

Introduction to Soft Computing
Prof. Debasis Samanta
Department of Computer Science & Engineering
Indian Institute of Technology, Kharagpur

Lecture – 37
Training ANNs (Contd.)

So, we are discussing Training Artificial Neural Networks.

(Refer Slide Time: 00:27)

The slide is titled "Learning of neural networks: Topics" and lists the following topics:

- ✓ Concept of learning
- ✓ Learning in
 - ✓ Single layer feed forward neural network
 - **Multilayer feed forward neural network**
 - Recurrent neural network

The slide also features logos for IIT Kharagpur and NPTEL Online Certification Courses, and a small video inset of the professor.


So, far we have discussed about so, how to train a neural net; it is basically based on the concept of learning. There are different learning techniques and we have considered the learning techniques again varies from different network architecture to different network architecture. In the last lectures we have discussed about how a single layer feed forward neural network can be trained. Today, we are going to discuss the multilayer feed forward neural network training and then the recurrent neural network training will be discussed.

Now, so, multilayer feed forward neural network training, basically, the similar approach to that of the single layer, but it follows more method more what is called a meticulous method, particularly it is if you consider the supervised training then it consider algorithms to train it and the popular algorithm in this regard is called back propagation algorithm.


(Refer Slide Time: 01:35)

Training multilayer feed forward neural network

- Like single layer feed forward neural network, supervisory training methodology is followed to train a multilayer feed forward neural network.
- Before going to understand the training of such a neural network, we redefine some terms involved in it.
- A block diagram and its configuration for a three layer multilayer FF NN of type $l - m - n$ is shown in the next slide.




IIT KHARAGPUR



NPTEL ONLINE CERTIFICATION COURSES

DEBASI DEPARTI IIT KH



So, today will discuss about the algorithm the whole. Now, before going to discuss about training a multilayer feed forward neural network for the simplicity of the discussion we will consider a multilayer feed forward neural network with the configuration $l-m-n$ that is called the $l-m-n$ network and it is basically a three layer feed forward neural network and l basically the number of neuron in the number of neurons in the first layer and m denotes the number of neuron in the hidden layer and n is the number of neuron in the output layer or in other words $l-m-n$ network is basically a network with l number of input and n number of output.

(Refer Slide Time: 02:28)

Specifying a MLFFNN

Input layer: N_1 , $[N_1]=l$


Hidden Layer: N_2 , $[N_2]=m$

Output Layer: N_3 , $[N_3]=n$


Linear transfer function: $f_1^i = (I_1, \theta_1)$

Log-Sigmoid transfer function: $f_2^m = (I_2, \alpha_2) = \frac{1}{1 + e^{-\alpha_2}}$

Tan-Sigmoid transfer function: $f_3^n = (I_3, \alpha_3) = \frac{e^{\alpha_3} - 1}{e^{\alpha_3} + 1}$




IIT KHARAGPUR



NPTEL ONLINE CERTIFICATION COURSES

DEBASI DEPARTI IIT KH



Now, let us see this figure, because our many discussion subsequently will be refer from this figure only. In this figure we have depicted the architecture of a l-m-n feed forward neural network and in this architecture. As you see this basically the input layer and this is the hidden layer and this is the output layer. So, the input layer network is called the N_1 the network which is there in hidden layer is called the N_2 and hidden layer that is there in output is called the N_3 .

So, we can see it is basically the cascading of 3 network N_1 , N_2 and N_3 one by one is a cascading in the sense that the. So, this is the input to the input layer and here we can consider l number of inputs the l number of inputs are denoted as $I_1, I_2 \dots I_l$ and then subscript denotes that it is a input layer. So, so these are the input basically and so, the set of input we can denote it is I and these are the perceptrons in this input layer l number of perceptrons are there. So, they are termed as I_1, I_2, \dots, I_l like this one. So, one indicate that it is a first layer and then next symbol increases that which make symbol if it is I_i indicate that is i-th perceptron in the first layer.

Now, the input to any perceptron is basically this one. For example, to this perceptron input is I_1 to this perception input is this one and. So, on now output of a perceptron in the input layer we denotes it is O_i ; O_i denotes the output of the input layer and here we can say that the output of the input layer is basically input to the hidden layer. So, input to the hidden layer we denoted I_H . Similarly, the output of the hidden layer we denoted O_H . So, these are the output of the hidden layer and the output of the hidden layer is basically output of the output layer and input of the output layer we denoted I_O . So, these are the basically input of the output layer and finally, this is the output layer is called the O .

So, here basically I and then O input and output through this network right it basically mapped from an input to output. So, this is idea about it. Now, so, this is the working of the multilayer feed forward neural network more specifically l-m-n network now here will consider few more things as I told you this is a network n and size of the network N_1 is l similarly, this is our network N_2 size of the network is m and this is a network N_3 size of the network is n. So, l-m-n like.

Now, in each perceptron here in the input layer they will follow each perceptron they will follow transfer function and then thresholding value for the simplicity for. So, this is

the idea. So, if this contains the first perceptron in the input layer we denote the transfer function as f_i . So, f_i this is the f_i means the transfer function of the i -th perceptron in the input layer this one.

So, this is basically it basically I_i , I_i and θ_i where θ_i is the threshold value for this perceptron. So, what I want to say is that, so, each neurons are characterized with their own transfer function and then thresholding value and then input when give it pass predict through this input it will produce output. Now, this output then will be fitted to all the neurons in the hidden layer. So, this output go to this layer this layer this perceptron and this perceptron and this one.

Now, whenever this output of the input layer goes to the goes does an input to the hidden layer they will be associated with weights. So, from the perceptron one in the input layer if it is goes to the perceptron one in the hidden layer we can say that it is weighted by V_{i1} . So, similarly we denote V_{ij} ; that means, it is a weight for the input when it passes from the i -th perceptron in the input layer to the j -th perceptron in the hidden layer, so, V_{ij} . So, this basically constitutes the weights for all signals that can be fed to the hidden layer and we can represent this weight by means of a matrix, V matrix that we have already discussed and it is similar the that of the single layer feed forward network. So, in this case the matrix is size l cross m . So, this is basically $l \times m$ matrix, l number of rows and m number of columns are their.

Next, the output of the hidden layer goes to the goes as an input to a where each perceptron in the output layer for example, if j -th new perceptron it is there in hidden layer so output from this perceptron goes there and goes all the perceptrons in the output layer. Now, likewise the input to the hidden layer here also all input to the output layer will be associated with weights. So, these are the weight matrix and this weight matrix is denoted by W_m and in this case the size of the weight matrix is m cross n , because from m number of perceptron it goes to the n number of perceptron in the output layer. So, m cross n .

So, it is a similar the V matrix that we have discussed in the previous in between input to hidden layer here also W matrix is between hidden layer to output layer. Now, again for each perceptron in hidden layer it will be characterized with transfer function and then thresholding value. So, if it is in the j -th layer j -th perceptron in this layer then we can

represent that f_j m and this basically the input to the j -th perceptron and this α is a threshold value we can assume here that all the transfer function that it basically here for each perceptron is denoted by this one it is basically log sigma transfer function and in this case we have considered a linear transfer function, that means, it is simply the pass input to the output.

So, so, these are the two transfer functions and threshold values are there in input layer and then output layer. Now, similarly the transfer function that we have considered here for any case perceptron in this layer is denoted by this one and here we have considered the log the tan sigma transfer function. So, for the sake of varieties we have disc[uss]- we have considered different transfer functions in the different perceptrons with the different threshold values in each layer.

So, this completes the description of the element network as a multilayer feed forward neural network. So, now, once this architecture is clear then we will be able to see how this network architecture can be trained. So, trained means there are many things are to be learned. So, far training is concerned the first of all how many numbers of perceptron should be there in the input layer that obviously, specified by the number of inputs of course, similarly number of perceptrons in the output layer it is also decided by the output.

So, these two things are very simple, right it is, but so, for the learning is concerned how many neurons are there should be in the hidden layer it also needs to be learned. So, the n this m value needs to be a plan. So, this is a another learning parameter and then we have considered the threshold valued in each perceptron in the input here. So, this values also needs to be learned, then α_j value for each perceptron also needs to be learned and then here also α values for each perceptron to be learned. So, these are the learning parameter therefore.

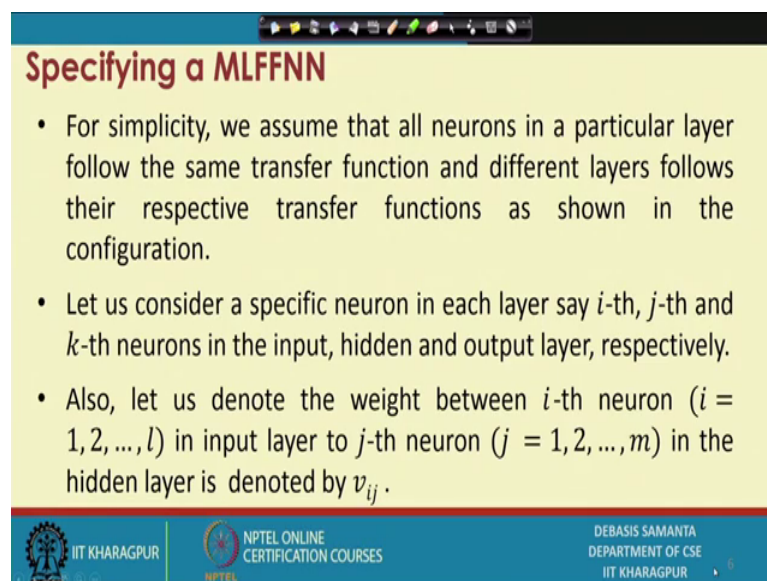
So, m to be learned θ_i values for each perceptron α_j values for each perceptron α_k values for each perceptron are to be learned. Another also the things to be learned that we have considered here in our discussion that this is a transfer function, but it can be other transfer function always as well. So, there many transfer function. So, it is also needs to be learned which transfer function will be better so far the accurate output is concerned. So, all the transfer function that we have discussed assume that these are

transfer function, but here also system should learn or the transfer function that it should be here. So, these are the different things to be learned.

So, these are the objective of learning. So, learning means we are fine I also forgot to mention one thing another learning parameter is V and W . So, these are most important metrics or parameters to be learn V , W . Now, in this discussion it is not possible to discuss all the learning parameters, but they can be learned in the same way the any one other parameter can be learned and for the simplicity in our discussion we will consider only how this network architecture should be trained. So, that it can learn the V matrix and W matrix for the application. So, we learn about how to how the network can be trained so that this network can learn V matrix and W matrix and following the same approach we can learn other network parameters.

So, so, this is the objective of the learning that we are going to discuss.

(Refer Slide Time: 13:12)



Specifying a MLFFNN

- For simplicity, we assume that all neurons in a particular layer follow the same transfer function and different layers follows their respective transfer functions as shown in the configuration.
- Let us consider a specific neuron in each layer say i -th, j -th and k -th neurons in the input, hidden and output layer, respectively.
- Also, let us denote the weight between i -th neuron ($i = 1, 2, \dots, l$) in input layer to j -th neuron ($j = 1, 2, \dots, m$) in the hidden layer is denoted by v_{ij} .

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES | DEBASIS SAMANTA DEPARTMENT OF CSE IIT KHARAGPUR

Now, fine and for the sake of discussion we will consider these are notation that we will follow so that we can understand the things completely. We can say any neuron in the input layer as the i -th neuron i is one to l . Similarly, any neuron in the hidden layer we denote it as j -th neuron and any neuron in the output layer we denote in the k -th neuron.

(Refer Slide Time: 13:43)

Specifying a MLFFNN

- The weight matrix between the input to hidden layer say V is denoted as follows.

$$V = \begin{bmatrix} v_{11} & v_{12} & \dots & v_{1j} & \dots & v_{1m} \\ v_{21} & v_{22} & \dots & v_{2j} & \dots & v_{2m} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ v_{i1} & v_{i2} & \dots & v_{ij} & \dots & v_{im} \\ v_{l1} & v_{l2} & \dots & v_{lj} & \dots & v_{lm} \end{bmatrix}$$

l x m

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES | DEBASIS SAMANTA DEPARTMENT OF CSE IIT KHARAGPUR

And, similarly there is a weight associated from i -th neuron in the input layer to the j -th neuron in the hidden layer, we denoted by a V matrix and the V matrix is like this. So, this is the usual V matrix that we have already used it in the singular feed forward neural network training.

So, it is basically v_{ij} denote the weights from the i -th neuron to the j -th neuron in the output layer to the j -th neuron in the hidden layer and their values of i will vary from one to l and values of j will vary from 1 to m and this way it is an l cross m matrix. Now, so, this is the matrix basically we have to learn it by means of training we should learn the different elements here in this matrix.

(Refer Slide Time: 13:43)

Specifying a MLFFNN

- Similarly, w_{jk} represents the connecting weights between j -th neuron ($j = 1, 2, \dots, m$) in the hidden layer and k -th neuron ($k = 1, 2, \dots, n$) in the output layer as follows.

$$W = \begin{bmatrix} W_{11} & W_{12} & \dots & W_{1k} & \dots & W_{1n} \\ W_{21} & W_{22} & \dots & W_{2k} & \dots & W_{2n} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ W_{j1} & W_{j2} & \dots & W_{jk} & \dots & W_{jn} \\ W_{m1} & W_{m2} & \dots & W_{mk} & \dots & W_{mn} \end{bmatrix}$$

m x n

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES | DEBASIS DEPARTI IIT KH

Now, likewise we denote the w_{jk} representing it is a weight from the j -th neuron to the k -th neuron in the hidden layer to the k -th neuron in the output layer. Now, all the weights that is there in this network can be represented by a matrix this is called the W matrix and w_{jk} , jk represents basically weight from the j -th neuron to the k -th neuron and it is therefore, a matrix of m cross n size where m is a number of neuron in the hidden layer and n is the number of neuron in the output layer. So, this matrix W also needs to be learned and by means of training process we have to learn this matrix.

(Refer Slide Time: 15:25)

Learning a MLFFNN

- Whole learning method consists of the following three computations:
 1. Input layer computation
 2. Hidden layer computation
 3. Output layer computation
- In our computation, we assume that $T = \langle T_i, T_j \rangle$ be the training set of size $|T|$.

Diagram: A circle labeled 'T' contains two smaller circles labeled 'T_i' and 'T_j'. Arrows point from 'T_i' and 'T_j' to a larger circle labeled 'I'. Below this, a circle contains the vector '⟨1, 0⟩'.

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES | DEBASIS DEPARTI IIT KH

Now, so far the training is concerned basically training here in the neural architecture needs a number of computation we can systemize the computational in the three steps method; first we have to compute the input layer, then hidden layer and output layer. So, we say that input layer computation, hidden layer computation and output layer computation. Now, so, far in our computation is concerned competition is based on some training set. So, here we denote the training set as T and T the training set consists of the input data and then output data for any input I which is belong to T I has an associated O , $I O$. So, it is basically I and O one sets, this sets is basically supervised training sets.

So, T is the training data here. So, given the training data right, we have to learn the different network parameter in this case V and W matrix. Now, to learn it or as a process of learning we have to learn we have to compute the different layers in each in the network.

(Refer Slide Time: 16:43)

Input layer computation

- Let us consider an input training data at any instant be $I^l = [I_1^l, I_2^l, \dots, I_i^l, I_l^l]$ where $I^l \in T_l = O^l$
- Consider the outputs of the neurons lying on input layer are the same with the corresponding inputs to neurons in hidden layer. That is, $O^l = I^l$
 $[l \times 1] = [l \times 1]$

[Output of the input layer]

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES | DEBASI DEPARTI IIT KH

And, so now, let us see first the input layer computation. Now, the input layer computation can be discussed like this say suppose any input I which is in T , T you have discussed about is the input set and this a I consists of l number of inputs for each neuron to the input layer. So, we can denoted I_1, I_2, \dots, I_l this one, ok. So, this basically is a one input that is there in the input set.

So, this is basically the one input which is belongs to this training set T and the output layer combination is prevail in this case this is because whatever because we have we

have considered the linear transformation. So, linear transformation means y equals to x . So, so the whatever the value of x will be directly pass to the y . So, in this is why the output layer combination is that if O is basically the output instance at any time then give then the it is input is I or in other words if I that mean if this is the input then it is output is also O it same because of linear transformation.

Now, all these things we can represent in terms of a matrix. So, all these right it can be represent in a matrix like I_1, I_2 and so on, so on. So, this is basically the matrix and as these l number of inputs datas are there so, this is basically l cross n matrix; similarly, I also l cross n matrix. So, what you can say that so, this is the input layer combination; that means, input to any perceptron in the input layer will be the output of that perceptron in that layer and this the any neurons input and corresponding output can be represented by means of this matrix formulation. This matrix of size l cross n l cross 1 and this matrix of size l cross 1 .

So, this is the idea about the input layer computation.

(Refer Slide Time: 19:07)

Input layer computation

- The input of the j -th neuron in the hidden layer can be calculated as follows.

$$I_j^H = [v_{1j}O_1^I + v_{2j}O_2^I + \dots + v_{lj}O_l^I + v_{lj}O_l^I]$$
 where $j = 1, 2, \dots, m$.
 [Calculation of input of each node in the hidden layer]
- In the matrix representation form, we can write

$$I^H = V^T \cdot O^I$$

$$[m \times 1] = [m \times l][l \times 1]$$

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES | DEBASI DEPARTI IIT KH

And, similarly we can say that if this is the output of a perceptron in the input layer then that will work as a input to the hidden layer. Now, if we consider any j -th neuron in the hidden layer then it is input can be considered as I_j^H . Now, this I_j^H can be expressed by this of product. So, basically it is from the first neuron to the j -th this is the symmetric

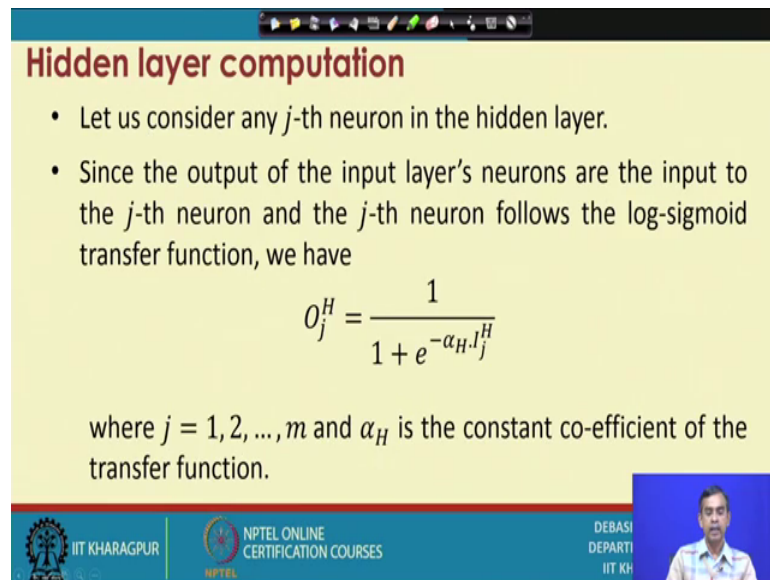
the what is called the weight values and then it basically the output of the first neuron in the input layer.

Similarly, the second neuron to the j -th and then output from the second neuron in the input layer and this is a v_{ij} the weight from the i -th to j -th and it is basically the output of the j -th neuron in the input layer. So, this concept is there and this basically is a summation of all the inputs with their weights it gives the input to any perceptron in the hidden layer. So, $I_j H$ is basically input to the i -th perceptron in the hidden layer and this basically the calculation of input to the hidden layer, ok.

And, now this expression this expression can be represented in the matrix form which is represented here. Here it basically $I H$ is basically all the inputs to the hidden layer it can be represented V transpose matrix and $O I$ matrix. So, $O I$ basically output of the input layer and V transpose matrix is a V matrix and it is transpose form. So, in other word, if $I H$ is basically m cross l matrix and V -th is the transpose V is a l cross m and it transpose from m cross l matrix and $O I$ is a l cross matrix. So, whole the things or whole this input layer computation can be expressed, in the form of a matrix representation. And, what we are observing is that all the calculations for example, input to the input layer or output of the input layer and input to the hidden layer is basically input layer calculation and all these computation can be expressed in the form of a matrix as it is here.

So, this is basically matrix representation and then matrix operation and then neural network training and then it is describing the model is nothing, but a matrix model or matrix formulation of the model.

(Refer Slide Time: 21:48)



Hidden layer computation

- Let us consider any j -th neuron in the hidden layer.
- Since the output of the input layer's neurons are the input to the j -th neuron and the j -th neuron follows the log-sigmoid transfer function, we have

$$O_j^H = \frac{1}{1 + e^{-\alpha_H \cdot I_j^H}}$$

where $j = 1, 2, \dots, m$ and α_H is the constant co-efficient of the transfer function.

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES | DEBASI DEPARTI IIT KH

So, this give the input layer combination and then we will come to the hidden layer combination. Now, we know exactly what is the input to any perceptron any j -th perceptron in the hidden layer now our task is to calculate what is the output of the j -th neuron in the hidden layer we represent the output of j -th neuron hidden layer as this form O_j^H and as you have discussed that the transfer function that it used is basically log sigmoid and it has this form and these are the α_H indicates that it is basically it is α_H better we can write that it is basically the threshold value of the j -th neuron in the hidden layer and I_j^H if it is a input. So, it is basically e to the minus α_H concept. So, it is the output. So, this output is basically for the j -th neuron in the hidden layer.

Now, the way we have expressed the matrix representation so, this calculation or the output combination also can be represented in the form of matrix and the matrix will look like this.

(Refer Slide Time: 22:55)

Hidden layer computation

- Note that all output of the nodes in the hidden layer can be expressed as a one-dimensional column matrix.

$$O^H = \begin{bmatrix} \dots \\ \dots \\ \vdots \\ 1 \\ \vdots \\ \dots \\ \dots \end{bmatrix}_{m \times 1}$$

The slide also features a video inset of a speaker and logos for IIT KHARAGPUR, NPTEL ONLINE CERTIFICATION COURSES, and DEBASI DEPARTI IIT KH.

It is basically that matrix we can say that O^H matrix is the O^H matrix this means the output of the hidden layer and output of the hidden layer because m number of perceptrons are there. So, it is a matrix of size m cross n . So, it basically includes the output for the first perceptron second perceptron and these the m -th perceptron. So, it basically matrix that mean all the outputs of the perceptrons in the hidden layer can be represented by this type of matrix. So, this is another matrix to compute the hidden layer.

Now, so, the hidden layer output is known, hidden layer input is known, input layer input is known, input layer output is also known.

(Refer Slide Time: 23:45)

Output layer computation

Let us calculate the input to any k -th node in the output layer. Since, output of all nodes in the hidden layer go to the k -th layer with weights $w_{1k}, w_{2k}, \dots, w_{mk}$, we have

$$I_k^O = w_{1k} \cdot o_1^H + w_{2k} \cdot o_2^H + \dots + w_{mk} \cdot o_m^H$$

where $k = 1, 2, \dots, n$.

In the matrix representation form, we have

$$I^O = W^T \cdot O^H$$
$$[n \times 1] = [n \times m][m \times 1]$$

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES | DEBASI DEPARTI IIT KH

Now, we are in a position to discuss about output layer computation. Now, in case of output layer we know output of the hidden layer works as an input to the output layer. Now, if there is any neuron say k -th neuron and we denote the I_k^O represents that it is basically input to the k -th neuron at the output layer. So, I_k^O is a input to the k -th neuron at the output layer.

Now, this input is basically is a summation is a sum of the products of the weight matrix corresponding to the output from all the perceptrons from the hidden layer. So, this can be expressed this is basically output from the first layer in the hidden layer, output from the second perceptron in the hidden layer and output from the m -th perceptron in the hidden layer and is multiplied by this product is the weight matrix weight values from the first neuron to the k -th neuron from the second neuron in the second layer to the k -th neuron in the output here and so on, so on.

So, this way this basically is an expression is the input to the k -th neuron at the output layer and there are n number of neuron. So, k will varies from 1 to n . So, this basically computes the input to the output layer. Now, this expression can be expressed in the in terms of matrix representation which is here. So, I^O denotes all the inputs at the output layer and this can be expressed the transposition of weight matrix and multiplied by the O^H , O is basically output of the hidden layer, this output of the hidden layer we have already learn how to get it. So, this way the matrix representation of the output layer,

input of input to the output layer can be obtained and we can represent it this matrix as this is a n cross 1 matrix, this is n cross m because W is a n cross m matrix and this is an m cross 1 matrix.

So, so, this basically the input computation at the output layer or is called the output layer computation. Now, we will discuss about the output of the output layer.

(Refer Slide Time: 26:03)

Output layer computation

Now, we estimate the output of the k -th neuron in the output layer. We consider the tan-sigmoid transfer function.

$$O_k = \frac{e^{\alpha_0 \cdot I_k^0} - e^{-\alpha_0 \cdot I_k^0}}{e^{\alpha_0 \cdot I_k^0} + e^{-\alpha_0 \cdot I_k^0}}$$

where $k = 1, 2, \dots, n$

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES | DEBASI DEPARTI IIT KH

Now, the similarly the way the output of the input layer hidden layer we have calculated in the same way we can express the output of the output layer. Now, we can see that we have already mentioned that the transfer function that we have used they are in output layer is basically they tan sigma transfer function which take this form. So, it basically the output of any k -th neuron in the output layer so, this is the output of any k -th neuron in the output layer and there are n number of neuron in the output layer so, k values varies from 1 to n .

(Refer Slide Time: 26:48)

Output layer computation

- Hence, the output of output layer's neurons can be represented as

$$O = \begin{bmatrix} \dots \\ \dots \\ \dots \\ \frac{e^{\alpha_0 \cdot I_k^0} - e^{-\alpha_0 \cdot I_k^0}}{e^{\alpha_0 \cdot I_k^0} + e^{-\alpha_0 \cdot I_k^0}} \\ \dots \\ \dots \\ \dots \end{bmatrix}_{n \times 1}$$

The slide includes logos for IIT KHARAGPUR, NPTEL ONLINE CERTIFICATION COURSES, and DEBASI DEPARTI IIT KH, along with a small video inset of the presenter.

Now, so, this expression is for a particular perceptron, now, as a whole for the entire network at the output side also can be expressed and that too can be expressed using the matrix representation. So, this is a matrix representation for the output layer., output layer here O denotes the all outputs that can be obtained from the output layer and this basically this is the entry of the output of the first neuron in the output layer and is a second neuron in the output layer and this is the nth nth neuron in the output layer.

So, the these things also can be represented by means of a matrix and here alpha O is denotes that what is the threshold values. Now, here if we can for the simplicity we can consider that only one values of alpha. So, alpha I is basically the threshold value in the input layer for all the perceptron, for the simplicity we assume it. Similarly, alpha H we do not the threshold values of all perceptrons in the hidden layer and alpha O denotes the threshold values of all perceptrons in the output layer.

Now, this is a just simply an assumption that we considered same values of threshold values for all neutrons neurons belongs to a particular layer, but in actual practice it is not necessarily the same we can consider the different threshold values for different perceptrons in the network, but it will increase the complexity it will demand more calculation so, otherwise it is not the issue it is only the issue of computation time.

Now, we have discussed about the different layer computation and all these layer computations will be used to train the network because this is the ok, is a mathematical

manipulation that how we can represents a neural network mathematically. Now, we have just now discussed that how the entire neural network can be represented mathematically and that mathematical representation in the form of a matrix representation. Now, in the next lecture will discuss about using these are the different calculation how we can train the network.

Now, for the training in multilayer feed forward neural network there are many training procedures, but in this lesson will discuss about a particular which is most popular the back-propagation algorithm. So, that we will be discuss in the next lecture slides.

Thank you.