NEURAL NETWORK & APPLICATIONS

NOTE:

WBUT course structure and syllabus of 8th Semester has been changed from 2014. **NEURAL NETWORK & APPLICATIONS [EC 802A]** has been ntroduced in th present curriculum as a new subject. We are providing chapterwise some model questions and answers along with the complete solutions of new university papers, so that students can get an idea about university questions patterns.

INTRODUCTION TO NEURAL NETWORKS

Multiple Choice Type Questions

1. In a neural net, iffor the training input vectors, the target output is not known, the training method idopted is called as [WBUT 2014, 2015]
   a) supervised traning
   b) unsupervised training
   c) reinforcementtraining
   d) none of these

Answer: (b)

2. The gradient descint rule mostly is used in [WBUT 2014]
   a) M-P Neural Leirning
   b) Hebb Neural Learning
   c) Back-Propagaion Neural Learning
   d) Adaline Neural Learning

Answer: (b)

3. ADALINE stands br [WBUT 2014, 2017]
   a) Additive Linear Neuron
   b) Adaptive Linear Neuron
   c) Associative Linear Neuron
   d) Adaptive De

Answer: (b)

4. Which of the following neural networks uses supervised learning? [WBUT 2014, 2015]
   a) simple recurrent network
   b) self-organizing feature map
   c) Hopfield network
   d) all of these

Answer: (b)

5. Which of the following algorithms can be used to train a single-layer feedforward network? [WBUT 2014, 2015, 2017]
   a) hard competitive learning
   b) soft competitive learning
   c) a genetic algorithm
   d) all of these

Answer: (d)

6. Supervised learning means [WBUT 2015]
   a) having a teacher
   b) having a class
   c) having a feedback
   d) none of these

Answer: (a)

7. Bias is [WBUT 2015]
   a) weight on a connection from a unit having activation 1
   b) weight on a network having activation 2
   c) weight on a function having activation 1
   d) none of these

Answer: (a)

8. Mc Culloch Model uses [WBUT 2015]
a) Sigmoid function
b) Step function
c) Signum function
d) Tan hyperbolic function
Answer: (b)

9. The competitive rule is suited for [WBUT 2015]
a) unsupervised network training
b) supervised network training
c) reinforced network training
d) none of these
Answer: (a)

10. Supervised learning algorithm continues until further change in [WBUT 2016]
a) weights
b) non-linearity
c) signal
d) learning-rate
Answer: (a)

11. The synapse of a neuron is modeled by a [WBUT 2016, 2018]
a) linear function
b) non-linear function
c) non-linear rough function
d) linear / non-linear function
Answer: (d)

12. Functional value of Bipolar sigmoid function is [WBUT 2016]
a) 0 to 1
b) –1 to 1
c) any positive value
d) none of these
Answer: (b)

13. The Hebbian rule is .....type of learning [WBUT 2017, 2018]
a) supervised b) unsupervised c) competitive d) reinforced
Answer: (a)

14. What are the advantages of neural network over conventional computers? [WBUT 2017]
(I) They have the ability to learn by example
(II) They are more faults tolerant
(III) They are suited for real time operation due to their 'computational' rates
a) (I) and (II) are true
b) (I) and (II) are true
c) (II) and (III) are true
d) all of these are true
Answer: (d)

15. Which of the following is/are true for neural networks? [WBUT 2017]
(I) The training time depends on the size of the network
(II) Neural networks can be simulated on a conventional computer
(III) Artificial neurons are identical in operation to biological ones
a) all of these are true
b) (II) is true
c) (II) and (III) are true
d) (I) and (II) are true
Answer: (a)

16. Artificial Neural Networks are inspired by [WBUT 2018]
a) Swarm intelligence
b) high speed parallel processor
c) human brain
d) All of these
Answer: (c)

17. Which of the following is an application of Neural Network? [MODEL QUESTION]
a) Sales forecasting
b) Data validation
c) Risk management
d) All of these
Answer: (d)

18. The Adaline neural network can be used as an adaptive filter for echo cancellation in telephone circuits. For the telephone circuit given in the above figure, which one of the following signals carries the corrected message sent from the human speaker on the left to the human listener on the right? (Assume that the person on the left transmits an outgoing voice signal and receives an incoming voice signal from the person on the right.) [MODEL QUESTION]
a) The outgoing voice signal, s.
b) The delayed incoming voice signal, n.
c) The contaminated outgoing signal, s + n0.
d) The output of the adaptive filter, y.
e) The error of the adaptive filter, " = s + n0 – y.
Answer: (e)

19. What is classification? [MODEL QUESTION]
a) Deciding which features to use in a pattern recognition problem
b) Deciding which class an input pattern belongs to
c) Deciding which type of neural network to use
Answer: (b)

20. What is a pattern vector? [MODEL QUESTION]
a) A vector of weights $w = [w1, w2, ..., wn]^T$ in a neural network.
b) A vector of measured features $x = [x1, x2, ..., xn]^T$ of an input example.
c) A vector of outputs $y = [y1, y2, ..., yn]^T$ of a classifier.
Answer: (b)

Short Answer Type Questions

1. Implement AND function using McCulloch Pitts neuron (take binary data). [WBUT 2014]

Answer:
The AND function returns a true value only if both the inputs are true, else it returns a false value. '1' represents true value '0' represents false value.
The truth table for AND function is,

x_1	x_2	y
1	1	1
1	0	0
0	1	0
0	0	0

A McCulloch-Pitts neuron to implement AND function is shown in Fig. 1. The threshold on unit Y is 2.

The output Y is, $Y = f(y_{in})$

The net input is given by

$$y_{in} = \sum_i \text{weights} * \text{input}$$

$$y_{in} = 1 * x_1 + 1 * x_2$$

$$y_{in} = x_1 + x_2$$

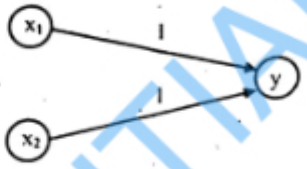


Fig: 1 McCulloch-Pitts neuron to perform logical AND function

From this the activations of output neuron can be formed.

$$Y = f(y_{in}) = \begin{cases} 1 & \text{if } y_{-in} \geq 2 \\ 0 & \text{if } y_{-in} < 2 \end{cases}$$

Now present the inputs

(i) $x_1 = x_2 = 1$, $y_{in} = x_1 + x_2 = 1 + 1 = 2$

$y = f(y_{in}) = 1$ since $y_{in} = 2$.

(ii) $x_1 = 1, x_2 = 0$, $y_{in} = x_1 + x_2 = 0 + 1 = 1$

$y = f(y_{in}) = 0$ since $y_{in} = 1 < 2$.

This is same when $x_1 = 0$ and $x_2 = 1$.

(ii) $x_1 = 0, x_2 = 0$, $y_{in} = x_1 + x_2 = 0 + 0 = 0$

Hence, $y = f(y_{in}) = 0$ since $y_{in} = 0 < 2$.

2. What is the necessity of an activation function? List commonly used activation functions. **[WBUT 2014, 2015]**

OR,

Discuss about the different activation functions used of training artificial neural networks. **[WBUT 2016]**

OR,

Discuss different Activation Functions that are used in Artificial Neural Network. **[WBUT 2017]**

Answer:

In a neural network each neuron has an activation function which species the output of a neuron to a given input. Neurons are *switches* that output a 1when they are sufficiently activated and a 0 when not.

Commonly used activation functions:

Step Function:

A step function is a function like that used by the original Perceptron. The output is a certain value, A1, if the input sum is above a certain threshold and A0 if the input sum is below a certain threshold. The values used by the Perceptron were A1 = 1 and A0 = 0.

Linear Combination:

A linear combination is where the weighted sum input of the neuron plus a linearly dependent bias becomes the system output.

Continuous Log-Sigmoid Function:

A log-sigmoid function, also known as a logistic function is given by the function

$$\sigma(t) = \frac{1}{1+e^{-\beta t}}$$

Softmax Function:

The softmax activation function is useful predominantly in the output layer of a clustering system. Softmax functions convert a raw value into a posterior probability. This provides a measure of certainty.

3. What is Adaline? Draw the model of an Adaline network. **[WBUT 2014]**

OR,

What is Adaline? What type of learning is used in Adaline? **[WBUT 2018]**

Answer:

ADALINE (**Adaptive Linear Neuron** or later **Adaptive Linear Element**) is an early single-layer artificial neural network and the name of the physical device that implemented this network.

Adaline is a single layer neural network with multiple nodes where each node accepts multiple inputs and generates one output. Given the following variables:

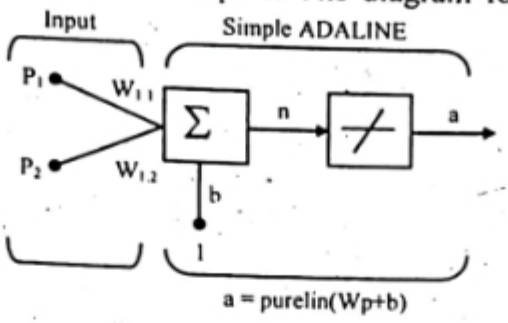
- x is the input vector
- w is the weight vector
- n is the number of inputs
- θ some constant
- y is the output of the model

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 further reduces to the dot product of x and w: $y = x \cdot w$.

Consider a single ADALINE with two inputs. The diagram for this network is shown below.



It uses supervised learning.

4. Implement XOR function using McCulloch-Pitts neuron (consider binary data). **[WBUT 2014, 2016]**

OR,

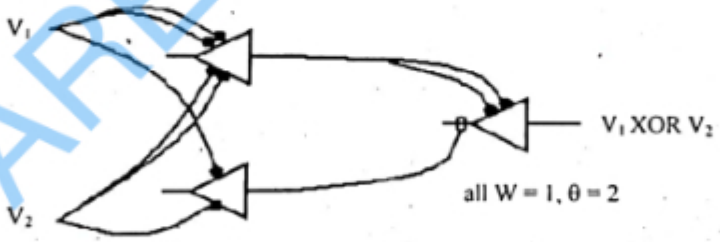
Design a Hebb net to implement logical AND function with bipolar inputs and target. **[WBUT 2017]**

Answer:

The **exclusive OR** (XOR) has the truth table:

V1	V2	XOR
0	0	0
0	1	1
1	0	1
1	1	0

It cannot be represented with a single neuron, but the relationship XOR = (V_1 OR V_2) AND NOT (V_1 AND V_2) suggests that it can be represented with the network. The network is as shown below:



5. What is the impact of weight in an artificial neural network? **[WBUT 2015]**

OR,

What is the role of weight and bias in an ANN model? **[WBUT 2016]**

How does a momentum factor make faster convergence of a network? **[WBUT 2015]**

Answer:

Individual nodes in a neural network emulate biological neurons by taking input data and performing simple operations on the data, selectively passing the results on to other neurons. The output of each node is called its "activation" (the terms "node values" and "activations" are used interchangeably here). Weight values are associated with each vector and node in the network, and these values constrain how input data (e.g., satellite

image values) are related to output data (e.g., land-cover classes). Weight values associated with individual nodes are also known as biases. Weight values are determined by the iterative flow of training data through the network (i.e., weight values are established during a training phase in which the network learns how to identify particular classes by their typical input data characteristics).

The gradient descent is very slow if the learning rate α is small and oscillates widely if α is too large. One very efficient and commonly used method that allows a larger learning rate without oscillations is by adding a momentum factor to the normal gradient descent method The momentum factor denoted by η $\varepsilon[0,1]$ and the value of 0.9 is often used for the momentum factor. A momentum factor can be used with either pattern by pattern updating or batch-mode updating. In case of batch mode, it has the effect of complete averaging over the patterns .Even though the averaging is partial in the pattern-by-pattern mode; it leaves some useful information for weight updating.

6. Define Delta rule. Write down the error function for delta rule.

[WBUT 2016, 2018]

Answer:

1st Part:

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 activation function in this case is called a linear activation function, in which the output node's activation is simply equal to the sum of the network's respective input/weight products. The strengths of network's connections (i.e., the values of the weights) are adjusted to reduce the difference between target and actual output activation (i.e., error).

2nd Part:

The Delta Rule employs the error function for what is known as gradient descent learning, which involves the modification of weights along the most direct path in weight-space to minimize error; change applied to a given weight is proportional to the negative of the derivative of the error with respect to that weight. The error function is commonly given as the sum of the squares of the differences between all target and actual node activations for the output layer. For a particular training pattern (i.e., training case), error is thus given by:

$$E_p = \frac{1}{2}\sum_n \left(t_{j_n} - a_{j_n}\right)^2$$

where E_p is total error over the training pattern, ½ is a value applied to simplify the function's derivative, n represents all output nodes for a given training pattern, t_{jn} represents the target value for node n in output layer j, and a_{jn} represents the actual activation for the same node. This particular error measure is attractive because its derivative, whose value is needed in the employment of the Delta Rule, is easily

calculated. Error over an entire set of training patterns (i.e., over one iteration, or epoch) is calculated by summing all E_p:

$$E = \sum_p E_p = \frac{1}{2}\sum_p \sum_n (t_{pn} - a_{pn})^2$$

where E is total error, and p represents all training patterns.

7. Discuss different categories of learning rules. **[WBUT 2017]**

Answer:

Different categories of learning rules are:

- **Supervised Learning:** The learning algorithm would fall under this category if the desired output for the network is also provided with the input while training the network. By providing the neural network with both an input and output pair it is possible to calculate an error based on it's target output and actual output. It can then use that error to make corrections to the network by updating it's weights.
- **Unsupervised Learning:** In this paradigm the neural network is only given a set of inputs and it's the neural network's responsibility to find some kind of pattern within the inputs provided without any external aid. This type of learning paradigm is often used in data mining and is also used by many recommendation algorithms due to their ability to predict a user's preferences based on the preferences of other similar users it has grouped together.
- **Reinforcement Learning:** Reinforcement learning is similar to supervised learning in that some feedback is given, however instead of providing a target output a reward is given based on how well the system performed. The aim of reinforcement learning is to maximize the reward the system receives through trial-and-error. This paradigm relates strongly with how learning works in nature, for example an animal might remember the actions it's previously taken which helped it to find food (the reward).

8. Compare biological neuron and ANN. **[WBUT 2017]**

Answer:

Artificial neural nets were originally designed to model in some small way the functionality of the biological neural networks which are a part of the human brain. Our brains contain about 10^{16} neurons. Each biological neuron consists of a cell body, a collection of dendrites which bring electrochemical information into the cell and an axon which transmits electrochemical information out of the cell.

A neuron produces an output along its axon i.e., it fires when the collective effect of its inputs reaches a certain threshold. The axon from one neuron can influence the dendrites of another neuron across junctions called synapses. Some synapses will generate a positive effect in the dendrite, i.e. one which encourages its neuron to fire, and others will produce a negative effect, i.e. one which discourages the neuron from firing. A single neuron receives inputs from perhaps 10^5 synapses and the total number of synapses in our brains may be of the order of 10^{16}. It is still not clear exactly how our brains learn and remember but it appears to be associated with the interconnections between the neurons (i.e. at the synapses).

Artificial neural nets try to model this low level functionality of the brain. This contrasts with high level symbolic reasoning in artificial intelligence which tries to model the high level reasoning processes of the brain. When we think we are conscious of manipulating concepts to which we attach names (or symbols) e.g. for people or objects. We are not conscious of the low level electrochemical processes which are going on underneath. The argument for the neural net approach to AI is that, if we can model the low level activities correctly, the high level functionality may be produced as an emergent property.
It can be seen from the above that there is an analogy between biological (human) and artificial neural nets. The analogy is summarized below.

Human	Artificial
Neuron	Processing Element
Dendrites	Combining Function
Cell Body	Transfer Function
Axons	Element Output
Synapses	Weights

However, it should be stressed that the analogy is not a strong one. Biological neurons and neuronal activity are far more complex than might be suggested by studying artificial neurons. Real neurons do not simply sum the weighted inputs and the dendritic mechanisms in biological systems are much more elaborate. Also, real neurons do not stay on until the inputs change and the outputs may encode information using complex pulse arrangement.

9. What is Boltzmann learning? How does it differ from Error-Correction learning?
[WBUT 2017]

Answer:
1st Part:
Boltzmann learning is statistical in nature, and is derived from the field of thermodynamics. It is similar to error-correction learning and is used during supervised training. In this algorithm, the state of each individual neuron, in addition to the system output, are taken into account. In this respect, the Boltzmann learning rule is significantly slower than the error-correction learning rule. Neural networks that use Boltzmann learning are called Boltzmann machines.
Boltzmann learning is similar to an error-correction learning rule, in that an error signal is used to train the system in each iteration. However, instead of a direct difference between the result value and the desired value, we take the difference between the probability distributions of the system.

2nd Part:
Boltzmann learning is similar to an error-correction learning rule, in that an error signal is used to train the system in each iteration. However, instead of a direct difference between the result value and the desired value, we take the difference between the probability distributions of the system. It is also significantly slower than the error-correction learning rule.

10. How neural network can be applied for pattern classification and clustering? **[WBUT 2017]**

Answer:

Classification

- the assignment of each object to a specific "class"
- We are provided with a "training set"
  - Recognizing printed or handwritten characters

Clustering

- Clustering requires grouping together objects that are similar to each other

11. a) What is delta learning rule? **[WBUT 2017]**
b) Compare delta learning rule and Perceptron learning rule.

Answer:

a) *Refer to Question No. 6(1st Part) of Long Answer Type Questions.*

b) There are two differences between the perceptron and the delta rule. The perceptron is based on an output from a step function, whereas the delta rule uses the linear combination of inputs directly. The perceptron is guaranteed to converge to a consistent hypothesis assuming the data is linearly separable. The delta rule converges in the limit but it does not need the condition of linearly separable data.

12. What are the parameters to increase efficiency Hebbian Synapse as a function of the correlation between the pre-synaptic and post-synaptic? How the parameters are influencing the Hebbian Synapse? **[WBUT 2018]**

Answer:

A Hebbian synapse is a synapse that uses a time-dependent, highly local, and strongly interactive mechanism to increase synaptic efficiency as a function of the correlation between the presynaptic and postsynaptic activities.

- **Time-dependent mechanism.** This mechanism refers to the fact that the modifications in a Hebbian synapse depend on the exact time of occurrence of the presynaptic and postsynaptic activities.
- **Local mechanism.** By its very nature, a synapse is the transmission site where information-bearing signals (representing ongoing activity in the presynaptic and postsynaptic units) are in spatiotemporal contiguity. This locally available information is used by a Hebbian synapse to produce a local synaptic modification that is input-specific. It is this local mechanism that enables a neural network made up of Hebbian synapses to perform unsupervised learning.
- **Interactive mechanism.** Here we note that the occurrence of a change in a Hebbian synapse depends on activity levels on both sides of the synapse. That is, a Hebbian form of learning depends on a "true interaction" between presynaptic and postsynaptic activities in the sense that we cannot make a prediction from either one of these two activities by itself. Note also that this dependence or interaction may be deterministic or statistical in nature.

- **Conjunctional or correlational mechanism.** One interpretation of Hebb's postulate of learning is that the condition for a change in synaptic efficiency is the conjunction of presynaptic and postsynaptic activities. Thus, according to this interpretation, the co-occurrence of presynaptic and postsynaptic activities (within a short interval of time) is enough to produce the synaptic modification. It is for this reason that a Hebbian synapse is sometimes referred to as a conjunctional synapse. For another interpretation of Hebb's postulate of learning, we may think of the interactive mechanism characterizing a Hebbian synapse in statistical terms. The correlation over time between presynaptic and postsynaptic activities is viewed as being responsible for a synaptic change.

13. What are differences between Supervised and Unsupervised Learning? How Reinforcement learning differs from Supervised Learning? **[WBUT 2018]**

Answer:

Refer to Question No. 7 of Short Answer Type Questions.

14. How Competitive Learning is different from Hebbian Learning? **[WBUT 2018]**

Answer:

Competitive learning is a form of unsupervised learning in artificial neural networks, in which nodes compete for the right to respond to a subset of the input data. The significant difference between competitive learning and Hebbian learning is in the number of active neurons at any one time. Whereas neural network based on Hebbian learning, several output neurons may be active simultaneously in competitive learning, only a single output neuron is active at any one time. According to this feature, competitive learning is highly suitable for discovering statistically salient features, which makes it useful for classification of input patterns.

15. Describe the main differences between the human brain and today's computers (such as your desktop PC) in terms of information processing.

[MODEL QUESTION]

Answer:

- The brain works in a highly parallel fashion, but in the PC, everything has to go through one or several processors.
- Neurons compute slowly (several ms per computation), electronic elements compute fast (<1 nanosecond per computation).
- The brain represents information in a distributed way, because neurons are unreliable and could die any time. Computer programs rely on every single bit to function properly, otherwise these programs would crash.
- Our brains change their connectivity over time to represent new information and requirements imposed on us. The connectivity between the electronic elements in a computer never changes unless we replace its components

16. Explain how we can use a layer of Adalines to perform classification for more than two classes. How are the units trained, and how do we interpret the units' output in production mode to determine the class of the current input?

[MODEL QUESTION]

Answer:

If there are K classes, then use K Adalines and train each one individually so that unit k outputs 1 if the input is from class k, and 0 (or -1 if you prefer) otherwise. In production mode, if exactly one unit outputs 1, then the input is estimated to be of class k. If no unit or multiple units output 1, there is a misclassification (the input is estimated to belong to none of the classes).

17. What is a training set and how is it used to train neural networks?

[MODEL QUESTION]

Answer:

Training set is a set of pairs of input patterns with corresponding desired output patterns. Each pair represents how the network is supposed to respond to a particular input. The network is trained to respond correctly to each input pattern from the training set. Training algorithms that use training sets are called supervised learning algorithms. We may think of a supervised learning as learning with a teacher, and the training set as a set of examples. During training the network, when presented with input patterns, gives 'wrong' answers (not desired output). The error is used to adjust the weights in the network so that next time the error was smaller.

This procedure is repeated using many examples (pairs of inputs and desired outputs) from the training set until the error becomes sufficiently small.

18. Describe the main steps of the supervised training algorithm.

[MODEL QUESTION]

Answer:

- Initially, set all the weights to some random values
- Repeat (for many epochs):
  a) Feed the network with an input from one of the examples in the training set
  b) Compute the error between the output of the network and the desired output
  c) Correct the error by adjusting the weights of the nodes
- Until the error is sufficiently small

19. When are neural networks a good choice for problem solving?

[MODEL QUESTION]

Answer:

Problems with the following characteristics are a good fit for neural networks:

a. Problems where we can't formulate an algorithmic solution.
b. Problems where we can get lots of examples of the behavior we require.
c. Problems where we need to pick out the structure from existing data.

20. Explain the difference between Supervised and unsupervised learning?

[MODEL QUESTION]

Answer:
The difference can be explained through an example.

- **Supervised learning:**

An archaeologist determines the gender of a human skeleton based on many past examples of male and female skeletons.

- **Unsupervised learning:**

The archaeologist determines whether a large number of dinosaur skeleton fragments belong to the same species or multiple species. There are no previous data to guide the archaeologist, and no absolute criterion of correctness.

21. What is a Neural Network? **[MODEL QUESTION]**

Answer:
A neural network is a powerful data modeling tool that is able to capture and represent complex input/output relationships. The motivation for the development of neural network technology stemmed from the desire to develop an artificial system that could perform "intelligent" tasks similar to those performed by the human brain. Neural networks resemble the human brain in the following two ways:

1. A neural network acquires knowledge through learning.
2. A neural network's knowledge is stored within inter-neuron connection strengths known as synaptic weights.

The true power and advantage of neural networks lies in their ability to represent both linear and non-linear relationships and in their ability to learn these relationships directly from the data being modeled. Traditional linear models are simply inadequate when it comes to modeling data that contains non-linear characteristics.

22. With a supervised learning algorithm, we can specify target output values, but we may never get close to those targets at the end of learning. Give two reasons why this might happen. **[MODEL QUESTION]**

Answer:
(i) data may be valid, and inconsistency results from a stochastic aspect of the task (or some aspect of the task is not modelled by the input data collected);
(ii) the data may contain errors - e.g. measurement errors or typographical errors

23. Differentiate using examples between the following types of neural network applications:
pattern association and forecasting. **[MODEL QUESTION]**

Answer:
Pattern association

- the presentation of an input sample should trigger the generation of a specific output pattern
  - Auto-associative (given a corrupted version of letters and recognizing the true one)
  - Hetero-associative (given features of a letter and associating them to the real letter)

Forecasting

- Redacting the behavior of stock market indices
- Though perfect prediction is hardly ever possible, neural networks can be used to obtain reasonably good predictions in a number of cases.

Long Answer Type Questions

1. Implement OR function with bipolar inputs and targets using ADALINE network initially all the weights are assumed to be small random values, say 0.1 and the learning rate 0.1. **[WBUT 2016]**

Answer:

Example:

An ADALINE for the OR function: bipolar inputs and targets

The logic function x_1 OR x_2 is defined by the following bipolar input and target patterns:

x_1	x_2	t
1	1	1
1	−1	1
−1	1	1
−1	−1	−1

Weights that minimize the total squared error for the bipolar form of the OR function are

$w_1 = \frac{1}{2}$

and

$$w_2 = \frac{1}{2},$$

With the bias

$$w_0 = \frac{1}{2}.$$

Thus, the separating line is

$$\frac{1}{2}x_1 + \frac{1}{2}x_2 + \frac{1}{2} = 0$$

2. Describe the salient features of McCulloch-Pitts neuon model. Design a McCulloch-Pitts neural net to realize the XOR function. **[WBUT 2017]**

Answer:

The McCulloch-Pitts neural model is also known as linear threshcd gate. It is a neuron of a set of inputs $I_1, I_2, I_3, ..., I_n$ and one output y. The linear threshol gate simply classifies the set of inputs into two different classes. Thus the output y is inary. Such a function can be described mathematically using these equations:

$$\text{Sum} = \sum_{i=1}^{N} I_i W_i \quad \ldots (1)$$

$$y = f(\text{Sum}) \quad \ldots (2)$$

$W_1, W_2, W_3, \ldots, W_n$ 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 **figure (a)** below. The symbolic representation of the linear threshold gate is shown in **figure (b)**.

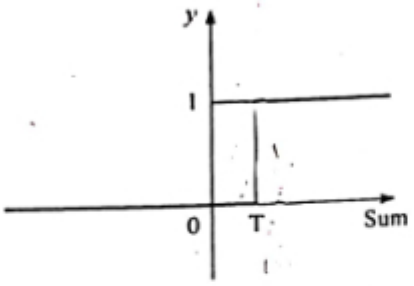


Fig. (a): Linear Threshold Function

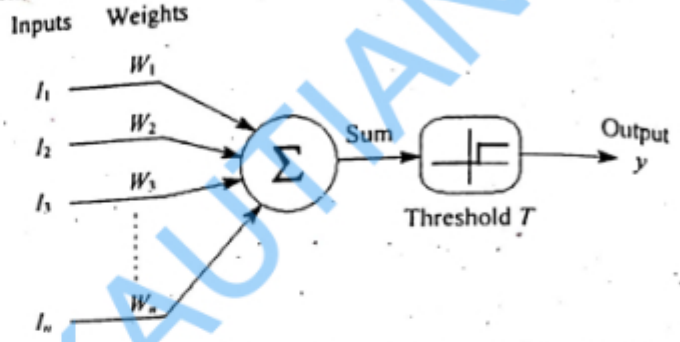


Fig. (b): Symbolic Illustration of Linear Threshold Gate

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.

3. Write short notes on the following [WBUT 2014]

a) Memory based learning [WBUT 2016]

b) Supervised learning [WBUT 2016]

c) Neural network architecture [WBUT 2017]

d) Gradient descent learning [WBUT 2017]

e) Competitive Learning [WBUT 2018]

f) Boltzman Learning [WBUT 2018]

g) Reinforcement leaning

Answer:

a) Memory based leaning:

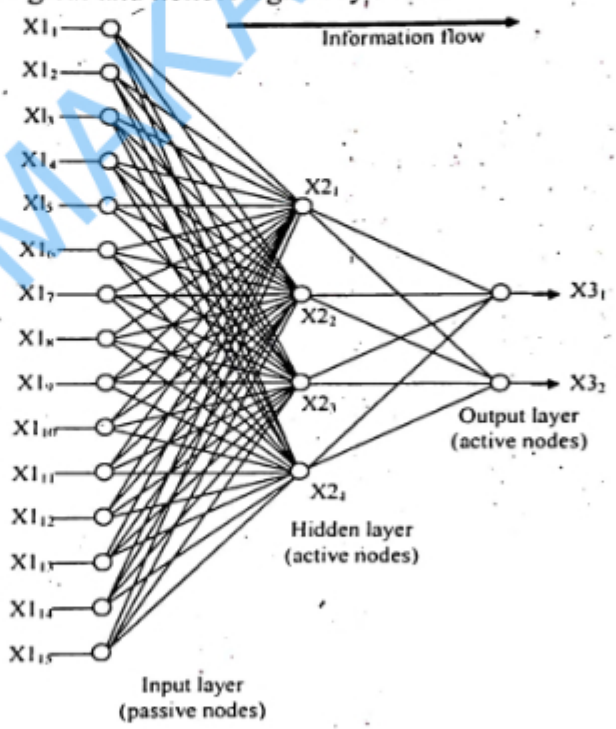
Memory Based Learning (MBL) is based on the idea that intelligent behavior can be obtained by analogical reasoning, rather than by the application of abstract mental rules as in rule induction and rule-based processing. In particular, MBL is founded in the hypothesis that the extrapolation of behavior from stored representations of earlier experience to new situations, based on the similarity of the old and the new situation, is of key importance. MBL algorithms take a set of examples (fixed-length patterns of feature-values and their associated class) as input, and produce a classifier which can classify new, previouly unseen, input patterns. MBL can in principle be applied to any kind of classification task with symbolic or numeric features and discrete (non-continuous) classes for which training data is available.

b) Supervised learning:

Supervised learning is the machine learning task of inferring a function from *supervised* training data. The training data consist of a set of *training examples*. In supervised learning, each example is a *pair* consisting of an input object (typically a vector) and a desired output value (also called the *supervisory signal*). A supervised learning algorithm analyzes the training data and produces an inferred function, which is called a *classifier* (if the output is discrete) or a *regression function* (if the output is continuous). The inferred function should predict the correct output value for any valid input object. This requires the learning algorithm to generalize from the training data to unseen situations in a "reasonable" way (see inductive bias). The parallel task in human and animal psychology is often referred to as concept learning.

c) Neural network architecture:

Humans and other animals process information with *neural networks*. These are formed from *trillions* of **neurons** (nerve cells) exchanging brief electrical pulses called **action potentials**. Computer algorithms that mimic these biological structures are formally called **artificial neural networks** to distinguish them from the squishy things inside of animals. However, most scientists and engineers are not this formal and use the term *neural network* to include both biological and nonbiological systems.



Neural network architecture: This is the most common structure for neural networks: three layers with full interconnection. The input layer nodes are passive, doing nothing but relaying the values from their single input to their multiple outputs. In comparison, the nodes of the hidden and output layers are active, modifying the signals in accordance with figure. The action of this neural network is determined by the weights applied in the hidden and output

d) Gradient descent learning:

Gradient descent is an optimization algorithm used to find the values of parameters (coefficients) of a function (f) that minimizes a cost function (cost).

Gradient descent is best used when the parameters cannot be calculated analytically (e.g. using linear algebra) and must be searched for by an optimization algorithm.
The procedure starts off with initial values for the coefficient or coefficients for the function. These could be 0.0 or a small random value.

coefficient = 0.0

The cost of the coefficients is evaluated by plugging them into the function and calculating the cost.

cost = f(coefficient) or **cost = evaluate(f(coefficient))**

The derivative of the cost is calculated. The derivative is a concept from calculus and refers to the slope of the function at a given point. We need to know the slope so that we know the direction (sign) to move the coefficient values in order to get a lower cost on the next iteration.

delta = derivative(cost)

Now that we know from the derivative which direction is downhill, we can now update the coefficient values. A learning rate parameter (alpha) must be specified that controls how much the coefficients can change on each update.

coefficient = coefficient – (alpha * delta)

This process is repeated until the cost of the coefficients (cost) is 0.0 or close enough to zero to be good enough.

e) Competitive Learning:

In competitive learning the following properties hold true:

- Nodes compete for inputs
- Node with highest activation is the winner
- Winner neuron adapts its tuning (pattern of weights) even further towards the current input
- Individual nodes specialize to win competition for a set of similar inputs
- Process leads to most efficient neural representation of input space
- Typical for unsupervised learning

f) Boltzman Learning:

Refer to Question No. 9 of Short Answer Type Questions.

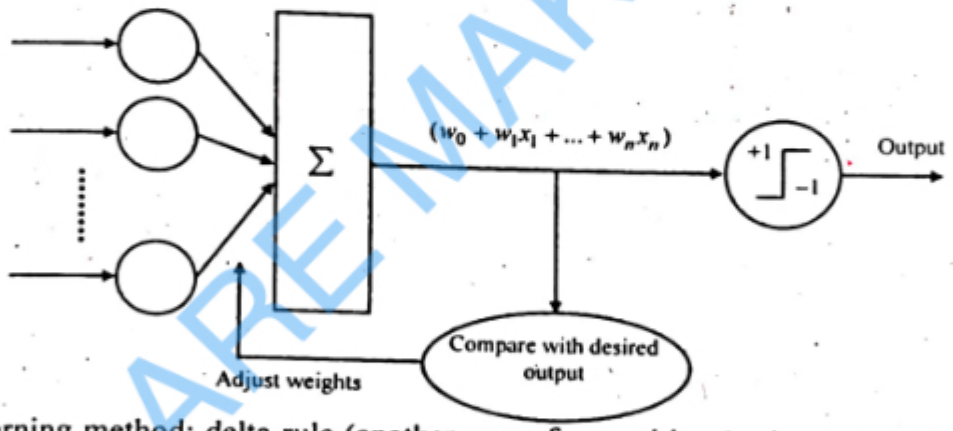
g) Reinforcement learning:

Reinforcement Learning is a type of Machine Learning, and thereby also a branch of Artificial Intelligence. It allows machines and software agents to automatically determine the ideal behaviour within a specific context, in order to maximize its performance. Simple reward feedback is required for the agent to learn its behaviour; this is known as the reinforcement signal. There are many different algorithms that tackle this issue. Reinforcement Learning is defined by a specific type of problem, and all its solutions are classed as Reinforcement Learning algorithms. In the problem, an agent is supposed decide the best action to select based on his current state. When this step is repeated, the problem is known as a Markov Decision Process. This automated learning scheme

implies that there is little need for a human expert who knows about the domain of application. Much less time will be spent designing a solution, since there is no need for hand-crafting complex sets of rules as with Expert Systems, and all that is required is someone familiar with Reinforcement Learning.
This automated learning scheme implies that there is little need for a human expert who knows about the domain of application. Much less time will be spent designing a solution, since there is no need for hand-crafting complex sets of rules as with Expert Systems, and all that is required is someone familiar with Reinforcement Learning. The possible applications of Reinforcement Learning are abundant, due to the generalness of the problem specification. As a matter of fact, a very large number of problems in Artificial Intelligence can be fundamentally mapped to a decision process. This is a distinct advantage, since the same theory can be applied to many different domain specific problem with little effort. In practice, this ranges from controlling robotic arms to find the most efficient motor combination, to robot navigation where collision avoidance behaviour can be learnt by negative feedback from bumping into obstacles. Logic games are also well-suited to Reinforcement Learning, as they are traditionally defined as a sequence of decisions.

4. What is the principle of learning of the Adaline? Fully explain the Adaline architecture and learning algorithm. **[MODEL QUESTION]**

Answer:



- Learning method: delta rule (another way of error driven), also called Widrow-Hoff learning rule
- Try to reduce the mean squared error (MSE) between the net input and the desired out put

Algorithm LMS-Adaline:

Start with a randomly chosen weight vector w_0:

Let $k = 1$;

while MSE is unsatisfactory and

computational bounds are not exceeded, do

Let i be an input vector

(chosen randomly or in some sequence)

for which d is the desired output value;
Update the weight vector to

$$w_k = w_{k-1} + \eta(d - w_{k-1}\cdot)i$$

Increment k :
end-while.

5. What is Hebbian Learning? Explain using mathematical terms. [MODEL QUESTION]

Answer:
Hebb's postulate of learning is the oldest and most famous of all learning rules:

1. If two neurons on either side of a synapse (connection) are activated simultaneously (i.e. synchronously), then the strength of that synapse is selectively increased.
2. If two neurons on either side of a synapse are activated asynchronously, then that synapse is selectively weakened or eliminated.

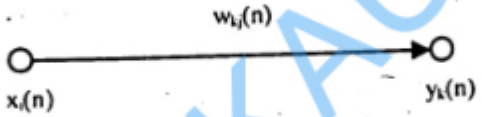


Fig: Synaptic connection

To formulate Hebb's postulate of learning in mathematical terms, consider a synaptic weight w_{kj} with presynaptic and postsynaptic activities denoted by x_j and y_k, respectively. According to Hebb's postulate, the adjustment applied to the synaptic weight w_{kj} at time n is $\Delta w_{kj}(n) = F(y_k(n), x_j(n))$.

As a special case we may use the activity product rule $\Delta w_{kj}(n) = \eta y_k(n) x_j(n)$

where η is a positive constant that determines the rate of learning. This rule clearly emphasizes the correlational nature of a Hebbian synapse.
From this representation we see that the repeated application of the input signal x_j leads to an exponential growth that finally drives the synaptic weight w_{kj} into saturation.

$$w_{kj}(n+1) = w_{kj}(n) + \eta y_k(n) x_j(n) = w_{kj}(n)\left(1 + \eta x_j^2(n)\right)$$

If x_j stays constant then

$$w_{kj}(n+N) = w_{kj}(n)\left(1 + \eta x_j^2\right)^N$$

To avoid such a situation from arising, we need to impose a limit on the growth of synaptic weights. One method for doing this is to introduce a nonlinear forgetting factor into the formula for the synaptic adjustment $\Delta w_{kj}(n)$. Specifically, we redefine $\Delta w_{kj}(n)$ as a generalized activity product rule:

$$\Delta w_{kj}(n) = \eta y_k(n) x_j(n) - \alpha y_k(n) w_{kj}(n) = \alpha y_k(n)\left[c x_j(n) - w_{kj}(n)\right]$$

where $c = \eta/\alpha$. If the weight $w_{kj}(n)$ increases to the point where $c x_j(n) - w_{kj}(n) = 0$ a balance point is reached and the weight update stops.

SINGLE-LAYER PERCEPTRONS

Multiple Choice Type Questions

1. A perceptron is [WBUT 2014]
   a) a single layer feed-forward neural network with preprocessing
   b) an autoassociative neural network
   c) a double layer autoassociative neural network
   d) all of these

Answer: (a)

2. In back-propagation algorithm _____ is propagated backward through the network. [WBUT 2014, 2016, 2018]
   a) error b) signal c) error + signal d) signal – error

Answer: (c)

3. A perceptron is: [WBUT 2015]
   a) a single layer feed-forward neural network with pre-processing
   b) an auto-associative neural network
   c) a double layer auto-associative neural network
   d) Hebb network

Answer: (a)

4. A 3-input neuron is trained to output a zero when the input is 110 and a one when the input is 111. After generalization, the output will be zero when and only when the input is [WBUT 2016]
   a) 000 or 110 or 011 or 101
   b) 010 or 100 or 110 or 101
   c) 000 or 010 or 110 or 100

Answer: (c)

5. The madaline network is [WBUT 2016, 2018]
   a) The combination of two single layered feed forward neural networks
   b) A type of multilayered feed forward neural network with multiple neurons in output layer
   c) The combination of adaline networks and multilayered feed forward network with one neuron in output layer
   d) A type of feedback network

Answer: (b)

6. Single layer Perceptron is used for [WBUT 2017, 2018]
   a) linear separability b) error minimization
   c) back propagation d) annealing

Answer: (a)

7. For a three input neuron representing a Perceptron where $\{x_1, x_2, x_3\} = \{0.8, 0.6, 0.4\}$ and weight $\{w_1, w_2, w_3\} = \{0.1, 0.3, -0.2\}$ and bias b=0.35 the output of neuron using bipolar sigmoid activation function is [WBUT 2017]
a) 0.265 b) 0.746 c) 0.346 d) 0.259

Answer: (a)

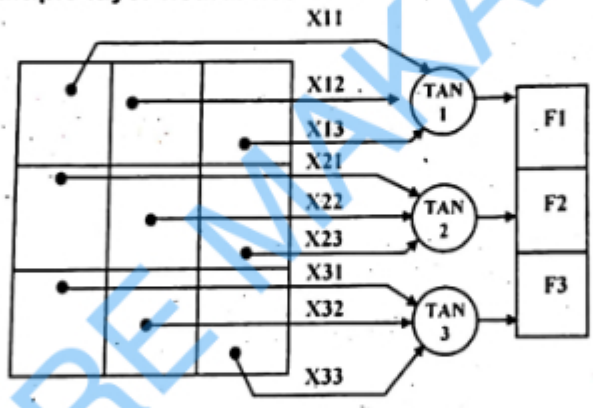
8. A 4-input neuron has weights 1, 2, 3 and 4. The transfer function is linear with the constant of proportionality being equal to 2. The inputs are 4, 10, 5 and 20 respectively. The output will be: [WBUT 2017]
a) 238 b) 76 c) 119 d) 328

Answer: (b)

9. For a four input (0, 0), (0, 1), (1, 0), (1, 1) neuron representing a Perceptron with $w1 = w2 = 1$ and $\theta = 1.5$ the classification does [WBUT 2018]
a) AND classifier b) OR classifier
c) XOR classifier d) None of these

Answer: (a)

10. The network of figure below is: [MODEL QUESTION]
a) a single layer feed-forward neural network
b) an autoassociative neural network
c) a multiple layer neural network



Answer: (a)

11. A single perceptron can compute the XOR function. [MODEL QUESTION]
a) True b) False

Answer: (b)

12. A perceptron adds up all the weighted inputs it receives, and if it exceeds a certain value, it outputs a 1, otherwise it just outputs a 0. [MODEL QUESTION]
a) True b) False
c) Sometimes – it can also output intermediate values as well
d) Can't say

Answer: (a)

13. "The XOR problem can be solved by a multi-layer perceptron, but a multi-layer perceptron with bipolar step activation functions cannot learn to do this." [MODEL QUESTION]
a) True b) False
Answer: (a)

14. The Perceptron Learning Rule states that "for any data set which is linearly separable, the Perceptron Convergence Theorem is guaranteed to find a solution in a finite number of steps." [MODEL QUESTION]
a) True b) False
Answer: (b)

15. A perceptron with a unipolar step function has two inputs with weights w1 = 0.5 and w2 = −0.2, and a threshold θ = 0.3 (θ can therefore be considered as a weight for an extra input which is always set to -1). For a given training example $x = [0,1]^T$, the desired output is 1. Does the perceptron give the correct answer (that is, is the actual output the same as the desired output)? [MODEL QUESTION]
a) Yes b) No
Answer: (b)

16. A perceptron is guaranteed to perfectly learn a given linearly separable function within a finite number of training steps. [MODEL QUESTION]
a) True b) False
Answer: (a)

17. In backpropagation learning, we should start with a small learning parameter η and slowly increase it during the learning process. [MODEL QUESTION]
a) True b) False
Answer: (b)

Short Answer Type Questions

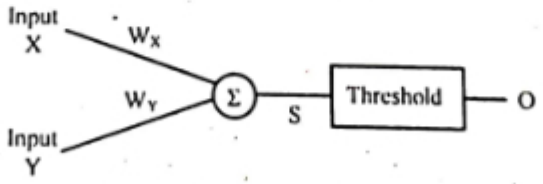
1. The Exclusive-OR function is not linearly separable and hence a single-layer perception cannot simulate it. Justify it. [WBUT 2014]

OR,

Define the single layer perceptron net and its linear separability. [WBUT 2016]

Answer:

Consider the two input neuron shown in figure below.



The output from the summing stage of the neuron is:

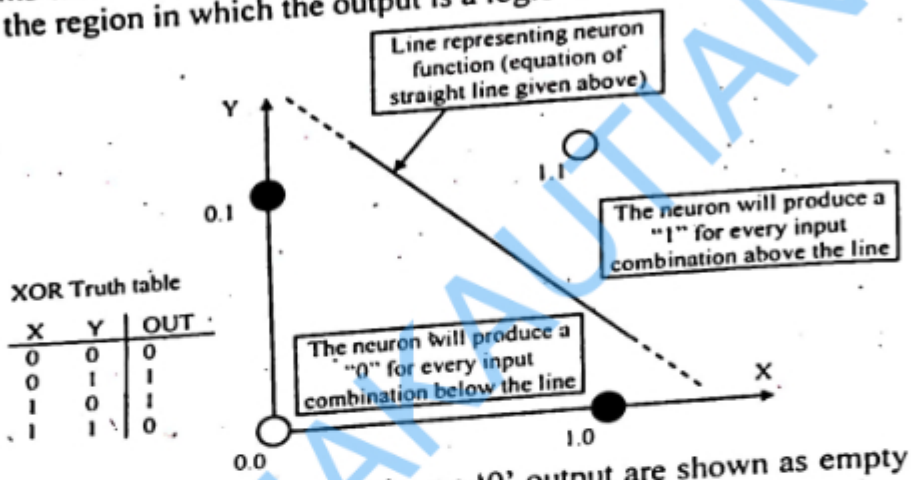
$$S = XW_X + YW_Y$$

We can re-arrange this equation into the equation of a straight line:

$$Y = -\frac{W_X}{W_Y}X + \frac{S}{W_Y}$$

which can be compared with $Y = mX + c$.

If the neuron is a simple threshold unit of the type shown above, then the physical meaning of this line is that it is the divider between the region in which the output is a logic '1' and the region in which the output is a logic '0' as shown in figure below.



XOR Truth table

X	Y	OUT
0	0	0
0	1	1
1	0	1
1	1	0

The inputs which we would like to produce a '0' output are shown as empty circles, and inputs which produce an output of '1' are shown as filled circles. It can be clearly seen, that no matter where the line is plotted on the graph, the '0' outputs cannot be separated from the '1' outputs by the line; and hence, a simple neuron cannot simulate a XOR gate. This class of problem is called Linear Separability and we say that the XOR function is not linearly separable by a single Perceptron type neuron.

2. A 4-input neuron has weights 0.1, 0.2, 0.3 and 0.4. The transfer function is linear with the constant of proportionality being equal to 5. The inputs are 5, 10, 15 and 20 respectively. Find the output. **[WBUT 2016]**

Answer:

The output is found by multiplying the weights with their respective inputs, summing the results and multiplying with the transfer function.

Therefore:

Output = 5 * (0.1*5 + 0.2*10 + 0.3*15 + 0.4*20) = 75.

3. Which type of Activation Function is commonly used in Back Propagation algorithm? **[WBUT 2018]**

Answer:

The activation function is non-linear and differentiable. A commonly used activation function is the logistic function: $1/(1+e^{-z})$

4. Explain briefly the difference between A perceptron and a feed-forward, back-propagation neural network. [MODEL QUESTION]

Answer:

A neural network consists of a collection of perceptrons organized in layers, so that the output of one layer is the input to the next layer. The perceptrons are modified so that the output is a continuous function of the inputs, rather than being a step function. The learning algorithm is similar in principle to the perceptron learning algorithm, but involves a system of message passing from the output layer backwards to the input layer.

5. Consider the following data set T. A and B are numerical attributes and Z is a Boolean classification. [MODEL QUESTION]

A	B	Z
1	2	T
2	1	F
3	2	T
1	1	F

a) Let P be the perceptron with weights $w_A = 2$, $w_B = 1$, and threshhold T=4.5. What is the value of the standard error function for this perceptron?

b) Find a set of weights and a threshold that categorizes all this data correctly.

Answer:

a) Perceptron P gets the first instance wrong, with an error of $|(2*1)+(1*2)-4.5|=0.5$ and the second instance wrong with an error of $|(2*2)+(1*1)-4.5|=0.5$. The total error is therefore 1.0.

b) $w_A= 0$, $w_B=1$, T=1.5 will do fine.

6. What is the advantage of Adalines over perceptrons? How is it achieved? [MODEL QUESTION]

Answer:

The advantage of Adalines is that they do not simply find any solution that leads to perfect classification of the training set, but they try to optimize their computation so that it works as best as possible with new (untrained) inputs. This is achieved by using a continuous, differentiable error function and minimizing this error using gradient descent. This error minimum is the best estimate for the optimal classification function with regard to the entire data set (of which the training data are usually only a small subset).

Long Answer Type Questions

1. List the stages involved in training of back propagation algorithm. [WBUT 2014]

Answer:

The back propagation-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 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. Subtract a ratio (percentage) of the gradient from the weight.

This ratio (percentage) influences the speed and quality of learning; it is called the *learning rate*. The greater the ratio, the faster the neuron trains; the lower the ratio, the more accurate the training is. 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.

2. a) Define perceptron learning rule. How the linear separability concept is implemented using perceptron network training? [WBUT 2015]

Answer:

1st Part:

Algorithm

Start with a randomly chosen weight vector w_0.

Let k=1;

While there exist input vectors that are misclassified by w_{k-1}, do

Let i_j be a misclassified input vector;

Let $x_k = class(i_j) \cdot i_j$, implying that $w_{k-1} \cdot x_k < 0$;

Update the weight vector to $w_k = w_{k-1} + \eta x_k$;

Increment k;

end-while;

For example, for some input i with class(i) = -1,

If $w \cdot i > 0$, then we have a misclassification.

Then the weight vector needs to be modified to $w + \Delta w$ with $(w + \Delta w) \cdot i < w \cdot i$ to possibly improve classification.

We can choose $\Delta w = -\eta i$, because

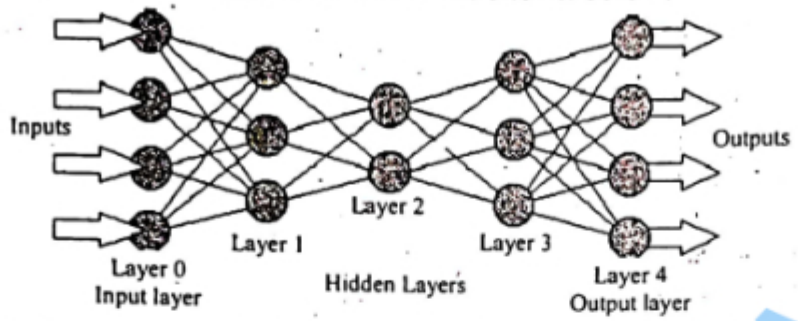
$(w + \Delta w) \cdot i = (w - \eta i) \cdot i = w \cdot i - \eta i \cdot i < w \cdot i$, and $i \cdot i$ is the square of the length of vector i and is thus positive.

If class(i) = 1, things are the same but with opposite signs; we introduce x to unify these two cases.

2nd Part:

Although the perceptron rule finds a successful weight vector when the training examples are linearly separable, it can fail to converge if the examples are not linearly separable –

e.g., the XOR function. This problem could be overcome by using more than one perceptron arranged in feed-forward networks as shown below.



Feed-forward networks have the following characteristics:

1. Perceptrons are arranged in layers, with the first layer taking in inputs and the last layer producing outputs. The middle layers have no connection with the external world, and hence are called hidden layers.
2. Each perceptron in one layer is connected to every perceptron on the next layer. Hence information is constantly "fed forward" from one layer to the next, and this explains why these networks are called feed-forward networks.
3. There is no connection among perceptrons in the same layer.

b) Implement OR function with binary inputs and bipolar targets using perceptron training algorithm upto 3 epochs. **[WBUT 2015]**

Answer:

The truth table for OR function with binary inputs and bipolar targets is given below.

x_1	x_2	t
1	1	1
1	0	1
0	1	1
0	0	−1

The perceptron network, which uses perceptron learning rule, is used to train the OR function. The network architecture is shown in figure.

The initial values of the weights and bias are taken as zero, i.e.,

$$w_1 = w_2 = b = 0$$

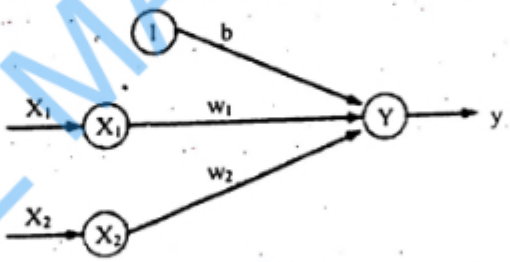
Also learning rate is 1 and threshold is 0.2. So, the activation function becomes

$$f(y_{in}) = \begin{cases} 1 & \text{if } y_{in} > 0.2 \\ 0 & \text{if } -0.2 \le y_{in} \le 0.2 \end{cases}$$

The network is trained as per the perceptron training algorithm.

The following table gives the network training for epochs.

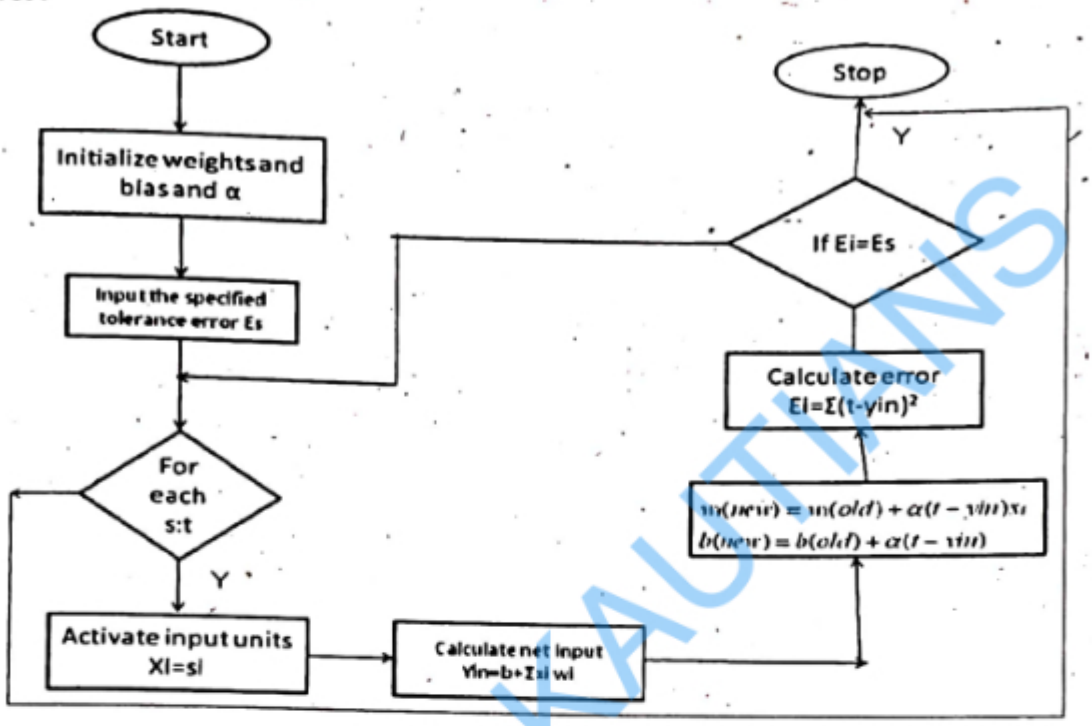
Input			Target	Net input	Calculated output	Weight changes			Weights		
x_1	x_2	1	(t)	(y_{in})	(y)	Δw_1	Δw_2	Δb	w_1 (0	w_2 0	b 0)
EPOCH-1											
1	1	1	1	0	0	1	1	1	1	1	1
1	0	1	1	2	1	0	0	0	1	1	1
0	1	1	1	2	1	0	0	0	1	1	0
0	0	1	−1	1	1	0	0	−1	1	1	0
EPOCH-2											
1	1	1	1	2	1	0	0	0	1	1	0
1	0	1	1	1	1	0	0	0	1	1	0
0	1	1	1	1	1	0	0	0	1	1	0
0	0	1	−1	0	0	0	0	0	1	1	−1
EPOCH-3											
1	1	1	1	1	1	0	0	0	1	1	−1
1	0	1	1	0	0	1	0	1	2	1	0
0	1	1	1	1	1	0	0	0	2	1	0
0	0	1	−1	0	0	0	0	−1	2	1	−1



Perception network for OR functio

3. a) Draw the flowchart for Adaline training process. **[WBUT 2015]**

Answer:



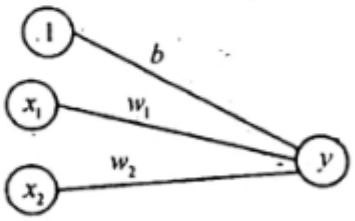
b) Use Adaline network to train ANDNOT function with bipolar inputs and targets. Perform 2 epochs of training. **[WBUT 2015]**

Answer:

The truth table for the ANDNOT function with bipolar inputs and targets is:

x_1	x_2	t
1	1	−1
1	−1	1
−1	1	−1
−1	−1	−1

The architecture is given by,



Initially the weights and bias are assumed a random value say 0.2. The learning rate is also set to 0.2.

The weights are calculated until the least mean square error is obtained.

The initial weights $w_1 = w_2 = b = 0.2$ and $x = 0.2$

The operations are carried out for 2 epochs, where the mean square error is minimized.

	Inputs			Target	Net		Weight changes			Weights			Error E
	x_1	x_2	b	t	y_{in}	$(t-y_{in})$	Δw_1	Δw_2	Δw_3	w_1	w_2	w_3	$(t-y_{in})^2$
Epoch 1	1	1	1	–1	0.6	–1.6	–0.32	–0.32	–0.32	–0.12	–0.12	–0.12	2.56
	1	–1	1	1	–0.12	1.12	0.22	–0.22	0.22	0.10	–0.34	0.10	1.25
	–1	1	1	–1	–0.34	–0.66	0.13	–0.13	–0.13	0.24	–0.48	–0.03	0.43
	–1	–1	1	–1	0.21	–1.2	0.24	0.24	–0.24	0.48	–0.23	–0.27	1.47 $E=5.71$
Epoch 2	1	1	1	–1	–0.02	–0.98	–0.195	–0.195	–0.195	0.28	–0.43	–0.46	0.95
	1	–1	1	1	0.25	0.76	0.15	–0.15	0.15	0.43	–0.58	–0.31	0.57
	–1	1	1	–1	–1.33	0.33	–0.065	0.065	0.065	0.37	–0.51	–0.25	0.106
	–1	–1	1	–1	–0.11	–0.90	0.18	0.18	–0.18	0.55	–0.33	0.43	0.8 $E=2.43$

Thus it is found that the weights at the end of Epoch 2 is approximately.

$w_1 = 0.55$, $w_2 = -0.33$, $b = -0.5$

By using the above weights, the LMS error is calculated

$y_{in} = b + \sum x_i w_i$

$E = (t - y_{in})^2$

Inputs			Net Input	t	$E=(t-y_{in})^2$
x_1	x_2	b	$w_1 = 0.55, w_2 = -0.33, b = -0.5$		
1	1	1	–0.5	–1	0.25
1	–1	1	+0.5	1	0.25
–1	1	1	–1.5	–1	0.25
–1	–1	1	–0.5	–1	0.25

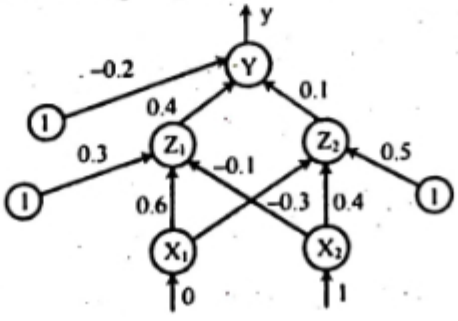
Thus the total error is,

$$E = \sum_{i=1}^{4} (t - y_{in})^2 = 0.25 + 0.25 + 0.25 + 0.25$$

$E = 1$

Depending upon the stopping conditions, the number of iterations of epochs takes place.

4. a) Using back-propagation network, find the new weights for the net shown in the following figure. It is presented with the input pattern [0, 1] and target output is 1. Use learning rate 0.25 and binary sigmoidal activation function. **[WBUT 2016]**



Answer:

The new weights are calculated based on the training algorithm. The initial weights are
[v11 v21 v01] = [0.6 -0.1 0.3],
[v12 v22 v02] = [-0.3 0.4 0.5] and
[w1 w2 w0] = [0.4 0.1 -0.2], and a learning rate of $\alpha = 0.25$.
The activation function used is binary sigmoidal activation function and is given by • f(x) = 1/[1 + e^(-1)].
Given the input sample [x1, x2] = [0 , 1] and target t = 1.
Compute the error portion δk : $\Delta k = (tk - yk).f'(yink)$
Now $f'(yin) = f(yin)[1-f(yin)] = 0.5227*[1-0.5227]$
$f'(yin) = 0.1191$
The changes in weights between hidden and output layer:
$\Delta w1 = \alpha. \delta 1.z1 = 0.25*0.1191 * 0.5498 = 0.0164$
$\Delta w2 = \alpha. \delta 1.z2 = 0.25*0.1191 * 0.7109 = 0.02117$
$\Delta w0 = \alpha. \delta 1 = 0.25*0.1191 = 0.02978$
Compute the error portion .δj between input and hidden layer (j = 1 to 2): .δj = .$\delta inj.f'(zinj)$ where $\delta inj = \sum \delta k.wjk$ • $\delta inj = \delta 1.wj1$ [only one output neuron]
$\delta in1 = \delta 1.w11 = 0.1191 *0.4 =0.04764$
$\delta in2 = \delta 1.w21 = 0.1191 *0.1 =0.01191$
Error, $\delta 1 = \delta in1.f'(zin1)$
$f'(zin1) = f(zin1)[1- f(zin1)] = 0.5498*[1-0.5498] = 0.2475$
$\delta 1 = \delta in1.f'(zin1) = 0.04764 * 0.2475 = 0.0118$
Error, $\delta 2 = \delta in2.f'(zin2)$
$f'(zin2) = f(zin2)[1- f(zin2)] = 0.71091*[1-0.7109] = 0.2055$
$\delta 2 = \delta in2.f'(zin2) = 0.01191* 0.2055 = 0.00245$
Now find the changes in weights between input and hidden layer:
$\Delta v11 = \alpha\delta 1.x1 = 0.25* 0.0118*0 = 0$
$\Delta v21 = \alpha\delta 1.x2 = 0.25* 0.0118*1 = 0.00295$
$\Delta v01 = \alpha\delta 1 = 0.25* 0.0118 = 0.00295$
$\Delta v12 = \alpha\delta 2.x1 = 0.25* 0.00245*0 = 0$
$\Delta v22 = \alpha\delta 2.x2 = 0.25* 0.00245*1 = 0.0006125$
$\Delta v02 = \alpha\delta 2 = 0.25* 0.00245 = 0.0006125$
Compute the final weights of the network:
v11(new) = v11(old) + Δv11 = 0.6 + 0 = 0.6
v12(new) = v12(old) + Δv12 = -0.3 + 0 = -0.3
v21(new) = v21(old) + Δv21 = -0.1 + 0.00295 = -0.09705 v22(new) = v22(old) + Δv22 = 0.4 + 0.0006125 = 0.4006125 w1(new) = w1(old) + Δw1 = 0.4 + 0.0164 = 0.4164
w2(new) = w2(old) + Δw2 = 0.1 + 0.02117 = 0.12117
v01(new) = v01(old) + Δv01 = 0.3 + 0.00295 = 0.30295
v02(new) = v02(old) + Δv02 = 0.5 + 0.0006125 = 0.5006125
w0(new) = w0(old) + Δw0 = -0.2 + 0.02978= -0.17022

b) How to choose the various parameters for a network trained by Backpropagation? **[WBUT 2016]**

Answer:
The number of layers and the number of processing elements per layer are important decisions. These parameters to a feedforward, back-propagation topology are also the most ethereal. There is no quantifiable, best answer to the layout of the network for any particular application. There are only general rules picked up over time and followed by most researchers and engineers applying this architecture to their problems.

Rule One: As the complexity in the relationship between the input data and the desired output increases, the number of the processing elements in the hidden layer should also increase.

Rule Two: If the process being modeled is separable into multiple stages, then additional hidden layer(s) may be required. If the process is not separable into stages, then additional layers may simply enable memorization of the training set, and not a true general solution effective with other data.

Rule Three: The amount of training data available sets an upper bound for the number of processing elements in the hidden layer(s). To calculate this upper bound, use the number of cases in the training data set and divide that number by the sum of the number of nodes in the input and output layers in the network. Then divide that result again by a scaling factor between five and ten. Larger scaling factors are used for relatively less noisy data. If you use too many artificial neurons the training set will be memorized. If that happens, generalization of the data will not occur, making the network useless on new data sets.

5. Using perceptron learning rule find the weights required to perform the following classifications:
Vector (1, 1, 1, 1), (–1, 1, –1, –1) and (1, –1, –1, 1) are members of class C1 (having target value 1), vectors (1, 1, 1, –1) and (1, 1, –1, 1) are members of class C2 (having target value –1). Use learning rate of 1 and starting weights of 0. Using perceptron training and vectors as input, test the response of the net.
[WBUT 2016]

Answer:
The updation is done according to perceptron learning rule,
If $y \neq t$, weight change, $\Delta w = \propto t x_i \;\&\; \Delta b = \propto t$
New weight are, $w_{(new)} = w_{(old)} + \Delta w$

$$b_{(new)} = b_{(old)} + \Delta b$$

If $t = y$, no weight change
By using the above, the below tabulation is formed, where, $y_{-in} = b + \sum_i x_i w_i$

and $y = f(y_{in})$ is the activation applied.

Input					Net Output		Target	Weight Changes					Weights				
x_1	x_2	x_3	x_4	1	y_{in}	y	t	Δw_1	Δw_2	Δw_3	Δw_4	Δb	w_1	w_2	w_3	w_4	b
													(0	0	0	0	0)
Epoch 1:																	
1	1	1	1	1	0	0	1	1	1	1	1	1	1	1	1	1	1
1	1	1	−1	1	3	1	−1	−1	−1	−1	1	−1	0	0	0	2	0
−1	1	−1	−1	1	0	0	1	−1	1	−1	−1	1	−1	1	−1	−1	1
1	−1	−1	1	1	−1	−1	−1	0	0	0	0	0	−1	1	−1	−1	1
Epoch 2:											**Initial →**		0	0	0	2	0
1	1	1	1	1	2	1	1	0	0	0	0	0	0	0	0	2	0
1	1	1	−1	1	−2	−1	−1	0	0	0	0	0	0	0	0	2	0
Initial →													−1	1	−1	−1	1
−1	1	−1	−1	1	5	0	0	0	0	0	0	0	−1	1	−1	−1	1
1	−1	−1	1	1	−1	−1	−1	0	0	0	0	0	−1	1	−1	−1	1

The final weights from Epoch 1 are used as the initial weights for Epoch 2. Thus the output is equal to target by training for suitable weights.

Testing the response of the net

The final weights are,

For the 1st set of input, $w_1 = 0, w_2 = 0, w_3 = 0, w_4 = 2, b = 0$, and

For the 2nd set of input, $w_1 = -1, w_2 = 1, w_3 = -1, w_4 = -1, b = 1$

The net input is, $y_{in} = b + \sum x_i w_i$

For the 1st set of inputs,

(i) (1 1 1 1 1)

$$y_{in1} = 0 + 0 \times 1 0 \times 1 + 0 \times 1 + 2 \times 1 = 2 > 0$$

Applying activation, $y_1 = f(y_{in1}) = 1$

(i) (1 1 1 −1 1)

$$y_{in2} = 1 + -1 \times 1 + 1 \times -1 + -1 \times -1 + -1 \times 1 + -1 \times 1 = -1 < 0$$

Applying activations, $y_2 = f(y_{in2}) = -1$

6. a) Describe the back propagation learning algorithm.
b) With a training dataset, illustrate the working of the back propagation learning algorithm.
c) What are the merits and demerits of back propagation algorithm? [WBUT 2017]

Answer:

a) The backpropagation algorithm is used in layered feed-forward ANNs. This means that the artificial neurons are organized in layers, and send their signals "forward", and then the errors are propagated backwards. The network receives inputs by neurons in the input layer, and the output of the network is given by the neurons on an output layer. There may be one or more intermediate hidden layers. The backpropagation algorithm uses supervised learning, which means that we provide the algorithm with examples of the inputs and outputs we want the network to compute, and then the error (difference

between actual and expected results) is calculated. The idea of the backpropagation algorithm is to reduce this error, until the ANN learns the training data. The training begins with random weights, and the goal is to adjust them so that the error will be minimal.

b) ***Refer to Question No. 1 of Long Answer Type Questions.***

c) ***Merits:***

1. The mathematical formula present here can be applied to any network and does not require any special mention of the features of the function to be learnt.
2. The computing time is reduced if the weights chosen are small at the beginning.

Demerits:

1. The number of learning steps may be high, and also the learning phase has intensive calculations.
2. The training may cause temporal instability to the system.

7. Using a single-node Perceptron with bipolar sigmoid function following classification problem is to be solved:

$X1 = (1, -2, 0, -1)^T$ $\quad d1 = 1$

$X2 = (0, 1.5, -0.5, -1)^T$ $\quad d2 = -1$

$X3 = (-1, 1, 0.5, -1)^T$ $\quad d3 = 1$

$W1 = (1, -1, 0, 0.5)$ $\quad \eta = 0.23$

Apply delta learning rule for one epoch to solve this problem. **[WBUT 2017]**

Answer:

In Delta Rule, we approximate the real concept for non-linearly separable perception.

For training:

$wx_1 = (1, 2.0, -0.5) \rightarrow \Sigma wx_1 = 2.5 \rightarrow (+ve) \rightarrow 1$

$wx_2 = (0, -1.5, 0, -0.5) \rightarrow \Sigma wx_2 = -2 \rightarrow (-ve) \rightarrow 0$

$wx_3 = (-1, -1, 0, -0.5) \rightarrow \Sigma wx_3 = -2.5 \rightarrow (-ve) \rightarrow 0$

$o_i \rightarrow$ target output, $t_i \rightarrow$ real output

$\therefore o_1 = 1, \; o_2 = 0, \; o_3 = 0$

$\eta = 0.2$

$d_1 = \underset{(+w)}{1} \quad d_2 = \underset{(-w)}{-1} \quad d_3 = \underset{(+w)}{1}$

$t_1 = 1, \; t_2 = 0, \; t_3 = 1$

$\therefore x_i$	o_i	t_i	$(t_i - x_i)$	Change in w_i	correctly classified
$[1, -2, 0, -1]$	1	1	0	No	yes
$[0, 1.5, -0.5, -1]$	0	0	0	No	yes
$[-1, 1, 0, 0.5]$	0	1	1	yes	No

$\therefore$ Need to correct x_3.

The changed $w_i = \eta \times (t_i - o_i) \times x_i = 0.2 \times 1 \times [-1, 1, 0, 0.5]$

$= 0.2(-1, 1, 0, 0.5) = (-0.2, 0.2, 0, 0.1)$

$\therefore w_1$ and w_2 remains the same as $W1$ and w_3 changes to [-0.2, 0.2, 0, 0.1]

8. a) What is Evaluation function? **[WBUT 2018]**

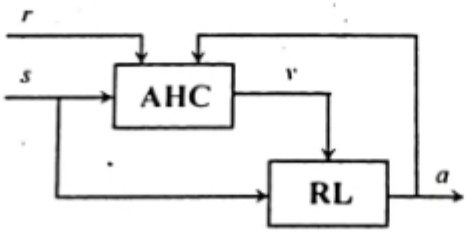
Answer:

An evaluation function, also known as a heuristic evaluation function or static evaluation function, is a function used by game-playing programs to estimate the value or goodness of a position in the minimax and related algorithms. The evaluation function is typically designed to prioritize speed over accuracy; the function looks only at the current position and does not explore possible moves

b) Explain Adaptive Heuristic Critic Learning. **[WBUT 2018]**

Answer:

The adaptive heuristic critic algorithm is an adaptive version of policy iteration in which the value-function computation is no longer implemented by solving a set of linear equations, but is instead computed by an algorithm called TD(0). A block diagram for this approach is given in Figure below. It consists of two components: a critic (labeled AHC), and a reinforcement-learning component (labeled RL). The reinforcement-learning component can be an instance of any of the k-armed bandit algorithms, modified to deal with multiple states and non-stationary rewards. But instead of acting to maximize instantaneous reward, it will be acting to maximize the heuristic value, v, that is computed by the critic. The critic uses the real external reinforcement signal to learn to map states to their expected discounted values given that the policy being executed is the one currently instantiated in the RL component.



c) Using Perceptron Convergence Theorem to find the weight vector required to perform following classifications:
Input Vector: (1, 0, 0), (1, 0, 1), (1, 1, 0), (1, 1, 1).
Decision Vector: (0, 0, 0, 1) **[WBUT 2018]**
Consider Learning Rate as unity.

Answer:

In Perceptron rule of learning weight change is given by $\Delta w_i = c(t - z)x_i$ where w_i is the weight from input i to perceptron node, c is the learning rate, t is the target for the current instance, z is the current output, and x_i is ith input

Perceptron Convergence Theorem is guaranteed to find a solution in finite time if a solution exists.

Let initial weights be 0. Since we start with random weights.we can ignore the threshold notion and use the bias as an extra available weight. (assuming -1 input). Thus, we assume a 3-input perceptron plus bias (it outputs 1 if net > 0, else 0).

Applying the learning rule, we compute the following table.

Pattern	Target	Weight Vector	Net($\sum w_i x_i$)	Output(z)	ΔWeight Vector
1 0 0 1	0	0 0 0 0	0	0	0 0 0 0
1 0 1 1	0	0 0 0 0	0	0	0 0 0 0
1 1 0 1	0	0 0 0 0	0	0	0 0 0 0
1 1 1 1	1	0 0 0 0	0	0	1 1 1 1
1 0 0 1	0	1 1 1 1	2	1	-1 0 0 -1
1 0 1 1	0	0 1 1 0	1	1	-1 0 -1 -1
1 1 0 1	0	-1 1 0 -1	-1	0	0 0 0 0
1 1 1 1	1	-1 1 0 -1	-1	0	1 1 1 1
1 0 0 1	0	0 2 1 0	0	0	0 0 0 0
1 0 1 1	0	0 2 1 0	1	1	-1 0 -1 -1
1 1 0 1	0	-1 2 0 -1	0	0	0 0 0 0
1 1 1 1	1	-1 2 0 -1	0	0	1 1 1 1
1 0 0 1	0	0 3 1 0	0	0	0 0 0 0
1 0 1 1	0	0 3 1 0	1	1	-1 0 -1 -1
1 1 0 1	0	-1 3 0 -1	1	1	-1 -1 0 -1
1 1 1 1	1	-2 2 0 -2	-2	0	1 1 1 1
1 0 0 1	0	-1 3 1 -1	-2	0	0 0 0 0
1 0 1 1	0	-1 3 1 -1	-1	0	0 0 0 0
1 1 0 1	0	-1 3 1 -1	1	1	-1 -1 0 -1
1 1 1 1	1	-2 2 1 -2	-1	0	1 1 1 1
1 0 0 1	0	-1 3 2-1	-2	0	0 0 0 0

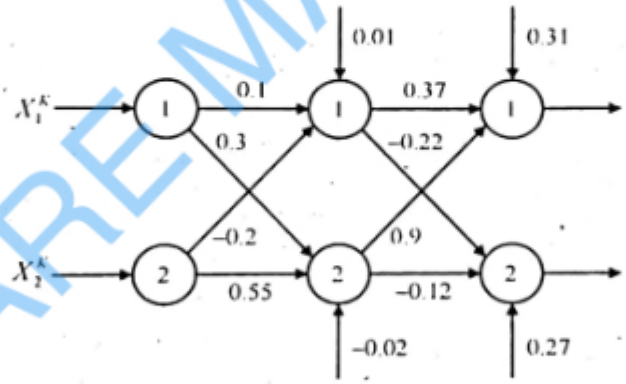
1 0 1 1	0	-1 3 2 -1	0	0	0 0 0 0
1 1 0 1	0	-1 3 2 -1	1	1	-1 -1 0 -1
1 1 1 1	1	-2 2 2 -2	0	0	1 1 1 1
1 0 0 1	0	-1 3 3 -1	-2	0	0 0 0 0
1 0 1 1	0	-1 3 3 -1	1	1	-1 0 -1 -1
1 1 0 1	0	-2 3 2 -2	-1	0	0 0 0 0
1 1 1 1	1	-2 3 2 -2	1	1	0 0 0 0
1 0 0 1	0	-2 3 2 -2	-4	0	0 0 0 0
1 0 1 1	0	-2 3 2 -2	-2	0	0 0 0 0
1 1 0 1	0	-2 3 2 -2	-1	0	0 0 0 0
1 1 1 1	1	-2 3 2 -2	1	1	0 0 0 0

So final weights are -2, 3, 2, -2.

9. Using Back Propagation algorithm find the new weights for the following network using both the pattern index. Consider the Learning rate and momentum 0.35 and 0.75 respectively.

Pattern Index	X_1^K	X_2^K	d_1^K	d_2^K
1	0.5	–0.5	0.9	0.1
2	–0.5	0.5	0.1	0.9

[WBUT 2018]



Answer:

For clarity of exposition we denote the weights from the input to hidden layer as m_{ih} and those from the hidden to the output layer as n_{hj}

Computation for Pattern 1

Table shows a single forward computation based on the presentation of Pattern 1 (iteration index $k = 1$)

A forward pass through network N^1 for Pattern 1

k	x_1^k	x_2^k	$\delta(x_1^k)$	$\delta(x_2^k)$	z_1^k	z_2^k	$\delta(z_1^k)$	$\delta(z_2^k)$	y_1^k	y_2^k	$\delta(y_1^k)$	$\delta(y_2^k)$
1	.5	-.5	.5	-.5	.16	-.145	.5399	.4638	.9271	.0955	.7164	.5238

The first step in the computation of weight changes is to compute the errors e_i^k at both output nodes:

$$e_1^1 = d_1^1 - s_1^1 = 0.9 - 0.7164 = 0.1836$$
$$e_2^1 = d_2^1 - s_2^1 = 0.1 - 0.5238 = -0.4238$$

Then compute $\delta_i^k s$ at each output neuron:

$$\delta_1^1 = 0.1836 \times 0.7164(1 - 0.7164) = 0.0373$$
$$\delta_2^1 = -0.4238 \times 0.5238(1 - 0.5238) = -0.1057$$

These deltas need to be propagated backwards to compute the scaled errors δH_h^k at a hidden neuron h as follows:

$$\delta H_1^1 = (0.0373 \times 0.37 + (-0.1057 \times -0.22)) \times 0.5399(1 - 0.5399) = 0.0092$$
$$\delta H_2^1 = (0.0373 \times 0.9 + (-0.1057 \times -0.12)) \times 0.4638(1 - 0.4638) = 0.0115$$

Using the correct deltas, we now compute the weight changes:
The following computations are done with up to 3 decimal places.

$$\Delta n_{01}^1 = 0.35 \times 0.0373 \times 1.0 = 0.013$$
$$\Delta n_{11}^1 = 0.35 \times 0.0373 \times 1.5399 = 0.007$$
$$\Delta n_{21}^1 = 0.35 \times 0.0373 \times 1.4638 = 0.006$$
$$\Delta n_{02}^1 = 0.35 \times -0.1057 \times 1.0 = -0.038$$
$$\Delta n_{12}^1 = 0.35 \times -0.1057 \times 0.5399 = -0.02$$
$$\Delta n_{22}^1 = 0.35 \times -0.1057 \times 0.4638 = -0.018$$
$$\Delta m_{01}^1 = 0.35 \times 0.0092 \times 1.0 = 0.003$$
$$\Delta m_{11}^1 = 0.35 \times 0.0092 \times 0.5 = 0.002$$
$$\Delta m_{21}^1 = 0.35 \times 0.0092 \times -0.5 = -0.002$$
$$\Delta m_{02}^1 = 0.35 \times -0.0115 \times 1.0 = 0.004$$
$$\Delta m_{12}^1 = 0.35 \times -0.0115 \times 0.5 = 0.002$$
$$\Delta m_{22}^1 = 0.35 \times 0.0115 \times -0.5 = -0.002$$

Finally, the weights are modified:

$$n_{01}^2 = 0.31 + 0.013 = 0.343$$
$$n_{11}^2 = 0.37 + 0.007 = 0.377$$
$$n_{21}^2 = 0.9 + 0.006 = 0.906$$

$n_{02}^2 = 0.27 - 0.038 = 0.232$

$n_{12}^2 = -0.22 - 0.01995 = -0.239$

$n_{22}^2 = -0.12 - 0.01747 = -0.138$

$m_{01}^2 = -0.01 + 0.00327 = 0.013$

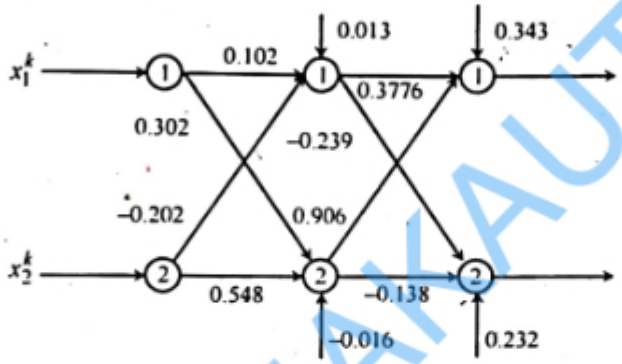
$m_{11}^2 = 0.1 + 0.00163 = 0.102$

$m_{21}^2 = -0.2 - 0.00163 = -0.202$

$m_{02}^2 = -0.02 + 0.00410 = -0.016$

$m_{12}^2 = 0.3 + 0.00205 = 0.302$

$m_{22}^2 = 0.55 - 0.00205 = 0.548$



Network N^2 after weight changes calculated from the first pattern presentation

Notice that since this is the first iteration there is no effect of the momentum term since the weight change on the previous iteration was zero.

Computation for Pattern 2

Table shows the results of a forward computation after presentation of Pattern 2

A forward pass through network N^2 for Pattern 2

k	x_1^k	x_2^k	$\delta(x_1^k)$	$\delta(x_2^k)$	z_1^k	z_2^k	$\delta(z_1^k)$	$\delta(z_2^k)$	y_1^k	y_2^k	$\delta(y_1^k)$	$\delta(y_2^k)$
2	-.5	.5	-.5	.5	.139	-.107	.465	.745	1.193	.018	.767	.505

As before, first compute the errors at each output node:

$e_1^2 = d_1^2 - s_1^2 = 0.1 - 0.767 = -0.667$

$e_2^2 = d_2^2 - s_2^2 = 0.9 - 0.505 = 0.395$

These errors are used to compute the deltas for the output and hidden weights:

$\delta_1^2 = -0.667 \times 0.767(1 - 0.767) = -0.1192$

$\delta_1^2 = 0.395 \times 0.505(1 - 0.505) = 0.099$

Once again propagating these deltas backwards yields δH_h^k for hidden units:

$\delta H_1^2 = (-0.1192 \times 0.377 + 0.099 \times -0.239) \times 0.465(1 - 0.465) = -0.017$

$$\delta H_2^2 = (-0.1192 \times 0.906 + 0.099 \times -0.138) \times 0.745(1 - 0.745) = -0.023$$

With these intermediate results we can calculate the weight changes for hidden-output neurons:

$$\Delta n_{01}^2 = 0.35 \times -0.1192 \times 1.0 = -0.0417$$
$$\Delta n_{11}^2 = 0.35 \times -0.1192 \times 0.465 = -0.019$$
$$\Delta n_{21}^2 = 0.35 \times -0.1192 \times 0.745 = -0.031$$
$$\Delta n_{02}^2 = 0.35 \times 0.099 \times 1.0 = 0.035$$
$$\Delta n_{12}^2 = 0.35 \times 0.099 \times 0.465 = -0.016$$
$$\Delta n_{22}^2 = 0.35 \times 0.099 \times 0.745 = 0.026$$
$$\Delta m_{01}^2 = 0.35 \times -0.017 \times 1.0 = -0.006$$
$$\Delta m_{11}^2 = 0.35 \times -0.017 \times -0.5 = 0.003$$
$$\Delta m_{21}^2 = 0.35 \times -0.017 \times 0.5 = -0.003$$
$$\Delta m_{02}^2 = 0.35 \times -0.023 \times 1.0 = -0.081$$
$$\Delta m_{12}^2 = 0.35 \times -0.023 \times -0.5 = 0.04$$
$$\Delta m_{22}^2 = 0.35 \times -0.023 \times 0.5 = -0.04$$

Finally, update the weights. This time, since there was a previous iteration where weight changes did take place, the momentum term comes into the update procedure:

$$n_{11}^3 = 0.377 - 0.019 + 0.75 \times 0.007 = 0.363$$
$$n_{21}^3 = 0.906 - 0.031 + 0.75 \times 0.006 = 0.88$$
$$n_{01}^3 = 0.343 - 0.0417 + 0.75 \times 0.013 = 0.311$$
$$n_{12}^3 = -0.239 + 0.015 + 0.75 \times -0.02 = -0.238$$
$$n_{22}^3 = -0.138 + 0.026 + 0.75 \times -0.018 = -0.1255$$
$$n_{02}^3 = 0.232 + 0.035 + 0.75 \times 0.038 = 0.239$$
$$m_{11}^3 = 0.102 + 0.003 + 0.75 \times 0.002 = 0.107$$
$$m_{21}^3 = -0.202 - 0.003 + 0.75 \times -0.002 = -0.207$$
$$m_{01}^3 = 0.013 - 0.006 + 0.75 \times 0.003 = 0.01$$
$$m_{12}^3 = 0.302 + 0.04 + 0.75 \times 0.002 = 0.344$$
$$m_{22}^3 = 0.548 - 0.04 + 0.75 \times -0.002 = 0.507$$
$$m_{02}^3 = -0.016 - 0.081 + 0.75 \times 0.004 = -0.094$$

This working completes one epoch of training over the pattern set

10. a) What is the advantage of using Pocket algorithm over Perceptron Convergence Theorem? **[WBUT 2018]**

Answer:

In perceptron convergence theorem the algorithm eventually finds a hyperplane that separates 2 classes of points, if such a hyperplane exists. Also, if no separating hyperplane exists, the algorithm does not have to converge and will iterate forever. In pocket algorithm one tracks the error the perceptron makes in each iteration and store the best weights found so far in a separate memory (pocket). Takes advantage of the properties of the perceptron algorithm in order to find better and better vector weights. Although the computational burden is rather high, a proper convergence theorem ensures that the pocket algorithm reaches an optimal configuration with probability one when the number of iterations grows indefinitely.

b) Determine the weights for XOR classification using RBF based Interpolation with fewer than Q basis functions.

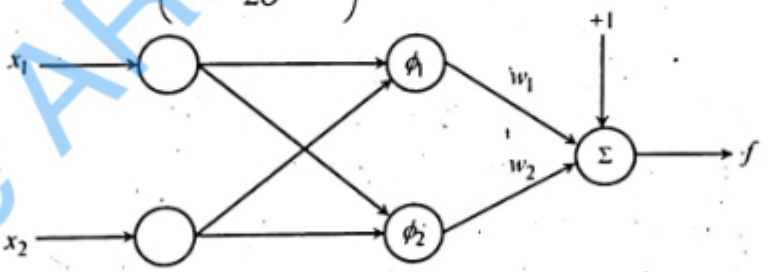
Point K	x_1	x_2	d
X_1	0	0	0
X_2	0	1	1
X_3	1	0	1
X_4	1	1	0

Consider localized Gaussian basis function where $\sigma = 1$. **[WBUT 2018]**

Answer:

$$\phi_1\left(\|X - X_1\|\right) = \exp\left(-\frac{\|X - X_1\|^2}{2\sigma^2}\right)$$

$$\phi_2\left(\|X - X_4\|\right) = \exp\left(-\frac{\|X - X_4\|^2}{2\sigma^2}\right)$$



Basic functions centered at data points 1 and 4

$$f(X) = w_0 + w_1 \exp\left(-\frac{\|X - X_1\|^2}{2}\right) + w_2 \exp\left(-\frac{\|X - X_4\|^2}{2}\right)$$

- We have the D, W, Φ vectors and matrices as shown alongside
- Pseudo inverse and weight vector

$$D = (0 \quad 1 \quad 1 \quad 0)^T$$

$$W = (w_0 \quad w_1 \quad w_2)^T$$

$$\Phi = \begin{pmatrix} 1.0000 & 1.0000 & 0.0183 \\ 1.0000 & 0.1353 & 0.1353 \\ 1.0000 & 0.1353 & 0.1353 \\ 1.0000 & 0.0183 & 1.0000 \end{pmatrix}$$

$$W = \begin{pmatrix} 1.3620 \\ -1.3375 \\ -1.3375 \end{pmatrix}$$

$$\Phi^{\dagger} = \begin{pmatrix} -0.1810 & 0.6810 & 0.6810 & -0.1810 \\ 1.1781 & -0.6688 & -0.6688 & 0.1594 \\ 0.1594 & -0.6688 & -0.6688 & 1.1781 \end{pmatrix}$$

11. Write short not on Error correction learning. **[WBUT 2018]**

Answer:

Error-Correction Learning, used with supervised learning, is the technique of comparing the system output to the desired output value, and using that error to direct the training. In the most direct route, the error values can be used to directly adjust the tap weights, using an algorithm such as the backpropagation algorithm. If the system output is y, and the desired system output is known to be d, the error signal e can be defined as:

$$e = d - y$$

Error correction learning algorithms attempt to minimize this error signal at each training iteration. The most popular learning algorithm for use with error-correction learning is the backpropagation algorithm.

12. Explain how you would train a neural network (in which each unit uses the sigmoid function as its activation function) to learn this function by describing in detail each of the steps below:

a) How many hidden layers would you use?

b) How many hidden units per layer?

c) How many connections would your net have?

d) Draw the neural net that would result from your choices above.

[MODEL QUESTION]

Answer:

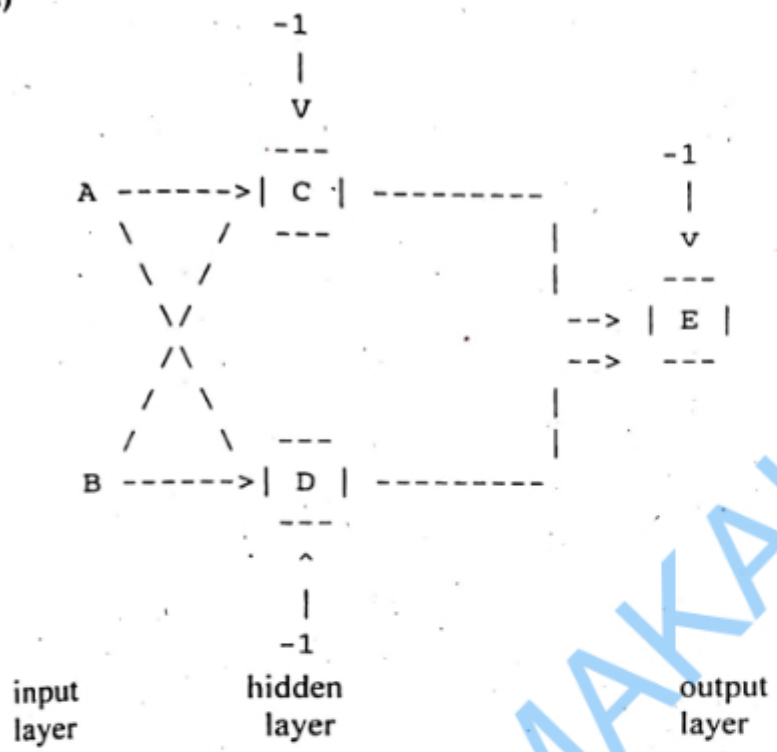
a) Only one hidden layer is needed since this is a boolean function and one hidden layer is sufficient to represent any boolean function.

b) Two hidden units since the function seems simple enough.

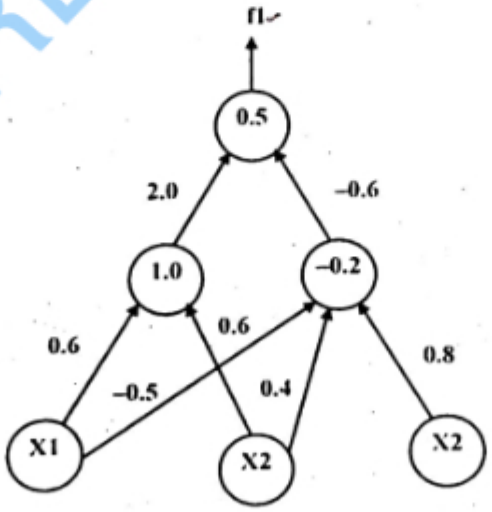
c) 4 connecting the input layer to the hidden layer + 2 connecting the hidden layer to the output layer.

The 3 links connecting the fixed input -1 to each of the units can also be taken into account here.

d)



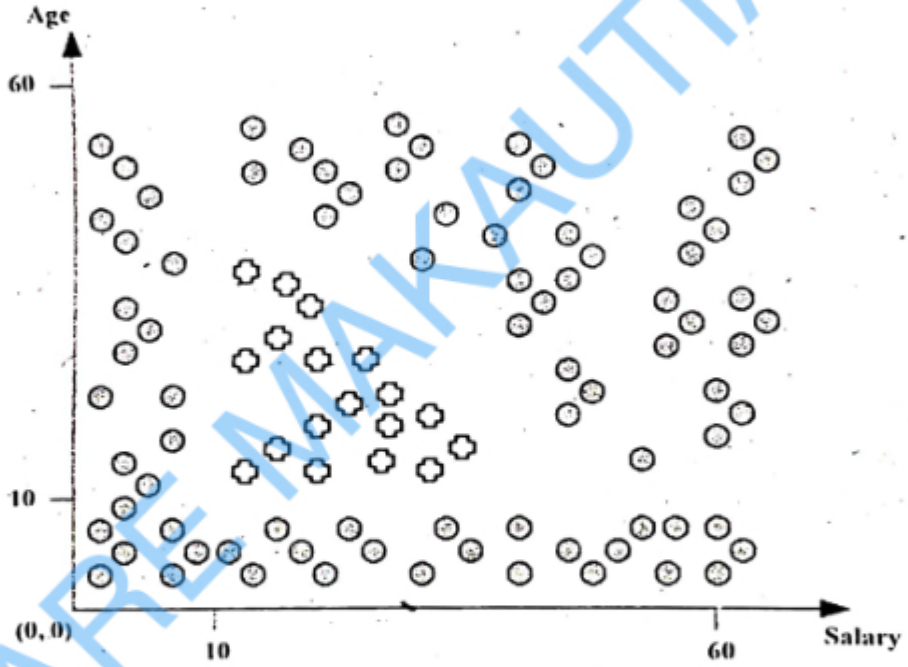
13. Given below is an artificial neural network (ANN) with three input nodes (X1, X2, X3), two hidden nodes, and one output node. The network uses simple threshold nodes (i.e., the node will output 1.0 if the sum of the weighted inputs is greater than or equal to the threshold, 0 otherwise). Show the output function f1 of this neural network. [MODEL QUESTION]



Answer:

x_0	x_1	x_2	f_1
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

14.



The diagram above classifies a group of people according to whether they enjoy playing arcade games (⬡) or not (◯). Each data point is a person surveyed, and the axes represent the age (in years) and the salary (in hourly wages) of each respondent.

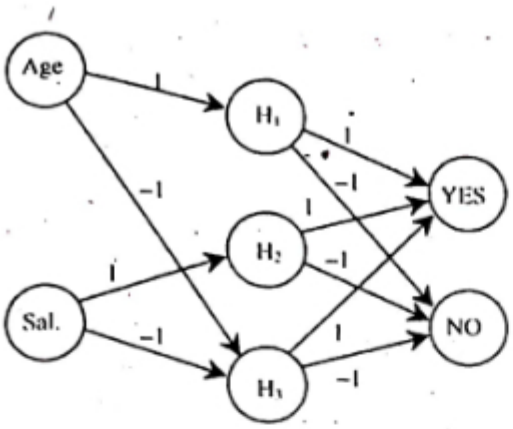
a) Can this data be classified using a neural network with no hidden layer? Explain why or why not.

b) Based on the diagram from the previous page, draw the simplest neural network that can classify this data, given inputs of age and salary, and outputs of YES (people who like games) and NO (people who don't like games). Indicate the weights and activation functions for each node in the network.

Answer:

a) This data cannot be classified using a neural network with no hidden layer (a perceptron), because perceptrons are only able to classify data that has a linear separation boundary.

b)



Activation function for H1: Unit step at x=10
Activation function for H2: Unit step at x=10
Activation function for H3: Unit step at x=-60

Activation function for YES: Unit step at x=3
Activation function for NO: Unit step at x=

RADIAL BASIS FUNCTION NETWORKS

Multiple Choice Type Questions

1. Gaussian Activation Function is used in [WBUT 2014, 2018]
a) Back Propagation b) Radial Basis Function
c) Self Organizing Feature Map d) Mexican Hat Net
Answer: (b)

2. Radial Basis Function Network (RBF) uses [WBUT 2015, 2016]
a) Sigmoidal function b) Gaussian function
c) Bipolar-Sigmoidal function d) None of these
Answer: (b)

3. In production mode, an RBF network with five hidden-layer units can create at most five different output vectors. [MODEL QUESTION]
a) True b) False
Answer: (b)

Short Answer Type Questions

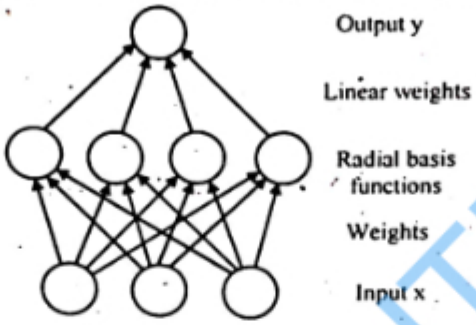
1. Discuss the architecture of radial neural network in the context of non-linearity. [WBUT 2017]

Answer:
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 $X \in \mathbb{R}^n$. The output of the network is then a scalar function of the input vector, $\varphi : \mathbb{R}^n \to \mathbb{R}$, and is given by $\varphi(X) = \sum_{i=1}^{N} a_i \rho(\|X - C_i\|)$ where N is the number of neurons in the hidden layer, C_i is the center vector for neuron i, and a_i is the weight of neuron i in the linear output neuron. Functions that depend only on the distance from a center vector are radially symmetric about that vector, hence the name radial basis function. In the basic form all inputs are connected to each hidden neuron. The norm is typically taken to be the Euclidean distance (although the Mahalanobis distance appears to perform better in general) and the radial basis function is commonly taken to be Gaussian $\rho(\|X - C_i\|) = \exp\left[-\beta \|X - C_i\|^2\right]$

The Gaussian basis functions are local to the center vector in the sense that $\lim_{\|x\| \to \infty} \rho(\|X - C_i\|) = 0$ i.e. changing parameters of one neuron has only a small effect for input values that are far away from the center of that neuron.

Given certain mild conditions on the shape of the activation function, RBF networks are universal approximators on a compact subset of $\mathbb{R}^n$. This means that an RBF network with enough hidden neurons can approximate any continuous function with arbitrary precision.

The parameters a_i, C_i and β_i are determined in a manner that optimizes the fit between φ and the data.



2. a) What are differences between Perceptron algorithm and RBF? [WBUT 2018]

Answer:

Radial basis function (RBF) networks are feed-forward networks trained using a supervised training algorithm. They are typically configured with a single hidden layer of units whose activation function is selected from a class of functions called basis functions. While similar to back propagation in many respects, radial basis function networks have several advantages. They usually train much faster than back propagation networks. They are less susceptible to problems with non-stationary inputs because of the behavior of the radial basis function hidden units.

The major difference between RBF networks and back propagation networks (that is, multi layer perceptron trained by Back Propagation algorithm) is the behavior of the single hidden layer. Rather than using the sigmoidal or S-shaped activation function as in back propagation, the hidden units in RBF networks use a Gaussian or some other basis kernel function. Each hidden unit acts as a locally tuned processor that computes a score for the match between the input vector and its connection weights or centers. In effect, the basis units are highly specialized pattern detectors. The weights connecting the basis units to the outputs are used to take linear combinations of the hidden units to product the final classification or output.

b) What is pseudo-inverse matrix in Interpolation based RBF function? [WBUT 2018]

Answer:

In RBF learning algorithm, are computed by means of the pseudo-inverse method.–For an example (x_i, d_i) consider the output of the network

$$y(x_i) \approx w_1\phi_1\left(\|x_i - t_1\|\right) + \cdots w_{m1}\phi_{m1}\left(\|x_i - t_{m1}\|\right)$$

We would like for each example $y(x_i)=d_i$, that is

$$w_1\phi_1\left(\|x_i - t_1\|\right) + \cdots w_{m1}\phi_{m1}\left(\|x_i - t_{m1}\|\right) \approx d_i$$

This can be re-written in matrix form for one example

$$\left[\phi_1(\|x_i - t_1\|)\ldots\phi_{m1}(\|x_i - t_{m1}\|)\right][w_1\ldots w_{m1}]^T = d_i$$

and

$$\begin{bmatrix} \phi_1(\|x_i - t_1\|)\ldots\phi_{m1}(\|x_i - t_{m1}\|) \\ \cdots \\ \phi_1(\|x_N - t_1\|)\ldots\phi_{m1}(\|x_N - t_{m1}\|) \end{bmatrix}[w_1\ldots w_{m1}]^T = [d_1\ldots.d_N]^T$$

for all the examples at the same time
Let

$$\Phi = \begin{bmatrix} \phi_1(\|x_1 - t_1\|) & \ldots & \phi_{m1}(\|x_1 - t_{m1}\|) \\ & \ldots.. & \\ \phi_1(\|x_N - t_1\|) & \ldots & \phi_{m1}(\|x_N - t_{m1}\|) \end{bmatrix}$$

then we can write

$$\Phi\begin{bmatrix} w_1 \\ \cdots \\ w_{m1} \end{bmatrix} = \begin{bmatrix} d_1 \\ \cdots \\ d_N \end{bmatrix}$$

If Φ^+ is the pseudo-inverse of the matrix we obtain the weights using the following formula

$$[w_1\ldots w_{m1}]^T = \Phi^+[d_1 - d_N]^T$$

3. What are the main features of radial basic function networks?

[MODEL QUESTION]

Answer:

The idea of Radial Basis Function (RBF) Networks derives from the theory of function approximation. Their main features are:

1. They are two-layer feed-forward networks.
2. The hidden nodes implement a set of radial basis functions (e.g. Gaussian functions).
3. The output nodes implement linear summation functions as in an MLP.
4. The network training is divided into two stages: first the weights from the input to hidden layer are determined, and then the weights from the hidden to output layer.
5. The training/learning is very fast.
6. The networks are very good at interpolation.

Long Answer Type Questions

1. What activation function is used in Radial Basic function? Write down and explain, the flow-chart of the training algorithm of RBF. Briefly explain the Wavelet Neural Network. **[WBUT 2014]**

Answer:

Radial basis function (RBF) is based on Gaussian Curve. It takes a parameter that determines the center (mean) value of the function used as a desired value (see Fig. 4). A radial function (RBF) is a real-valued function whose value depends only on the distance from the origin, so that $g(x) = g(|x|)$

or alternatively on the distance from some other point c, called a center, so that $g(x, c) = g(|x - c|)$.

Sums of radial basis functions are typically used to approximate given functions. This approximation process can also be interpreted as a simple type of neural network. RBF are typically used to build up function approximations of the form

$y(x) = \sum w_i g(|x - c|)$ where the approximating function $y(x)$ is represented as a sum of N radial basis functions, each associated with a different center c_i, and weighted by an appropriate coefficient w_i. The weights w_i can be estimated using the matrix methods of linear least squares, because the approximating function is linear in the weights.

The flowchart of training algorithm of RBF is given below:

The training is started in the hidden layer with an unsupervised learning algorithm. The training is continued in the output layer with a supervised learning algorithm. Simultaneously, we can apply supervised learning algorithm to the hidden and output layers for fine-tuning of the network. The training algorithm is given as follows.

Step 0: Set the weights to small random values.

Step 1: Perform Steps 2–8 when the stropping condition is false.

Step 2: Perform Steps 3–7 for each input.

Step 3: Each input unit (x_i for all $i = 1$ to n) receives input signals and transmits to the next hidden layer unit.

Step 4: Calculate the radial basis function.

Step 5: Select the centers for the radial basis function. the centers are selected from the set of input vectors. It should be noted that a sufficient number of centers have to be selected to ensure adequate sampling of the input vector space.

Step 6: Calculate the output from the hidden layer unit:

$$v_i(x_i) = \frac{\exp\left[-\sum_{j=1}^{l}(x_{ji} - \hat{x}_{ji})^2\right]}{\sigma_i^2}$$

where $\hat{x}_{ji}$ = center of the RBF unit for input variables

σ_i = width of i^{th} RBF unit

x_{ji} = j^{th} variable of input pattern.

Step 7: Calculate the output of the neural network:

$$y_{net} = \sum_{i=1}^{k} w_{im} v_i(x_i) + w_0$$

where k = number of hidden layer nodes (RBF function)

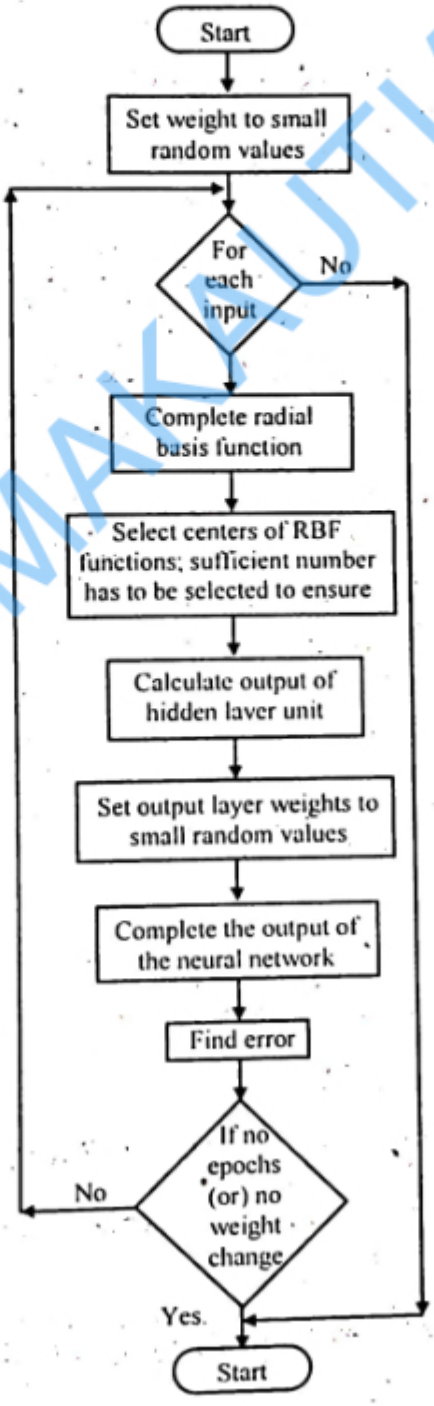
y_{net} = output value of m^{th} node in output layer for the n^{th} incoming pattern

w_{im} = weight between i^{th} RBF unit and m^{th} output node

w_0 = biasing term at n^{th} output node

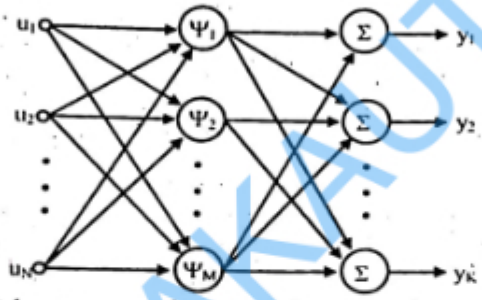
Step 8: Calculate the error and test for the stopping condition. The stopping condition may be number of epochs or to a certain extent weight change.

Thus, a network can be trained using RBFN.



Wavelet neural networks combine the theory of wavelets and neural networks into one. A wavelet neural network generally consists of a feed-forward neural network, with one hidden layer, whose activation functions are drawn from an orthonormal wavelet family. One application of wavelet neural networks is that of function estimation. Given a series of observed values of a function, a wavelet network can be trained to learn the composition of that function, and hence calculate an expected value for a given input.

The structure of a wavelet neural network is very similar to that of a $\left(1+\frac{1}{2}\right)$ layer neural network. That is, a feed-forward neural network, taking one or more inputs, with one hidden layer and whose output layer consists of one or more linear combiners or summers (Figure shown below). The hidden layer consists of neurons, whose activation functions are drawn from a wavelet basis. These wavelet neurons are usually referred to as wavelons.



There are two main approaches to creating wavelet neural networks.

- In the first the wavelet and the neural network processing are performed separately. The input signal is first decomposed using some wavelet basis by the neurons in the hidden layer. The wavelet coefficients are then output to one or more summers whose input weights are modified in accordance with some learning algorithm.
- The second type combines the two theories. In this case the translation and dilation of the wavelets along with the summer weights are modified in accordance with some learning algorithm

2. Describe the architecture and training process of Radial Basis Function Network. **[WBUT 2015]**

Answer:

Refer to Question No. 1(1st & 2nd Part) of Long Answer Type Questions.

3. a) Write down the learning process of Radial Basis function network. **[WBUT 2016]**

Answer:

Refer to Question No. 1(2nd Part) of Long Answer Type Questions.

b) What are the differences between RBF and MLP? **[WBUT 2016]**

Answer:

- **MLP:** uses dot products (between inputs and weights) and sigmoidal activation functions (or other monotonic functions such as ReLU) and training is usually done through backpropagation for all layers (which can be as many as you want). This type of neural network is used in deep learning with the help of many techniques (such as dropout or batch normalization);
- **RBF:** uses Euclidean distances (between inputs and weights, which can be viewed as centers) and (usually) Gaussian activation functions (which could be multivariate), which makes neurons more locally sensitive. Thus, RBF neurons have maximum activation when the center/weights are equal to the inputs (look at the image below). Due to this property, RBF neural networks are good for novelty detection (if each neuron is centered on a training example, inputs far away from all neurons constitute novel patterns) but not so good at extrapolation. Also, RBFs may use backpropagation for learning, or hybrid approaches with unsupervised learning in the hidden layer (they usually have just 1 hidden layer). Finally, RBFs make it easier to grow new neuronsduring training.

4. Write short note on Wavelet neural network. **[WBUT 2015, 2016, 2018]**

Answer:

Refer to Question No. 1(3rd Part) of Long Answer Type Questions.

ASSOCIATIVE MEMORY NETWORKS

Multiple Choice Type Questions

1. Sigmoid is a _____ type of non-linearity. [WBUT 2014, 2016, 2018]
   a) hard b) soft c) rough d) soft and rough
Answer: (b)

2. An auto-associative network is [WBUT 2014, 2017, 2018]
   a) a neural network that contains no loops
   b) a neural network that contains feedback
   c) a neural network that has only one loop
   d) none of these
Answer: (b)

3. Discrete Hopfield net is the example of [WBUT 2015, 2016, 2018]
   a) feed forward network b) feedback network
   c) recurrent network d) both (b) and (c)
Answer: (c)

4. A Hopfield network has 20 units. How many adjustable parameters does this network contain? [WBUT 2015]
   a) 95 b) 190 c) 200 d) 380
Answer: (b)

5. The input-output patterns of Hebb Net is? [WBUT 2016]
   a) Binary b) Bipolar c) Any value
Answer: (c)

6. Which of the following is true? [WBUT 2017]
Single layer associative neural networks do not have the ability to:
(i) perform pattern recognition
(ii) find the parity of a picture
(iii) determine whether two or more shapes in a picture are connected or not
   a) (II) and (III) are true b) (II) is true
   c) (I) and (III) are true d) all of these are true
Answer: (a)

7. The Error Correction learning is.....type of learning [WBUT 2017]
   a) supervised b) unsupervised c) reinforced d) competitive
Answer: (a)

8. A Hopfield network has 20 units. How many adjustable parameters does this network contain? [WBUT 2017]
   a) 95 b) 190 c) 200 d) 380
Answer: (b)

9. The Hopfield Network has only a single layer of neurons. [MODEL QUESTION]
a) True b) False
Answer: (a)

10. After training a Self-Organizing Map, output neurons that win for similar inputs are usually far apart from each other in the map. [MODEL QUESTION]
a) True b) False
Answer: (b)

11. A Boltzmann machine where all the variables are observable can only capture second order statistics (means and covariances) between the variables. [MODEL QUESTION]
a) True b) False
Answer: (a)

12. How many hidden layers are there in an autoassociative Hopfield network? [MODEL QUESTION]
a) None (0) b) One (1) c) Two (2) d) Three (3)
Answer: (a)

13. Which of the following statements is NOT true for a self-organizing feature map (SOFM)? [MODEL QUESTION]
a) The size of the neighbourhood is decreased during training
b) The SOFM training algorithm is based on soft competitive learning
c) The network can grow during training by adding new cluster units when required
d) The cluster units are arranged in a regular geometric pattern such as a square or ring
e) The learning rate is a function of the distance of the adapted units from the winning unit
Answer: (c)

14. Which application in intelligent mobile robots made use of a self-organizing feature map? [MODEL QUESTION]
a) Goal finding b) Path planning
c) Wall following d) Route following
e) Gesture recognition
Answer: (d)

Short Answer Type Questions

1. Describe the discrete Hopfield Net and its training algorithm. [WBUT 2014, 2018]
Answer:
A **Hopfield network** is a form of recurrent artificial neural network invented by John Hopfield in 1982. Hopfield nets serve as content-addressable memory systems with binary threshold nodes. They are guaranteed to converge to a local minimum, but convergence to a false pattern (wrong local minimum) rather than the stored pattern

(expected local minimum) can occur. Hopfield networks also provide a model for understanding human memory.

The units in Hopfield nets are binary threshold units, i.e. the units only take on two different values for their states and the value is determined by whether or not the units' input exceeds their threshold. Hopfield nets normally have units that take on values of 1 or -1, and this convention will be used throughout the article. However, other literature might use units that take values of 0 and 1.

Training a Hopfield net involves lowering the energy of states that the net should "remember". This allows the net to serve as a content addressable memory system, that is to say, the network will converge to a "remembered" state if it is given only part of the state. The net can be used to recover from a distorted input to the trained state that is most similar to that input. This is called associative memory because it recovers memories on the basis of similarity. For example, if we train a Hopfield net with five units so that the state (1, 0, 1, 0, 1) is an energy minimum, and we give the network the state (1, 0, 0, 0, 1) it will converge to (1, 0, 1, 0, 1). Thus, the network is properly trained when the energy of states which the network should remember are local minima.

2. What is a Hopfield net? State the types of Hopfield network. What is the energy function of discrete Hopfield net? **[WBUT 2015]**

Answer:

1st Part:

A Hopfield network is a form of recurrent artificial neural network invented by John Hopfield in 1982. Hopfield nets serve as content-addressable memory systems with binary threshold nodes. They are guaranteed to converge to a local minimum, but convergence to a false pattern (wrong local minimum) rather than the stored pattern (expected local minimum) can occur. Hopfield networks also provide a model for understanding human memory.

2nd Part:

There are two types of operations: auto-association and hetero-association. The first being when a vector is associated with itself, and the latter being when two different vectors are associated in storage.

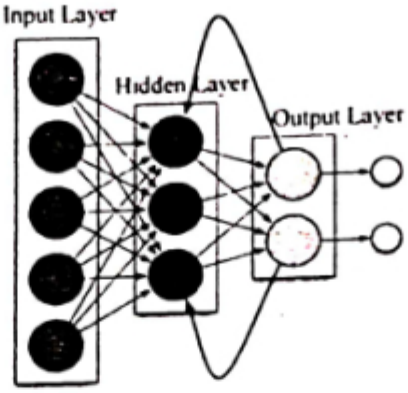
3rd Part: $E = -0.5\left[\sum_{i=1}^{N}\sum_{j=1}^{N} w_{ij}x_i x_j\right.$

3. With a diagram compare multilayer feed forward and recurrent network. **[WBUT 2017]**

Answer:

A recurrent neural network (RNN) is a class of artificial neural network where connections between units form a directed cycle. This allows it to exhibit dynamic temporal behavior. Unlike feedforward neural networks, RNNs can use their internal memory to process arbitrary sequences of inputs. This makes them applicable to tasks such as unsegmented, connected handwriting recognition or speech recognition.

The diagram describes a RNN.



4. What problem could occur in SOM learning if you use a very small neighborhood? Why? **[MODEL QUESTION]**

Answer:

It is possible that neurons in separate areas of the output layer become selective for the same type of input. Because there are no global interactions between neurons in the output layer, separate "islands" of similar selectivity can develop without ever "meeting" each other and merging.

5. Hopfield networks are most often used for auto-association. What does that mean, and why can this be useful at all? **[MODEL QUESTION]**

Answer:

Auto-association is a network's ability for a certain number of stored patterns, whenever they are input to the network, to produce the same pattern as its output. This is useful if the network is still able to output precisely the stored pattern even if the input pattern is noisy, i.e., slightly deviates from the stored one. In this case, the network can be used to remove noise from known input patterns.

6. Describe the relationship between the Self-Organising Map algorithm, and the Learning Vector Quantisation algorithm. **[MODEL QUESTION]**

Answer:

In order to use Learning Vector Quantisation (LVQ), a set of approximate reconstruction vectors is first found using the unsupervised SOM algorithm. The supervised LVQ algorithm is then used to fine-tune the vectors found using SOM.

7. Briefly explain sigmoid neurons. **[MODEL QUESTION]**

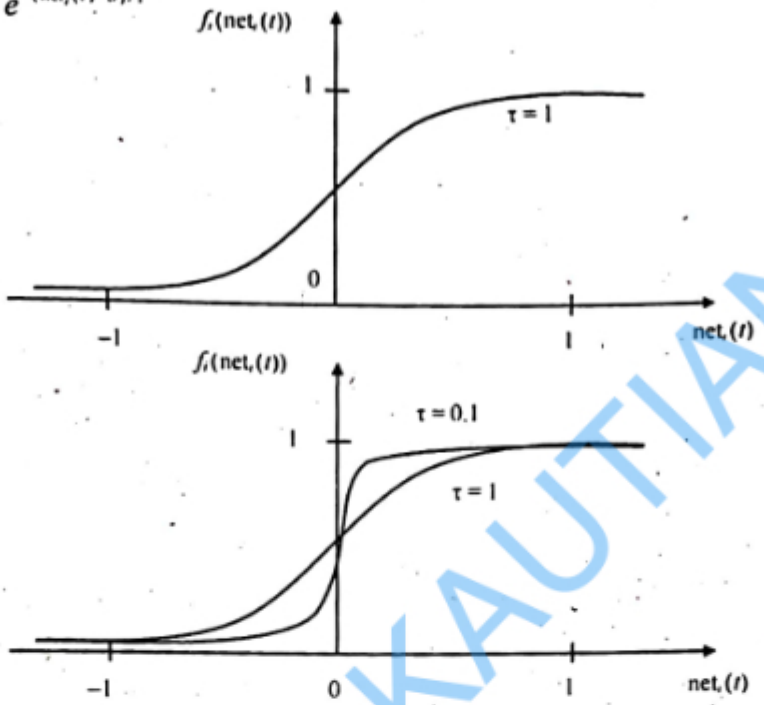
Answer:

Sigmoidal neurons accept any vectors of real numbers as input, and they output a real number between 0 and 1.

Sigmoidal neurons are the most common type of artificial neuron, especially in learning networks.

A network of sigmoidal units with m input neurons and n output neurons realizes a network function $f: R^m \rightarrow (0,1)^n$

$$f_i(\text{net}_i(t)) = \frac{1}{1+e^{-(\text{net}_i(t)-\theta)/\tau}}$$



The parameter τ controls the slope of the sigmoid function, while the parameter θ controls the horizontal offset of the function in a way similar to the threshold neurons.
In backpropagation networks, we typically choose $\tau = 1$ and $\theta = 0$.

8. List two difference between Hopfield and iterative auto associative net.
[MODEL QUESTION]

Answer:
The two main differences between Hopfield and iterative auto associative net are that, in the Hopfield net

- Only one unit updates its activation at a time,
- Each unit continues to receive an external signal in addition to the signal from the other units in the net.

Long Answer Type Questions

1. a) Describe Hebb network training algorithm using flowchart.
[WBUT 2014, 2015, 2017]

Answer:
The training algorithm of Hebb network is given below:
Step 0: First initialize the weights. Basically in this network they may be set to zero, i.e., $w_i = 0$ for $i = 1$ to n where "n" may be the total number of input neutrons.
Step 1: Steps 2-4 have to be performed for each input training vector and target output pair, $s : t$.

Step 2: Input units activations are set. Generally the activation function of input layer is identity function.

$$x_i = s_i \quad \text{for } i = 1 \text{ to } n.$$

Step 3: Output units activations are set.

$$y = t$$

Step 4: Weight adjustments and bias adjustments are performed:

$$w_i(\text{new}) = w_i(\text{old}) + x_i y$$

$$b(\text{new}) = b(\text{old}) + y$$

the above five steps complete the algorithmic process. In Step 4, the weight updation formula can also be given in vector form as

$$w(\text{new}) = w(\text{old}) + xy$$

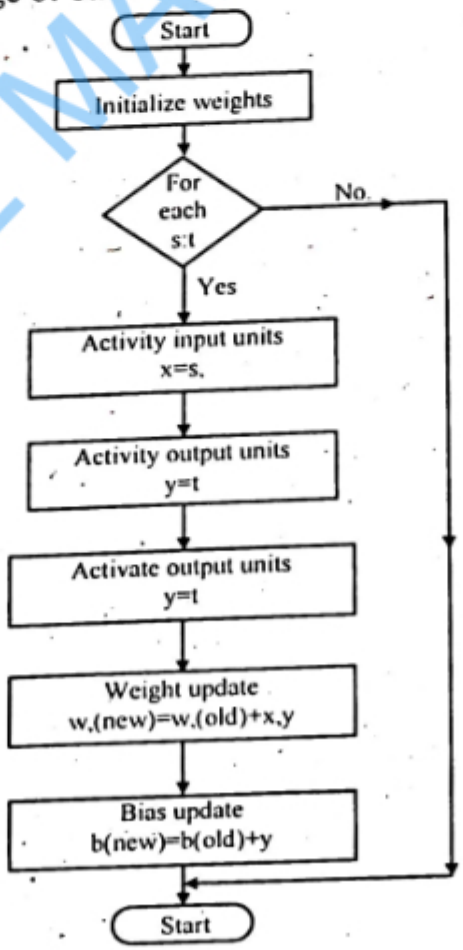
Here the change in weight can be expressed as

$$\Delta w = xy$$

As a result,

$$w(\text{new}) = w(\text{old}) + \Delta w$$

The Hebb rule can be used for pattern association, pattern categorization, pattern classification and over a range of other areas.

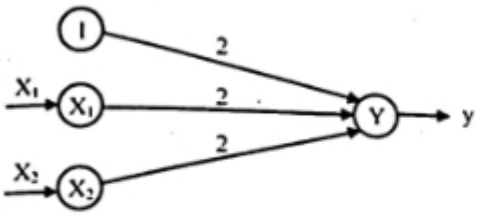


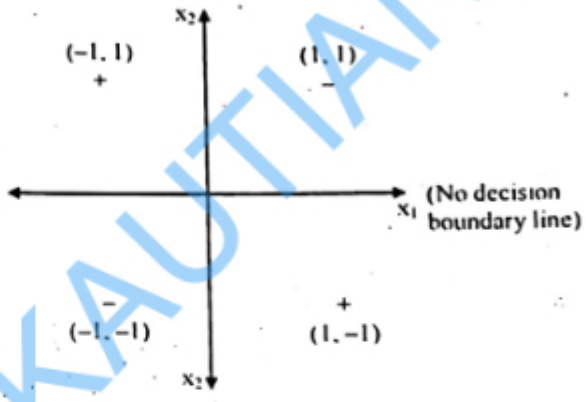
b) Use the Hebb rule method to implement XOR function (take bipolar inputs). **[WBUT 2014]**

Answer:

The training patterns for an XOR function are shown below:

Inputs			Target
x_1	x_2	b	y
1	1	1	−1
1	−1	1	1
−1	1	1	1
−1	−1	1	−1





Decision boundary for XOR function

Here, a single-layer network with two input neutrons, one bias and one output neutron is considered.

In this case also, the initial weights are assumed to be zero.

$w_1 = w_2 = b = 0$

By using the Hebb training algorithm, the network is trained and the final weights are calculated as shown in the following table.

Inputs				Weight changes			Weights		
x_1	x_2	b	y	Δw_1	Δw_2	Δb	w_1	w_2	b
							(0	0	0)
1	1	1	−1	−1	−1	−1	−1	−1	−1
1	−1	1	1	1	−1	1	0	−2	0
−1	1	1	1	−1	1	1	−1	−1	1
−1	−1	1	−1	1	1	−1	0	0	0

2. Show and analyse the architecture of Discrete Bidirectional Associative Memory (DBAM) network. Explain how weights are determined? List the activation functions used in DBAM. Also explain the testing algorithm of DBAM. [WBUT 2014]

Answer:

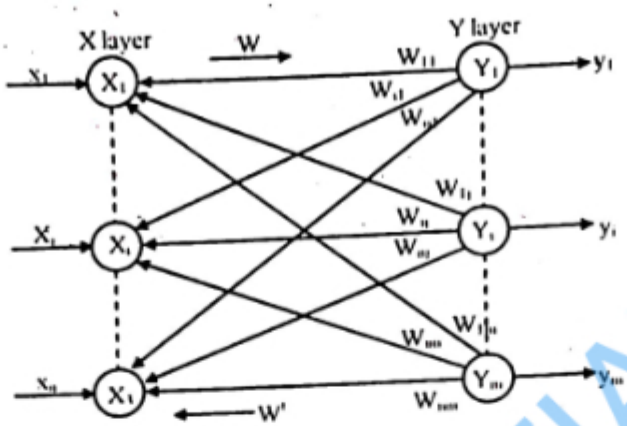


Fig: 1

The structure of discrete BAM is same as shown in Fig. 1. When the memory neutrons are being activated by putting an initial vector at the input of a layer, the network evolves a two-pattern stable state with each pattern at the output of one layer. Thus, the network involves two layers of interaction between each other. The two bivalent forms of BAM are found to be related with each other, i.e., binary and bipolar. The weights in both the cases are found as the sum of the outer products of the bipolar form of the given training vector pairs. In case of BAM, a definite nonzero threshold is assigned. Thus, the activation function is a step function, with the defined nonzero threshold. When compared, to the binary vectors, bipolar vectors improve the performance of the net to a large extent.

Determination on Weights: Let the input vectors be denoted by $s(p)$ and target vectors by $t(p), p = 1, \ldots\ldots P$. Thus the weight matrix to store a set of input and target vectors, where

$$s(p) = (s_1(p), \ldots\ldots, s_i(p), \ldots\ldots, s_n(p)),$$

$$t(p) = (t_1(p), \ldots\ldots, t_j(p), \ldots\ldots, t_m(p)),$$

can be determined by Hebb rule training algorithm.

In case of input vectors being binary, the weight matrix $W = \{w_{ij}\}$ is given by

$$w_{ij} = [2s_i(p) - 1][2t_j(p) - 1]$$

On the other hand, when the input vectors are bipolar, the weight matrix $W = \{w_{ij}\}$ can be defined as,

$$w_{ij} = \sum_{p=1}^{P} s_i(p) t_j(p)$$

The weights matrix in both the cases is going to be in bipolar form neither the input vectors are in binary or not. The formulas mentioned above can be directly applied to the determination of weights of a BAM.

Activation Functions of BAM: The step activation function with a nonzero threshold is used as the activation function for discrete BAM network. The activation function is based on whether the input target vector pairs used are binary or bipolar.

The activation function for the Y layer.

1. with binary input vectors is

$$y_j = \begin{cases} 1 \text{ if } y_{inj} > 0 \\ y_j \text{ if } y_{inj} = 0 \\ 0 \text{ if } y_{inj} < 0 \end{cases}$$

2. with bipolar input vectors is

$$y_j = \begin{cases} 1 \text{ if } y_{inj} > \theta_j \\ y_j \text{ if } y_{inj} = \theta_j \\ -1 \text{ if } y_{inj} < \theta_j \end{cases}$$

The activation function for the X layer

1. with binary input vectors is

$$x_i = \begin{cases} 1 \text{ if } x_{ini} > 0 \\ x \text{ if } x_{ini} = 0 \\ -1 \text{ if } x_{ini} < 0 \end{cases}$$

2. with bipolar input vectors is

$$x_i = \begin{cases} 1 \text{ if } x_{ini} > \theta_i \\ x \text{ if } y_{inj} = \theta_i \\ -1 \text{ if } y_{ini} < \theta_i \end{cases}$$

It may be noted that if the threshold value is equal to that of the net input calculated, then the previous output value calculated is left as the activation of that unit. At a particular time instant, signals are sent only from one layer to the other and not in both the directions.

Testing Algorithm for Discrete BAM: The testing algorithm is used to test the noisy patterns entering into the network. Based on the training algorithm, weights are determined, by means of which net input is calculated for the given test pattern and activations is applied over it, to recognize the test patterns. The testing algorithm for the net is as follows:

Step 0: Initialize the weights to store p vectors. Also initialize all the activations to zero.

Step 1: Perform Steps 2–6 for each testing input.

Step 2: Set the activations of X layer to current input pattern, i.e., presenting the input pattern x to X layer and similarly presenting the input pattern y to Y layer. Even through, it is bi-directional memory, at one time step, signals can be sent from only one layer. So, either of the input patterns may be the zero vector.

Step 3: Perform Steps 4–6 when the activations are not converged.

Step 4: Update the activations of units in Y layer. Calculate the net input.

$$y_{in_j} = \sum_{i=1}^{n} x_i w_{ij}$$

Applying the activation, we obtain

$$y_j = f\left(y_{in_j}\right)$$

Send this signal to the X layer.

Step 5: Update the activations of units in X layer.
Calculate the net input.

$$x_{in_j} = \sum_{j=1}^{m} y_j w_{ij}$$

Apply the activations over the net input.

$$x_i = f\left(x_{in_i}\right)$$

Send this signal to the Y layer.
Step 6: Test for convergence of the net. The convergence occurs if the activation vectors x and y reach equilibrium. If this occurs then stop, otherwise, continue.

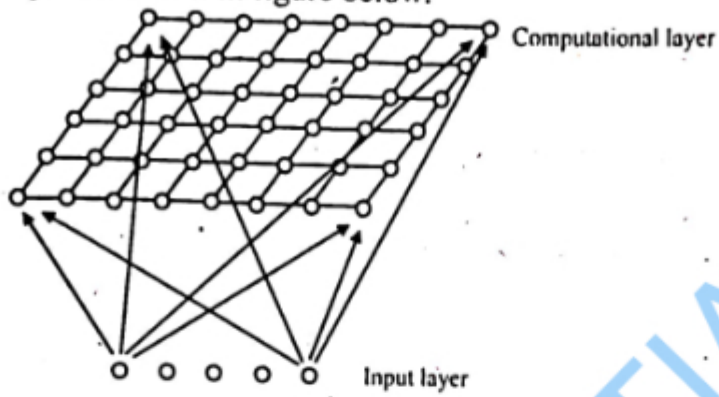
3. a) What do you understand by Kohonen Self-organization Feature Map? Draw and explain its architecture.
b) Construct a Kohonen Self-organization Feature map to cluster four vectors [0 0 1 1], [1 0 0 1] [0 1 0 1], [1 1 1 1].
The maximum number of clusters to be found is 2 and assume random initial weights. **[WBUT 2014]**

OR.

a) Explain the Kohonen Self-organization Feature Map architecture with diagram.
b) Construct a Kohonen Self-organization Feature map to cluster four vectors [0 0 1 1], [10 0 1], [010 1], [1111] into two classes and start with any random weights. **[WBUT 2018]**

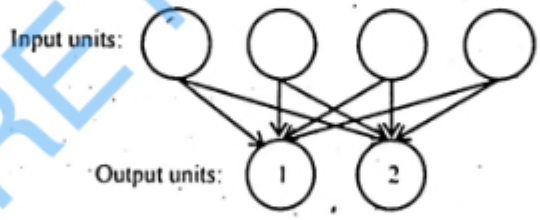
Answer:
a) Kohonen Self-Organizing Maps (or just Self-Organizing Maps, or SOMs for short) are a type of neural network. Self-Organizing Maps are aptly named. "Self-Organizing" is because no supervision is required. SOMs learn on their own through unsupervised competitive learning. "Maps" is because they attempt to map their weights to conform to the given input data. The nodes in a SOM network attempt to become like the inputs presented to them. In this sense, this is how they learn. They can also be called "Feature Maps", as in Self-Organizing Feature Maps. Retaining principle 'features' of the input data is a fundamental principle of SOMs, and one of the things that makes them so valuable. Specifically, the topological relationships between input data are preserved when mapped to a SOM network.

Architecture: This has a feed-forward structure with a single computational layer of neurons arranged in rows and columns. Each neuron is fully connected to all the source units in the input layer as shown in figure below.



Each node has a specific topological position (an x, y coordinate in the lattice) and contains a vector of weights of the same dimension as the input vectors. That is to say, if the training data consists of vectors, V_i of n dimensions: $V_1, V_2, V_3, \ldots\ldots, V_n$.
Then each node will contain a corresponding weight vector W of n dimensions: $W_1, W_2, W_3, \ldots\ldots, W_n$.
The lines connecting the nodes in the figure are only there to represent adjacency and do not signify a connection as normally indicated when discussing a neural network. There are no lateral connections between nodes within the lattice.

b) The network architecture is as shown below:



Training samples
i1: (0, 0, 1, 1)
i2: (1, 0, 0, 1)
i3: (0, 1, 0, 1)
i4: (1, 1, 1, 1)

Euclidean Distances Between Data Samples

	i1	i2	i3	i4
i1	0			
i2	2	0		
i3	2	2	0	
i4	2	2	2	0

Let neighborhood = 0
- Only update weights associated with winning output unit (cluster) at each iteration.
Learning rate $\eta(t) = 0.6$; $1 <= t <= 4$ $\eta(t) = 0.5\ \eta(1)$; $5 <= t <= 8$ $\eta(t) = 0.5\ \eta(5)$; $9 <= t <= 12$ etc.

Initial weight matrix (random values between 0 and 1)

$$\begin{cases}\text{Unit 1:} \\ \text{Unit 2:}\end{cases}\begin{bmatrix}.2 & .6 & .5 & .9 \\ .8 & .4 & .7 & .3\end{bmatrix}$$

$d^2 = (\text{Euclidean distance})^2 = \sum_{k=1}^{n} (i_{l,k} - w_{j,k}(t))^2$

Weight update: $w_j(t+1) = w_j(t) + \eta(t)(i_l - w_j(t))$

We now calculate the weight updates for the first four st eps

Training sample: i1

– Unit 1 weights • d2 = $(.2-0)^2 + (.6-0)^2 + (.5-1)^2 + (.9-1)^2 = .66$
– Unit 2 weights • d2 = $(.8-0)^2 + (.4-0)^2 + (.7-1)^2 + (.3-1)^2 = 1.18$
– Unit 1 wins
– Weights on winning unit are updated
– Giving an updated weight matrix:
new unit1 weights
= [.2 .6 .5 .9] + 0.6([0 0 1 1] – [.2 .6 .5 .9])
= [.08 .24 .8 .96]

Unit1: .08 .24 .8 .96
Unit2: .8 .4 .7 .3

Training sample: i2

– Unit 1 weights • d2 = $(.08-1)^2 + (.24-0)^2 + (.8-0)^2 + (.96-1)^2 = .7032$
– Unit 2 weights • d2 = $(.8-1)^2 + (.4-0)^2 + (.7-0)^2 + (.3-1)^2 = 1.18$
– Unit 1 wins
– Weights on winning unit are updated
– Giving an updated weight matrix:
new– unit1 – weights
= [.08 .24 .8 .96] + 0.6([1 0 0 1] – [.08 .24 .8 .96])
= [.2 .096 .32 .936]

Unit1: .2 .096 .32 .936
Unit2: .8 .4 .7 .3

Training sample: i3

– Unit 1 weights • d2 = $(.2-0)^2 + (.096-1)^2 + (.32-0)^2 + (.936-1)^2 = .146416$
– Unit 2 weights • d2 = $(.8-0)^2 + (.4-1)^2 + (.7-0)^2 + (.3-1)^2 = 1.98$
– Unit 1 wins
– Weights on winning unit are updated
– Giving an updated weight matrix:
new – unit1 – weights
= [.2 .096 .32 .936] + 0.6([0 1 0 1] – [.2 .096 .32 .936]

= [.08 .0424 .128 .9744]

Unit1: .08 .0424 .128 .9744
Unit2: .8 .4 .7 .3

Training sample: i4
– Unit 1 weights • d2 = $(.08-1)^2 + (.0424-1)^2 + (.128-1)^2 + (.9744-1)^2 = 1.6781$
– Unit 2 weights • d2 = $(.2-1)^2 + (.6-1)^2 + (.5-1)^2 + (.9-1)^2 = 1.06$
– Unit 2 wins
– Weights on winning unit are updated
– Giving an updated weight matrix:
new – unit2 – weights
= [.8 .4 .7 .3] + 0.6([1 1 1 1] – [.8 .4 .7 .3])
= [.92 .76 .88 .72]

Unit1: .08 .0424 .128 .9744
Unit2: .92 .76 .88 .72

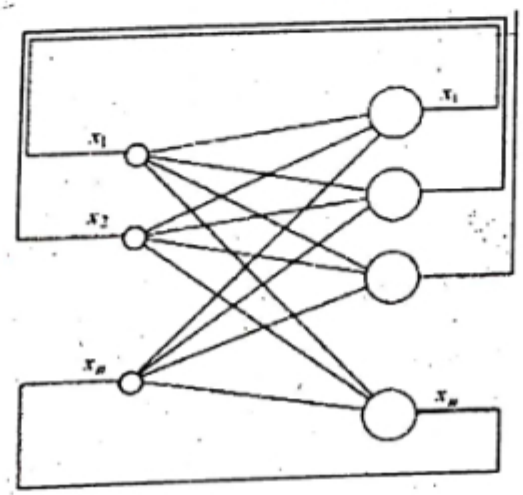
4. a) How associative memory network is related to content addressable memories (CAM)? Draw the architecture of an auto associative network. [WBUT 2015]

Answer:

1st Part:

In a content-addressable memory (CAM) which is also know as associative memory network some additional logic is provided inside the memory that allows a fast access of any stored information by using only the contents of a supplied input keyword. For an imperfect input pattern, associative memory has the capability to recall the stored pattern correctly by performing a collective relaxation search. Associative memories can be either heteroassociative or autoassociative, where the input and output vectors range over different vector spaces or over the same vector space.

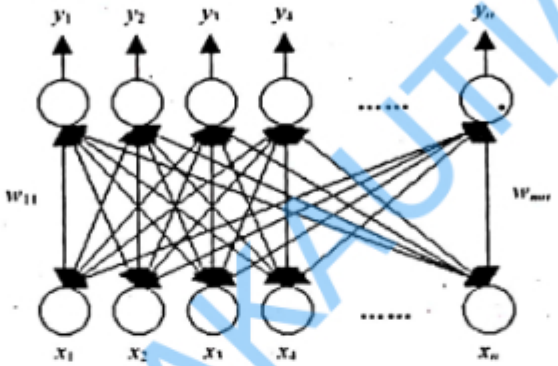
2nd Part



b) What is a bi-directional associative memory network? List the activation functions used in BAM network. **[WBUT 2015]**

Answer:

In network structure of the *Bidirectional Associative Memory* connections are bidirectional, i.e., $w_{ij} = w_{ji}$, for $i = 1, 2, \ldots, m$ and $j = 1, 2, \ldots, n$. Also, the units in both layers serve as both input and output units depending on the direction of propagation. Propagating signals from the X layer to the Y layer makes the units in the X layer act as input units while the units in the Y layer act as output units. The same is true for the other direction, i.e., propagating from the Y layer to the X layer makes the units in the Y layer act as input units while the units in the X layer act as output units. Below is an illustration of the BAM architecture.



The activation function for a BAM is the appropriate step function depending on whether binary or bipolar vectors are used.

For binary input vectors, the activation function for the Y-layer is:

$$y_j = \begin{cases} 1 & \text{if } y_{inj} > 0 \\ y_j & \text{if } y_{inj} = 0 \\ 0 & \text{if } y_{inj} < 0 \end{cases}$$

and the activation for the X-layer is:

$$x_i = \begin{cases} 1 & \text{if } X_{ini} > 0 \\ x_i & \text{if } X_{ini} = 0 \\ 0 & \text{if } X_{ini} < 0 \end{cases}$$

For bipolar input vectors, the activation function for the Y-layer is:

$$y_j = \begin{cases} 1 & \text{if } y_{inj} > \theta_j \\ y_j & \text{if } y_{inj} = \theta_j \\ -1 & \text{if } y_{inj} < \theta_j \end{cases}$$

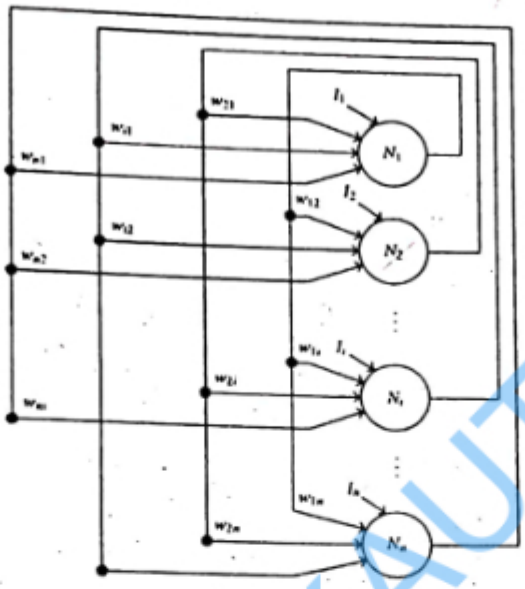
and the activation function for the X-layer is:

$$x_i = \begin{cases} 1 & \text{if } X_{inj} > \theta_j \\ x_i & \text{if } X_{inj} = \theta_j \\ 0 & \text{if } X_{inj} < \theta_j \end{cases}$$

c) Draw the architecture of discrete Hopfield net and state the testing algorithm used in discrete Hopfield net. **[WBUT 2015]**

Answer:

1st Part:



Architecture:

- single layer (units serve as both input and output)
- nodes are threshold units (binary or bipolar)
- weight: fully connected, symmetric, and zero diagonal

  $w_{ij} = w_{ji}$

  $w_{ii} = 0$
- x_i are external inputs, which may be transient or permanent.

2nd Part: ***Refer to Question No. 1 of Short Answer Type Questions.***

5. a) Consider a Kohonen net with two cluster units and three input units (0.9, 0.7, 0.6) and (0.4, 0.3, 0.5). Find the winning cluster unit for the input vector (0.4, 0.2, 0.1). Use learning rate of 0.2. Find the new weights for the winning unit. **[WBUT 2016]**

Answer:

Step 1: Initial weight matrix

$$\begin{bmatrix} 0.9 & 0.4 \\ 0.7 & 0.3 \\ 0.6 & 0.5 \end{bmatrix}$$

Initial radius $R = 0$, Initial learning rate: $\alpha = 0.2$

Step 2: Begin training.

Step 3: For the input vector (0.4, 0.2, 0.1) do Steps 4-6

Step 4: Calculate squared Euclidean distance

$$D(j) = \sum (w_{ij} - x_i)^2$$

$$D(1) = (0.9-0.4)^2 + (0.7-0.2)^2 + (0.6-0.1)^2 = 0.75$$

$$D(2) = (0.4-0.4)^2 + (0.3-0.2)^2 + (0.5-0.1)^2 = 0.17$$

Step 5: The input vector is closest to output node 2. Since (D(2)) is minimum J=2

$$w_{12(new)} = w_{12(old)} + \alpha\left[x_1 - w_{22},(\text{old})\right] = 0.4 + 0.2(0.4-0.4) = 0.4$$

$$w_{12(new)} = w_{22(old)} + \alpha\left[x_2 - w_{22},(\text{old})\right] = 0.3 + 0.2(0.2-0.3)$$

$$w_{32(new)} = 0.28$$

$$w_{32(new)} = w_{32(old)} + \alpha\left[x_3 - w_{32},(\text{old})\right] = 0.5 + 0.2(0.1-0.5) = 0.42$$

This gives the new updated weight matrix

$$\begin{bmatrix} 0.9 & 0.4 \\ 0.7 & 0.28 \\ 0.6 & 0.42 \end{bmatrix}$$

b) Use an autoassociative network to store a vector s(p) = [1 –1 1 1]: Examine whether the associative net accurately retrieves the input vector [1 –1 –1 1].

[WBUT 2016]

Answer:

Recall:

Example of auto-associative memory

- A single pattern s(p) = [1 -1 1 1] is stored. We calculate the weights by Hebbian rule-outer product as given below.

  Outer product of s, $W = [s^T]_{4\times1}[s]_{1\times4}$

1	–1	1	1
–1	1	–1	–1
1	–1	1	1
1	–1	1	1

Now we need to examine whether **vector [1 –1 –1 1]** is retrieved accurately.

Now,

(1 -1 -1 1).W = (2 -2 2 2) → (1 -1 1 1) which retrieves accurately

6. a) Describe the architecture of Kohonen self-organizing feature maps.

[WBUT 2017]

Answer:

Refer to Question No. 3(a) of Long Answer Type Questions.

b) Consider a Kohonen net with two cluster units and five input units. The weight vectors for the cluster units are:
w_1 = (1.0, 0.9, 0.7, 0.3, 0.2)
w_2 = (0.6, 0.7, 0.5, 0.4, 1.0)
Find the winning cluster units for the input pattern X = (0.0, 0.2, 0.1, 0.2, 0.0). Using a learning rate 0.2, find the new weights for the winning units. [WBUT 2017]

Answer:

Step 1: Initial weight matrix

$$\begin{pmatrix} 1.0 & 0.6 \\ 0.9 & 0.7 \\ 0.7 & 0.5 \\ 0.3 & 0.4 \\ 0.2 & 1.0 \end{pmatrix}$$

Initial Radius R = 0, Learning rate = 0.2

Step 2: Begin Training

Step 3: for the input vector X = (0.0, 0.2, 0.1, 0.2, 0.0) do:

Step 4: Calculate squared Euclidian distance

$D(1) = (1.0 - 0.0)^2 + (0.9 - 0.2)^2 + (0.7 - 0.1)^2 + (0.3 - 0.2)^2 + (0.2 - 0.0)^2$
$= 1 + 0.49 + 0.36 + 0.01 + 0.04 = 1.9$

$D(2) = (0.6 - 0.0)^2 + (0.7 - 0.2)^2 + (0.5 - 0.1)^2 + (0.4 - 0.2)^2 + (1.0 - 0.0)^2$
$= 0.36 + 0.25 + 0.16 + 0.04 + 1 = 1.81$

The input vector is closest to output node 2 since D(2) is minimum.

$W_{12(new)} = W_{12(old)} + 0.2(X_1 - W_{12(old)}) = 0.6 + 0.2*(0 - 0.6) = 0.48$
$W_{22(new)} = W_{22(old)} + 0.2(X_1 - W_{22(old)}) = 0.7 + 0.2*(0.2 - 0.7) = 0.6$
$W_{32(new)} = W_{32(old)} + 0.2(X_1 - W_{32(old)}) = 0.5 + 0.2*(0.1 - 0.5) = 0.42$
$W_{42(new)} = W_{42(old)} + 0.2(X_1 - W_{42(old)}) = 0.4 + 0.2*(0.2 - 0.4) = 0.36$
$W_{52(new)} = W_{52(old)} + 0.2(X_1 - W_{52(old)}) = 1.0 + 0.2*(0.0 - 1.0) = 0.8$

7. Write short notes on the following:

a) Hamming network [WBUT 2014, 2015]
b) Learning vector quantization [WBUT 2014, 2015]
c) Image restoration based on Associative Memory [WBUT 2014]
d) Boltzmann machine [WBUT 2014, 2015]
e) Simulated annealing [WBUT 2015, 2016, 2017]
f) Gibbs sampling [WBUT 2017]
g) Associative Memory Neural Network [WBUT 2017]
h) Markov chains [WBUT 2017]

Answer:

a) Hamming network:

The Hamming network is a straightforward associative memory. It calculates the Hamming distance between the input pattern and each memory pattern and selects the memory with the smallest Hamming distance. The network output is the index of a prototype pattern and thus the network can be used as a pattern classifier. The Hamming

network is a used as the classical Hamming decoder or Hamming associative memory. It provides the minimum-Hamming-distance solution.

The Hamming network has a *J-N-J* layered architecture, as illustrated in Fig. 1.

The activation function at each of the units in each layer, denoted by vector ϕ is the signum function and $\theta^{(2)}$ is a vector comprising the biases for all neurons at the hidden layer. The weights in the third layer

$$T = [t_{kl}], k, l = 1, \ldots\ldots, N .$$

The third layer is called the memory layer, each of whose neurons corresponds to a prototype pattern. The input and hidden layers are feedforward, fully connected, while each hidden node has a feedforward connection to its corresponding node in the memory layer. Neurons in the memory layer are fully interconnected and form a competitive subnetwork known as the MAXNET. The MAXNET responds to an input pattern by generating a winner neuron through iterative competitions. The Hamming network is implicitly recurrent due to the interconnections in the memory layer.

The second layer generates matching scores that are equal to J minus the Hamming distances to the stored patterns, that is,

$$J - d_H(x, x_i), i = 1, \ldots\ldots, N ,$$

for pattern x. These matching scores range from 0 to J. The unit with the highest matching score corresponds to the stored pattern that best matches the input. The weights between the input and hidden layers and the biases of the hidden layer are respectively, set as

$$w_{ij} = \frac{x_{j,i}}{2}, \ \theta_j^{(2)} = \frac{J}{2}, \ j = 1, \ldots\ldots, N, i = 1, \ldots\ldots, J .$$

All the thresholds and the weights t_{kl} in the MAXNET are fixed. The thresholds are set as zero. The weights from each node to itself are set as unity and weights between nodes are inhibitory, that is,

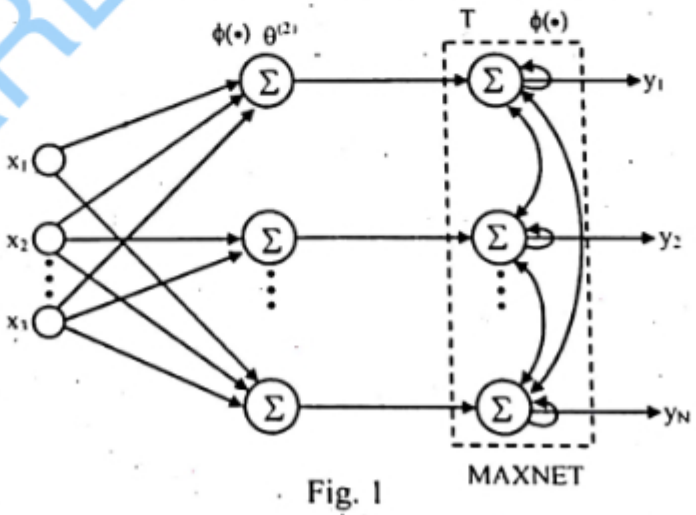


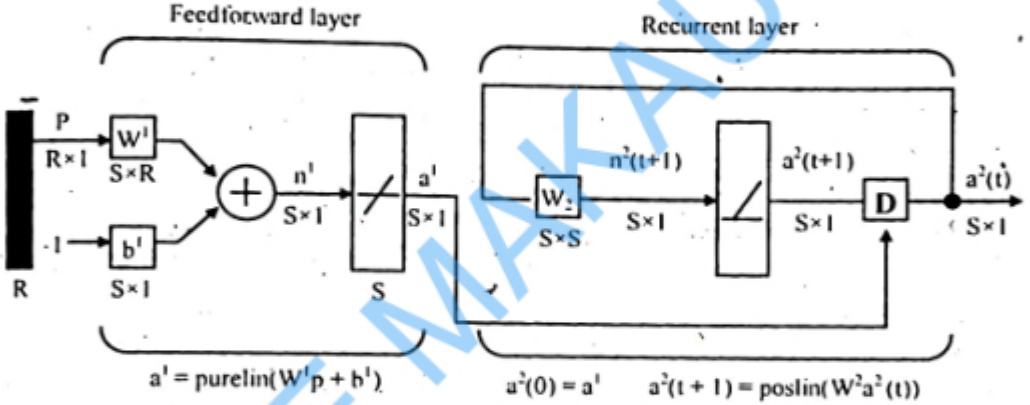
Fig. 1

b) Learning vector quantization:

A **Learning Vector Quantization** (LVQ) system is represented by prototypes $W = (w(i), \ldots\ldots, w(n))$. which are defined in the feature space of observed data. In winner-take-all training algorithms one determines, for each data point, the prototype which is closest to the input according to a given distance measure. The position of this so-called winner prototype is then adapted, i.e. the winner is moved closer if it correctly classifies the data point or moved away if it classifies the data point incorrectly.

An advantage of LVQ is that it creates prototypes that are easy to interpret for experts in the respective application domain. LVQ systems can be applied to multi-class classification problems in a natural way. It is used in a variety of practical applications,

A key issue in LVQ is the choice of an appropriate measure of distance or similarity for training and classification. Recently, techniques have been developed which adapt a parameterized distance measure in the course of training the system, see LVQ can be a source of great help in classifying text documents.



c) Image restoration based on Associative Memory:

Image restoration is the operation of taking a corrupted/noisy image and estimating the clean original image.

An associative memory is a content-addressable structure that maps specific input representations to specific output representations. It is a system that "associates" two patterns (X, Y) such that when one is encountered, the other can be recalled.

Typically, $X \hat{I} \{-1, +1\}^m$, $Y \hat{I} \{-1, +1\}^n$ and m and n are the length of vectors X and Y, respectively. The components of the vectors can be thought of as pixels when the two patterns are considered as bitmap images.

The Hopfield model which is an associative memory model can be used for image restoration. The Hopfield model computes its output recursively in time until the system becomes stable. The Hopfield model consists of a single layer of processing elements where each unit is connected to every other unit in the network other than itself. The units in the Hopfield model act as both input and output units. Since the Hopfield model is an autoassociative memory model, patterns, rather than associated pattern pairs, are stored in memory. After encoding, the network can be used for decoding. Decoding in the Hopfield model is achieved by a collective and recursive relaxation search for a stored pattern given an initial stimulus pattern. Given an input pattern X, decoding is

accomplished by computing the net input to the units and determining the output of those units using the output function to produce the pattern X'. The pattern X' is then fed back to the units as an input pattern to produce the pattern X". The pattern X" is again fed back to the units to produce the pattern X'". The process is repeated until the network stabilizes on a stored pattern where further computations do not change the output of the units. If the input pattern X is an incomplete pattern or if it contains some distortions, the stored pattern to which the network stabilizes is typically one that is most similar to X without the distortions. This feature is called pattern completion and is very useful in many image restoration applications.

d) Boltzmann machine:

A **Boltzmann machine** is a type of stochastic recurrent neural network, which can be seen as the stochastic, generative counterpart of Hopfield nets. They were one of the first examples of a neural network capable of learning internal representations, and are able to represent and (given sufficient time) solve difficult combinatoric problems. They are theoretically intriguing because of the locality and Hebbian nature of their training algorithm, and because of their parallelism and the resemblance of their dynamics to simple physical processes. Due to a number of issues discussed below, Boltzmann machines with unconstrained connectivity have not proven useful for practical problems in machine learning or inference, but if the connectivity is properly constrained, the learning can be made efficient enough to be useful for practical problems.

The Boltzmann machine would theoretically be a rather general computational medium. For instance, if trained on photographs, the machine would theoretically model the distribution of photographs, and could use that model to, for example, complete a partial photograph.

Unfortunately, there is a serious practical problem with the Boltzmann machine, namely that it seems to stop learning correctly when the machine is scaled up to anything larger than a trivial machine. This is due to a number of effects, the most important of which are:

- the time the machine must be run in order to collect equilibrium statistics grows exponentially with the machine's size, and with the magnitude of the connection strengths
- connection strengths are more plastic when the units being connected have activation probabilities intermediate between zero and one, leading to a so-called variance trap. The net effect is that noise causes the connection strengths to follow a random walk until the activities saturate.

e) Simulated annealing:

Simulated annealing is a method for finding a good (not necessarily perfect) solution to an optimization problem. If you're in a situation where you want to maximize or minimize something, your problem can likely be tackled with simulated annealing.

The traveling salesman problem is a good example: the salesman is looking to visit a set of cities in the order that minimizes the total number of miles he travels. As the number

of cities gets large, it becomes too computationally intensive to check every possible itinerary. At that point, you need an algorithm.

There are many optimization algorithms, including hill climbing, genetic algorithms, gradient descent, and more. Simulated annealing's strength is that it avoids getting caught at local maxima - solutions that are better than any others nearby, but aren't the very best.

A high-level overview of the basic algorithm is given below.

1. First, generate a random solution
2. Calculate its cost using some cost function you've defined
3. Generate a random neighboring solution
4. Calculate the new solution's cost
5. Compare them:
   - If $c_{new} < c_{old}$: move to the new solution
   - If $c_{new} > c_{old}$: *maybe* move to the new solution
6. Repeat steps 3-5 above until an acceptable solution is found or you reach some maximum number of iterations.

f) Gibbs sampling:

Gibbs sampling is a Markov chain Monte Carlo (MCMC) algorithm for obtaining a sequence of observations which are approximated from a specified multivariate probability distribution, when direct sampling is difficult. This sequence can be used to approximate the joint distribution (e.g., to generate a histogram of the distribution); to approximate the marginal distribution of one of the variables, or some subset of the variables (for example, the unknown parameters or latent variables); or to compute an integral (such as the expected value of one of the variables). Typically, some of the variables correspond to observations whose values are known, and hence do not need to be sampled.

Gibbs sampling is commonly used as a means of statistical inference, especially Bayesian inference. It is a randomized algorithm (i.e. an algorithm that makes use of random numbers), and is an alternative to deterministic algorithms for statistical inference such as the expectation-maximization algorithm (EM).

The idea in Gibbs sampling is to generate posterior samples by sweeping through each variable (or block of variables) to sample from its conditional distribution with the remaining variables fixed to their current values. For instance, consider the random variables X_1, X_2, and X_3. We start by setting these variables to their initial values $x_1^{(0)}$, $x_2^{(0)}$ and $x_3^{(0)}$ (often values sampled from a prior distribution q). At iteration i,, we sample $x_1^{(i)} \sim p(X_1 = x_1 | X_2 = x_2^{(i-1)}, X_3 = x_3^{(i-1)})$, sample $x_2 \sim p(X_2 = x_2 | X_1 = x_1^{(i)}, X_3 = x_3^{(i-1)})$, and sample $x_3 \sim p(X_3 = x_3 | X_1 = x_1^{(i)}, X_2 = x_2^{(i)})$.

This process continues until "convergence" (the sample values have the same distribution as if they were sampled from the true posterior joint distribution).

g) Associative Memory Neural Network:

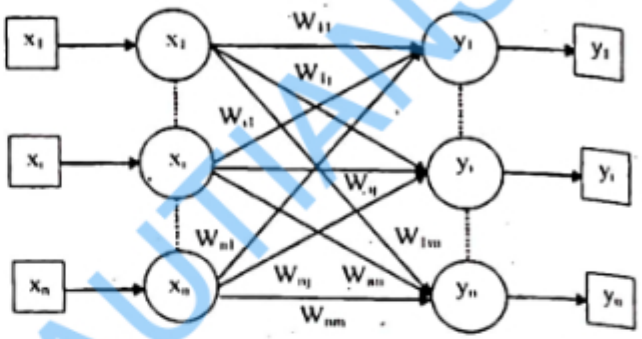
These kinds of neural networks work on the basis of pattern association, which means they can store different patterns and at the time of giving an output they can produce one of the stored patterns by matching them with the given input pattern. These types of

memories are also called **Content-Addressable Memory** (CAM). Associative memory makes a parallel search with the stored patterns as data files.
Following are the two types of associative memories we can observe:

- Auto Associative Memory
- Hetero Associative memory

Auto Associative Memory

This is a single layer neural network in which the input training vector and the output target vectors are the same. The weights are determined so that the network stores a set of patterns.



Architecture

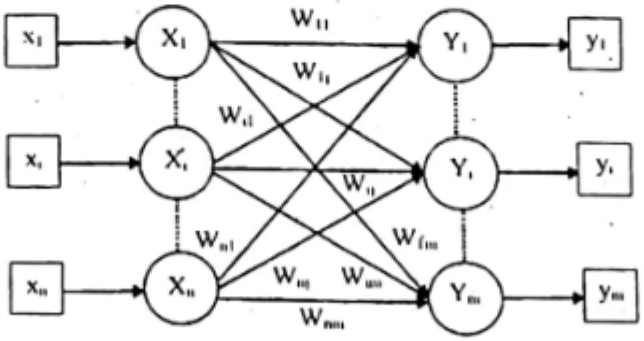
As shown in the following figure, the architecture of Auto Associative memory network has '***n***' number of input training vectors and similar '***n***' number of output target vectors.

Hetero Associative memory

Similar to Auto Associative Memory network, this is also a single layer neural network. However, in this network the input training vector and the output target vectors are not the same. The weights are determined so that the network stores a set of patterns. Hetero associative network is static in nature, hence, there would be no non-linear and delay operations.

Architecture

As shown in the following figure, the architecture of Hetero Associative Memory network has '***n***' number of input training vectors and '***m***' number of output target vectors.



h) Markov chains:

A Markov chain is a mathematical system that undergoes transformation from one state to another on a state space (e.g. over successive time instants). Such a sequence of states is characterized by a Markovian state transition probability, i.e. the probability of occupying the next state only depends on the current (and potentially previous m) states. One does not have to know the entire state sequence to be able to compute the probability

distribution for the next state. Thus, each state transition can be stochastically described using a state transition matrix that maps the probability of occupying the new state based on the current (+ potentially previous few) states.

Formally, a Markov chain is collection of random variables $\{X_t\}$ (where the index t runs through 0, 1, ...) having the property that, given the present, the future is conditionally independent of the past.

In other words,

$$P(X_t = j \mid X_0 = i_0, X_1 = i_1, \ldots, X_{t-1} = i_{t-1}) = P(X_t = j \mid X_{t-1} = i_{t-1}).$$

If a Markov sequence of random variates x_n take the discrete values $a_1, \ldots, a_n$, then

$$P(x_n = a_{i_n} \mid x_{n-1} = a_{i_{n-1}}, \ldots, x_1 = a_{i_1}) = P(x_n = a_{i_n} \mid x_{n-1} = a_{i_{n-1}})$$

and the sequence x_n is called a Markov chain.

8. a) SOMs can reduce the dimensionality of a given data space. Explain what that means. Please give an example of how this capability can be used for practical applications.

b) Why are SOMs interesting for researchers who study biological nervous systems? **[MODEL QUESTION]**

Answer:

a) The map layer of an SOM is typically one- or two-dimensional, whereas the input space for the SOM usually has many more dimensions. Nevertheless, the SOM tries as much as possible to establish a topology conserving mapping such that inputs from neighboring regions in the input space will make neighboring neurons (or even the same neuron) in the map layer win the competition. In this way, the high-dimensional input space is mapped onto a lower dimensional output space. One example is the visual representation of large data sets, for example, image or document databases. Using an SOM, such high-dimensional spaces can be mapped, for example, onto a two-dimensional space in which similar images or documents are usually close to each other. The user can browse through this space to find a desired item much more easily than through the original, high-dimensional space.

b) The topology-conserving property of the SOM between its input and output spaces is in fact a very common feature of biological neural networks. For instance, neighboring areas in our sensory cortex respond when our ring finger or our middle finger on our right hand is being touched. However, the area responding to touch in our left foot is far away from the first two areas. The development of this mapping and its adaptation to changes (my favorite example: losing one of your fingers) is assumed to be accomplished by competitive mechanisms as in the SOM. SOMs have been used to predict the development of neural connections in our brain with good accuracy.

9. a) Explain how it can be shown that for a constant input, Hopfield networks are guaranteed to reach a stable state after a finite number of iterations.

b) Why do we demand for auto-association that all wii = 0? **[MODEL QUESTION]**

Answer:

a) First, we introduce an energy function in such a way that it decreases whenever the state of the network gets closer to one of the stored patterns. Every possible state has a particular energy value. Then we show that whenever the Hopfield rule demands that the state of a neuron switches, the network energy decreases. Since in the discrete Hopfield network there is only a finite number of possible states, it means that after a finite number of switches a state will be reached in which no further switches can occur because none of them would lower the energy.

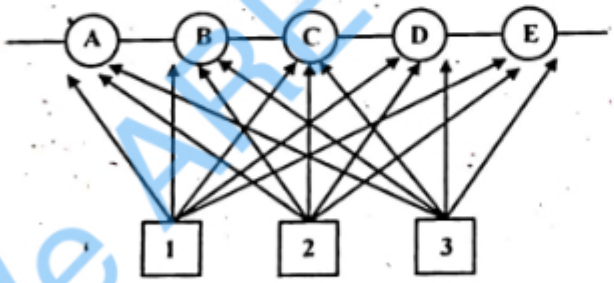
b) If we did not demand this, the trivial solution $w_{ii} = 1$ and $w_{ij} = 0$ for $i \neq j$ could be chosen, which formally leads to auto-association for all input patterns but does not store any pattern information and cannot be used for error correction. With $w_{ii} = 0$ we ensure that the pattern information (the correlation between pairs of pixels) is stored in the connections between neurons.

10. Compute the weight matrix for a 4-neuron Hopfield net with the single fundamental memory $\xi_1 = [1,-1,-1,1]$ stored in it. **[MODEL QUESTION]**

Answer:

$$W = \xi_1^T \xi_1 - I_4 = \begin{bmatrix} 0 & -1 & -1 & 1 \\ -1 & 0 & 1 & -1 \\ -1 & 1 & 0 & -1 \\ 1 & -1 & -1 & 0 \end{bmatrix}$$

11. Below is a diagram of a self–organising map: **[MODEL QUESTION]**



By looking at the diagram answer the following questions:

a) How many input nodes does this SOM have?

b) How many output nodes does this SOM have?

c) The input to an SOM can be represented by a point in an m–dimensional space (or m–dimensional vector). How many dimensions are in the space that this SOM is analysing?

d) How many weights does each of the output nodes have?

e) The output nodes are organised in a lattice. How many dimensions does the output lattice of this SOM have?

f) How many output nodes can fire simultaneously?

g) Is it important what value the output node sends when it fires?

h) Is there any limit on how many data points (input patterns) this SOM can analyse?
i) How many clusters can this SOM detect in the input data?
j) If node D is the winner, which output nodes are its immediate neighbours?
Answer:
a) Three input nodes: node 1, 2 and 3.
b) Five output nodes: node A, B, C, D and E
c) The number of dimensions corresponds to the number of input nodes. Thus, this SOM analyses points in three–dimensional space. Each point is represented by three coordinates: x = (x1, x2, x3).
d) The number of weights of every output node corresponds to the number of input nodes. Thus, each output node has three weights: w1, w2 and w3. The weights of each of the output nodes fixate particular points in the input space: w = (w1, w2, w3).
e) The output nodes A, B, C, D and E are all positioned along a line. So, the output lattice is one–dimensional
f) Only one of the output nodes can fire at a time.
g) No. In SOM it is only important which of the output nodes fires. It does not matter what value.
h) There is no such limit. You can feed as much data as you wish into an SOM. The input space can be infinite.
i) Because the number of output nodes is five, this SOM cannot distinguish between more than five different clusters. However, it is possible that there will be fewer than five clusters detected in the data.
j) The neighbours are node C and E

12. An insurance company with several thousands of customers has decided to analyse its customers in order to understand better why they buy the policy. The company collected data about its customers for the last 2 years. Each customer's profile was stored electronically in a database and fed into a data warehouse, where it was assessed on 50 parameters. Discuss in a formBIS32267 of essay how a self–organising map (SOM) could be used for this analysis. Why would the results, produced by an SOM, be particularly useful for the reports presented to strategic managers? **[MODEL QUESTION]**

Answer:

- Based on description above, the centralized data warehouse contains information about thousands of customers, each evaluated on 50 parameters. Thus, the **input** dataset has several thousands points in 50-dimensional **space**.
- The SOM would need to have **50 inputs** and the output lattice would have arbitrary number of nodes, but probably high enough to distinguish between many groups of customers.
- The SOM can show the distribution of customers and their **clusters** on this output map.
- Because SOM shows results on one or two dimensional feature map, they can easily be included into a report and help to **visualize** the results of the analysis.

- The individual groups of customers, discovered by SOM, can then be further investigated and new products and promotions can be designed specifically for these groups.
- Data analysis considering long periods of time (i.e., 2 years in this case) and decisions on new products and promotions are more likely to be the responsibility strategic management.

13. a) Calculate the weight matrix for a Hopfield network to store two patterns [1 −1 1 −1] and [−1 −1 −1 1].
b) A pattern [−1 −1 1 −1] is presented to the network, then the nodes of the network are updated until a steady state is reached. What is the final state of the network? **[MODEL QUESTION]**

Answer:

a) $W = \begin{bmatrix} 0 & 0 & 2 & -2 \\ 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & -2 \\ -2 & 0 & -2 & 0 \end{bmatrix}$

b) [1 −1 1 −1]

14. What are the main similarities and differences between feed–forward neural networks and self–organising maps? **[MODEL QUESTION]**

Answer:

Similarities are:

- Both are feed-forward networks (no loops)
- Nodes have weights corresponding to each link
- Both networks require training.

The main differences are:

- Self-organising maps (SOM) use just a single output layer, they do not have hidden layers.
- In feed-forward neural networks (FFNN) we have to calculate weighted sums of the nodes. There are no such calculations in SOM, weights are only compared with the input patterns using Euclidean distance.
- In FFNN the output values of nodes are important and they are defined by the activation functions. In SOM nodes do not have any activation functions and the output values are not important.
- In FFNN all the output nodes can fire, while in SOM only one.
- The output of FFNN can be a complex pattern consisting of the values of all the output nodes. In SOM we only need to know which of the output nodes is the winner.
- Training of FFNN usually employs supervised learning algorithms, which require a training set. SOM use unsupervised learning algorithm. There are, however, unsupervised training methods for FFNN as well.

APPLICATIONS

Multiple Choice Type Questions

1. SVM is based on **[WBUT 2014, 2018]**

a) supervised learning **b) unsupervised learning**

c) reinforcement learning **d) both (a) & (b)**

Answer: (a)

Short Answer Type Questions

1. For each of the following cases, state whether it would be best to use the primal or dual SVM formulation.

a) We apply a feature transformation that maps the input data into a feature space with infinite dimension

b) We apply a feature transformation that doubles the dimension of the input data. The input data has billions of training examples and is linearly separable.

[MODEL QUESTION]

Answer:

a) **Dual:** The primal would have an infinite number of components in the weight vector w and be unsolvable.

b) **Primal:** The dual formulation would have billions of α variables, and if the data is linearly separable we do not need an ϵ parameter for each data point in the primal.

2. Describe the competitive process of the Self-Organising Map algorithm.

[MODEL QUESTION]

Answer:

Let m denote the dimension of the input pattern $x = [x1, x2, ..., xm]^T$

The weight vector for each of the neurons in the SOM also has dimension m. So for neuron j, the weight vector will be:

$wj = [wj1, wj2, ..., wjm]^T$

For an input pattern x, compute the norm difference $\|wj-x\|$ for each map neuron j, and choose the smallest value: this is the winning neuron. Let i(x) denote the index of the winning neuron (and also the output of a trained SOM).

3. You have an application problem for which you need to decide whether to use a Two-Layer Neural Net or a Support Vector Machine with a non-linear Kernel. Name one argument in favor of the Net, and one other argument in favor of the SVM.

[MODEL QUESTION]

Answer:

Pro SVM:

- no local optima
- have to pick fewer parameters (e.g. layers, number of nodes) than in net
- easier to do get estimate of generalization error, since number of support vectors is an upper bound on leave-one-out error

Pro Net:

- network structure allows modelling of prior knowledge
- can control size of the net, while size of SVM classifier grows with the number of training

Long Answer Type Questions

1. Write short note on SVM based learning **[WBUT 2014]**

OR,

Support vector machine **[WBUT 2015]**

OR,

SVM classifier **[WBUT 2016, 2018]**

Answer:

Support vector machines (SVM) are supervised learning models with associated learning algorithms that analyze data and recognize patterns, used for classification and regression analysis. Given a set of training examples, each marked as belonging to one of two categories; an SVM training algorithm builds a model that assigns new examples into one category or the other, making it a non-probabilistic binary linear classifier. An SVM model is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible. New examples are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall on.

In addition to performing linear classification, SVMs can efficiently perform a non-linear classification using what is called the kernel trick, implicitly mapping their inputs into high-dimensional feature spaces.

SVMs can be used to solve various real world problems:

- SVMs are helpful in text and hypertext categorization as their application can significantly reduce the need for labeled training instances in both the standard inductive and transductive settings.
- Classification of images can also be performed using SVMs. Experimental results show that SVMs achieve significantly higher search accuracy than traditional query refinement schemes after just three to four rounds of relevance feedback.
- SVMs are also useful in medical science to classify proteins with up to 90% of the compounds classified correctly.
- Hand-written characters can be recognized using SVM

2. Write a MATLAB program for perceptron net for an AND function with bipolar inputs and targets. **[MODEL QUESTION]**

Answer:

The truth table for the AND function is given as

X_1	X_2	Y
−1	−1	−1
−1	1	−1
1	−1	−1
1	1	1

The MATLAB program for the above table is given as follows.

Program

```
%Perceptron for AND funtion
clear;
clc;
x=[1 1 -1 -1;1 -1 1 -1];
t=[1 -1 -1 -1];
w=[0 0];
b=0;
alpha=input('Enter Learning rate=');
theta=input('Enter Threshold value=');
con=1;
epoch=0;
while con
    con=0;
    for i=1:4
        yin=b+x(1,i)*w(1)+x(2,i)*w(2);
        if yin>theta
            y=1;
        end
        if yin <=theta & yin>=-theta
            y=0;
        end
        if yin<-theta
            y=-1;
        end
        if y-t(i)
            con=1;
            for j=1:2
                w(j)=w(j)+alpha*t(i)*x(j,i);
            end
            b=b+alpha*t(i);
        end
    end
    epoch=epoch+1;
end
disp('Perceptron for AND funtion');
disp(' Final Weight matrix');
disp(w);
```

2. Write a MATLAB program for perceptron net for an AND function with bipolar inputs and targets. **[MODEL QUESTION]**

Answer:

The truth table for the AND function is given as

X_1	X_2	Y
−1	−1	−1
−1	1	−1
1	−1	−1
1	1	1

The MATLAB program for the above table is given as follows.

Program

```
%Perceptron for AND funtion
clear;
clc;
x=[1 1 -1 -1;1 -1 1 -1];
t=[1 -1 -1 -1];
w=[0 0];
b=0;
alpha=input('Enter Learning rate=');
theta=input('Enter Threshold value=');
con=1;
epoch=0;
while con
    con=0;
    for i=1:4
        yin=b+x(1,i)*w(1)+x(2,i)*w(2);
        if yin>theta
            y=1;
        end
        if yin <=theta & yin>=-theta
            y=0;
        end
        if yin<-theta
            y=-1;
        end
        if y-t(i)
            con=1;
            for j=1:2
                w(j)=w(j)+alpha*t(i)*x(j,i);
            end
            b=b+alpha*t(i);
        end
    end
    epoch=epoch+1;
end
disp('Perceptron for AND funtion');
disp(' Final Weight matrix');
disp(w);
```

```
disp('Final Bias');
disp(b);
```

Output

```
Enter Learning rate=1
Enter Threshold value=0.5
Perceptron for AND funtion
Final Weight matrix
     1     1
Final Bias
    -1
```

QUESTION 2014

Group – A
(Multiple Choice Type Questions)

1. Choose the correct alternatives for any *ten* of the following:

i) A perceptron is
- ✓a) a single layer feed-forward neural network with preprocessing
- b) an autoassociative neural network
- c) a double layer autoassociative neural network
- d) all of these

ii) Artificial neural networks are inspired by
- a) swarm intelligence
- ✓b) human brain
- c) high speed parallel processors
- d) all of these

iii) In a neural net, if for the training input vectors, the target output is not known, the training method adopted is called as
- a) supervised training
- ✓b) unsupervised training
- c) reinforcement training
- d) none of these

iv) Sigmoid is a _____ type of non-linearity.
- a) hard
- ✓b) soft
- c) rough
- d) soft and rough

v) In back-propagation algorithm _____ is propagated backward through the network.
- a) error
- b) signal
- ✓c) error + signal
- d) signal – error

vi) An auto-associative network is
- a) a neural network that contains no loops
- ✓b) a neural network that contains feedback
- c) a neural network that has only one loop
- d) none of these

vii) The gradient descent rule mostly is used in
- a) M-P Neural Learning
- ✓b) Hebb Neural Learning
- c) Back-Propagation Neural Learning
- d) Adaline Neural Learning

viii) Gaussian Activation Function is used in
- a) Back Propagation
- ✓b) Radial Basis Function
- c) Self Organizing Feature Map
- d) Mexican Hat Net

ix) ADALINE stands for
- a) Additive Linear Neuron
- ✓b) Adaptive Linear Neuron
- c) Associative Linear Neuron
- d) Adaptive De

x) Which of the following neural networks uses supervised learning?
- a) simple recurrent network
- ✓b) self-organizing feature map
- c) Hopfield network
- d) all of these

xi) Which of the following algorithms can be used to train a single-layer feedforward network?

a) hard competitive learning
b) soft competitive learning
c) a genetic algorithm
✓d) all of these

xii) SVM is based on

✓a) supervised learning
b) unsupervised learning
c) reinforcement learning
d) both (a) & (b)

Group – B
(Short Answer Type Questions)

2. Describe the discrete Hopfield Net and its training algorithm.
See Topic: ASSOCIATIVE MEMORY NETWORKS, Short Answer Type Question No. 1.

3. Implement AND function using McCulloch Pitts neuron (take binary data).
See Topic: INTRODUCTION TO NEURAL NETWORKS, Short Answer Type Question No. 1.

4. The Exclusive-OR function is not linearly separable and hence a single-layer perception cannot simulate it. Justify it.
See Topic: SINGLE-LAYER PERCEPTRONS, Short Answer Type Question No. 1.

5. What is the necessity of an activation function? List commonly used activation functions.
See Topic: INTRODUCTION TO NEURAL NETWORKS, Short Answer Type Question No. 2.

6. What is Adaline? Draw the model of an Adaline network.
See Topic: INTRODUCTION TO NEURAL NETWORKS, Short Answer type Question No. 3.

Group – C
(Long Answer Type Questions)

7. a) Describe Hebb network training algorithm using flowchart.
b) Use the Hebb rule method to implement XOR function (take bipolar inputs).
c) List the stages involved in training of back propagation algorithm.
a) See Topic: ASSOCIATIVE MEMORY NETWORKS, Long Answer Type Question No. 1(a).
b) See Topic: ASSOCIATIVE MEMORY NETWORKS, Long Answer Type Question No. 1(b).
c) See Topic: SINGLE-LAYER PERCEPTRONS, Long Answer Type Question No. 1.

8. What activation function is used in Radial Basic function? Write down and explain, the flow-chart of the training algorithm of RBF. Briefly explain the Wavelet Neural Network.
See Topic: RADIAL BASIS FUNCTION NETWORKS, Long Answer Type Question No. 1.

9. Show and analyse the architecture of Discrete Bidirectional Associative Memory (DBAM) network. Explain how weights are determined? List the activation functions used in DBAM. Also explain the testing algorithm of DBAM.
See Topic: ASSOCIATIVE MEMORY NETWORKS, Long Answer type Question No. 2.

10. a) What do you understand by Kohonen Self-organization Feature Map? Draw and explain its architecture.

b) Construct a Kohonen Self-organization Feature map to cluster four vectors [0 0 1 1], [1 0 0 1] [0 1 0 1], [1 1 1 1].
The maximum number of clusters to be found is 2 and assume random initial weights.
See Topic: ASSOCIATIVE MEMORY NETWORKS, Long Answer Type Question No. 3.

11. Write short notes on any *three* of the following:
   a) SVM based learning
   b) Memory based learning
   c) Hamming network
   d) Learning vector quantization
   e) Image restoration based on Associative Memory
   f) Boltzmann machine

a) See Topic: APPLICATIONS, Long Answer Type Question No. 1.
b) See Topic: INTRODUCTION TO NEURAL NETWORKS, Long Answer Type Question No. 3(a).
c) See Topic: ASSOCIATIVE MEMORY NETWORKS, Long Answer Type Question No. 7(a).
d) See Topic: ASSOCIATIVE MEMORY NETWORKS, Long Answer Type Question No. 7(b).
e) See Topic: ASSOCIATIVE MEMORY NETWORKS, Long Answer Type Question No. 7(c).
f) See Topic: ASSOCIATIVE MEMORY NETWORKS, Long Answer Type Question No. 7(d).

QUESTION 2015

Group – A
(Multiple Choice Type Questions)

1. Answer any *ten* questions:

i) Discrete Hopfield net is the example of
   a) feed forward network
   b) feedback network
   ✓c) recurrent network
   d) both (b) and (c)

ii) In a neural net, critic information is used in category of learning known as
   a) supervised training
   ✓b) unsupervised training
   c) reinforcement training
   d) none of these

iii) A perceptron is:
   ✓a) a single layer feed-forward neural network with pre-processing
   b) an auto-associative neural network
   c) a double layer auto-associative neural network
   d) Hebb network

iv) Which of the following neural networks uses supervised learning?
   a) simple recurrent network
   ✓b) self-organizing feature map
   c) hopfield network
   d) all of the these

v) Supervised learning means
   ✓a) having a teacher
   b) having a class
   c) having a feedback
   d) none of these

vi) Bias is
- ✓a) weight on a connection from a unit having activation 1
- b) weight on a network having activation 2
- c) weight on a function having activation 1
- d) none of these

vii) Mc Culloch Model uses
- a) Sigmoid function
- ✓b) Step function
- c) Signum function
- d) Tan hyperbolic function

viii) The competitive rule is suited for
- ✓a) unsupervised network training
- b) supervised network training
- c) reinforced network training
- d) none of these

ix) Redial Basis Function Network (RBF) uses
- a) Sigmoidal function
- ✓b) Gaussian function
- c) Bipolar-Sigmoidal function
- d) None of these

x) Which of the following algorithms can be used to train a single-layer feed forward network?
- a) Hard competitive learning
- b) Soft competitive learning
- c) A genetic algorithm
- ✓d) All of these

xi) A Hopfield network has 20 units. How many adjustable parameters does this network contain?
- a) 95
- ✓b) 190
- c) 200
- d) 380

Group – B
(Short Answer Type Questions)

2. What is the necessity of activation function? List commonly used activation functions.
See Topic: INTRODUCTION TO NEURAL NETWORKS, Short Answer Type Question No. 2.

3. Implement XOR function using McCulloch-Pitts neuron (consider binary data).
See Topic: INTRODUCTION TO NEURAL NETWORKS, Short Answer Type Question No. 4.

4. What is Adaline? Draw the model of an Adaline Network.
See Topic: INTRODUCTION TO NEURAL NETWORKS, Short Answer Type Question No. 3.

5. What is the impact of weight in an artificial neural network? How does a momentum factor make faster convergence of a network?
See Topic: INTRODUCTION TO NEURAL NETWORKS, Short Answer Type Question No. 5.

6. What is a Hopfield net? State the types of Hopfield network. What is the energy function of discrete Hopfield net?
See Topic: ASSOCIATIVE MEMORY NETWORKS, Short Answer Type Question No. 2.

Group – C
(Long Answer Type Questions)

7. a) Define perceptron learning rule. How the linear separability concept is implemented using perceptron network training?
b) Implement OR function with binary inputs and bipolar targets using perceptron training algorithm upto 3 epochs.

See Topic: SINGLE-LAYER PERCEPTRONS, Long Answer Type Question No. 2.

8. a) Draw the flowchart for Adaline training process.
b) Use Adaline network to train ANDNOT function with bipolar inputs and targets. Perform 2 epochs of training.

See Topic: SINGLE-LAYER PERCEPTRONS, Long Answer Type Question No. 3.

9. a) How associative memory network is related to content addressable memories (CAM)? Draw the architecture of an auto associative network.
b) What is a bi-directional associative memory network? List the activation functions used in BAM network.
c) Draw the architecture of discrete Hopfield net and state the testing algorithm used in discrete Hopfield net.

See Topic: ASSOCIATIVE MEMORY NETWORKS, Long Answer Type Question No. 4.

10. a) Describe the architecture and training process of Radial Basis Function Network.
b) Explain the Hebb rule training algorithm used in pattern association.

a) See Topic: RADIAL BASIS FUNCTION NETWORKS, Long Answer Type Question No. 2.
b) See Topic: ASSOCIATIVE MEMORY NETWORKS, Long Answer Type Question No. 1(a).

11. Write short notes on any *three* of the following:
   a) Wavelet neural network
   b) Support vector machine
   c) Learning vector Quantization
   d) Simulated annealing
   e) Boltzman machine
   f) Hamming Network

a) See Topic: RADIAL BASIS FUNCTION NETWORKS, Long Answer Type Question No. 4.
b) See Topic: APPLICATIONS, Long Answer Type Question No. 1.
c) See Topic: ASSOCIATIVE MEMORY NETWORKS, Long Answer Type Question No. 7(b).
d) See Topic: ASSOCIATIVE MEMORY NETWORKS, Long Answer Type Question No. 7(c).
e) See Topic: ASSOCIATIVE MEMORY NETWORKS, Long Answer Type Question No. 7(d).
f) See Topic: ASSOCIATIVE MEMORY NETWORKS, Long Answer Type Question No. 7(a).

QUESTION 2016

GROUP – A
(Multiple Choice Type Questions)

1. Choose the correct alternatives for the following:

i) A 3-input neuron is trained to output a zero when the input is 110 and a one when the input is 111. After generalization, the output will be zero when and only when the input is
a) 000 or 110 or 011 or 101
b) 010 or 100 or 110 or 101
✓c) 000 or 010 or 110 or 100

ii) Discrete Hopfield net is the example of
a) feed forward network
b) feedback network
✓c) recurrent network
d) none of these

iii) Supervised learning algorithm continues until further change in
✓a) weights
b) non-linearity
c) signal
d) learning-rate

iv) Sigmoid is a ..... type of non-linearity.
a) hard
✓b) soft
c) rough
d) soft and rough

v) In back-propagation algorithm, ... is propagated backward through the network.
a) error
b) signal
✓c) error + signal
d) signal – error

vi) The madaline network is
a) The combination of two single layered feed forward neural networks
✓b) A type of multilayered feed forward neural network with multiple neurons in output layer
c) The combination of adaline networks and multilayered feed forward network with one neuron in output layer.
d) A type of feedback network.

vii) The synapse of a neuron is modeled by a
a) linear function
b) non-linear function
c) non-linear rough function
✓d) linear / non-linear function

viii) Functional value of Bipolar sigmoid function is
a) 0 to 1
✓b) –1 to 1
c) any positive value
d) none of these

ix) The input-output patterns of Hebb Net is?
a) Binary
b) Bipolar
✓c) Any value

x) Radial Basis Function Network (RBF) uses
a) Sigmoidal function
✓b) Gaussian function
c) Bipolar-sigmoidal function
d) Linear function

Group – B
(Short Answer Type Questions)

2. a) What is the role of weight and bias in an ANN model?
See Topic: INTRODUCTION TO NEURAL NETWORKS, Short Answer Type Question No. 5 (1st Part).

b) A 4-input neuron has weights 0.1, 0.2, 0.3 and 0.4. The transfer function is linear with the constant of proportionality being equal to 5. The inputs are 5, 10, 15 and 20 respectively. Find the output.
See Topic: SINGLE-LAYER PERCEPTRONS, Short Answer Type Question No. 2.

3. Implement XOR function using McCulloch Pitts neuron (take binary data).
See Topic: INTRODUCTION TO NEURAL NETWORKS, Short Answer Type Question No. 4.

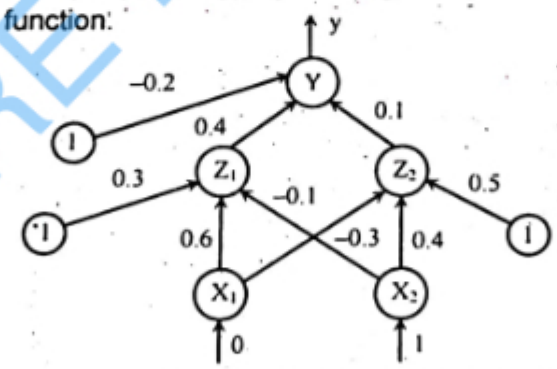
4. Define the single layer perceptron net and its linear separability.
See Topic: SINGLE-LAYER PERCEPTRONS, Short Answer Type Question No. 1.

5. Discuss about the different activation functions used of training artificial neural networks.
See Topic: INTRODUCTION TO NEURAL NETWORKS, Short Answer Type Question No. 2.

6. Define Delta rule. Write down the error function for delta rule.
See Topic: INTRODUCTION TO NEURAL NETWORKS, Short Answer Type Question No. 6.

Group – C
(Long Answer Type Questions)

7. a) Using back-propagation network, find the new weights for the net shown in the following figure. It is presented with the input pattern [0, 1] and target output is 1. Use learning rate 0.25 and binary sigmoidal activation function.



b) How to choose the various parameters for a network trained by Backpropagation?
See Topic: SINGLE-LAYER PERCEPTRONS, Long Answer Type Question No. 4(a) & (b).

8. Consider a Kohonen net with two cluster units and three input units (0.9, 0.7, 0.6) and (0.4, 0.3, 0.5). Find the winning cluster unit for the input vector (0.4, 0.2, 0.1). Use learning rate of 0.2. Find the new weights for the winning unit.
See Topic: ASSOCIATIVE MEMORY NETWORKS, Long Answer Type Question No. 5(a).

9. a) Using perceptron learning rule find the weights required to perform the following classifications:
Vector (1, 1, 1, 1), (–1, 1, –1, –1) and (1, –1, –1, 1) are members of class C1 (having target value 1), vectors (1, 1, 1, –1) and (1, 1, –1, 1) are members of class C2 (having target value –1). Use learning rate of 1 and starting weights of 0. Using perceptron training and vectors as input, test the response of the net.
See Topic: SINGLE-LAYER PERCEPTRONS, Long Answer Type Question No. 5.

b) Write down the learning process of Radial Basis function network.
c) What are the differences between RBF and MLP?
b) & c) See Topic: RADIAL BASIS FUNCTION NETWORKS, Long Answer Type Question No. 3(a) & (b).

10. a) Implement OR function with bipolar inputs and targets using ADALINE network initially all the weights are assumed to be small random values, say 0.1 and the learning rate 0.1.
See Topic: INTRODUCTION TO NEURAL NETWORKS, Long Answer Type Question No. 1.

b) Use an autoassociative network to store a vector s(p) = [1 –1 1 1]. Examine whether the associative net accurately retrieves the input vector [1 –1 –1 1].
See Topic: ASSOCIATIVE MEMORY NETWORKS, Long Answer Type Question No. 5(b).

11. Write short notes on any three of the following:
a) Supervised learning
b) Wavelet Neural Network
c) Neural network architecture
d) SVM classifier
e) Simulated annealing.

a) See Topic: INTRODUCTION TO NEURAL NETWORKS, Long Answer Type Question No. 3(b).
b) See Topic: RADIAL BASIS FUNCTION NETWORKS, Long Answer Type Question No. 4.
c) See Topic: INTRODUCTION TO NEURAL NETWORKS, Long Answer Type Question No. 3(c).
d) See Topic: APPLICATIONS, Long Answer Type Question No. 1.
e) See Topic: ASSOCIATIVE MEMORY NETWORKS, Long Answer Type Question No. 7(e).

QUESTION 2017

Group – A
(Multiple Choice Type Questions)

1. Choose the correct alternatives for any *ten* of the following:
i) The Hebbian rule is ……type of learning
✓a) supervised b) unsupervised c) competitive d) reinforcer

ii) What are the advantages of neural network over conventional computers?
(I) They have the ability to learn by example
(II) They are more faults tolerant
(III) They are suited for real time operation due to their 'computational' rates
a) (I) and (II) are true b) (I) and (II) are true
c) (II) and (III) are true ✓d) all of these are true

iii) Single layer Perceptron is used for
✓a) linear separability
b) error minimization
c) back propagation
d) annealing

iv) For a three input neuron representing a Perceptron where $\{x_1, x_2, x_3\} = \{0.8, 0.6, 0.4\}$ and weight $\{w_1, w_2, w_3\} = \{0.1, 0.3, -0.2\}$ and bias b=0.35 the output of neuron using bipolar sigmoid activation function is
✓a) 0.265
b) 0.746
c) 0.346
d) 0.259

v) Which of the following is/are true for neural networks?
(I) The training time depends on the size of the network
(II) Neural networks can be simulated on a conventional computer
(III) Artificial neurons are identical in operation to biological ones
✓a) all of these are true
b) (II) is true
c) (II) and (III) are true
d) (I) and (II) are true

vi) An auto associative network is
a) a neural network that contains no loops
✓b) a neural network that contains feedback
c) a neural network that has only one loop
d) none of these

vii) ADALINE stands for
✓a) Adaptive Linear Neuron
b) Additive Linear Neuron
c) Associative Linear Neuron
d) Adaptive Derivative Linear Neuron

viii) A 4-input neuron has weights 1,2,3 and 4. The transfer function is linear with the constant of proportionality being equal to 2. The inputs are 4, 10, 5 and 20 respectively. The output will be
a) 238
✓b) 76
c) 119
d) 328

ix) Which of the following is true?
Single layer associative neural networks do not have the ability to
(I) perform pattern recognition
(II) find the parity of a picture
(III) determine whether two or more shapes in a picture are connected or not
✓a) (II) and (III) are true
b) (II) is true
c) (I) and (III) are true
d) all of these are true

x) The Error Correction learning is.....type of learning
✓a) supervised
b) unsupervised
c) reinforced
d) competitive

xi) which of the following algorithms can be used to train a single-layer feed forward network?
a) Hard competitive learning
b) Soft competitive learning
c) A genetic algorithm
✓d) All of these

xii) A Hopfield network has 20 units. How many adjustable parameters does this network contain?
a) 95
✓b) 190
c) 200
d) 380

Group – B
(Short Answer Type Questions)

2. Discuss different categories of learning rules.
See Topic: INTRODUCTION TO NEURAL NETWORKS, Short Answer Type Question No. 7.

3. Compare biological neuron and ANN.
See Topic: INTRODUCTION TO NEURAL NETWORKS, Short Answer Type Question No. 8.

4. Describe Hebb network training algorithm using flowchart.
See Topic: ASSOCIATIVE MEMORY NETWORKS, Long Answer Type Question No. 1(a).

5. What is Boltzmann learning? How does it differ from Error-Correction learning?
See Topic: INTRODUCTION TO NEURAL NETWORKS, Short Answer Type Question No. 9.

6. How neural network can be applied for pattern classification and clustering?
See Topic: INTRODUCTION TO NEURAL NETWORKS, Short Answer Type Question No. 10.

Group – C
(Long Answer Type Questions)

7. a) Describe the back propagation learning algorithm.
b) With a training dataset, illustrate the working of the back propagation learning algorithm.
c) What are the merits and demerits of back propagation algorithm?
See Topic: SINGLE-LAYER PERCEPTRONS, Long Answer Type Question No. 6(a), (b) & (c).

8. a) Describe the salient features of McCulloch-Pitts neuron model. Design a McCulloch-Pitts neural net to realize the XOR function.
See Topic: INTRODUCTION TO NEURAL NETWORKS, Long Answer Type Question No. 2.

b) Design a Hebb net to implement logical AND function with bipolar inputs and target.
See Topic: INTRODUCTION TO NEURAL NETWORKS, Short Answer Type Question No. 4.

9. a) Describe the architecture of Kohonen self-organizing feature maps.
b) Consider a Kohonen net with two cluster units and five input units. The weight vectors for the cluster units are:
w_1 = (1.0, 0.9, 0.7, 0.3, 0.2)
w_2 = (0.6, 0.7, 0.5, 0.4, 1.0)
Find the winning cluster units for the input pattern X = (0.0, 0.2, 0.1, 0.2, 0.0). Using a learning rate 0.2, find the new weights for the winning units.
See Topic: ASSOCIATIVE MEMORY NETWORKS, Long Answer Type Question No. 6(a) & (b)

c) Discuss the architecture of radial neural network in the context of non-linearity.
See Topic: RADIAL BASIS FUNCTION NETWORKS, Short Answer Type Question No. 1.

10. a) What is delta learning rule?
b) Compare delta learning rule and Perceptron learning rule.
a) & b) See Topic: INTRODUCTION TO NEURAL NETWORKS, Short Answer Type Question No. 11(a) & (b).

c) With a diagram compare multilayer feed forward and recurrent network.
See Topic: ASSOCIATIVE MEMORY NETWORKS, Short Answer Type Question No. 3.

d) Using a single-node Perceptron with bipolar sigmoid function following classification problem is to be solved:

$X1 = (1, -2, 0, -1)^T$ $\quad d1 = 1$

$X2 = (0, 1.5, -0.5, -1)^T$, $\quad d2 = -1$

$X3 = (-1, 1, 0.5, -1)^T$ $\quad d3 = 1$

$W1 = (1, -1, 0, 0.5)$ $\quad \eta = 0.23$

Apply delta learning rule for one epoch to solve this problem.
See Topic: SINGLE-LAYER PERCEPTRONS, Long Answer Type Question No. 7.

11. Write the short notes any *three* of the following:
a) Gradient descent learning
b) Simulated annealing
c) Competitive learning
d) Gibbs sampling
e) Associative Memory Neural Network
f) Markov chains

a) See Topic: INTRODUCTION TO NEURAL NETWORKS, Long Answer Type Question No. 3(d).
b) See Topic: ASSOCIATIVE MEMORY NETWORKS, Long Answer Type Question No. 7(e).
c) See Topic: INTRODUCTION TO NEURAL NETWORKS, Long Answer Type Question No. 3(e).
d) See Topic: ASSOCIATIVE MEMORY NETWORKS, Long Answer Type Question No. 7(f).
e) See Topic: ASSOCIATIVE MEMORY NETWORKS, Long Answer Type Question No. 7(g).
f) See Topic: ASSOCIATIVE MEMORY NETWORKS, Long Answer Type Question No. 7(h).

QUESTION 2018

Group – A
(Multiple Choice Type Questions)

1. Choose the correct alternatives for *any ten* of the following:

i) Artificial Neural Networks are inspired by
a) Swarm intelligence
b) high speed parallel processor
✓c) human brain
d) All of these

ii) Discrete Hopfield net is the example of
a) feed forward network
✓b) recurrent network
c) feedback network
d) None of these

iii) The Hebbian rule is _____ type of learning.
✓a) supervised
b) competitive
c) unsupervised
d) reinforced

iv) Sigmoid is a _____ type of non-linearity.
a) hard
b) rough
✓c) soft
d) Both (b) and (c)

v) In Back-propagation algorithm, _____ is propagated backwards through the network.
a) error b) signal ✓c) error + signal d) signal – error

vi) Single layer Perceptron is used for
✓a) linear separability b) error minimization
c) back-propagation d) annealing

vii) Gaussian Activation Function is used in
a) Back Propagation ✓b) Radial Basis Function
c) Self Organizing Feature Map d) Mexican Hat Net

viii) The Madaline network is
a) the combination of two single layered feed forward neural network
✓b) a type of multilayered feed forward neural network with multiple neurons in output layer
c) the combination of Adaline networks and multilayered feed forward network with one neuron in output layer
d) a type of feedback network

ix) An auto associative network is
a) a neural network that contains no loops ✓b) a neural network that contains feedback
c) a neural network that contains only one loop d) None of these

x) SVM is based on
✓a) Supervised learning b) Reinforcement learning
c) Unsupervised learning d) Both (a) and (b)

xi) The synapse of a neuron is modelled by a
a) linear function b) non-linear function
c) non-linear rough function ✓d) linear/non-linear function

xii) For a four input (0, 0), (0, 1), (1, 0), (1, 1) neuron representing a Perceptron with $w1 = w2 = 1$ and $\theta = 1.5$ the classification does
✓a) AND classifier b) OR classifier c) XOR classifier d) None of these

Group – B
(Short Answer Type Questions)

2. Discuss different Activation Functions that are used in Artificial Neural Network. Which type of Activation Function is commonly used in Back Propagation algorithm?

1st part: See Topic: INTRODUCTION TO NEURAL NETWORKS, Short Answer Type Question No. 2.

2nd Part: See Topic: SINGLE-LAYER PERCEPTRONS, Short Answer Type Question No. 3.

3. What are the parameters to increase efficiency Hebbian Synapse as a function of the correlation between the pre-synaptic and post-synaptic? How the parameters are influencing the Hebbian Synapse?

See Topic: INTRODUCTION TO NEURAL NETWORKS, Short Answer Type Question No. 12.

4. What are differences between Supervised and Unsupervised Learning? How Reinforcement learning differs from Supervised Learning?

See Topic: INTRODUCTION TO NEURAL NETWORKS, Short Answer Type Question No. 13.

5. Use Cover's Theorem to solve classification problem of AND gate.

Wrong Question

6. Describe Hopfield Net and its training algorithm.

See Topic: ASSOCIATIVE MEMORY NETWORKS, Short Answer Type Question No. 1.

Group – C
(Long Answer Type Questions)

7. a) What is Evaluation function?

b) Explain Adaptive Heuristic Critic Learning.

c) Using Perceptron Convergence Theorem to find the weight vector required to perform following classifications:

Input Vector: (1, 0, 0), (1, 0, 1), (1, 1, 0), (1, 1, 1).

Decision Vector: (0, 0, 0, 1)

Consider Learning Rate as unity.

See Topic: SINGLE-LAYER PERCEPTRONS, Long Answer Type Question No. 8(a), (b) & (c)

8. a) What is Adaline? What type of learning is used in Adaline?

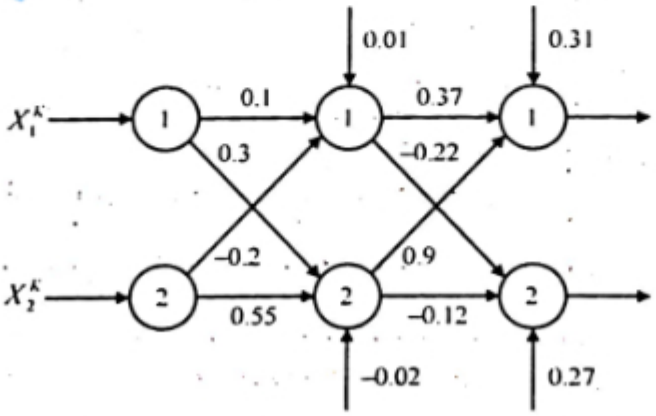
See Topic: INTRODUCTION TO NEURAL NETWORKS, Short Answer Type Question No. 3.

b) What is Delta Rule? Write the error function for the Delta Rule.

See Topic: INTRODUCTION TO NEURAL NETWORKS, Short Answer Type Question No. 6.

c) Using Back Propagation algorithm find the new weights for the following network using both the pattern index. Consider the Learning rate and momentum 0.35 and 0.75 respectively.

Pattern Index	X_1^K	X_2^K	d_1^K	d_2^K
1	0.5	–0.5	0.9	0.1
2	–0.5	0.5	0.1	0.9



See Topic: SINGLE-LAYER PERCEPTRONS, Long Answer Type Question No. 9.

9. a) What are differences between Perceptron algorithm and RBF?
b) What is pseudo-inverse matrix in Interpolation based RBF function?
a) & b) See Topic: RADIAL BASIS FUNCTION NETWORKS, Short Answer Type Question No. 2(a) & (b).

c) What is the advantage of using Pocket algorithm over Perceptron Convergence Theorem?
d) Determine the weights for XOR classification using RBF based Interpolation with fewer than Q basis functions.

Point K	x_1	x_2	d
X_1	0	0	0
X_2	0	1	1
X_3	1	0	1
X_4	1	1	0

Consider localized Gaussian basis function where $\sigma = 1$.
c) & d) See Topic: SINGLE-LAYER PERCEPTRONS, Long Answer Type Question No. 10(a) & (b).

10. a) Explain the Kohonen Self-organization Feature Map architecture with diagram.
b) Construct a Kohonen Self-organization Feature map to cluster four vectors [0 0 1 1], [10 0 1], [010 1], [1111] into two classes and start with any random weights.
a) & b) See Topic: ASSOCIATIVE MEMORY NETWORKS, Long Answer Type Question No. 3(a) & (b).

c) How Competitive Learning is different from Hebbian Learning?
See Topic: INTRODUCTION TO NEURAL NETWORKS, Short Answer Type Question No. 14.

11. Write short notes on any *three*:
a) Boltzman Learning
b) SVM classifier
c) Wavelet Neural Network
d) Error correction learning
e) Reinforcement learning

a) See Topic: INTRODUCTION TO NEURAL NETWORKS, Long Answer Type Question No. 3(f).
b) See Topic: APPLICATIONS, Long Answer Type Question No. 1.
c) See Topic: RADIAL BASIS FUNCTION NETWORKS, Long Answer Type Question No. 4.
d) See Topic: SINGLE-LAYER PERCEPTRONS, Long Answer Type Question No. 11.
e) See Topic: INTRODUCTION TO NEURAL NETWORKS, Long Answer Type Question No. 3(g).