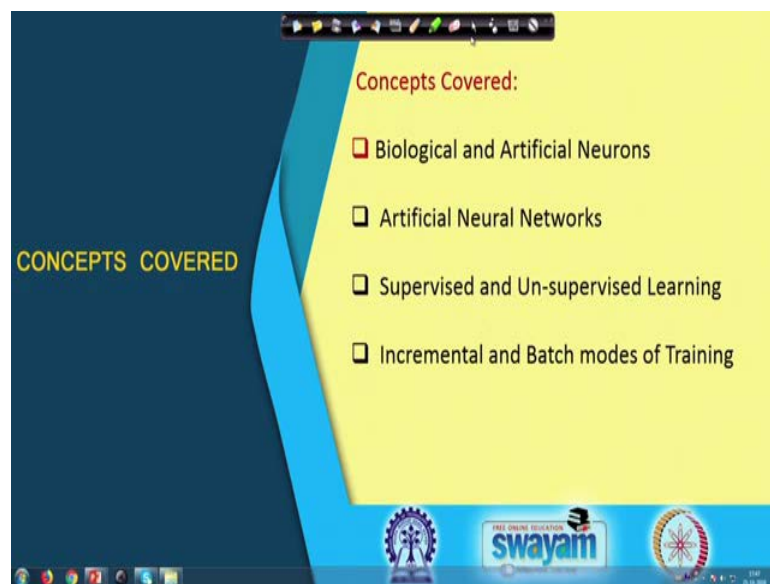


Fuzzy Logic and Neural Networks
Prof. Dilip Kumar Pratihari
Department of Mechanical Engineering
Indian Institute of Technology, Kharagpur

Lecture – 20
Introduction to Neural Networks

Now, we are going to start with another topic, that is, Introduction to Neural Networks. So, we are going to enter another region, that is artificial neural networks.

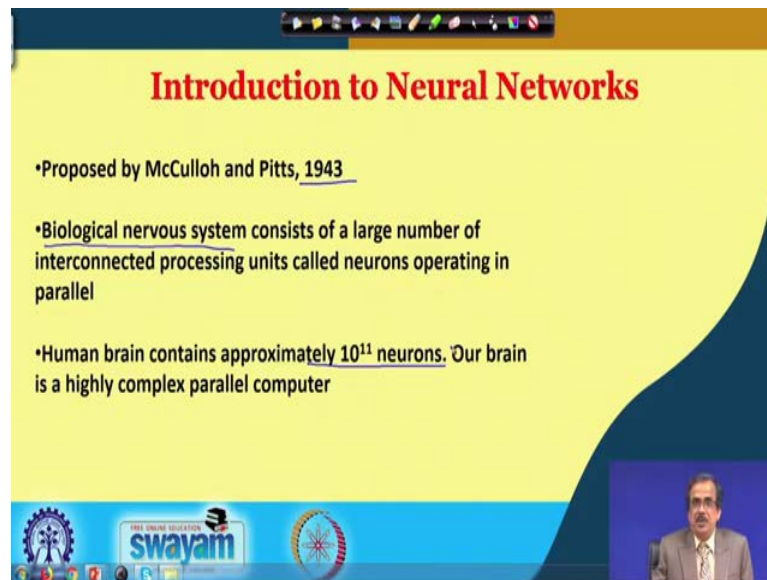
(Refer Slide Time: 00:31)



Now, the purpose of these artificial neural networks is how to model the human brain in the artificial way. Now, these are the topics actually which we are going to discuss. So, at first I will give a brief introduction to the working principle of a biological neuron and I will try to design an artificial neuron based on the working principle of the biological neuron. And, after that, we are going to discuss like how to design one artificial neural network, which consists of a large number of neurons.

Then, we will be discussing the principle of supervised and unsupervised learning and at the end, we will try to define like what do you mean by incremental and batch modes of training. Now, let us start with actually the biological neuron.

(Refer Slide Time: 01:33)



Introduction to Neural Networks

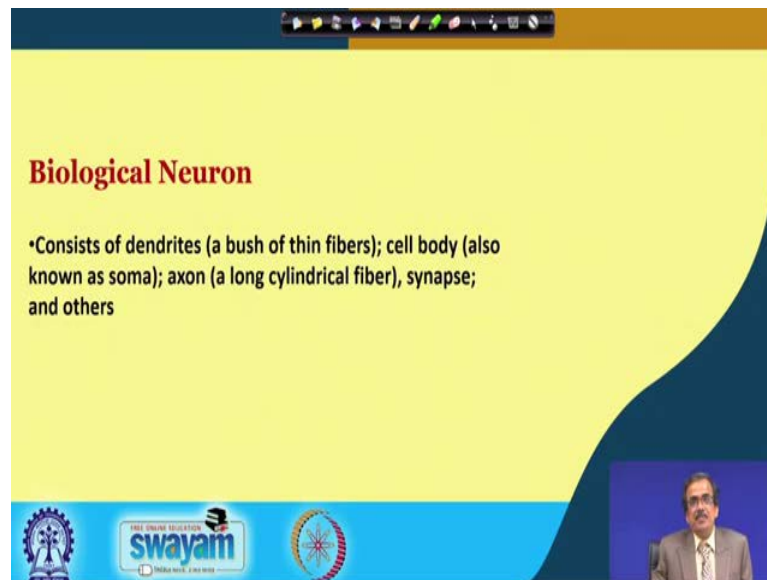
- Proposed by McCulloch and Pitts, 1943
- Biological nervous system consists of a large number of interconnected processing units called neurons operating in parallel
- Human brain contains approximately 10^{11} neurons. Our brain is a highly complex parallel computer

THE ONLINE EDUCATION swayam

Now, before that let me tell you that the concept of neural networks actually was proposed in the year 1943 by McCulloch and Pitts. And, here actually, what we do is, we try to copy everything from the biological neurons or biological nervous system. Now, if you see the biological nervous system, it consists of a large number of neurons and these neurons are working in parallel.

Now, the average human brain contains approximately 10^{11} neurons. And, of course, it varies from person to person and these neurons are working in parallel and that is why, our brain is a highly complex parallel computer.

(Refer Slide Time: 02:33)



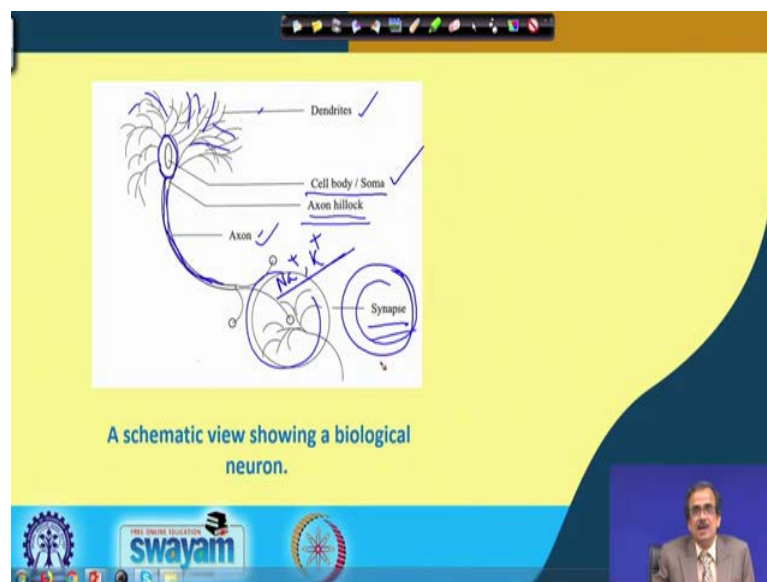
Biological Neuron

- Consists of dendrites (a bush of thin fibers); cell body (also known as soma); axon (a long cylindrical fiber), synapse; and others

The slide features a yellow background with a blue header and footer. The footer includes the Swayam logo and a small video inset of a man in a suit.

Now, let us see the working principle of a particular biological neuron. Now, if you see a biological neuron, it consists of, in fact, say a bush of thin fibers and those are known as dendrites.

(Refer Slide Time: 02:43)



A schematic view showing a biological neuron.

The diagram labels the following parts: Dendrites, Cell body / Soma, Axon hillock, Axon, and Synapse. Each label has a checkmark next to it. The diagram shows a neuron with a cell body, dendrites, and a long axon ending in a synapse. The Swayam logo and presenter video are at the bottom.

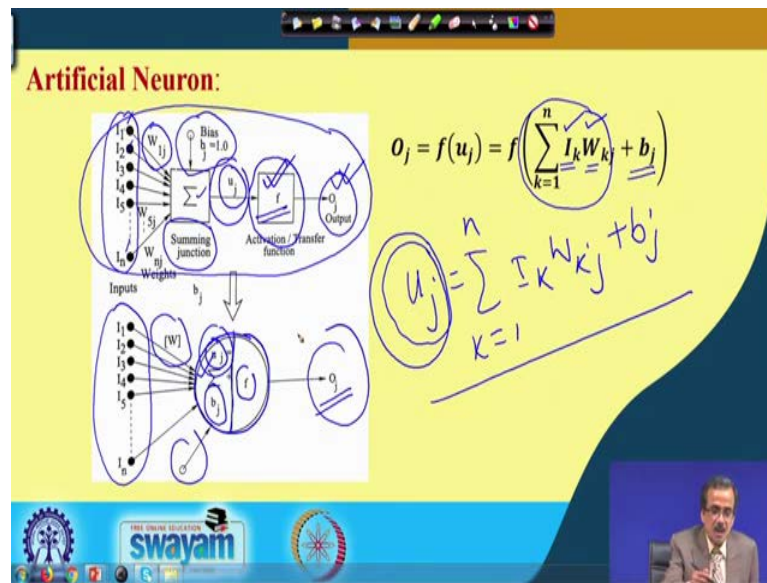
So, these are nothing but the dendrites and then, we have got a one long cylindrical fiber and that is called the axon. So, we have got this particular axon and we have got the cell body or the soma and of course, we have got this particular the synapse.

Now, let me explain the working of each of these particular components of the neurons for example, a particular neuron will try to collect information from the neighboring neuron, now this neighboring neuron could be here. Now, it will try to collect information from the neighboring neurons with the help of these thin fibers, those are nothing but the dendrites and it will collect all such information and in the cell body or the soma, the information will be collected and this particular collected information will pass through the axon and then, it will go to the junction between so this particular neuron and the next neuron and that particular junction is known as actually the synapse. So, a biological neuron consist of dendrites, it has got axon, it has got cell body or soma and it has got this particular the synapse and the junction between the axon and the cell body is known as the axon hillock.

Now, what happens? So, the information will be collected from the neighboring neurons with the help of dendrites and it will be collected here, then it will pass through the axon then, here actually in the synapse, there will be actually some sort of transfer of information. Now, in biological neuron, the transfer of information takes place through the difference in ion concentration, for example, there is a difference of sodium ion concentration, potassium ion concentration. And, due to this particular sodium ion and potassium ion concentration, some part of the information, the collected information will be passed to the next neuron through the synapse. So, this is the way actually, one biological neuron works.

Now, in an artificial neuron, the same working principle has been copied in the artificial way.

(Refer Slide Time: 05:21)



Now, let us see, how to copy this particular principle in the artificial way. Now, just like our biological neuron, it collects information with the help of dendrites. So, these are all collected information, for example, say I_1 , I_2 up to say I_n . So, these are all inputs, these inputs are nothing but the collected information and these inputs will be multiplied by the corresponding connecting weights, that is denoted by your the W . So, W is going to represent actually the connecting weights. So, what we do is, we multiply a particular input with its corresponding connecting weights.

So, we try to find out I_k multiplied by W_{kj} and we just sum them up. So, all such things are summed up. So, here this summed up value is going to enter and we add some bias value, that is your b_j just a small value. So, that particular bias value is going to be added here.

So, I will be getting this particular u_j and this $u_j = \sum_{k=1}^n I_k W_{kj} + b_j$. So, this is nothing but is your u_j , now if you remember this particular u_j , as if it is passing through the action of the biological neuron. So, this u_j , let me repeat is u_j is going to pass through the long cylindrical fiber that is nothing but the axon and then it will go to the synapse and in the synapse, there will be some transfer function. And, here actually this is going to represent the synapse of the biological neuron, and that is nothing but here, we have got one activation function or the transfer function.

So, this particular unit, the input, that is u_j is going to enter through the transfer function and consequently, I will be getting this particular output and this output is nothing but the amount of information which is going to enter the second neuron. So, this is the way actually, we passed the information in biological nervous system from one neuron to the next neuron and the same thing actually it has been copied here, in the artificial way, in the artificial neuron.

Now, this indicates actually the summing junction and this is almost similar to your cell body or soma of the biological neuron. Now, this particular thing, this artificial neural generally, we try to represent with the help of actually one circle. So, this circle is going to represent one artificial neuron and this circle has got two compartments, the first part, we have got u_j , that is nothing but we have got the summing junction and here, we are going to add this particular the bias value; that means, this is nothing but u_j . Now, this u_j is going to pass through the transfer function and the transfer function is here and it passes through the transfer function, I will be getting this particular output, that is denoted by O_j .

Now, actually to represent this artificial neuron generally we use this type of circle and there are some inputs coming. So, these are the all inputs, each of the inputs will be multiplied by its corresponding connecting weight, the bias value will be added, we will be getting u_j ; u_j will pass through the transfer function and consequently, I will be getting this particular the output. This is the way actually, we get the output for a particular neuron from this input.

Now, this is the way actually, one artificial neuron works and it has been copied from the biological neuron.

(Refer Slide Time: 09:45)

Types of transfer functions:

- Hard-limit /
- Linear /
- Log-sigmoid /
- Tan-sigmoid, and others /

The slide features a yellow background with a blue header and footer. The footer includes the Swayam logo and a small video inset of a man in a suit.

Now, if you see the literature, we use different types of transfer function. For example, say we use some sort of hard limit transfer function, we also use some sort of linear transfer function; we use sigmoid transfer function and tan sigmoid transfer function. Now, here actually, I am just going to explain each of this particular transfer function.

(Refer Slide Time: 10:11)

Transfer functions

Hard-limit TF:

$$O = \begin{cases} 0.0, & \text{if } u < 0.0 \\ 1.0, & \text{otherwise} \end{cases}$$

Handwritten notes: *perception neuron*, *0.0*, *1.0*

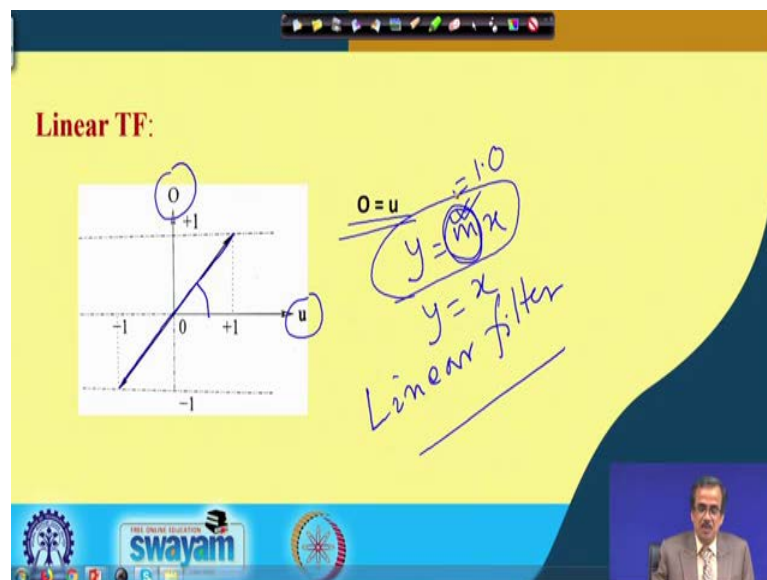
The slide features a yellow background with a blue header and footer. The footer includes the Swayam logo and a small video inset of a man in a suit.

So, this indicates actually, the hard limit transfer function. Now, let us try to understand, what do you mean by this hard limit transfer function.

Now, if you remember the input for the transfer function is nothing but u and output is nothing but O . Now, the output will become 0.0, if input u is found to be less than 0.0; that means, if it is negative, if the input u is negative, my output will become equal to 0, otherwise the output will be actually 1.0. So, for this type of transfer function, there are two outputs, it is either 0 or it is 1.0.

So, this type of transfer function is known as the hard limit transfer function and this type of transfer function, we use, in fact, in a special type of neuron, that is called the perceptron neuron. So, we use this type of hard limit transfer function in perceptron neuron. So, this is the way actually, this hard limit transfer function works.

(Refer Slide Time: 11:33)

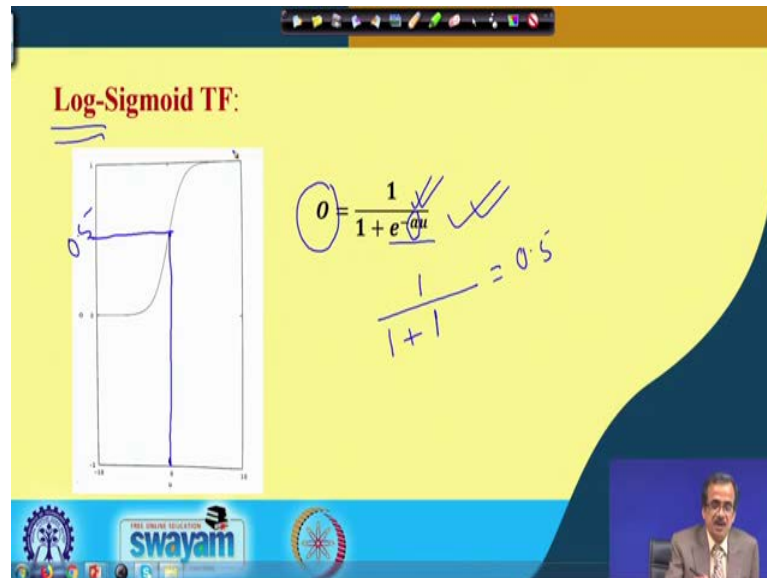


The next is the linear transfer function now, here you can see once again. So, u indicates the input, O is nothing but the output and this is actually the transfer function. So, $y = mx$. So, y equals to say mx state line and here let me consider m is equals to say 1.0. So, if y equals to x , if m equals to 1.0. So, this becomes y equals to x ; that means, this particular angle is your 45 degrees. That means, output is nothing but the input. So, $O = u$. So, y equals to x .

So, the output is same as actually the input and this type of transfer function is used in linear filter. So, we use this type of transfer function in linear filter and if I use $y = mx$, I can also find out what should be the optimal value for this particular m . So, m could be 1, it could be less than 1. So, we can find out what should be the value for this m , during

optimization, we can also find out what should be the most appropriate value for this particular m.

(Refer Slide Time: 12:53)

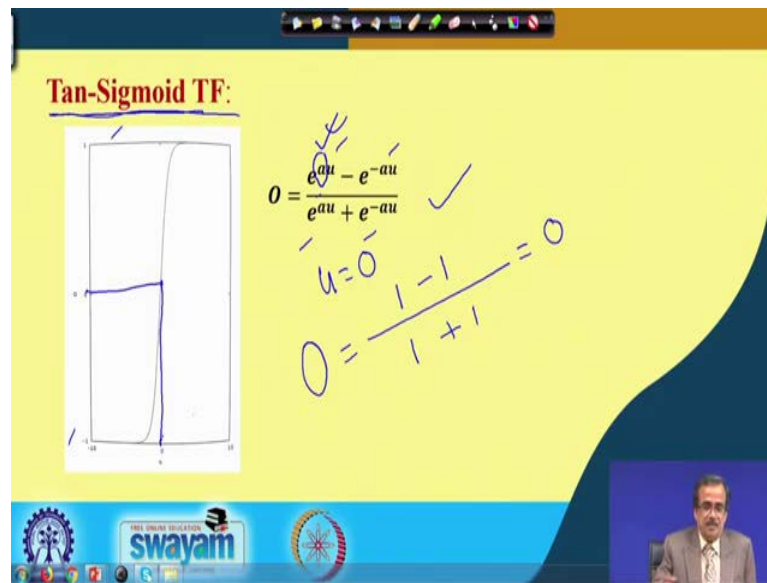


Now, then comes your the log sigmoid transfer function. So, this is the mathematical expression for the log sigmoid transfer function.

Now, here, $O = \frac{1}{1 + e^{-au}}$, now a is actually the coefficient, which decides what should be the slope of this particular curve. Now, if I take the higher value of a, this particular curve will be very steep, and if I take low value, I will be getting some sort of flatter distribution for this particular log sigmoid transfer function. Now, let us see what happens, if I consider u is equal to 0.

Now, if I put u equals to 0. So, this will become 1 divided by 1 plus e raise to the power 0 and that is nothing but 1. So, this is 1 divided by 2 that is 0.5. So, corresponding to u equals to 0. So, I will be getting this is 0.5 and here, this value of your output, it varies from 0 to 1 because this is log sigmoid, so it cannot be negative. So, it varies from 0 to 1 and this is actually a non-linear distribution for this transfer function and this is very frequently used in artificial neural networks.

(Refer Slide Time: 14:33)

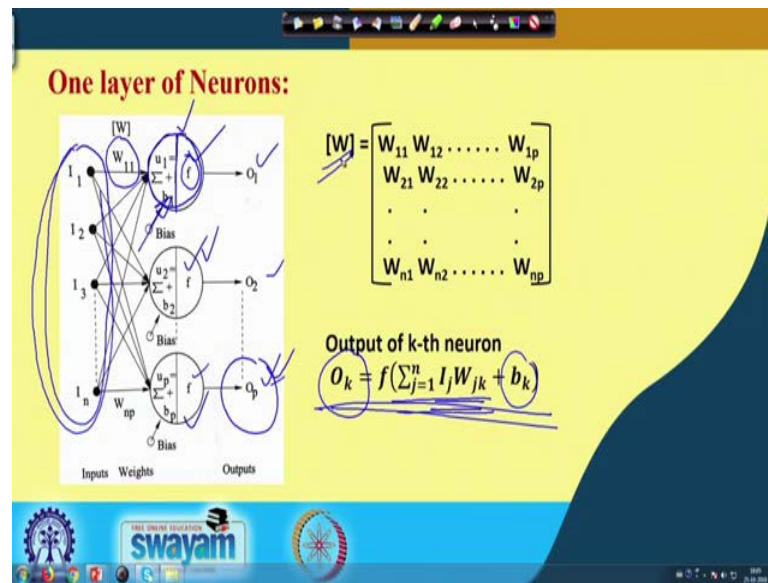


Now, then comes here the tan sigmoid transfer function. Now, for this particular tan sigmoid transfer function, this is the mathematical expression, that is $O = \frac{e^{au} - e^{-au}}{e^{au} + e^{-au}}$.

And, once again, a is going to decide what should be the slope of this particular distribution. So, the higher the value of a the steeper will be this particular curve and vice-versa, and supposing that if I put say u is equals to 0 if, I put u equals to 0. So, the output O will become what? This will become your e raise to the power 0 is 1 minus e raise to the power 0 that will become 1 and here I will be getting 1 plus say 1 and this will become equal to 0.

So, corresponding to this particular u equals to 0. So, there is a possibility I will be getting the output is equals 0 and here, the output actually will vary between minus 1 to plus 1 and this is actually your the tan-sigmoid. So, output varies from -1 to + 1. So, these are the actually some of the very popular transfer functions, which are generally used in neural networks.

(Refer Slide Time: 16:07)



And, now I am just going to consider say one layer of neurons. Now, a particular layer of neurons will consist of a number of neurons for example, say. So, this particular layer is considered, so this is consisting of say 1, 2 and. So, I have got say p number of neurons and if I concentrate on each of the neuron you can see that it has got two compartments, one is the summing junction, so this is the summing junction and this is nothing but the transfer function.

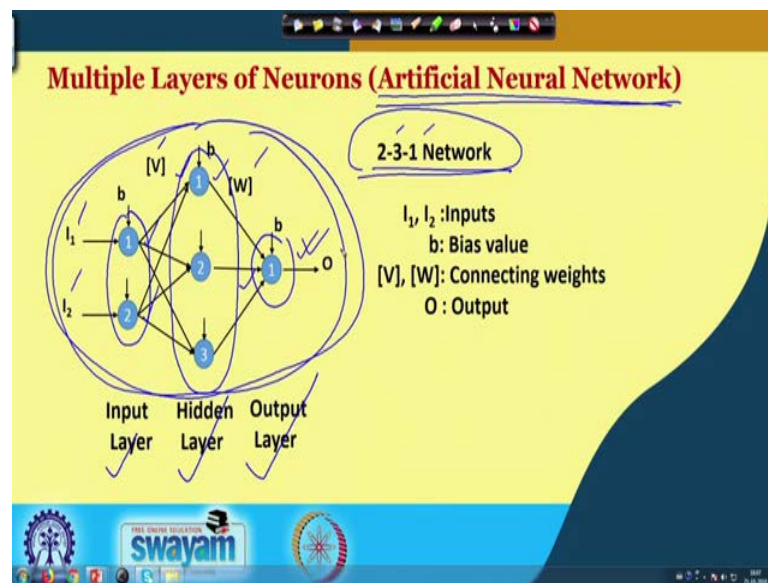
Now, what you do is. So, I have got a large number of inputs say n number of inputs I_1, I_2 up to I_n . So, what I do is, these particular inputs will be multiplied by the connecting weight, the corresponding connecting weights, that is, W and those things will be summed up here. Now, if you just sum them here, this is actually what is happening. So, I_j multiplied by W_{jk} . So, this particular input will be multiplied by the corresponding connecting weights and those things will be summed up here. And, I am just going to add the bias value say this particular b , then it will pass through the transfer function. So, this particular transfer function and I will be getting the output.

Now, it shows what happens for a particular neuron, that is the k -th neuron, for example, this is the p -th neuron. So, I will be getting by following this particular principle, what should be the output of the p -th neuron. So, I have got the set of inputs say n inputs, I know the connecting weights, I know the transfer function, I know the bias values. So, I will be able to find out what should be the output of a particular neuron, for example, this

is the output of the p -th neuron and in one layer, you have got a large number of neurons for example, say 10 neurons or say 20 neurons, and so on.

So, we should be able to find out, what should be the output for each of the neurons lying in this particular layer. And, as I mentioned like this, W indicates actually the connecting weights and the values of the connecting weights actually will vary from say - 1 to + 1 in the normalized scale. Now, this shows actually one layer of neurons.

(Refer Slide Time: 18:47)



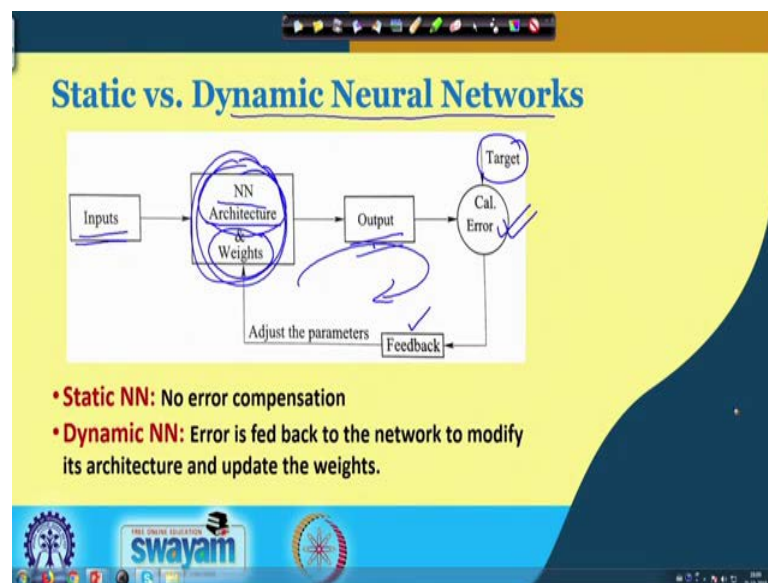
And, now I am just going to show you a few layers of neurons and that is nothing but an artificial neural network, which consists of a number of layers, for example, say here for this particular simple artificial neural network, which have been considered. So, it has got three layers one is called the input layer, we have got the hidden layer and we have got this particular output layer.

Now, for simplicity, we have considered only 2 neurons in the input layer and we have considered 3 neurons in the hidden layer and we have considered only 1 neuron in the output layer. And, that is why, this is nothing but a 2-3-1 network. That means, there are 2 neurons in the input layer, 3 neurons on the hidden layer and 1 neuron on the output layer. So, this is also known as a 2-3-1 network and this is nothing but actually one artificial neural network.

Now, as I told, there are two inputs I_1 and I_2 , V and W indicate the connecting weights. So, V indicates the connecting weights between the input and hidden layers and W matrix indicates the connecting weights between the hidden and the output layer. And, at each of the neurons, we have got the transfer function, ok. So, I will be able to find out what should be the output here, I should also be able to calculate, what should be the input here, I will be able to calculate the output here, then the input here and I can also find out what should be the final output of this particular network.

All such things will be discussing in much more details. Now, here, what I want to say, one artificial neural network consists of a number of layers and each layer contains a number of neurons, and there will be some sort of connectivity and that is why, starting from the input, I will be getting finally, some output here. So, this is actually the way, one artificial neural network looks like.

(Refer Slide Time: 21:09)



Now, I am just going to explain the concept of static versus the dynamic neural networks. Now, if I see the static network means there is no error compensation, there is no chance of feedback, and if I consider the dynamic network there will be a chance of error feedback and there will be a chance of further improvement.

Now, this schematic view shows one dynamic network, now, the performance of a particular neural network depends on the architecture of that or the topology of this particular network. That means, what is the number of layers, how many neurons are

going to be present at each of the layers that indicates actually the architecture or topology of this particular network. And, the performance of this particular network of course, depends on the connecting weights values and moreover, it depends on the coefficient of transfer functions, the different transfer functions used in different layers.

So, the performance depends on the architecture, weights, coefficients of transfer function and all such things, the movement we pass one set of inputs. So, I will be getting some output here, now this particular output will be compared with the target output just to find out what should be this particular error. Now, this error is feedback for the adjustment of these particular parameters and this process will go on and go on. And, through a number of iterations, there is a possibility that will be getting some artificial neural network, which will be able to make the prediction for a set of inputs very accurately. Now, this is the way actually, we can develop the dynamic neural networks.

Now, these things actually we will be discussing in much more details in future.

(Refer Slide Time: 23:23)

Training of Neural Networks

- **Supervised Learning / Learning with Teacher**
The outputs of the network are compared with the corresponding target values and the error is calculated. It is then fed back to the network for updating of the same.
- **Un-Supervised Learning / Learning without Teacher**
Competition, cooperation and updating

swayam

So, let me discuss a little bit, how to design and develop this particular dynamic network. Now, if you want to design and develop the dynamic network. So, what will have to do is we will have to train the network. So, we will have to use some optimizer or we will have to use some sort of learning tool. So that we can design and develop that particular the neural network so that it can predict the output for a set of inputs very accurately.

Now, if you see the literature. In fact, we have got two types of learning, now one is called actually the supervised learning and another is called, in fact, your unsupervised learning. Now, if you see the supervised learning and the unsupervised learning. So, we will just try to find out the difference between the supervised learning and the unsupervised learning; now the supervised learning is also known as learning with a teacher.

Now, if the students make mistake the teachers are there to make them correct and actually, there will be some sort of a feedback and there will be some sort of error compensation. And, due to this error compensation or the feedback, there will be actually some sort of supervised learning.

Now, actually, what you do is, in supervised learning, for a set of inputs, we calculate the output, now these calculated output is compared with the target output to find out what should be the error. Now, this particular error is feedback for actually the adjustment of the parameters of the network, so that this particular network can predict as accurately as possible. So, in supervised learning actually, we have got the provision for error compensation, we have got the provision for the feedback and that is why, this particular network, which will be trained using the supervised learning will become more efficient and more accurate, I should say. On the other hand, we have got some sort of unsupervised learning. Now, for supervised learning, what I need is, some well defined training scenarios, some known input-output relationships. Now, if you have got some pre-collected input-output relationships, we can carry out supervised learning with the help of that particular the training data, but supposing that we do not have the training data, now if you do not have the training data, then how to find out.

So, this type of the network, which will be able to predict as accurately as possible; so we are in trouble, now to solve that particular problem actually, what we do is, we use the principle of your un-supervised learning and that is nothing but the learning without a teacher. Now, here, actually we use the principle of competition and through the competition one winner will be selected or declared, there will be cooperation with the between the winner and the surroundings. And, there will be further updating and through this principle of the competition, cooperation and updating, in fact, we can implement the unsupervised learning or unsupervised training to the network.

Now, this particular things, in fact, both supervised as well as unsupervised learning will be discussed in much more details later on, in this particular the course; now let me concentrate for the time being a little bit on supervised learning.

(Refer Slide Time: 27:37)

Incremental vs. Batch Modes of Training

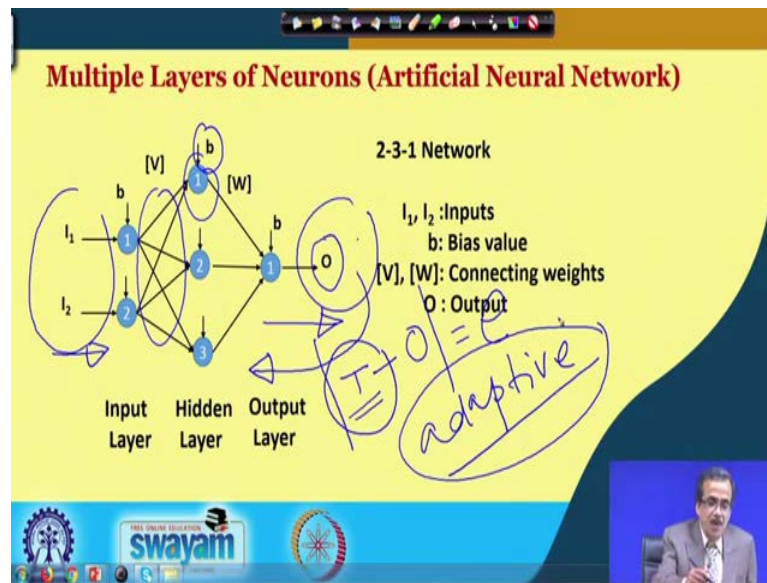
- **Incremental Training / On-line Training:**
A particular training scenario is passed through the network, output(s) is/are calculated and then error is determined by comparing it/them with the target(s). The error is propagated to modify the network

The slide features a yellow background with a blue wave graphic on the right side. At the bottom, there is a blue banner with the Swayam logo and a small video inset of a man in a suit.

So, if you see the literature. So, we have got two types of training used in supervised learning. Now, one is called the incremental training and another is called the batch mode of training. Now, these incremental training is also known as the online training.

Now, let us try to understand the principle of your incremental training or the online training. Now, to understand this, let me go back a few slides back on, let me concentrate on this particular slide a little bit.

(Refer Slide Time: 28:17)



Now, here actually what you do is, during the incremental training, what I am doing, I am passing say one set of inputs, now depending on the connecting weights the coefficient of the transfer function or the nature of the transfer function, I will be getting some output here. So, output will be calculated. Now, this particular calculated output, we will be comparing with the target output denoted by T and we will try to find out the error. So, by comparing the target and this particular calculated output, we try to find out this particular error.

Now, based on this particular error, what you do is, we try to go for the feedback circuit; that means, this error will be propagated back, so that we can modify the values of the connecting weights, the coefficients of the transfer function, the bias values and all such things. So, I pass one training scenario calculate the output, determine the error and this particular error will be feedback for the purpose of updating this particular network, that is nothing but is your incremental training or online training.

Now, let me make it more clear, supposing that I have got 1000 training scenarios, collected known input-output relationships, now what we do is, out of the 1000 training scenarios, the first one, the first training scenario means known input-output relationship. So, inputs we are passing, I will be able to calculate the output. So, this calculated output will be compared with the target value, I will be able to find out, what will be the error

based on this particular error, we update this particular network, that is nothing but is your incremental training or online training.

Next, we go for the second training scenario, once again we repeat the process, once again we modify the network then, I go for the third training scenario repeat the process and this is actually the principle of your incremental training or the online training. Now, if you see the computational complexity of this particular training, it is computationally faster. But, there is a possibility that you may not get a very adaptive network, now by adaptive network I mean a network, which will be able to provide some sort of adaptive solutions, even for some unknown test scenarios.

So, if I follow the incremental training or the online training, we may not get a very adaptive network, but its computational complexity is much less. Now, I am just going to explain the principle of another training and that is called the batch mode of training.

(Refer Slide Time: 31:31)

Incremental vs. Batch Modes of Training (cont.)

- **Batch Training / Off-line Training:**

The whole training set consisting of a large number of scenarios, is passed through the network and an average error in predictions is determined. The network will be updated based on this average error.

Handwritten notes on the slide include:

- e_1 , e_2 , epoch
- e_{1000}
- adaptive
- \bar{e}

The slide also features the Swayam logo and a small video inset of a speaker in the bottom right corner.

Now, let me take the same example that I have got a set of 1000 training scenarios like the known input-output training scenarios. Now out of 1000, the first one I pass through this particular network and I will be getting some output. So, you store that particular output and this output you compare with the target output of the first training scenario, find out the error and this particular error actually, you save it.

And, then you go to the second training scenario. So, once again, I am going to pass the set of inputs corresponding the second training scenario and I will be getting some output, this output will be compared to the target output and I will be getting some error supposing that corresponding to the first training scenario, say I have got some error. So, that is say denoted by e_1 corresponding to the second training scenario the error which I am getting say e_2 and I am passing all the training scenarios. So, supposing that I am passing all the 1000 training scenarios.

So, I will be getting this particular error, you find out the average error, that is, \bar{e} , you sum them up divided by one thousand that will become your average error and based on this particular average error, you update the network only once. That is actually, the principle of your batch mode of training or the offline training and once again let me repeat the batch mode of training.

So, this particular updating of the network is done after passing all the training scenarios and based on the average effect, we are going to update this particular the network and as you consider the average effect, while updating the network, there is a possibility that this particular network will become very adaptive.

So, this adaptiveness or adaptability we can achieve, if I just go for the batch mode of training and which is nothing but is your offline training. Now, here, we should take one precaution like what should be the number of training scenarios during the batch mode of training, actually that we will have to follow very meticulously. Now, supposing that, I have got, say capital say X number of the design parameters in a network. Now, these particular design parameters include the number of connecting weights, the number of bias values, the coefficient of transfer function and all such things; and supposing that I have got capital X number of design variables corresponding to the network.

So, I will have to select the number of training scenario in such a way that the number of training scenario becomes at least equal to X or greater than X . So, the number of training scenarios should be either equal to a capital X or it should be slightly more than capital X , but it cannot be less than capital X .

Now, if I consider the number of training scenario is a less than capital X , there is a chance that there will be some sort of under-training and this particular network may not be able to predict very accurately and that is why, actually, for the batch mode of training

so, we will have to use a large number of training scenarios. Now, the way I mentioned that after passing all the training scenarios, we do the updating only once and that is called actually one epoch of this particular training.

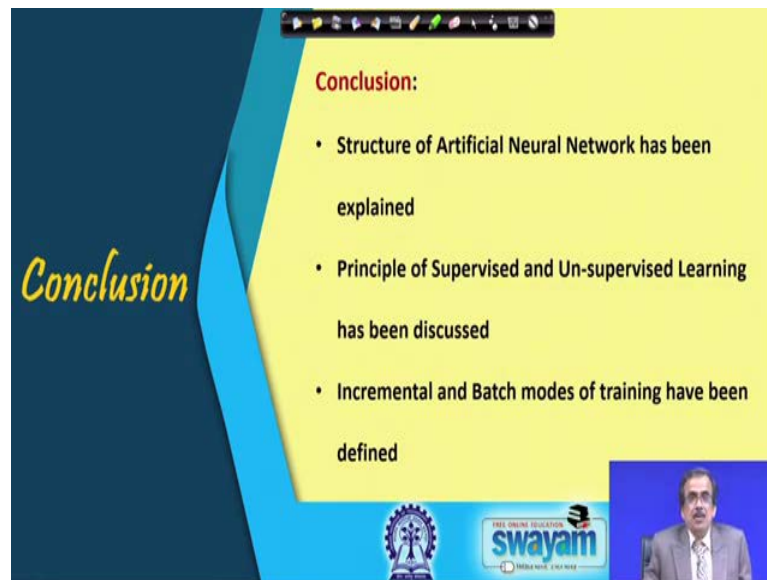
So, if I consider the batch mode of training, after passing all the training scenarios, we do the updating once, that actually concludes one epoch of this particular batch mode of training. So, this is the way, actually we implement the batch mode of training or the off-line training.

(Refer Slide Time: 36:05)



Now, regarding the reference, the textbook for this particular course is Soft Computing: Fundamentals and Applications. So, we will have to see it for more details.

(Refer Slide Time: 36:21)



Conclusion

Conclusion:

- Structure of Artificial Neural Network has been explained
- Principle of Supervised and Un-supervised Learning has been discussed
- Incremental and Batch modes of training have been defined

swayam

Now, let me conclude, whatever I discussed in this particular the lecture. Actually, we started with the working principle of one biological neuron and we tried to design one artificial neuron just by copying the functions of or the working principle of that particular biological neuron. Then, we in fact, discuss or introduced the structure of an artificial neural network, we discuss the principle of the supervised and unsupervised training. And, these things will be discussed in much more details in future. Then, we concentrated on the concept of the incremental and batch modes of training generally used in supervised training for the neural networks.

Thank you.