Mathematical Modelling and Simulation of Chemical Engineering Process

Professor Sourav Mondal
Department of Chemical Engineering
Indian Institute of Technology, Kharagpur

Lecture – 59
Artificial Neural Network

Hello everyone. Today we are going to talk about the neural network systems. As I have said that this week we are studying all the different types of stochastic models. And neural network has been, though it is quite old in some sense the original model was developed almost like 40 years back. But, with the increase in the computational power, these days this becomes quite popular. So, now, this neural network system is actually inspired from the way how information is communicated or transferred in the biological neuron or in the human nervous system. But, this has nothing to do with the biological or physiological neural activity.

It is only the structure that represents or has similarity to the neural networks that we have in in the human body or in human living animals. This neural network has been used to model or to correlate or to formulate a process model, where there is no available physical law or physical relationships or fear some physical description towards the problem.

So, you can think of this that model as sort of blackbox model, which can relate my input and the output, without much requirements or knowledge of the process background. So, absolutely the neural network models or the response surface methodologies that we have studied in the last couple of classes, thus tell you anything; I mean, not a single piece of information related to the physics of the system.

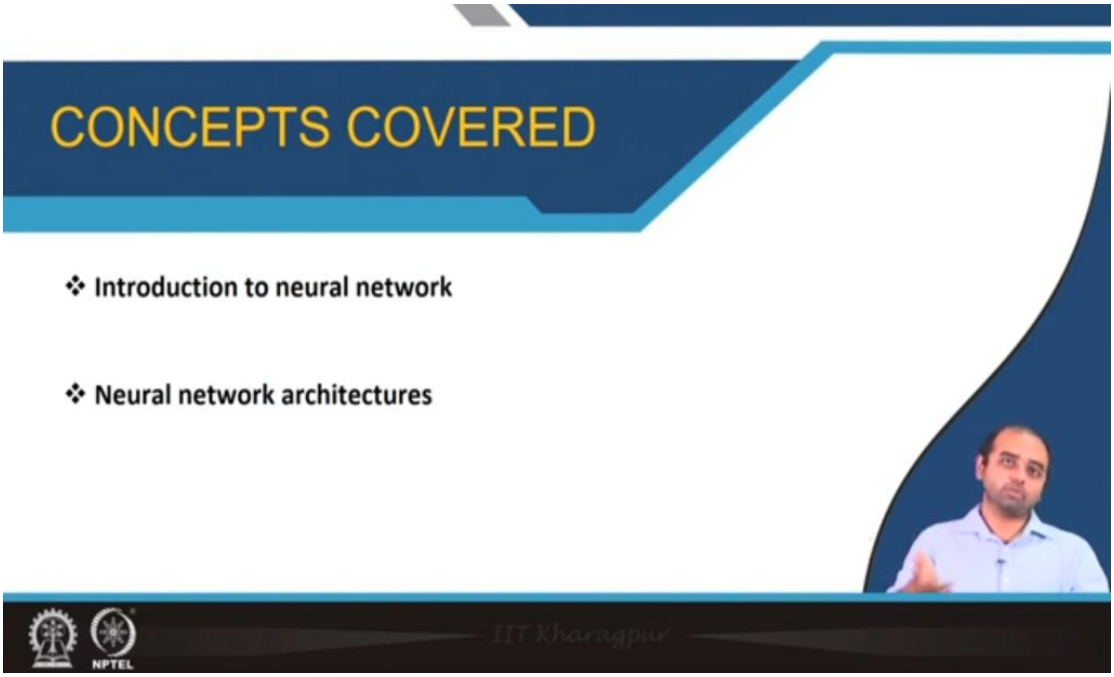
But, essentially if you want to find a model description or a sort of correlation that can relate your inputs to a response, this is where this plays a role; or this is where this is useful. Trying to have a correlation between the input variable; I mean from the input variables to the output. And using that model we can predict to some other unknown results, or some unknown value process parameter values for which the output is not known. So, a classical example these days that we see is that this COVID infection. So, in the pandemic, the modelling, the trend of the infection spread is very very vital, so as to make policy decisions to take administrative preparations to have non-clinical interventions; this is very essential.

Now, I mean, this COVID infection, although it is transmitted through the humans, or through air whatever it is; but this is impacted or influenced by so many variables, geographical diversity, immunity of the individuals, vaccination of the population, then, the variants you have. It is driven by so much of factor that is almost impossible to track the effect of one of these particular factors into the problem, or get a physical description of this.

So, in these situations, it is often useful that if you can have a model that only on the knowledge of whatever has been developed or known for certain. So, for example, the relation between or the observed using the observed values of, let us say the number of days with the infections that is growing. So, from this information, can you predict or forecast future such infections? This is where neural networks come in very handy.

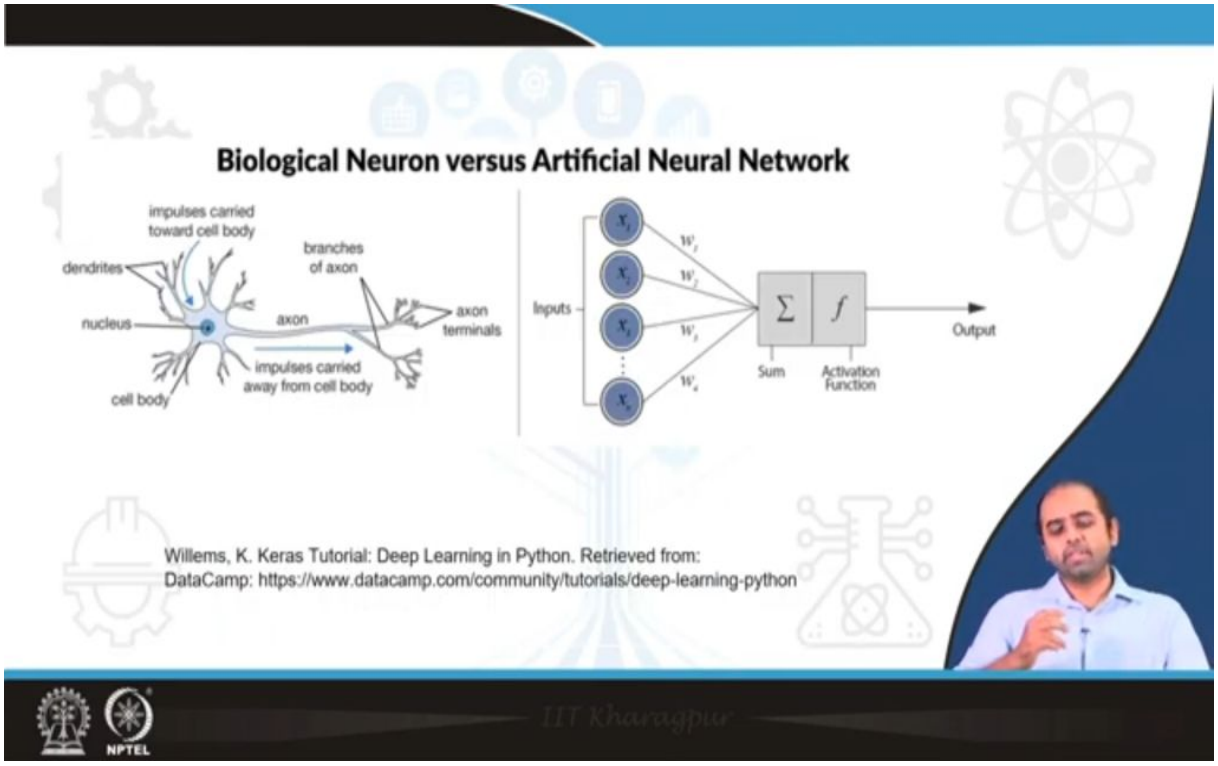
So, as to process this information or use this information as a new input, and to deliver; I mean, I mean information means both the input as well as the output, I mean whatever is known; so, everything it has to be fed. And using that, it will try to develop a correlation or a mathematical description with which we can predict for unknown set of values. So, in this case the timeline is the unknown, I mean the future is unknown. So, that is how this is useful or that is where it is more relevant and powerful in its approach. Now, what essentially is the neural networks?

(Refer Slide Time: 04:57)



So, we will talk about the neural network background, its structures or architectures in this class. And then in the next class, we will talk about more on the algorithm or the mathematical background of these model features.

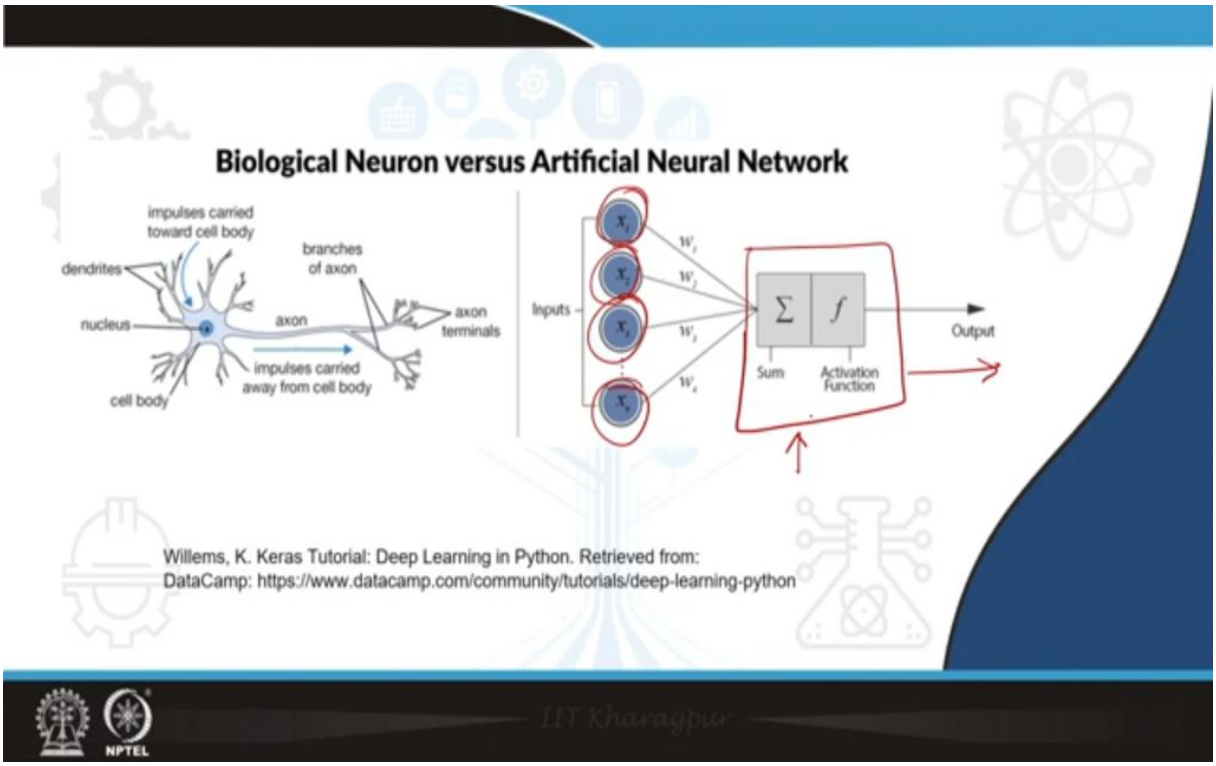
(Refer Slide Time: 05:11)



So, essentially, the idea of the neural network is is derived actually from, as I said, from how the nervous system is actually in our body works. It is to process, assimilate process, memorize information, and using that it is somewhat the system or the brain actually learns. And then, it can provide solutions, or it can provide answers to some of the unknown situations. So, from your past learning or experience or your training, you can handle an unknown situation; and that is how this any mathematical model is something what we expect them to do. So, any mathematical model will have certain unknown parameters, if at all; and that is the case.

So, in the response surface methodology also have seen that the regression coefficients are the unknown, and those coefficients are to be determined from the known values. So, that is what we can think of as a sort of learning or training exercise. And then the model is capable or equipped to predict for some unknown parameters; that is what something exactly how our human nervous system or human psychology actually works.

(Refer Slide Time: 06:31)

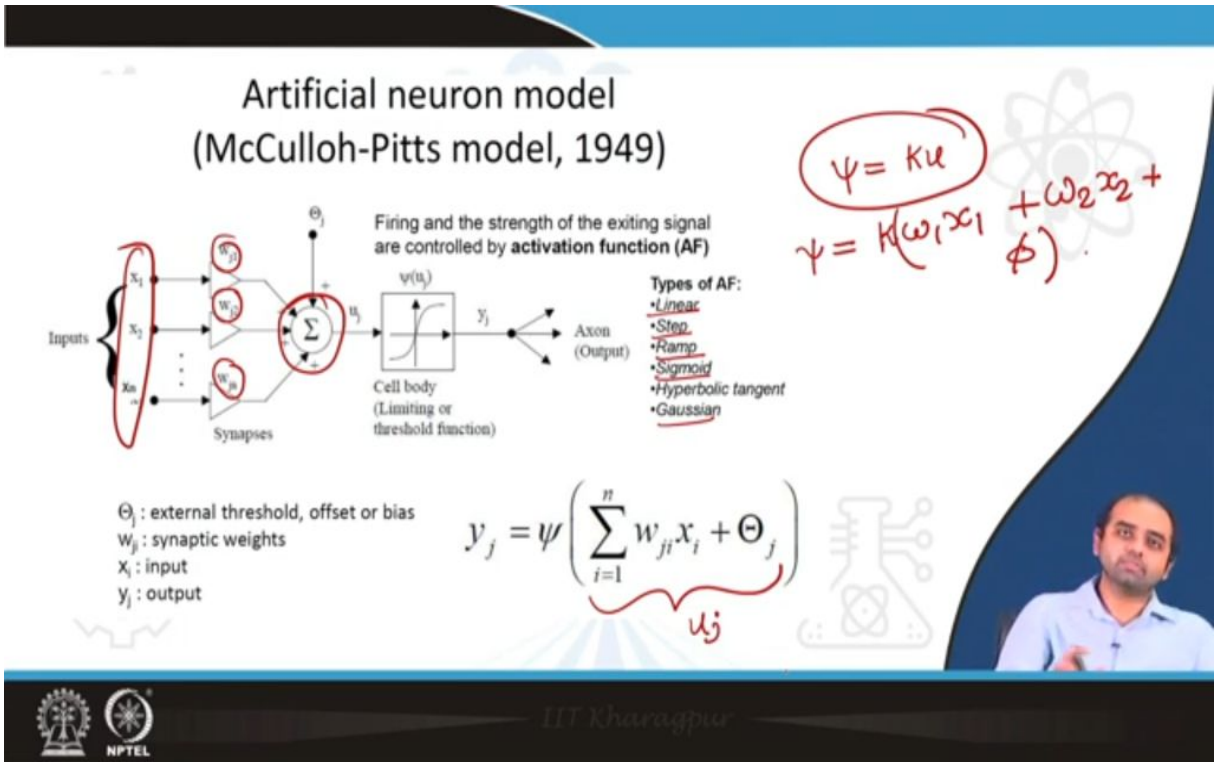


So, there are some more similarities to the this artificial neural network with respect to a biological neuron. So, in a biological neuron, we have something known as the axon and something known as the dendron or the dendrites. So, these dendrites are the these are sort of the input part, which then the information or whatever these impulses that is actuated by the or, or are accumulated by the dendrons is actually passed through the axons. And then it goes to the synaptic branches and then go to the next neuron. That is how it works out to a central nervous system; that is how it generally works.

In the same way, the structure here also works in the same way for a neural network, is that you have these input nodes, the once on the blue sides that you see is essentially the input nodes in this case. So, these are the input nodes, which collects the information or which gathers the information, which is then after some processing is transferred into this block.

Where this information is assimilated, it is processed with an activation function generating an output. So, whatever this structure that we have here for the processing of the inputs is something very hard of the neural network; and that is something also similar to what the biological neuron actually works.

(Refer Slide Time: 08:04)



So, let us look into the neural network model and this model is in fact, proposed almost 70 years back by McCulloh-Pitts, 1949. So, what it tells you that let us say the input variables are defined by x and the output is y. So, these inputs once they are collected, they are they are essentially processed through the synapses, so as to increase or or have some this scaling, or some sort of processing to them. So, this W are essentially known as the synaptic weights. So, that these weights actually help the different these inputs to be balanced, or you can say they are scaled appropriately.

Let us say these three inputs one is mass, one is for example, density; sorry, one is mass, one is let say volume, and one is temperature. So, if I feed these two into my system, it may happen that the mass is too large, and volume is too small. So, if it is treating something in terms of absolute value, then it would be too much, it would be almost inappropriate to balance; or to put in a relation where the different inputs are of different levels, or different dimensions, or different scales. So, it is very essential to scale them appropriately, either to remove their dimensions and make them almost uniform dimensions, or to actually scale their different magnitudes.

It is same as how we try to do non dimensionalization of an equation; so, the different terms are of order 1. In the same way these are scaling these weights, W are sort of scalings which helps to process these inputs in the appropriate level; and so, all of them should be in appropriate level before we can feed into the function. So, this whatever this activation function, this big block that we see that activation function here, actually expect its input in in at the same levels; they should not be having very different levels or very different order of magnitude, in fact. So, that is very crucial for the for the problem. So, this is what this is weights W actually does.

It is a sort of scaling that is applied to each of these variables; and such that everything is in the same order, and it is fed into the function. So, that all the parameters have equal levels of sensitivity in the system. And once it is then summed down this in this block, it is actually summation of all the parameters; or the summation of all the weighted I would say variable or the input variables.

Let us say, we call that as u; and to that there is some bias also is added. Bias is something that you get an output, something like if you get an output without any input. So, that is what we call as bias or thresholding; that is added and the entire cumulative thing is not processed through an activation function.

So, this entire part is actually u in this problem, that is what you see, phi of u. Now, these activation functions could be of different types. It could be linear, it could be step, it could be RAM, it could be sigmoid, Gaussian, several types are there this activation function. So, if it is linear, if it is linear; so, it means it is something like this.

Or essentially, if y u is of very, I mean, it is a summation of different variables. It is essentially, u k into w1x1, then w2x2 plus phi; some, some bias into the system, something like this. That this phi actually works; so this is a linear activation function. So, once this activation function is applied to the cumulative input, you get your output.

Now, output could be single valued, output could be multiple outputs also. So, if there are more than one outputs, they will then you will need more than one activation function. So, this is the simplest neural network design which, which is actually easy to understand. And actually, this is one of the simplest and starting description of the neural network.

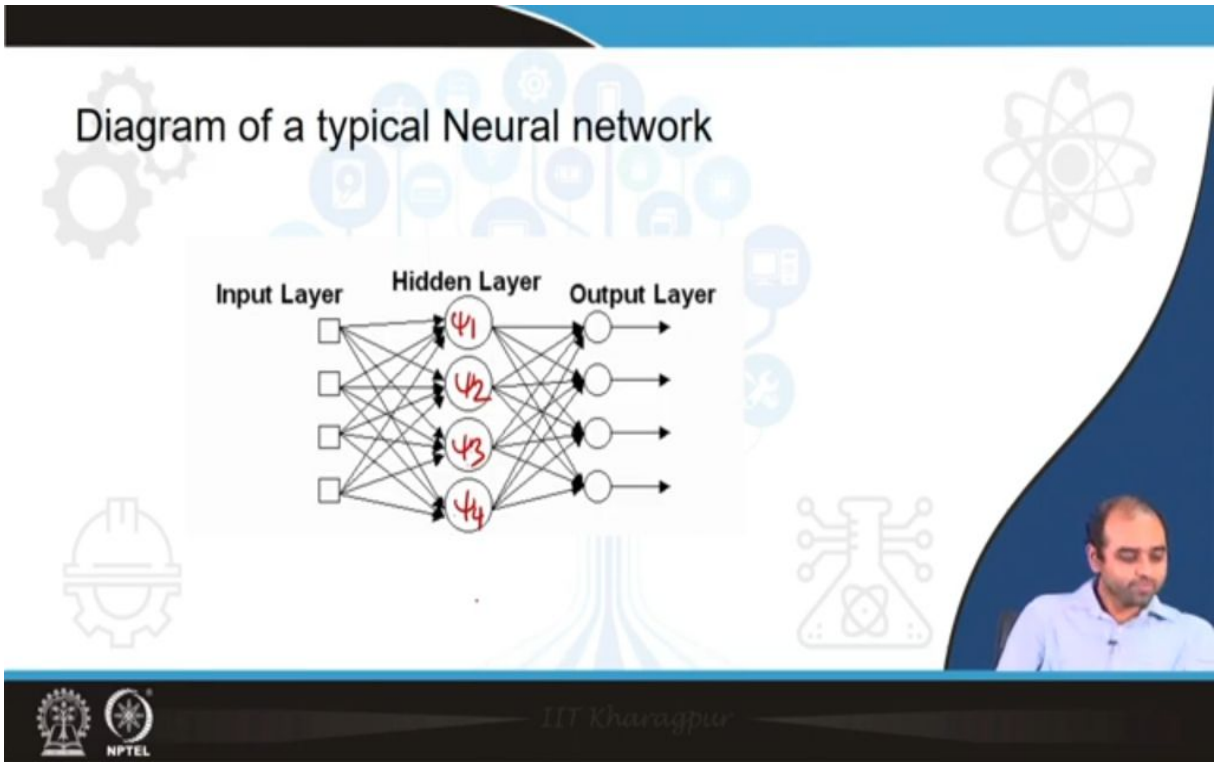
There could be other different types, we will talk about them; we will talk about the different architectures of the neural network. But, just to give you an understanding about what does the this the base case, or the starting foundational scenario of the neural network looks like. I will repeat this once again you have the inputs, they are designated as x1 to xn; they are processed or scaled with respect to the weights; and weights is something which is unknown for the problem. Generally, that is the case.

You can also try to have your own weights depending on what kind of function you are implementing there, and the activation function. But, generally weights are considered to be unknown and they are automatically optimized, such that the sensitivity of the different input variable variations are actually captured in the output. In that way you can, the more the variation that you have in your inputs, the better would be the this optimization of your weights (optimal). These weights are actually the unknown in this equation. So, then you pass it through the activation, I mean then you sum them up in the summation all the processed as scaled input variables plus the bias.

Then, you process it and then you get your activation function, essentially to process the cumulative input to produce your axon; sorry, to produce your output. So, in this case, these in the weights w1, w2, these are essentially the unknowns in this case. And if you are choosing a linear activation function, you can clearly see that for this problem, this is equivalent to the first order model, is not it?

So, there is a co-relation between the neural network with the response surface or the classical regression equations. If you are choosing a parabolic activation function, then it becomes the second order model. Other functions will lead to several other types of the functionality or the regression equation. But, this is the heart of the neural network. Of course, several variations are exist and things can get complicated; but this is the basic case of the neural network design.

(Refer Slide Time: 14:47)



Now, let me let us talk about this another typical type of neural networks. And this is something what normally you can expect to get is that, there may be more than one output as I was saying. So, if there are more outputs, the internal layer that we have. So, internal layer means where you have the processing through the activation function; that is something we call the hidden layer.

So, how many such activation blocks do you keep is something. So, more the activation number or activation layer blocks the more the number of weights, as well as those linear proportionality constants and all those things, if you are choosing a linear model or nonlinear model, whatever.

So, the number of unknowns will increase as you increase the number of layers or the number of nodes in the layer also. So, each each activation block or each block in this hidden layer have information of all the inputs; that is how it is designed. And all of them produces outputs are all have their information to be passing to the output layer also.

So, these individual blocks can be considered as like psi1, psi2 different activation functions, psi4. So, the more number of nodes you add, the more number of unknowns you are injecting into your system. So, the idea is that network which works best with the least number of nodes,

or these least number of activation block is more reliable. Why? Mathematically speaking, this will have less number of unknowns into your system.

The more number of unknowns that you inject into your system, the (number); so, it is something like degrees of freedom. So, if the more number of unknowns you have, the degrees of freedom of the system also increases. So, you are you are you are trying to estimate more number of unknown parameters from the same data set.

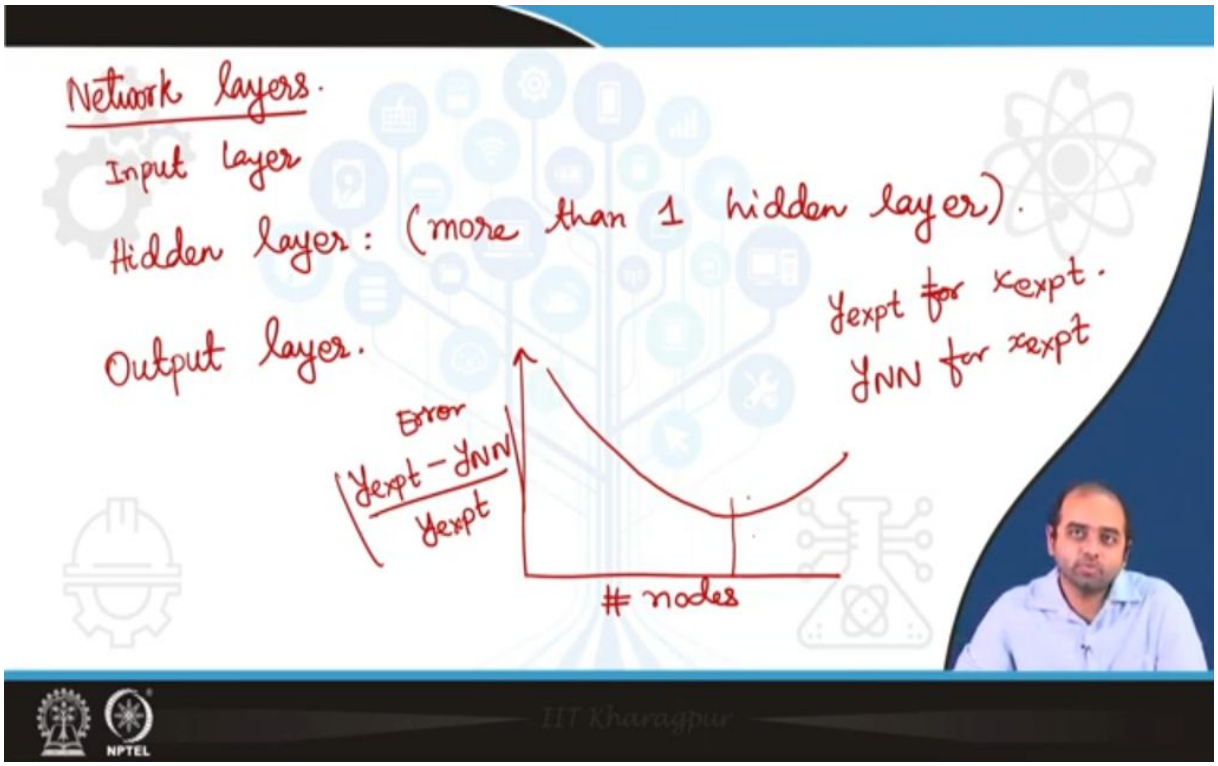
So, if let us say the number of unknowns is 10 and you are having 100 data points or 100 known solutions; so, approximately for each data for each node or for each unknown. Let us say each node will also have if you have four units like four unknowns, or four weights, those are which are unknown; so almost 40 unknown parameters into your system over 100 data points.

So, approximately is 2.5 number of data that you have per evaluating per unknown; so, that is too low that is too low. So, if you are having let us say only one unknown, which will have four inputs, it will have four unknowns. So, almost 25 data points would be there for evaluating or estimating or regressing that unknown parameter quite well.

So, the idea is that we should try to attain to work with minimum number of nodes getting a reasonably accurate result; that is mathematically more accurate, reliable, and something to be appreciated. Why I said this? Because, we always see that the tendency to have a better accurate result, of course, with the known data set that you are using to optimize your weight to have more number of nodes.

So, of course, if you have more number of nodes, you will have better accuracy in the result. Let us say, if you have a straight line, a straight line can be modeled or can be estimated or modeled with only two unknowns; but you can overestimate it, you can use like 10 unknowns. You can fit a 10th order polynomial to a straight line; but that is mathematically not feasible or not a good thing, or that is less accurate you would say; so, same thing happens here. Now, input layers.

(Refer Slide Time: 18:47)



Let us understand, what are the different network layers in this neural network architecture input layer. So, it is where the activity of the inputs or the information from the inputs is actually processed, or the raw information, or the raw values that are there is actually processed. We are scaled with the help of the weights; then you have the hidden layer or the internal layer. So, this is again dependent on the activity of the input and all it is. What it is does? It does the summation of all the inputs or processed inputs, it applies the activation function and everything; and then you have the output layer where it generates the output.

Generally, this is the normal structure of the neural network is having these three layers; generally, you have one hidden layer, one input, and one output layer. But, you can have more than one hidden layer. So, more than one hidden layer means there is interconnectivity of the information between the layers.

More than one hidden layer is possible and each layer can have different number of nodes. So, there is interconnectivity of the information between the layers for further processing to generate the output; but again the less number of hidden layers you have the better easier result. Now, the

number of layers or number of nodes in the hidden layer is something which is not known a priori.

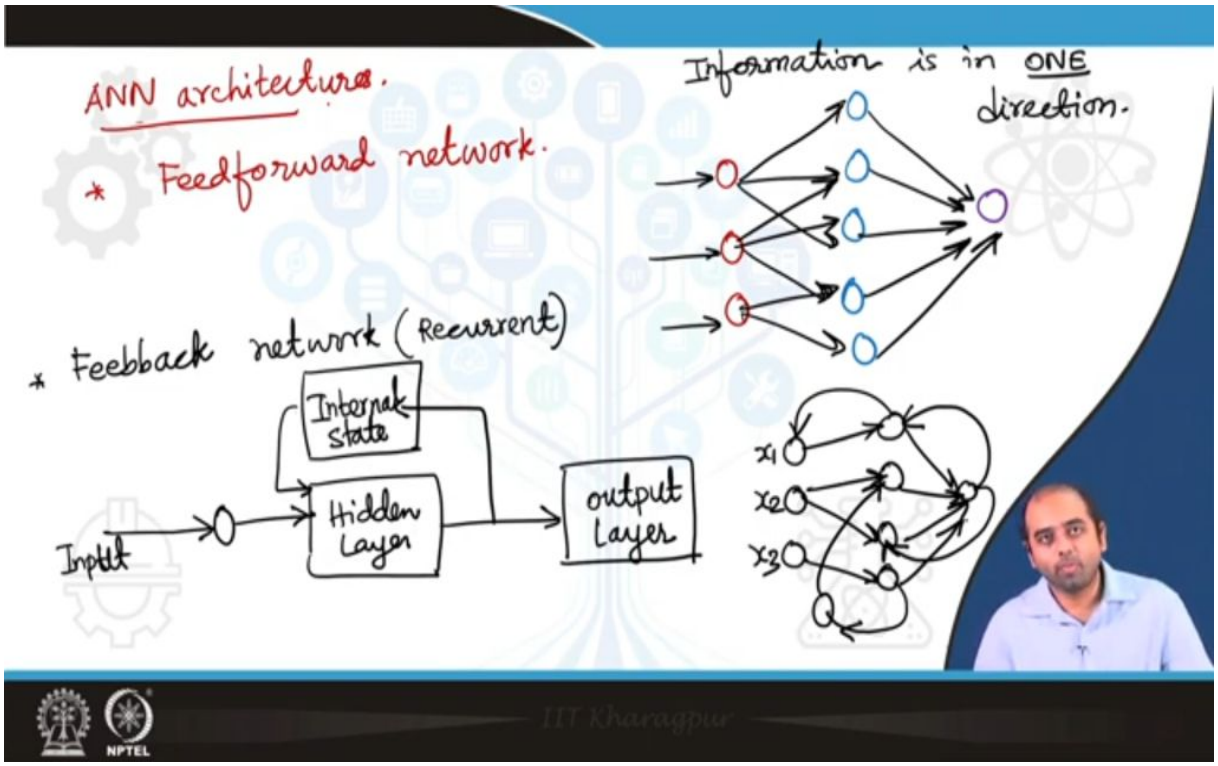
So, what ideally one expect to do is that first you start with a single hidden layer, and then you try to make a plot of the error. So, what is this error? I will tell you with respect to number of nodes. So, this error is the relative difference of the neural network output with respect to the known values; that is what we call an error.

So, of course, using these known values only the weights are optimized. So, it is like the regressed equation or the solution or the output of the regressed equation for the known inputs, for which the output is already known from the experiments or from some other information. So, let us say you have the experimental expt is known for the inputs.

So, you want to predict neural network output for the same quantities of the data for which you have the response is known to you. So, error is actually for the same experimental data. It is like you can see the something like the relative error; this is something can also make a square of it. This error is something that you wish to see that (ha), I mean with changing the number of nodes, which should be decreasing; and at some point, it will again start to increase which is like overfitting. So, what is the lowest error that you can achieve for one hidden layer?

So, these ones you fix it, then you go for more number of layers, if required in your problem and do the same exercise. First of all, one layer try to optimize the number of nodes. And if you see the known number of nodes are too much, then you go for hidden two hidden layers, and see if things are improving or not with less number of nodes. Now, let us come to the ANN architecture.

(Refer Slide Time: 22:51)



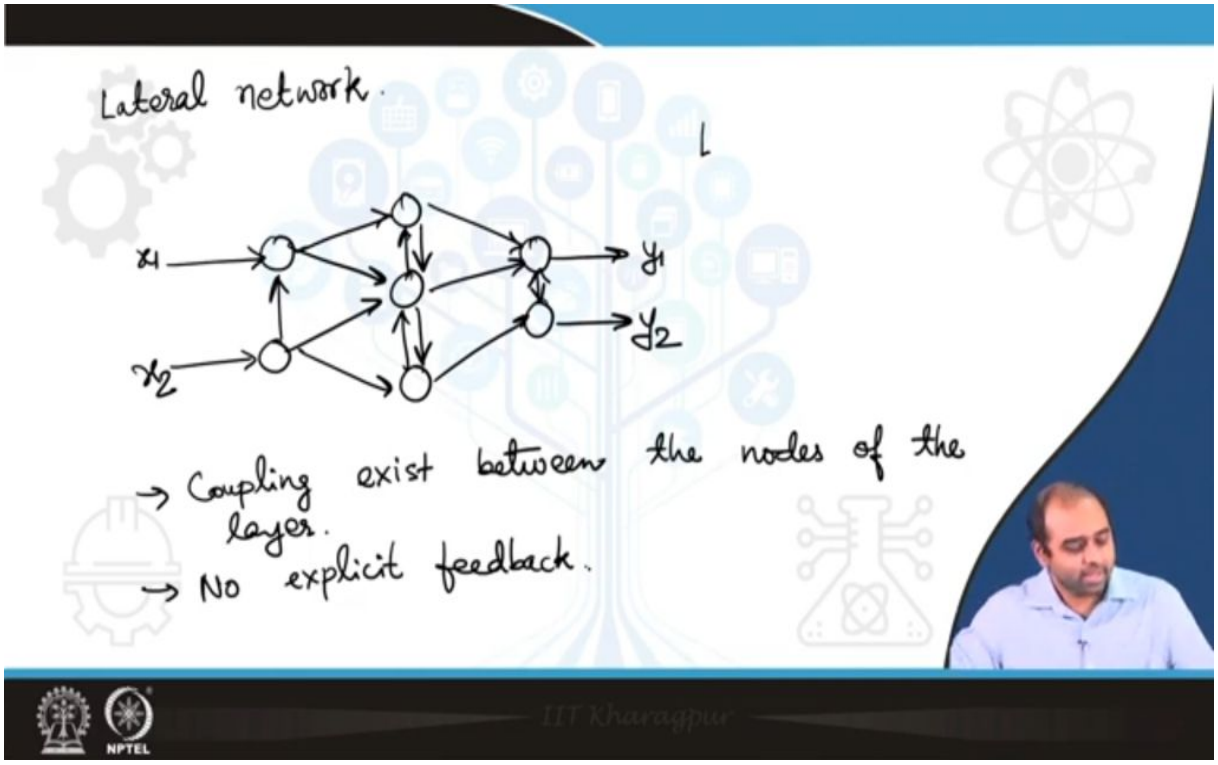
So, there are something known as the feed forward network. So, the feed forward network is that where information only flows in one direction. So, feed forward network you can think of like these are the inputs; then you have, let us say the hidden layer nodes, and this is the output. So, whatever information that is coming to these input nodes is getting passed on to these nodes whatever way, let us say the connection we make into these nodes. And it is always information is in one direction; in this case it is from left to right whatever is processing. But, feedback networks, there is another type called feedback; feedback network are also known as recurrent network.

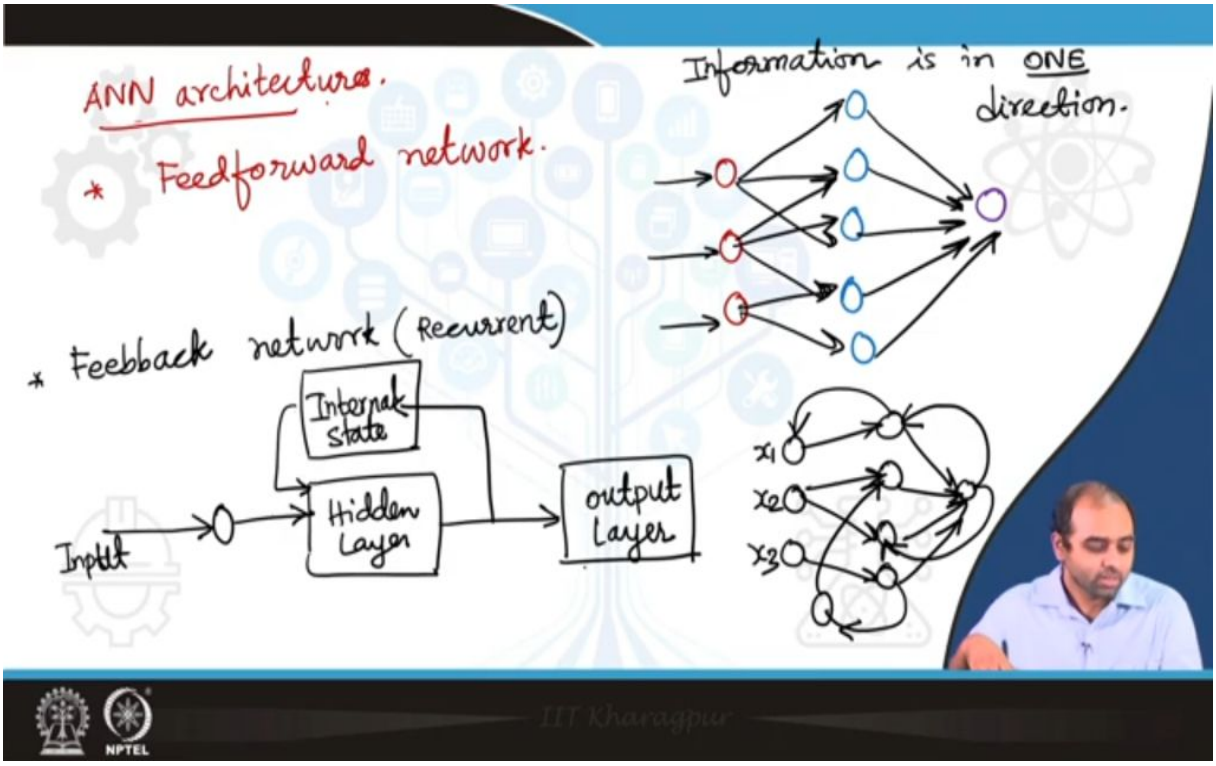
Here it is slightly more complex and something beyond; I mean working with feedback network is beyond the scope of this course actually. But, I will try to give you a hint about how does things look like in the feedback; I mean what is the structure. So, you have the input, it goes to the input layer; then you have the hidden layer, it goes to the output. But, there is a backflow of information or recycling of information. Let us say this is another internal state through an intermediate layer or a state, and it is fed again to the hidden layer. So, part of the output, part of the output is actually reused or recycled back into the hidden layer; so, information is not always unidirectional.

So, common example like if you have the same kind of structure; so, it is like let us say this is the input, this is the hidden layer. So, this is like x1, x2, x3; so, this goes there, this goes there. So, this goes here like this. But, a part of the intermediate layer, I mean this goes back to a hidden state, something like this, and goes back there maybe.

There is this sort of back connection into here or this can also make a back connection here; so, there is backflow of information also. So, part of the output or part of the internal hidden result is also reused due to process the information in between the hidden layers. So, this is what is known as the feedback network and these are generally not very common, and only in special circumstances this is used.

(Refer Slide Time: 27:02)





Then you have the, for example, this time varying predictions or forecasting feedback networks is sometimes used. For example, prediction of the future stock market, you know this changes that is something would need the knowledge of the output, the neural network outputs to again pre-process itself, and validate itself for the future process. Lateral network means there is no backflow, but there is interconnection between the nodes; for example, like this. Let us say, you have internal nodes when not only information is passing between the nodes, but among the nodes also; so, this is something like lateral network.

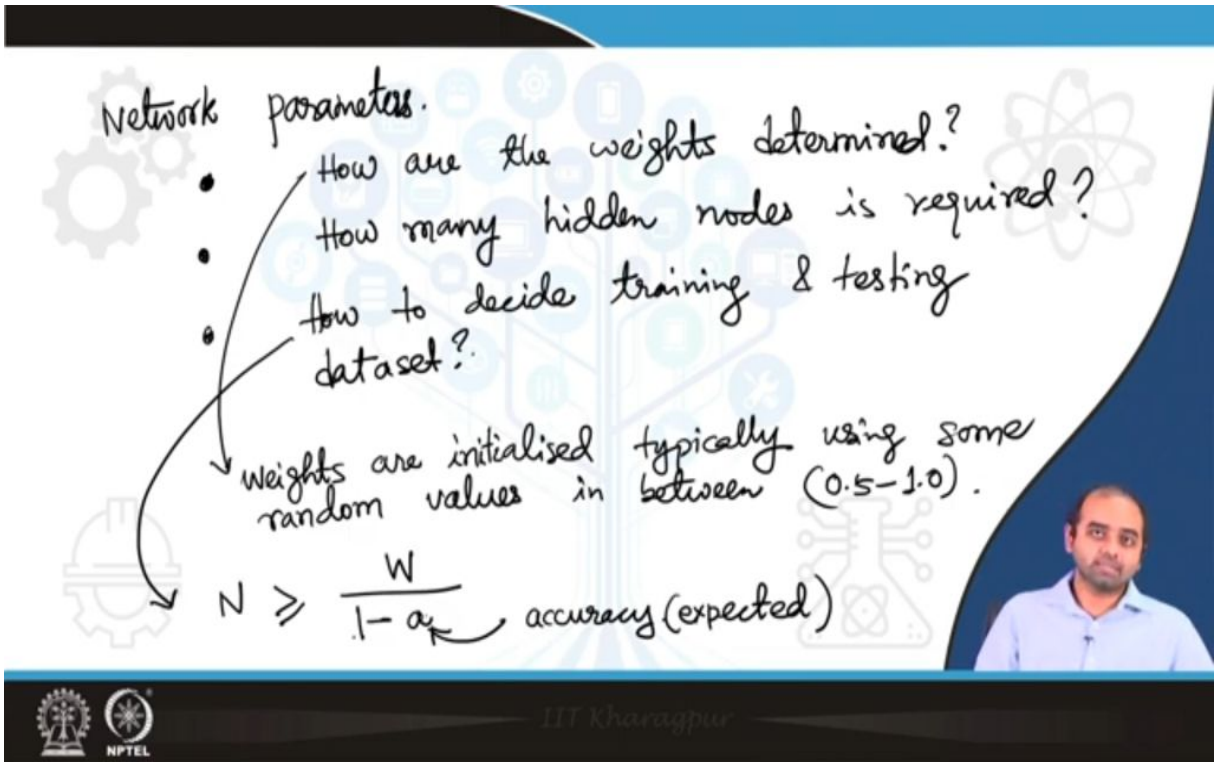
So, this there is coupling between the nodes that is what we mean by lateral network. So, coupling exist between the nodes of the layer; no explicit feedback essentially, because there is no bad connection. So, this sort of a compromise or something intermediate between the feedback and the feed forward network.

So, feedback I also must say, the feedback is something very relevant for process control. I mean not relevant, but it is having an analogy to the process controls. In process control if you recall, you also have this feedback network, which is very common. Because, you always need feedback of the final input to adjust your set point changes or the controller to take action; and

they are the most stable. Feed forward is also there in this control system, ratio control cascade controls and all are feed forward controllers.

So, in feed forward always information proceeds in one direction. I do not know whether lateral kind of network do exist in control system or the explicit analogy, if it is there. What about the network parameter? So, the key question that we want to ask here is that how does the network weights actually.

(Refer Slide Time: 30:04)



So, the network parameters are very essential to be estimated. So, the question is, how are the weights determined? How many hidden layers I talked about, how do you determine that how many hidden layers or hidden nodes is required? How to decide, how to decide training and testing data set? So, in general, these the weights initially are initialized using some guess value. So most, in most cases, whatever neural network platform you are using to simulate for or to develop this model, for have some inbuilt strategy to initialize the weights. And these weights are typically in the values of point 5 or 1; so, this is how the weights are initialized.

And but how the weights are determined is something probably we will talk in the next class. And that is the whole actually the the the background of the neural network, and that is what it is

based upon. But, the weights please note the weights are initialized. So, weights are initialized typically, initialized typically using some random values.

Again, we are having random number, some random values in between point 5 and 1; that is generally the case. Weights can also be initialized as a part of the process, so you can have fixed networks where the weights are fixed by initialization, on the the way that are initialized or fixed; it can also be adaptive.

So, during the, so it is the iterative process. So, what does this iteration do? The iteration actually helps, it is the regression iteration, by iteration the values of the weights are optimized. So, if you find that during the iteration, the weights become too much further away or the weights are not getting optimized; or you are not getting the final output close to the actual known output.

Then, the weights need to be reinitialized. We find that the initialization was itself a problem; that is why it is not progressing towards solution, error is actually diverging. It is not coming closer not reducing. So, in that case weight needs to be re-initialized; that is normally the case how most mostly this is how it is designed for.

And they are (initial) this is called adaptive weight theory initialization. How many hidden nodes are required? I already told you calculate the error, and then make a plot with the number of nodes; and see in each case the error is minimum. How to decide on the training and the testing data set? So, typically the number of weights that you have, so deciding the number.

Sorry, the number of weights, the the the way is that if you are expecting the total number of data points should be greater than equal to the number of weights that you have, divided by 1 minus a. And what is this a? a is the accuracy that you are looking for; expected accuracy that you are looking for.

So, if you want your result to be we more than 90 percent accurate or a to be point 9; I mean 90 percent accurate of the actual value that you have. Then you need, then you need the a is point 9; then you need 10 times of the number of weights. So, if you are choosing four nodes, then the total number of known data point should be 10 times; and the known data set is something should be available to you.

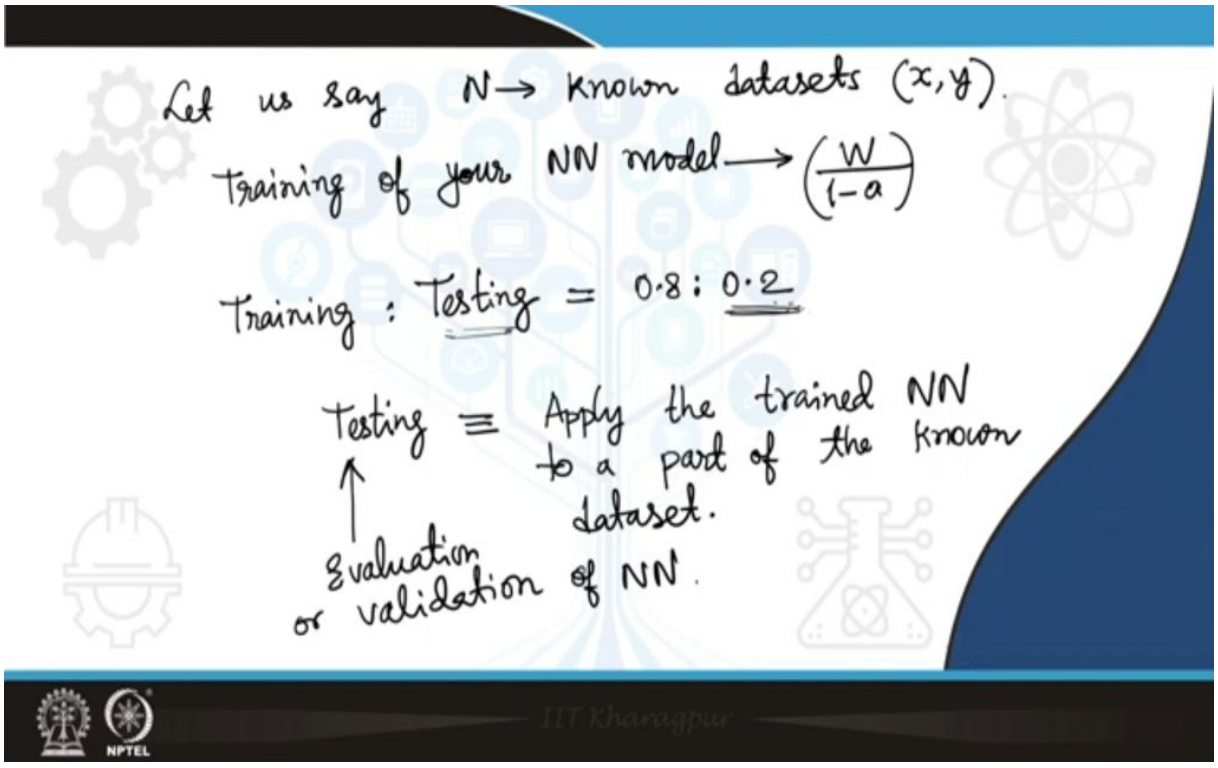
So, what do you mean by training and testing? So, if you have a data set where you have a set of inputs, you have the set of outputs. We generally, divide these data set, we do not utilize the entire data set for the regression or the optimization of the weights; which is called as training. So, training, learning are same thing.

So, essentially learning is very common in the neural network terminology. So, learning is actually a process of iteration with which you are trying to find optimized value of your weights; weights are essentially the unknowns in this parameter. So, all the unknown parameters of your equation, finally it is mathematically linear-nonlinear whatever this equation that relates the input and output; so these weights are actually unknowns in the parameter. So, essentially training is a process or iterative process with which you try to optimize the weights of your unknown parameters. I mean, the weights are unknown parameters of your neural network model.

So, training, supervising weights are quite technical jargons in neural network, learning. So, that is what we mean that this training or learning is essentially trying to optimize the weights of the this neural network or this regression equation, that can relate the input and the output. So, how many number of data points you need?

This is the equation or this is the kind of relation that needs to be followed. So, if you want 90 percent accuracy, generally you need 10 times the number of unknowns. So, if you have 10 nodes in your problem, the minimum data points that is needed is 100; we achieved 90 percent accuracy. So, let us say if you have; so that is the minimum number of data set. So, how to (deter), so what is testing by the way then? So, let us say you have n number of known data sets.

(Refer Slide Time: 37:01)



Let us say, you have n number of data points; you have n number of known data sets; data set means set of input and output. Now, for this, these are the known data that you have; you want the data to also. This what you call learning or training of your neural network, training of your neural network model. So, you should make sure that it satisfies the criteria, 1 minus W by point; sorry, not 1 minus W; W by 1 minus a, this criterion has to be satisfied. And if it is more than that, that is then it is good; but at least this much amount of data you need. So, you, so the total data if it is possible, whatever the total number of data that you have.

You also want it to keep some of these data to check for the final output. In that case, we treat the inputs for that case to the model as unknown. So, we do not try to train our model or supervise our model, but we want the result. But, since these results are already known, because they form it is part of a data set; so, we can easily compare them. So, generally the ratio of training to testing data set, is training to testing; we do not want to utilize if there are 100 data points. Let us say we and we see that the minimum requirements from this criteria W by 1 minus a is say 60 is sufficient.

We have only six nodes; and we have one to achieve 90 percent accuracy; so, this is sufficient to have 60 nodes. So, if you have 100, we do not want to use the entire 100. And the idea is that the ratio of training is to testing, the best is to point 8 to point 2; so, you divide that by 80-20 ratio. So, out of this 100, you can utilize 80 for training; or this determination of the weights by regression, iteration and everything, integration of the regressive equation regression equation. And remaining 20 use for testing. So, what is this testing? See, for these entire data sets we will have all the values of x and y known.

But, in this testing this whatever trained neural, so what is this testing? So, you apply the trained, apply the trained neural network to a part of the known data set; and then try to see how it is performing. You may calculate the error et-cetera and on that part; because for that those data are fed to the system to generate the output, as if this is unknown for the system. After the thing is trained and the weights are finalized all the unknowns are determined. I want to feed some data into the system for which I want to know the output. And this is since it is a part of the known data set; the output is already known.

The actual output although is known; so, I can easily compare. So, this testing is for evaluation or validation of the neural network, that how good are the predictions and this is very vital. The better the testing results that you have the more reliable is your neural network as simple as that. So, the minima, generally the recommended practice is 18 to 20. But, if you want to have more reliance on your; let us see you have a huge data. And there is there is a lot of data set, then you find that even with 70 percent of data is sufficient for training, then you can use almost 30 percent of the data for your training, testing.

Testing you can do with 10 percent also; but the, if there are a lot of different patterns in the output that you expect, then you need more data sets and more number of known data points for testing also. So, more the number you feed for testing, the better is the reliability of your model or the validation of your model. So that is what we mean by training and testing.

So, how you do the training? How do we estimate the this regression coefficients or the unknowns in the neural network model is something we will talk about in the next class. And will see the different algorithms that is used to calculate the unknowns. So, see you everyone in the next class. Please stay tuned. Thank you.