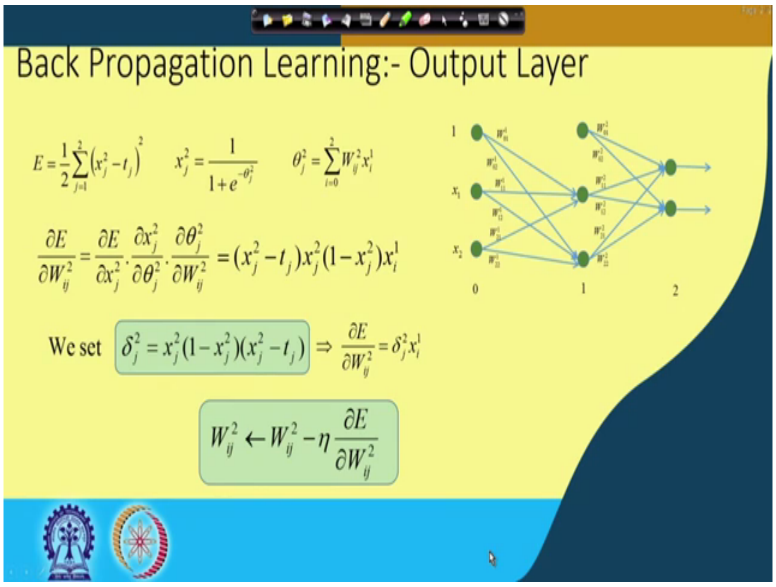
Deep Learning

Prof. Prabir Kumar Biswas
Department of Electronics and Electrical Communication Engineering
Indian Institute of Technology, Kharagpur

Lecture – 26
Backpropagation Learning - Example II

Hello welcome to the NPTEL online certification course on Deep Learning. You remember in the previous class we were discussing about the back propagation learning with respect to some examples. So, what I wanted to do is, I wanted to explain how the back propagation learning algorithm works with the help of example to have clear understanding of the back progression learning.

(Refer Slide Time: 01:05)



Back Propagation Learning:- Output Layer

$$E = \frac{1}{2}\sum_{j=1}^{2}\left(x_j^2 - t_j\right)^2 \qquad x_j^2 = \frac{1}{1+e^{-\theta_j^2}} \qquad \theta_j^2 = \sum_{i=0}^{2} W_{ij}^2 x_i^1$$

$$\frac{\partial E}{\partial W_{ij}^2} = \frac{\partial E}{\partial x_j^2}\cdot\frac{\partial x_j^2}{\partial \theta_j^2}\cdot\frac{\partial \theta_j^2}{\partial W_{ij}^2} = (x_j^2 - t_j)x_j^2(1-x_j^2)x_i^1$$

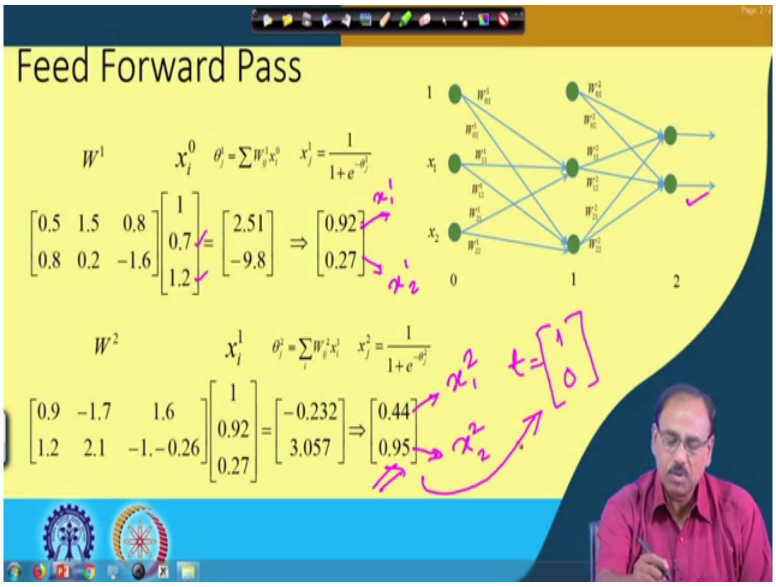
We set $\boxed{\delta_j^2 = x_j^2(1-x_j^2)(x_j^2 - t_j)} \Rightarrow \frac{\partial E}{\partial W_{ij}^2} = \delta_j^2 x_i^1$

$$\boxed{W_{ij}^2 \leftarrow W_{ij}^2 - \eta \frac{\partial E}{\partial W_{ij}^2}}$$

And the point till which we completed was this that when you go for back propagation of the error at the output layer; that means, now I want to update the weight victors connecting the nodes from the hidden layer to the output layer. The example that we are considering is 2 layer neural network or 2 layer feed forward neural network. So, when you go for updation of the connection weights as the error can be computed only at the output. I cannot compute an error at the output of the any of the hidden layer nodes, because I do not know what are the targets or what are the target vectors at the outputs of the hidden layer nodes. But given the class belongingness of the training vector I have, what is or I know what should be my target output. So, any deviation of the output that you get from the network from the target output that is my error.

So, the back propagation learning algorithm tries to adjust the weight vectors, so that the error is deduced or the error is minimized and for that you start from the output layer node. So, in the previous class what you have discussed is that, how do you adjust the weights at the output layer nodes using the error information or by back propagating the derivative of the error information with respect to the weight vectors or the weight components from the hidden layer node to the output layer node. So, for that we had defined this various terms as is shown in this particular diagram.

(Refer Slide Time: 02:58)



Feed Forward Pass

$W^1 \quad x_i^0 \quad \theta_j^1 = \sum W_{ij}^1 x_i^0 \quad x_j^1 = \frac{1}{1+e^{-\theta_j^1}}$

$$\begin{bmatrix} 0.5 & 1.5 & 0.8 \\ 0.8 & 0.2 & -1.6 \end{bmatrix} \begin{bmatrix} 1 \\ 0.7 \\ 1.2 \end{bmatrix} = \begin{bmatrix} 2.51 \\ -9.8 \end{bmatrix} \Rightarrow \begin{bmatrix} 0.92 \\ 0.27 \end{bmatrix} \begin{matrix} \rightarrow x_1^1 \\ \rightarrow x_2^1 \end{matrix}$$

$W^2 \quad x_i^1 \quad \theta_j^2 = \sum_i W_{ij}^2 x_i^1 \quad x_j^2 = \frac{1}{1+e^{-\theta_j^2}}$

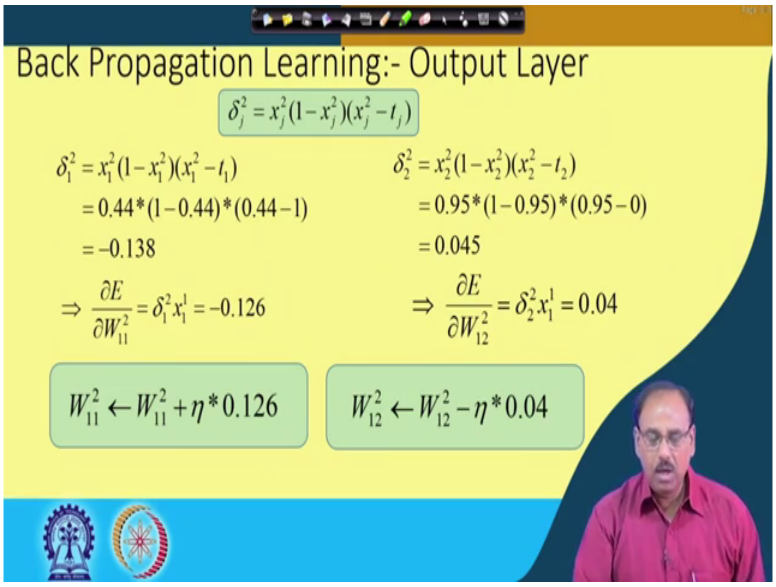
$$\begin{bmatrix} 0.9 & -1.7 & 1.6 \\ 1.2 & 2.1 & -1.-0.26 \end{bmatrix} \begin{bmatrix} 1 \\ 0.92 \\ 0.27 \end{bmatrix} = \begin{bmatrix} -0.232 \\ 3.057 \end{bmatrix} \Rightarrow \begin{bmatrix} 0.44 \\ 0.95 \end{bmatrix} \begin{matrix} \rightarrow x_1^2 \\ \rightarrow x_2^2 \end{matrix} \quad t = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

Now you also remember that in the forward pass we have computed the outputs of every node in the hidden layer, we have also computed the output of every node in the output layer in fact, that is from there only, I know what is my error. So, the outputs of the nodes in the hidden layer are as given here that x 1 1 is 0.92. So, this is x 1 1 which is the output of the hidden layer node and similarly x 2 1 which is output of the second node in the hidden layer is 0.27.

And I said that I cannot compute the error in the hidden layer, because though I know from the forward pass that what are my outputs, but I do not know what are the targets right, so I cannot compute error in the hidden layer. Another same manner I have computed output of the output layer nodes so, this is my x 1 2 that is the output of the first node in the output layer which is 0.44 and 0.95 is x 2 2 which is the output of the second node in the output layer. And we also said that the vector 0.7 1.2 that we considered this vector is taken from category one. So, as a result my target is nothing, but 1 0. This is my target vector, but this is the actual vector that I got.

So, the difference between the target vector and the actual vector that gives me an error and by using the backward learning mechanism, we have to reduce or minimize this error. That means, this output should be as close as possible to the target output and that is what we are doing using the backward learning algorithm. So, now, let us see so, as we have seen before that how this backward learning algorithms. So, I need to com compute these terms del j 2 that is for j equal to 1 and 2 so, I have to compute del 1 2 x 1 1. So, for various combinations of j and I have to compute this derivative of E with respect to the connection weights, and for this I need the values that we have computed in the forward pass.

(Refer Slide Time: 05:55)



Back Propagation Learning:- Output Layer

$$\boxed{\delta_j^2 = x_j^2(1-x_j^2)(x_j^2-t_j)}$$

$$\delta_1^2 = x_1^2(1-x_1^2)(x_1^2-t_1)$$
$$= 0.44*(1-0.44)*(0.44-1)$$
$$= -0.138$$
$$\Rightarrow \frac{\partial E}{\partial W_{11}^2} = \delta_1^2 x_1^1 = -0.126$$

$$\delta_2^2 = x_2^2(1-x_2^2)(x_2^2-t_2)$$
$$= 0.95*(1-0.95)*(0.95-0)$$
$$= 0.045$$
$$\Rightarrow \frac{\partial E}{\partial W_{12}^2} = \delta_2^2 x_1^1 = 0.04$$

$$\boxed{W_{11}^2 \leftarrow W_{11}^2 + \eta * 0.126}$$

$$\boxed{W_{12}^2 \leftarrow W_{12}^2 - \eta * 0.04}$$

So, let us see how it is done. So, I define del j 2 as it is already defined which is x j 2 into 1 minus x j 2 into x j 2 minus t j. So, you remember x j 2 minus t j is the error of error component the j th error component of the output error vector. So, for different values of j I compute delta 1 2; so, delta 1 2 is nothing, but x 1 2 into 1 minus x 1 2 into x 1 2 minus t 1 and as we have just seen that what is my x 1 2, x 1 2 is 0.44 as is obvious from here, x 1 2 is 0.44 and x 2 2 is 0.95.

So, by using this I go for this computation that delta 1 2 is equal x 1 2 into 1 minus x 1 2 into x 1 2 minus t 1, x 1 2 is 0.44 and t 1 is 1, because the input vector that I am considering I am considering that to belong to plus 1, so t 1 equal to 1. So, as a result when you compute this delta 1 2 it simply becomes minus 1.38, just this computation will give you and in the same manner when I compute delta 2 2 which is x 2 2 into 1 minus x 2 2 into x 2 minus t 2.

So, remember that my x 2 2 was 0.95 and t 2 is equal to 0 because the input vector belongs to plus. 1; so, my t 1 is 1 in the target vector and t 2 is 0 in the target vector. So, accordingly as x 2 2 is equal to 0.95 when I compute delta 2 2 delta 2 2 comes out to be 0.045. So, simply by gradient descent algorithm the updated delta 1 1 2 is nothing, but delta 1 1 2 minus the updated W 1 23 is nothing, but W 1 12 minus some eta times del E del W 1 12 and as I have computed delta 1 2 as minus 0.138. So, del E del W 1 12 it is simply delta 1 2 into x 1 1, that is the output of the first node in the hidden layer that is the x 1 1.

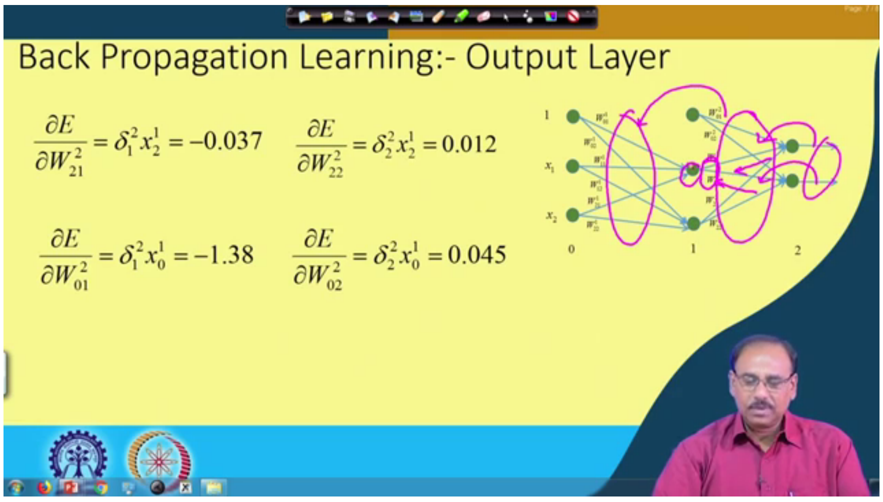
So, del E del W 1 12 which is delta 1 2 into x 1 1 which is nothing, but minus 0.126. So, what is x 1 1 again x 1 1 we have computed in the forward pass as you see from here that x 1 1 is nothing, but this value 0.92 and x 2 1 is 0.27. So, using these values I am computing these quantities. So, this del E del W 1 1 2 is delta 1 2 delta 1 2 is minus 0.138 multiplied by x 1 1 that gives you minus 0.126, and in the same manner del E del W 1 22 which is delta 2 2 times x 1 1 which becomes 0.04.

So, as a result the updated weights that is W 1 12 which is the connection weight from the first node in the hidden layer to the first node in the output layer that is simply becomes W 1 12 plus eta which is a hyper parameter indicating the learning rate eta times 0.126. You find that del E del W 1 1 2 was negative, my learning algorithm the or uptation rule is actually W 1 12 minus eta times del E del W 1 12 as W 1 12 is negative. So, this del E del W 1 12 absolute value of that is being added to W 1 12. Similarly, W 1 22 which is the connection weight from the first node in the hidden layer to the second load of the output layer that becomes W 1 22 minus eta times 0.04.

Now you note an interesting point over here, that is after updation W 1 1 which is the connection weight from first node in the hidden layer to the first node in the output layer that is being increased whereas, the connection weight from the first node in the hidden layer to the second node in the output layer is being decreased. And you remember that my target vector was 1 0 whereas, the output that actually I have obtained after this forward pass is 0.44 and 0.95. So, that clearly says that I should update the weights in such a way that the first component of the output vector should increase and the second component of the output vector should reduce. That means the output of the first node in the output layer should increase and the output of the second node in the output layer should reduce, and you see that what is this W 1 1 and W 1 2? W 1 1 is the connection weight from a hidden layer node to the first node in the output layer and W 1 2 is the connection weight from the same hidden layer node which is the first node in the hidden layer to the second node in the output layer.

So, if I increase W 1 1 then effectively I am adding more values to the output of W 1 to the output of the first node in the output layer and if I reduce W 1 2, as this is weighting at this is multi being multiplied with the output of the first node in the hidden layer. So, the effective contribution as the weight is reduced effective contribution to the output of the second node of the output layer that is been reduced. So, effectively what we are trying to do is we are trying to increase by doing this we are trying to increase this value; this is we are trying to increase and this value we are trying to reduce. That means, we are trying to move more towards our target vectors. So, this is how it is done in the output layer.

(Refer Slide Time: 13:55)



Back Propagation Learning:- Output Layer

$$\frac{\partial E}{\partial W_{21}^2} = \delta_1^2 x_2^1 = -0.037 \qquad \frac{\partial E}{\partial W_{22}^2} = \delta_2^2 x_2^1 = 0.012$$

$$\frac{\partial E}{\partial W_{01}^2} = \delta_1^2 x_0^1 = -1.38 \qquad \frac{\partial E}{\partial W_{02}^2} = \delta_2^2 x_0^1 = 0.045$$

Similarly, in the same manner I can compute what is W 2 1 or the updation of W 2 1; that means, the connection weight from the second node of the hidden layer to the first node in the output layer this is what is W 2 12 and also W 2 22 which is the connection weight from the second node in the hidden layer to the second node in the output layer. So, in the same manner by computing this delta 1 2 I can and delta 2 2 and making use of the outputs of the previous layer nodes or hidden layer nodes I can compute what is del E del W 2 12, I can also compute what is del E del W 2 22.

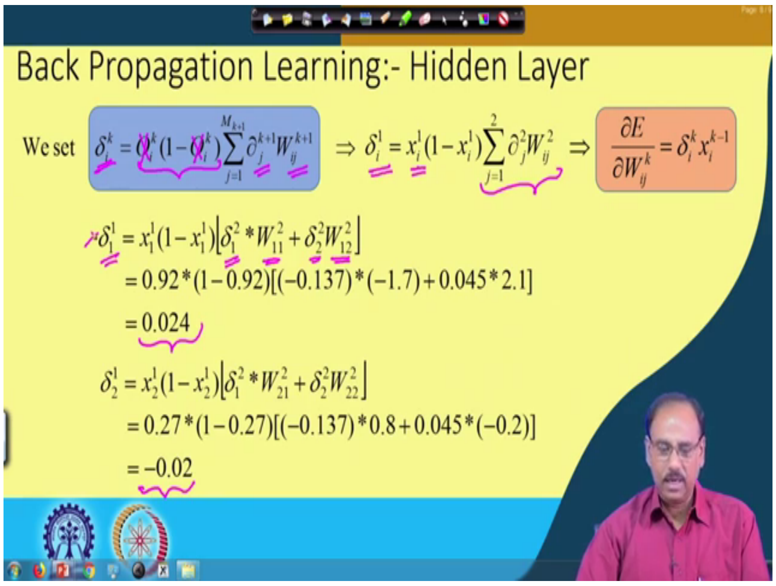
And here again you find that del E del W 2 12 comes out to be negative which is minus 0.037 and also del E del W 2 22 that is the connection weight from the second node in the hidden layer to the second load to the output layer that is also positive. So, the connection weight to the first node in the output layer is negative, connection weight to the second node of the output layer is positive. That means, when you go for weight updation W 2 12 will be incremented by a factor by a value which is eta times 0.037. This is the amount which will be added to W 2 12 and similarly when you update W 2 22 eta times 0.012 will be subtracted from W 2 22; that means, the value of W 2 22 will be reduced whereas, value of W 2 12 will be increased. So, this is a vector which will be increased whereas, this is a vector which will be reduced.

So, again as we have explained before that this increment of W 2 1 and reduction of W 2 2 tries to make the output of the first node higher and the output of the second node lower, and that is what we should do to reach our target vector which is 1 0. So, in the same manner the bias term bias is nothing, but W 0 12 which is the bias of the first node in the second layer or in output layer and W 0 22 this is the bias of the second node in the output layer.

So, in the same manner you find that del E del W 0 12 that is also negative that is minus 1.38 and del E del W 0 22 that is positive which is 0.045. So, all of these values will come based on the computations that you have done in the forward pass. So, this also says the bias term that is W 0 1 once it is updated it will be incremented by an amount eta times 1.38 whereas, W 0 22 will be decremented by amount eta times 0.45. So, this again tries to increase the output of the first node of the output layer and tries to reduce the output of the second node of the output layer because W 01 is being incremented by eta times 1.38, whereas W 02 is reduced by 0.045 ok.

So, this is how the nodes in the, or the weights in the output layer will be updated and while doing so we have propagated the effect of error computed at the output in the backward direction. Now this error has to be propagated further inside to update the weights from the input layer to the hidden layer, and when we you do that you find that coming to a particular node you are propagating the weight from the first node, you are also propagating the weight from the second node. So, these two propagating the error from the first node and propagating error from the second node; so, these two error terms are to be added together to consider what is the total back propagated error term at every node in the hidden layers.

(Refer Slide Time: 19:14)



Back Propagation Learning:- Hidden Layer

We set $\boxed{\delta_i^k = \cancel{O}_i^k(1-\cancel{O}_i^k)\sum_{j=1}^{M_{k+1}} \partial_j^{k+1} W_{ij}^{k+1}} \Rightarrow \delta_i^1 = x_i^1(1-x_i^1)\underbrace{\sum_{j=1}^{2} \partial_j^2 W_{ij}^2} \Rightarrow \boxed{\frac{\partial E}{\partial W_{ij}^k} = \delta_i^k x_i^{k-1}}$

$$\delta_1^1 = x_1^1(1-x_1^1)\left[\delta_1^2 * W_{11}^2 + \delta_2^2 W_{12}^2\right]$$
$$= 0.92*(1-0.92)[(-0.137)*(-1.7)+0.045*2.1]$$
$$= \underbrace{0.024}$$

$$\delta_2^1 = x_2^1(1-x_2^1)\left[\delta_1^2 * W_{21}^2 + \delta_2^2 W_{22}^2\right]$$
$$= 0.27*(1-0.27)[(-0.137)*0.8+0.045*(-0.2)]$$
$$= \underbrace{-0.02}$$

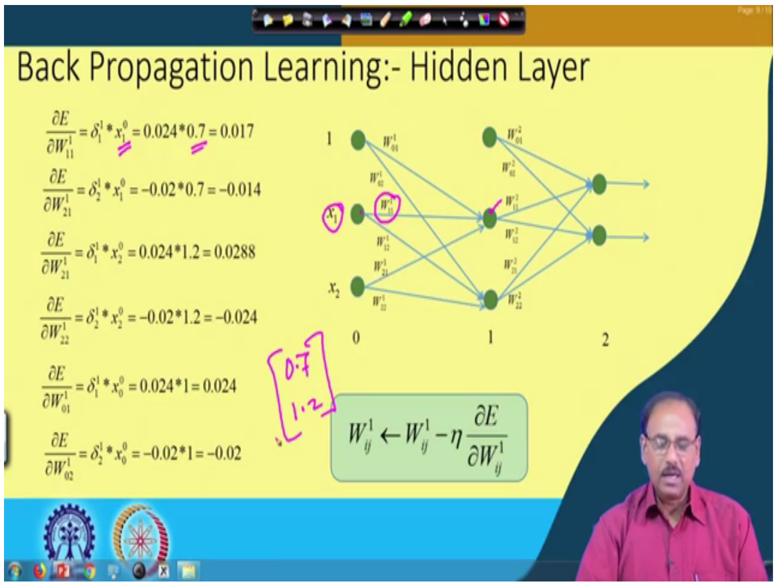
So, for that again we have seen before that how this is done. We had defined an additional term if you remember that when we discussed about the back propagation algorithm that for every k th layer node we have defined, and back propagated error term which is given by delta i k which was 0 i k sorry in this particular case our we are considering O i to be x i. So, please read this as O i has to be x i, O i k has to be x i k. So, please read this as x i k. So, my delta i k will be x i k into 1 minus x i k into sum of delta j k plus 1 W i k k plus 1.

So, you remember that this W i j k plus 1 is the connection weight from the i th node in the k th layer to the j th node in the k plus first layer, and this delta j k plus 1 is the error term which is propagated up to the j th node of the k plus first layer. So, this is being weighted by the corresponding weight and you are adding them together to find out what is the propagated error at the i th node and for doing this all the nodes to which the i th node has feed the input; that means, all the js from all of them. Now you are accumulating the error by weighting them by the corresponding connection weight and adding them together. And then multiplying that with a local derivative, local derivative of the output of the i th node which is x i k into 1 minus x i k right.

So once you have this, you define it with respect to k now this k in our case, because we are considering a 2 layer network the value of k is equal to 1. So, what we have to find out is, what is delta i 1 and as per this definition this delta i 1 is nothing, but x i 1 into 1 minus x i 1. So, this x i 1 is the output of the i th node in the hidden layer. So, this x i 1 into 1 minus x i 1 into some of all the back propagated errors from all the nodes to which the i th node has fed the input. So, that is what gives us delta i 1. So, given this and all the pre computed values all the computed values that we have computed during the forward pass I can find out what is delta 1 1, because I know what is the x 1 1 from the forward pass I know what is delta 1 12 I know what is delta 1 22. Because that is what we had initialized or any intermediate stage in any step of weight iteration these are the updated weights, those are also known. I also know what is delta 1 2 that is the back propagated error from the next node the nodes in the next layers.

Similarly, I also know what is delta 2 2 that is the; what is the back propagated error up to again a node in the next year. So, if you compute this you find that delta 1 11 delta 1 1 that comes out to be 0.24. And in the same manner you find that delta 2 1 that comes out to be minus 0.02, right. So, these are the back propagated errors to the first node in the hidden layer and to the second node in the hidden layer. So, this is the back propagated error to the first node in the hidden layer and this is the back propagated error to the second node in the hidden layer. So, using this now I can go for updation of the weight vectors from the input layer nodes to the hidden layer nodes.

(Refer Slide Time: 23:42)



Back Propagation Learning:- Hidden Layer

$$\frac{\partial E}{\partial W_{11}^1} = \delta_1^1 * x_1^0 = 0.024*0.7 = 0.017$$

$$\frac{\partial E}{\partial W_{21}^1} = \delta_2^1 * x_1^0 = -0.02*0.7 = -0.014$$

$$\frac{\partial E}{\partial W_{21}^1} = \delta_1^1 * x_2^0 = 0.024*1.2 = 0.0288$$

$$\frac{\partial E}{\partial W_{22}^1} = \delta_2^1 * x_2^0 = -0.02*1.2 = -0.024$$

$$\frac{\partial E}{\partial W_{01}^1} = \delta_1^1 * x_0^0 = 0.024*1 = 0.024$$

$$\frac{\partial E}{\partial W_{02}^1} = \delta_2^1 * x_0^0 = -0.02*1 = -0.02$$

$$\begin{bmatrix} 0.7 \\ 1.2 \end{bmatrix}$$

$$W_{ij}^1 \leftarrow W_{ij}^1 - \eta \frac{\partial E}{\partial W_{ij}^1}$$

So, how do I do it, again by using the same expressions your del E del W 1 11; what is del W 1 11? It is the connection weight from the first node, the connection weight from the first node in the input layer to the first node in the hidden layer. So, that is W 1 11; so, this is the one. So, I can compute del E del W 1 11 as delta 1 1 times x 1 0. What is the x 1 0? x 1 0 is supposed to be the output of the first node in the 0 th layer. And in our case in this case the 0 th layer is the input layer and what we said earlier is that empty node in the input layer gives you an identity function; that means, whatever is the input it simply passes that to the output. So, this x 1 0 is nothing but the first component of the input vector which is x 1, because that is the input vector the first component of the input vector.

So, this first component of the input vector if you remember, we considered that to be 0.7. So, my input vector was 0.7 and 1.2 that was the input vector. So, using this I can compute what is del E del W 1 11,I can compute what is del E del W 2 11 which comes out to be minus 0.014. I can compute what is del E, Del W 2 11 which comes out to be 0.288. I can compute del E del W 2 21 which comes out to be minus 0.024. I can compute what is del E? Del 0 11 now del E del W 0 11, W 0 11 if you remember it is representing the bias of the first node in the input layer, so which comes out to be again 0.24. I can also compute del E del W 0 21, W 0 21 is the bias to the second node. So, I can also compute del E del W 0 21 which comes out to be minus 0.2.

And once I compare all this my updation rule weight updation rule is as before del i j 1 gets W i j 1 simply gets W i j 1 minus eta times del E del W i j 1, right. So, you find that by forwarding the backward direction by back propagation of the error terms from the output layer, I can update the weight vectors even in between the hidden layers and this can proceed further. So, instead of two layer network if I take a three layer network again the error will be propagated from the second hidden layer to the first hidden layer and it can be used in a similar manner for multilayer networks.

So, here you find that what we have done, we have just illustrated with the help of examples that in the network layer. At the network level how the back propagation algorithm is can be implemented to update the weight vectors. And you find that when you use this back propagation algorithms the values that you are using at every layer are actually computed in the forward layer, actually computed in the forward pass. So, it will be wise if you save all the values that we have computed in the forward pass for reducing the computation during the back propagation learning work. And each of these works that you have done that independent every node is independent of the other nodes. So, it is highly parallelism. So, I will stop here today. We will continue with these discussions even at the node level in our next lectures.

Thank you.