can be used, as, e.g., center of gravity, bisector of area, and mean of maximum, to obtain a crisp numerical output value).

- It can work with less precise inputs.
- It doesn‟t need fast processors.
- It is more robust than other non-linear controllers.

Ques 8 b) Explain four arithmetic operations on closed intervals with examples.
Ans. Arithmetic operations on α-cuts which are a subject of interval analysis of classical mathematics. Fuzzy number can uniquely be represented by their α-cuts. Classical interval analysis
Four operations are

- +,-,·,/
- Let $*$ denote any of the 4 operations→
- $[a,b]*[d,e]=\{f*g \mid a\le f\le b, d\le g\le e\}$
- Except $[a,b]/[d,e]$ when $0\in[d,e]$
- The result of an operations on closed intervals is a closed interval.
- $[a,b]+[d,e]=[a+b,b+e]$
- $[a,b]-[d,e]=[a-b,b-e]$
- $[a,b]\bullet[d,e]=[\min(ad,ae,bd,be),\max(ad,ae,bd,be)]$

- $[a,b]/[d,e] = [a,b] \bullet [1/e, 1/d] = [\min(\frac{a}{d}, \frac{a}{e}, \frac{b}{d}, \frac{b}{e}), \max(\frac{a}{d}, \frac{a}{e}, \frac{b}{d}, \frac{b}{e})]$
- where $0 \notin [d,e]$
- [2,5]+[1,3]=[3,8]
- [2,5]-[1,3]=[-1,4]
- [3,4]·[2,2]=[6,8]
- [4,10]/[1,2]=[2,10]

Ques 9 a) Compare and contrast traditional and genetic algorithm. State the importance of genetic algorithm.

Ans: A genetic algorithm deals with a set (a population) of possible solutions (individuals) of a problem. Each individual is a point in the search space, so we can think of the genetic algorithm as a multi-point optimization technique for multi-dimensional spaces. The majority of traditional optimization methods explore 1, 2, or 3 points in the search space on each iteration. Traditional methods require a starting point to begin the optimization. The choice of a starting point plays a significant role in finding a good solution to the problem with a large number of local optima. Genetic algorithms, which offer many solutions and can search multiple points simultaneously, do not suffer as much from this drawback. If a genetic algorithm is used to optimize a function of continuous variables, it does not work with the problem variables themselves.

- GA's work with string coding of variables instead of variables. so that coding discretising the search space even though the function is continuous.
- GA's work with population of points instead of single point.
- In GA's previously found good information is emphasized using reproduction operator and propagated adaptively through crossover and mutation operators.
- GA does not require any auxiliary information except the objective function values.
- GA uses the probabilities in their operators.
- Discrete functions can also be handled easily.
- Larger likelihood of getting a global solution
- Multiple optimals can be found easily
- No information is needed for problem domain (like slope etc.).

Ques 9 b) Explain genetic algorithm in terms of reproduction, selection, evaluation and replacement.

Ans. Selection is the stage of a genetic algorithm in which individual genomes are selected from a population for later breeding. The fitness function is evaluated for each individual, providing fitness values, which are then normalized. Normalization means dividing the fitness value of each individual by the sum of all fitness values, so that the sum of all resulting fitness values equals 1.The population is sorted by descending fitness values. A random number R between 0 and 1 is chosen. The selected individual is the first one whose accumulated normalized value is greater than R. During each successive generation, a proportion of the existing population is selected to breed a new generation. Individual solutions are selected through a fitness-based process, where fitter solutions (as measured by a fitness function) are typically more likely to be selected. Certain selection methods rate the fitness of each solution and preferentially select the best solutions. The fitness function is defined over the genetic representation and measures the quality of the represented solution. The fitness function is always problem dependent. For instance, in the knapsack problem one wants to maximize the total value of objects that can be put in a knapsack of some fixed capacity.

Reproduction is an identical copy of the individual selected is carried over into the next generation: survival of the fittest. Fitness proportionate reproduction is the reproduction of chromosomes selected stochastically from the population. According to Koza, "the operation of fitness proportionate reproduction for the genetic programming paradigm is the basic engine of Darwinian reproduction and survival of the fittest". In other words, the selection of individual an chromosome is based on a probability that is relative to that chromosomes relative fitness within its population. To generate a second generation population of solutions from those selected through a combination of genetic operators: crossover (also called recombination), and mutation.

In **Evaluation** process the population of strings using the given fitness function is measured. The evaluator decodes a chromosome and assigns it a fitness measure. The evaluator is the only link between a classical GA and the problem it is solving. This is another advantage of genetic algorithms is their inherently parallel nature, i.e., the evaluation of individuals within a population can be conducted simultaneously, as in nature. Typical optimization methods can not deal with such types of problem. In this case, it may be necessary to forgo an exact evaluation and use an approximated that is computationally efficient. It is apparent that amalgamation of approximate models may be one of the most promising approaches to convincingly use GA to solve complex real life problems.

Ques 10) Write short notes on Roulette wheel selection, Random selection, Tournament selection and Boltzmann selection?

Ans: Roulette wheel selection: Roulette wheel is the simplest selection approach. In this method all the chromosomes (individuals) in the population are placed on the roulette wheel according to their fitness value. Each individual is assigned a segment of roulette wheel. The size of each segment in the roulette wheel is proportional to the value of the fitness of the individual - the bigger the value is, the larger the segment is. Then, the virtual roulette wheel is spinned. The individual corresponding to the segment on which roulette wheel stops are then selected. The process is repeated until the desired number of individuals is selected. Individuals with higher fitness have more probability of selection. This may lead to biased selection towards high fitness individuals. It can also possibly miss the best individuals of a population. There is no guarantee that good individuals will find their way into next generation. Roulette wheel selection uses exploitation technique in its approach. The individuals are mapped to contiguous segments of a line, such that each individual's segment is equal in size to its fitness. A random number is

generated and the individual whose segment spans the random number is selected. The process is repeated until the desired number of individuals is obtained (called mating population).

Tournament selection: Tournament selection is probably the most popular selection method in genetic algorithm due to its efficiency and simple implementation. In tournament selection, n individuals are selected randomly from the larger population, and the selected individuals compete against each other. The individual with the highest fitness wins and will be included as one of the next generation population. The number of individuals competing in each tournament is referred to as tournament size, commonly set to 2 (also called binary tournament). Tournament selection also gives a chance to all individuals to be selected and thus it preserves diversity, although keeping diversity may degrade the convergence speed. The tournament selection has several advantages which include efficient time complexity, especially if implemented in parallel, low susceptibility to takeover by dominant individuals, and no requirement for fitness scaling or sorting. In tournament selection, larger values of tournament size lead to higher expected loss of diversity. The larger tournament size means that a smaller portion of the population actually contributes to genetic diversity, making the search increasingly greedy in nature. There might be two factors that lead to the loss of diversity in regular tournament selection; some individuals might not get sampled to participate in a tournament at all while other individuals might not be selected for the intermediate population because they lost a tournament.

Random selection: a population of individuals selected from the search space, often in a random manner, serves as candidate solutions to optimize the problem. The individuals in this population are evaluated through ("fitness") adaptation function. A selection mechanism is then used to select individuals to be used as parents to those of the next generation. These individuals will then be crossed and mutated to form the new offspring. The next generation is finally formed by an alternative mechanism between parents and their offspring. This process is repeated until a certain satisfaction condition.

Boltzmann selection: Simulated annealing is a method of function minimization or maximization .This method simulates the process of slow cooling of molten metal to achieve the minimum function value in a minimization problem. The cooling phenomenon is simulated by controlling a temperature like parameter introduced with the concept of Boltzmann probability distribution so that a system in thermal equilibrium at a temperature T has its energy distributed probability according to $P(E)=\exp(-E/kT)$ Where 'k' is Boltzmann constant. This expression suggests that a system at a high temperature has almost uniform probability of being at any energy state, but at a low temperature it has a small probability of being at a high energy state. Therefore, by controlling the temperature T and assuming search process follows Boltzmann probability distribution, the convergence of the algorithm is controlled.