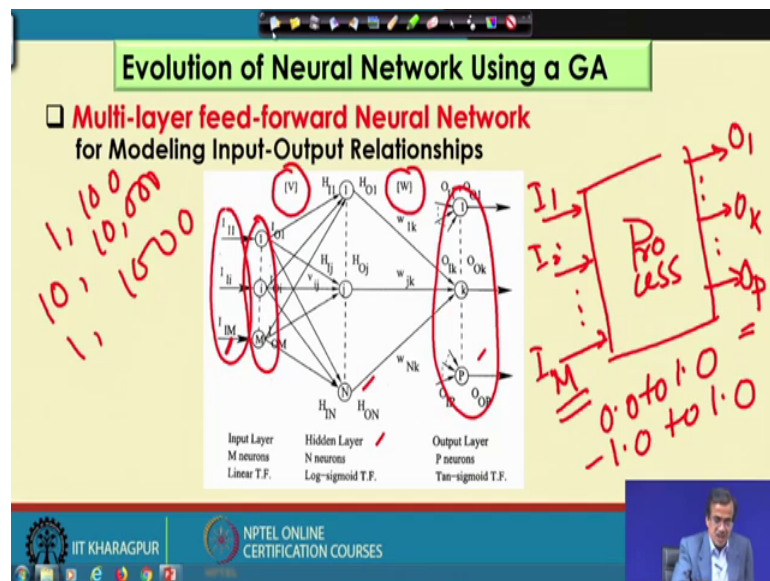


**Traditional and Non-Traditional Optimization Tools**  
**Prof. D. K. Pratihari**  
**Department of Mechanical Engineering**  
**Indian Institute of Technology, Kharagpur**

**Lecture – 38**  
**Genetic Algorithm as Evolution Tool (Contd.)**

So, our aim is to represent the input output relationship of a process.

(Refer Slide Time: 00:19)



Having like  $M$  inputs, and the  $P$  outputs. Now let me draw the block diagram of this particular process. So, this is the process say, and we have got the inputs like  $I_1, I_2, \dots, I_M$ , and we have got like  $M$  such inputs, and we have got  $P$  number of outputs. So, this is  $O_1$ , then comes  $O_k$ , and we have got the last is your  $O_P$ . So, we have got  $M$  number of inputs and  $P$  number of outputs.

Now to represent that on this particular the hidden layer, we use capital  $M$  number of the neurons. So, it is having capital  $M$  number of neurons, and to represent the output and the output layer. So, we use capital  $P$  number of neurons. So, the number of neurons in the input layer, and the output layer those are kept fixed. The only thing which you can do is we can change the number of hidden layers, and we can also change the number of neurons to be present on this particular the hidden layer, that decides actually the topology of this particular the network.

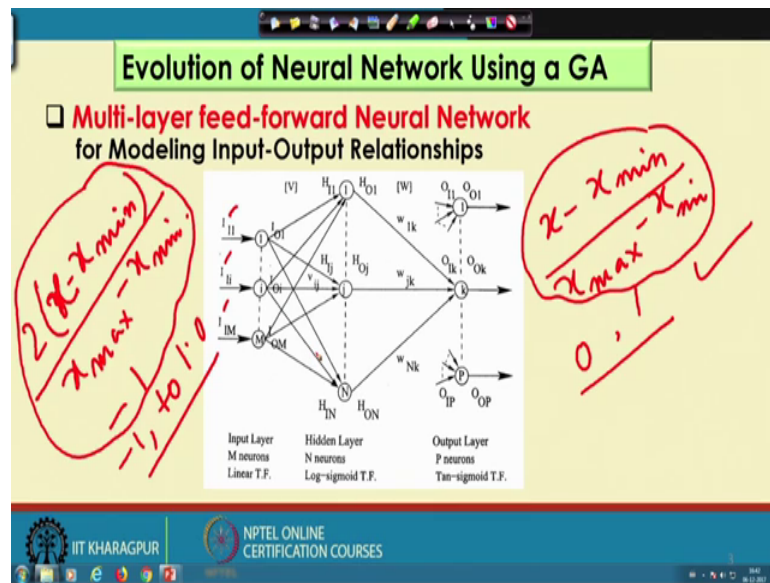
Now here let me assume that I am going to consider only 1 hidden layer, and this particular hidden layer is going to contain capital N number of neurons. Now and as I told that these particular connecting weights that is V and W should lie in the range of like either 0 to 1 0 to 1.0 or from minus 1 to 1.0; that means, in the normalized scale, and the starting values of these particular the connecting weights are generated at random using the random number generator, and through a large number of iteration, we are going to update those connecting weight. So, that we can get very accurate input output relationship.

Now before I go for that actually, I will have to or concentrate on this particular the inputs like how to pass this set of inputs to the network, now we have got capital M number of these inputs. So, all the inputs may not have the same range for variation for example, say the input the first input may vary in the range of say 1 to 100 the I-th 1 the I-th input may vary from say 10 to 10000, and this M-th input may vary from say 1 to 1000.

So, the range for this particular variation for the different inputs are not the same, and this is going to create some problem, and we will not be able to actually determine the very accurate this input output relationship, and that is why actually what we will have to do is, we will have to do something to express these particular inputs in the normalized scale.

Now, let us see how to express this particular the inputs in the normalized scale. To express the input in the normalized scale, either in the scale of 0 to 1 or from minus 1 to plus 1 we use some relationship like this.

(Refer Slide Time: 03:51)



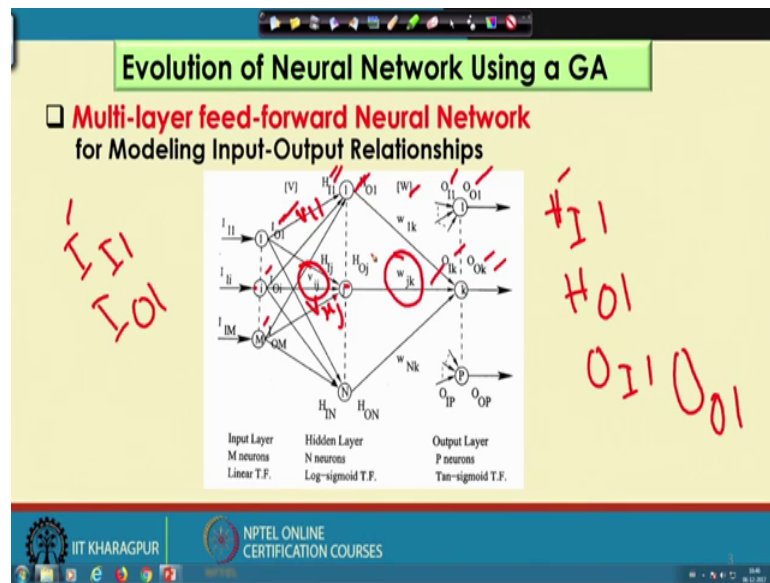
For example, say  $x$  minus  $x$  minimum divided by  $x$  maximum minus  $x$  minimum. Now if I put  $x$  equals to  $x$  minimum. So, I will be getting 0 and if I put  $x$  equals to  $x$  maximum. So, I will be getting 1. So, using this particular the relationship I can represent so, these particular inputs in the scale of 0 to 1.

Now, similarly if I want to represent in the scale of minus 1 to plus 1 so, we will have to use 1 rule that is your 2 into  $x$  minus  $x$  minimum divided by  $x$  maximum minus  $x$  minimum minus 1. Now if put  $x$  equals to  $x$  minimum. So, I will be getting 0 here. So, this is going to give minus 1, and if I put  $x$  equals to  $x$  maximum. So, I will be getting 2 minus 1 that is plus 1. So, this will vary from minus 1 to plus 1. So, if I use this particular formula. So, I will be getting these particular inputs in the scale of minus 1 to plus 1, and if I use these I will be getting this particular the inputs in the scale of 0 to 1.

Now, supposing that I know the normalized values of each of these particular the inputs, now I am passing those normalized values through, this particular the network, and let us see what happens to this particular the network. Now here actually I am just going to pass, all such inputs in the normalized scale through this input layer. So, I will be getting some output here.

Now, these particular outputs will be multiplied by the corresponding connecting weights, and those things will be summed up here.

(Refer Slide Time: 05:40)



So, I will be getting the input for this particularly the hidden layer, and here we have got a transfer function. So, I will be getting this output here, and once you have got this particular output of the hidden layer. So, once again I multiply with the corresponding connecting weight, we sum them up and we will be getting the input for this particular the output layer, and using this transfer function I will be getting this particular the output. Now these things whatever I told, I am just going to discuss in much more details.

Now here let us see how to find out the outputs corresponding to a set of inputs, now before I go for that I should spend some time on this particular nomenclature which you are using here. Now here if you see I have written  $I_1$ . So, this is actually  $I_1$  now this,  $I_1$  indicates the input of the first neuron lying on this particular the input layer, at this  $O_1$  that is  $O_1$  is nothing, but the output of the first neuron lying on this particular the input layer.

Now, similarly if I see this  $I_1$  so, this  $I_1$  is nothing, but the input of the first neuron lying on the hidden layer, then your  $O_1$  is the output of the first neuron lying on this particular the hidden layer, then comes this particular symbol that is  $O_1$  is nothing, but the input of the first neuron lying on the output layer, and this  $O_1$  is nothing, but the output of the first neuron lying on this particular the output layer.

Now, let me let me repeat for this particular the general case for example, say  $I$  that is the input of the  $I$ -th neuron lying on the input layer  $O$ . So, this is the input of the sorry output of the  $I$ -th neuron lying on the input layer, then comes here  $H_{ij}$  that is the input of the  $J$ -th neuron lying on the hidden layer,  $H_{Oj}$  that is the output of the  $J$ -th neuron lying on the hidden layer.

Then comes  $O_{Ik}$  that is input of the  $K$ -th neuron lying on the output layer and  $O_{Ok}$  that is nothing, but the output of the  $K$ -th neuron lying on the output layer, and here these connecting weights between  $A_1$ , and  $A_1$  is denoted by  $V_{11}$ . Similarly between  $M$  and  $j$  so, this is nothing, but  $V_{Mj}$ , and here between  $I$  and  $j$ . So, this is denoted by  $V_{Ij}$ .

Similarly, the connecting weights between the  $j$ -th neuron so, so  $j$ -th neuron lying on the hidden layer, and the  $k$ -th neuron lying on this particular output layer is denoted by  $W_{jk}$ . So, these are actually the notations which we are going to use in this type of the network ; that means, for 1 set of inputs like how to reach this output layer, and how to get this particular your the output. Now let us try to see this particular thing.

(Refer Slide Time: 09:33)

**Connecting Weights**

$$[V] = \begin{bmatrix} V_{11} & \dots & V_{1j} & \dots & V_{1N} \\ \vdots & \dots & \vdots & \dots & \vdots \\ \vdots & \dots & \vdots & \dots & \vdots \\ V_{i1} & \dots & V_{ij} & \dots & V_{iN} \\ \vdots & \dots & \vdots & \dots & \vdots \\ \vdots & \dots & \vdots & \dots & \vdots \\ V_{M1} & \dots & V_{Mj} & \dots & V_{MN} \end{bmatrix} \quad [W] = \begin{bmatrix} W_{11} & \dots & W_{1K} & \dots & W_{1P} \\ \vdots & \dots & \vdots & \dots & \vdots \\ \vdots & \dots & \vdots & \dots & \vdots \\ W_{j1} & \dots & W_{jK} & \dots & W_{jP} \\ \vdots & \dots & \vdots & \dots & \vdots \\ \vdots & \dots & \vdots & \dots & \vdots \\ W_{N1} & \dots & W_{NK} & \dots & W_{NP} \end{bmatrix}$$

*M x N*      *N x P*

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

Now here as I told the connecting weights that is  $V$  is denoted by this particular matrix. Now here in this particular matrix we have got like,  $M$  number of rows, and we have got like your  $N$  number of columns. So, this is nothing, but so, this is nothing but your the  $M$  cross  $N$  matrix. Now similarly we have got the  $V$  this  $W$  matrix that is nothing but the

connecting weights between the hidden layer, and the output layer and this particularly if you see the dimension so, here we have got like your so, we have got this particular N. So, N number of rows, and we have got P number of columns. So, here the dimension is nothing, but N cross P.

So, these are the dimension of this particular the connecting weights matrix, and if you see the individual value like  $V_{11}$  or say  $V_{1j}$ . So, this will lie either in between 0 and 1 or from in the range of minus 1 to the plus 1, and as I told that initially those connecting weights are selected at random, and then we try to modify.

(Refer Slide Time: 10:53)

**Forward Calculations**

- Step 1:** Calculation of the outputs of input layer  
 $I_{O_i} = I_{I_i}$ ,  
 where  $i = 1, 2, \dots, M$ , y = u
- Step 2:** Calculation of the inputs of hidden layer  
 $H_{I_j} = v_{1j} I_{O_1} + \dots + v_{Mj} I_{O_M}$ ,  
 where  $j = 1, 2, \dots, N$
- Step 3:** Calculation of the outputs of hidden neurons  
 $H_{O_j} = \frac{1}{1 + e^{-a_1 H_{I_j}}}$ ,  
 where  $a_1$  represents the coefficients of TF

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

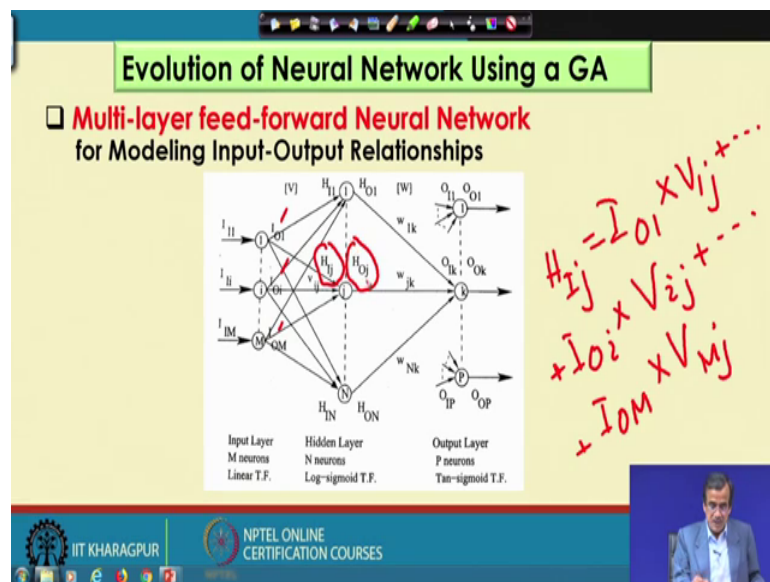
Now let us see how to carry out. So, this type of the forward calculation. Now the step 1 calculation of the output of the input layer, now if we remember on the input layer we are using the linear transfer function, and that is your  $y$  is nothing, but  $u$  now if use linear transfer function that is output is nothing, but input so, I can write down the input of the  $I$ -th neuron lying on these particular input layer is nothing, but the output of the  $I$ -th neuron lying on this particular input layer.

Where  $I$  varies from 1 2 up to  $M$  so, this capital  $M$  is actually the total number of inputs and that is nothing, but the total number of neurons lying on the input layer. So, this output is nothing, but the input because we are considering the linear transfer function like output equals to input that is  $y$  is nothing, but  $u$  sort of thing that is  $y$  is nothing, but  $u$  output is nothing but input.

Now next is the step 2. So, we calculate the inputs of the hidden layer that is  $H_{Ij}$ , now this  $H_{Ij}$  is nothing, but the input of the  $j$ -th neuron lying on the hidden layer and that is nothing, but the output of the first neuron lying on the hidden layer that is  $I_{O1}$  multiplied by  $V_{1j}$  plus a few terms are there next is  $V_{ij} I_{Oi}$  plus, there are a few other terms and the last term will be  $V_{Mj}$  multiplied by  $I_{OM}$  where  $j$  varies from 1 2 up to  $N$ .

Now, let us try to see from this particular the diagram. Now if I see from the diagram. So, this is something like this, like I am just going to find out what is this  $V_{Ij}$  so, if I want to find out.

(Refer Slide Time: 13:05)



So, we have already got this particular output we have got, now I am trying to find out what is this  $H_{Ij}$ . So, this  $H_{Ij}$  this  $H_{Ij}$  will be nothing, but so this particular output that is  $I_{O1}$  multiplied by. So, this connecting weights that is  $V_{1j}$  that is  $V_{1j}$  plus a few terms are there, then this middle term that is your  $I_{Oi} I_{Oi}$  multiplied by so, this particular  $V_{ij}$ , there are a few other terms, and this last term will be your  $I_{OM} I_{OM}$  multiplied by  $V_{Mj}$ .

So, this is nothing, but your this  $H_{Ij}$  that is the input of the  $j$ -th neuron lying on this particular the hidden layer, and once you got it and I know the transfer function that is the log sigmoid transfer function which I am going to use. So, very easily I can find out

what should be the output of the j-th neuron lying on the hidden layer. So, that I am going to find out mathematically.

Now once you have got this particular the input of the hidden layer, now let us try to find out the output of the hidden layer. Now if we want to find out the output of the hidden layer, it is very simple because we know this particular the transfer function that is nothing, but your the log sigmoid transfer function. So, here we try to find out the output of the j-th neuron lying on the hidden layer is nothing, but 1 divided by 1 plus e raised to the power minus a 1 multiplied by H Ij or a 1 represents the coefficient of this particular the transfer function. So, this a 1 is nothing, but the coefficient of the transfer function.

(Refer Slide Time: 15:11)

**Forward Calculations**

- **Step 1:** Calculation of the outputs of input layer  

$$I_{O_i} = I_{j_i}$$
 where  $i = 1, 2, \dots, M$
- **Step 2:** Calculation of the inputs of hidden layer  

$$H_{ij} = v_{1j} I_{O_1} + \dots + v_{ij} I_{O_i} + \dots + v_{Mj} I_{O_M}$$
 where  $j = 1, 2, \dots, N$
- **Step 3:** Calculation of the outputs of hidden neurons  

$$H_{O_j} = \frac{1}{1 + e^{-a_1 H_{I_j}}}$$
 where  $a_1$  represents the coefficients of TF

So, this is actually the coefficient of the transfer function, and the out input is this H Ij, and this is nothing but the output. So, this is actually this log sigmoid transfer function which I have already discussed.

Now using this log sigmoid transfer function, I can find out like what should be the output for this particular the jth neuron lying on the hidden layer. And once you have got this particular this output of the hidden neuron.



(Refer Slide Time: 15:46)

• **Step 4:** Calculation of the inputs of output layer


$$O_{IK} = w_{1K} H_{O1} + \dots + w_{jK} H_{Oj} + \dots + w_{NK} H_{ON},$$

where  $K = 1, 2, \dots, P$

• **Step 5:** Calculation of the outputs of Output layer

$$O_{OK} = \frac{e^{a_2 O_{IK}} - e^{-a_2 O_{IK}}}{e^{a_2 O_{IK}} + e^{-a_2 O_{IK}}},$$

Where  $a_2$  indicates the coefficient of transfer function



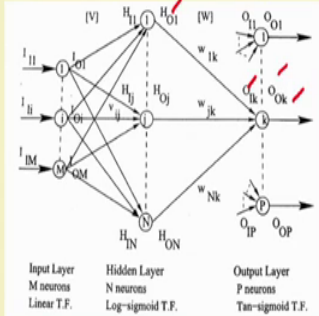
Now we are in a position to find out to calculate the input of the output layer, now this input of the output layer is nothing, but  $O_{IK}$ , and that is nothing, but  $w_{1K}$  multiplied by  $H_{O1}$  plus a few terms are here plus  $w_{jK}$  multiplied by  $H_{Oj}$  plus a few other terms are there, and the last term is  $w_{NK} H_{ON}$   $K$  varies from 1 2 up to  $P$ .

Now let us try to see from the diagram or or from the figure of this particular the network, now supposing that I am just going to calculate what should be this particular  $O_{IK}$ .

(Refer Slide Time: 16:30)


**Evolution of Neural Network Using a GA**

□ **Multi-layer feed-forward Neural Network for Modeling Input-Output Relationships**



Input Layer: M neurons, Linear T.F.  
 Hidden Layer: N neurons, Log-sigmoid T.F.  
 Output Layer: P neurons, Tan-sigmoid T.F.

*Handwritten notes:*  
 $O_{IK} = H_{O1} \times w_{1K} + \dots + H_{Oj} \times w_{jK} + \dots + H_{ON} \times w_{NK}$



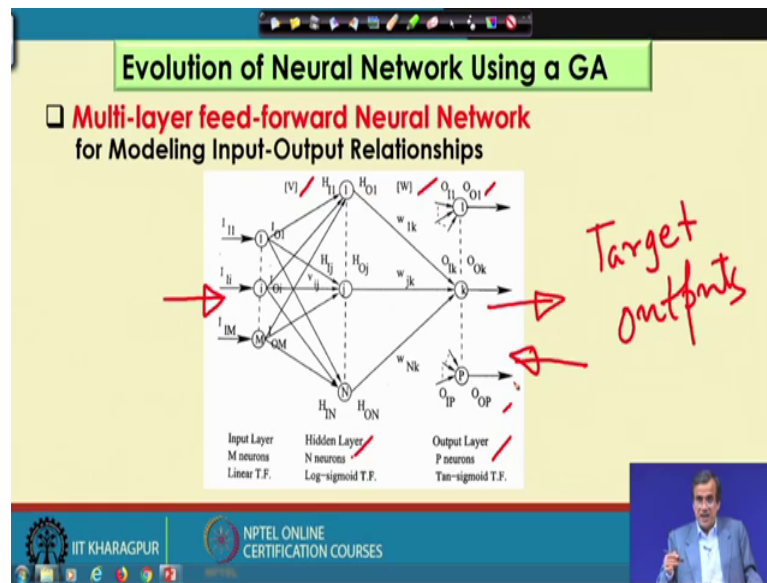
Now this  $O_k$  that is nothing, but the input of the  $k$ -th neuron lying on the output layer, and this is nothing but  $H_1 O_1 + H_2 O_2$  multiplied by  $W_{1k}$  plus we have got a few terms I am not writing, the next is  $H_3 O_3 + H_4 O_4$  multiplied by  $W_{2k}$  a few terms are there, and the last term will be your  $H_N O_N$  multiplied by  $W_{Nk}$ .

So this is nothing, but the input of the  $k$ -th neuron lying on the output layer, and now we have got the tan sigmoid transfer function, we know the mathematical expression for this tan sigmoid transfer function. So, using that tan sigmoid transfer function. So, I can find out what should be the output that is the output of the  $k$ -th neuron lying on this particular the output layer.

Now, the same thing whatever I discussed here. So, I have written it on the slide. So, let us try to see that yes step 5 calculation of the output the out of the output layer that is  $O_k$  is nothing, but  $e^{2 O_k} - e^{-2 O_k}$  divided by  $e^{2 O_k} + e^{-2 O_k}$ . Now here. So, this is nothing, but the input that is the input of the  $k$ -th neuron lying on the output layer, and this is the output, and this  $2$  is nothing, but the coefficient of the transfer function.

So, using this I can find out like what should be the output for a set of input, and I can find out all the outputs of the output neurons lying on the output layer, and once I have got this particular your this output, now we can compare. So, just to find out like whether we have we are able to make the prediction correctly or there is some error. Now till now whatever I have discussed. So, I have passed 1 set of inputs here.

(Refer Slide Time: 18:55)



And using these transfer function, then connecting weights and everything. So, here I am getting the set of outputs.

For example say I am getting O OK K varies from 1 to up to P. So, we have got P number of output neurons, and we have got some the target output. So, this particular the output the calculated output will be compared with the target outputs, and through this comparison, we will try to find out like what is the error or what is the deviation in prediction, and if I know this particular the deviation in prediction. So, that particular deviation prediction I will just pass it in the backward direction for the purpose of training of this particular network. So, that so, this particular network should be able to make the prediction ah more accurately.

Now, let us see using the principle of genetic algorithm. So, how we can evolve, the different parameters the design parameters of this particular network for example, the connecting weights V W the coefficient of transfer function like log sigmoid, and tan sigmoid transfer function those things could be the design variables, because the performance of this particular network depends on this particular the design variable.

So, we will have to find out the suitable values for this particular design variable, or optimal values of this particular design variable. So, that this particular network can make prediction more accurately; that means, for a set of inputs. So, it should be able to predict the set of outputs accurately.

Now let us see how can it predict, now I am just going to discuss.

(Refer Slide Time: 21:06)

• Step 6: Calculation of error in prediction

Error at K-th output neuron

$$E_k = \frac{1}{2} (T_{OK} - O_{OK})^2$$

Error in prediction considering all the output neurons

$$E = \sum_{K=1}^P \frac{1}{2} (T_{OK} - O_{OK})^2$$

P denotes the number of outputs of the network

*Handwritten notes:*  
Steepest Descent  
 $E_k = |T_{OK} - O_{OK}|$

So, that particular thing like how to predict, and how to determine this particular the different connecting weights the modified values of the connecting weights, and the modified values of the coefficient of transfer function, so, that this particular network can make the prediction more accurately.

Now, step 6 we will try to find out the error in prediction. Now here T is nothing, but the target output of the K-th neuron lying on the your the output layer. So, this is actually nothing but the target output, and this O O K is nothing, but the calculated output of the K-th neuron lying on this particular the output layer. So, this difference like T O K minus O O K is going to give some sort of deviation or the error. Now this particular error it could be either positive or negative depending on which 1 is larger and that is why to make it positive actually what we are doing, we are using this particular the square term.

So, we generally try to represent this particular error that is E K is nothing, but half multiplied by T O K minus O O K square. So, this is the way actually we represent this particular the error, we can also represent in a slightly different form for example, say we can represent like this like E K is nothing, but the mod value of the difference of T OK minus O OK, and this mod value will be positive, and that could be actually the error at K-th output neuron.

Now, once you have got the error in K-th output neuron. Now I can I can consider all the output neurons, just to find out what is the total error considering all the output neuron. So, this total error is denoted by E and that is nothing, but summation K equals to 1 to P. So, P is nothing, but the total number of neurons lying on the output layer, half T O K minus O O K square or P denotes the number of outputs of this particular the network.

So, using this actually I can find out like what should be this particular the total error in prediction, and once I have got this particular total error. Now I will have to minimize I can take the help of some sort of optimizer, now ah if you see the literature many people tried, in fact to reduce this particular error, using the principle of the back propagation algorithm which works based on the steepest descent algorithm.

Now this steepest descent algorithm its principle I have already discussed in much more detail. So, using the steepest descent method so, I can minimize. So, this particular error, and I can find out what should be the updated value for the connecting weights the coefficient of transfer function and so, on.

Now, this is 1 way of doing now I am not going for that the reason is actually is very simple, and these I have already discussed, because for the steepest descent method as it works based on the concept of gradient there is a chance of local minima. So, there is a possibility that this particular optimal solution is going to hit that local minima, and we may not get a globally optimal or the globally minimum network which will be able to predict so, this particular the outputs for a set of inputs very accurately.

Now just to overcome that particular problem. So, what we are going to do is we are going to use a genetic algorithm, and using this particular genetic algorithm let us see like how to how to evolve a network. So, that this particular network will be able to make the prediction accurately or a set of inputs the output responses, now before I proceed further let me let me discuss 1 more thing and that is as follows like.

(Refer Slide Time: 25:43)

• Step 6: Calculation of error in prediction

Error at K-th output neuron

$$E_k = \frac{1}{2} (T_{ok} - O_{ok})^2$$

Error in prediction considering all the output neurons

$$E = \sum_{k=1}^P \frac{1}{2} (T_{ok} - O_{ok})^2$$

P denotes the number of outputs of the network

*[Handwritten notes in red:  $[V], [W], b=0.001, a_1, a_2$ ]*

Supposing that say I am getting some error here on the output layer, so I am getting, so some error. And let us try to investigate why do you get this type of error who are responsible for this particular the error. Now if you see the design variables the design variables are nothing, but the connecting weights that is V between the input and the hidden layer, the connecting weights W that is between the hidden and the output layer. the coefficient of transfer function that is a 1 and a 2 for the hidden layer, and this is for the output layer, and sometimes you use some set of bias value, and here for simplicity I did consider the bias bias is actually a small value might be 0.001 something like this ok.

So, this is a fixed value. So, the performance of the network depends on all such parameters. So, these are all design variables, and if I want to get the optimal network corresponding to which I will be getting the minimum error in prediction. So, definitely I will have to use some the principle of optimization to to derive this particular network, or to evolve this particular the network.

now here so, this is actually your so, this is the function of so, many variables, now for simplicity let me assume that that this is a function of only 2 variables say V and W. So, let me assume that it is a function of V and W, and if it is a function of V and W. So, this is a function of 2 variable. So, very easily I can plot the error function, the error surface I can plot. So, if this is your V matrix, and this is the W, and if I plot the error here that is a. So, there is a possibility that I will be getting this type of error surface in 3 d.

So, our aim is to reach a particular solution, which corresponds to the minimum error in prediction. So, this is actually the 3 d of this particular errors surface. So, there is a possibility that initially I will start from here, or I will start from here, then gradually this particular algorithm is going to reach this particular optimal solution that is the solution corresponds to the minimum error in prediction.

Now let us see how this particular task can be implemented using the principle of genetic algorithm so, that the genetic algorithm can evolve, so this type of suitable network which will be able to predict the outputs for the set of inputs ah very accurately.

Thank you.