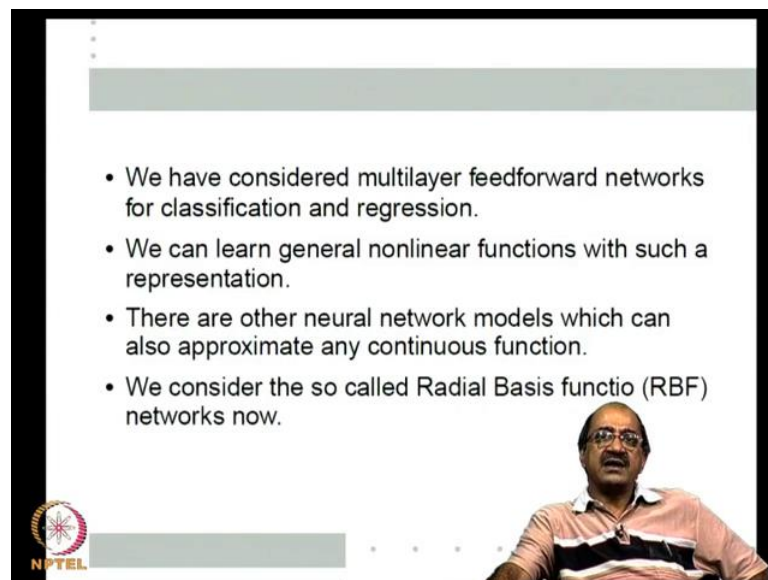


Pattern Recognition
Prof. P. S. Sastry
Department of Electronics and Communication Engineering
Indian Institute of Science, Bangalore

Lecture - 30
Radial Basis Function Networks; Gaussian RBF networks

Welcome to this next lecture in this course on pattern recognition. We have been considering in the last few classes on neural network models specifically what we called the multilayer feed forward networks for models. We have looked at these multilayer feed forward networks both for classification and regression.

(Refer Slide Time: 00:38)



- We have considered multilayer feedforward networks for classification and regression.
- We can learn general nonlinear functions with such a representation.
- There are other neural network models which can also approximate any continuous function.
- We consider the so called Radial Basis function (RBF) networks now.

Basically these are a good class of parameterized non-linear functions, so these models are good for learning general non-linear functions for classification as well as regression problems. We have seen that these networks can represent any continuous function to an arbitrary approximation on a compact set. So, in that sense, it is a very good parameterized class for non-linear functions. We have seen that we can very efficiently learn the weights or the parameters in this network. Essentially these networks are trained, so that we minimize the empirical error or empirical risk corresponding to squared error loss function.

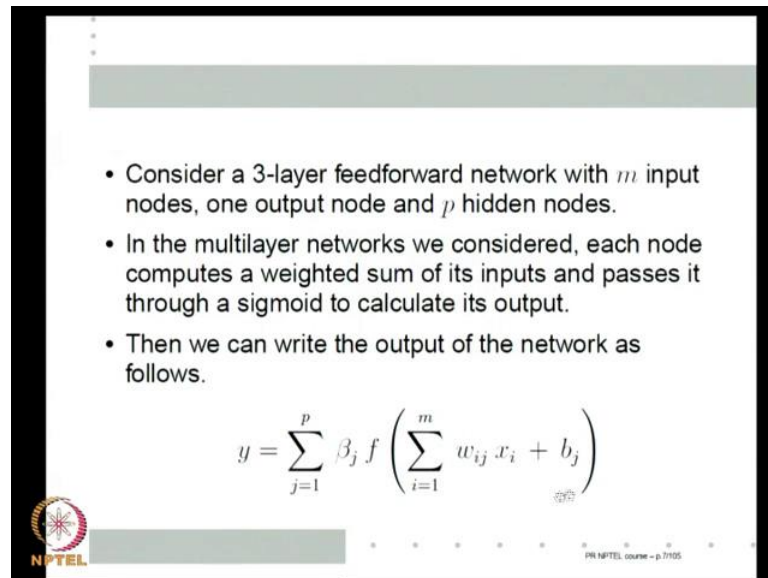
So, we take the square of the output of the network and the desired output. This is what we want to minimize and to find the... To minimize this we use gradient descent. To find

the partial derivatives of this error with respect to various weights in the network we use back propagation. So, this multilayer feed forward networks trained with back propagation or very nice general techniques for learning non-linear classifiers and regression functions.

Now, the particular structure that we considered are not the only kind of neural network structures that we can have, so the multilayer feed forward networks where each unit has a (()) activation function. Each unit actually takes a weighted sum of the inputs is receiving and passes through a sigmoid, that kind of structure. What we call the multilayer feed forward networks is only one of many other neural network structures that one can use for classification regression.

And what it means is is only one of the many structures which are just as good for representing any non-linear function. So, there are other neural network structures or you know neural network models, which are also good at approximating any continuous function. The reason why we said the multilayer feed forward networks are good for classification regression is that they can well approximate any continuous function and hence they are good models for parameterizing non-linear functions similar properties also help the other neural network model structures. So, for this course we will just look at one other such neural network model, what we call the radial basis function networks. So, that is what we are going to study in this class and possibly next class.

(Refer Slide Time: 03:15)



• Consider a 3-layer feedforward network with m input nodes, one output node and p hidden nodes.

• In the multilayer networks we considered, each node computes a weighted sum of its inputs and passes it through a sigmoid to calculate its output.

• Then we can write the output of the network as follows.

$$y = \sum_{j=1}^p \beta_j f \left(\sum_{i=1}^m w_{ij} x_i + b_j \right)$$

NPTEL © NPTEL course - p 1105

So, to understand what this radial basis function network is, let us start from the network models that we already know. Let us say you have a three layer feed forward network with m input nodes one output node and p hidden nodes by now we know these structures. So, that is why I am not putting any figures this time, so in the multi networks we know each node computes a weighted sum of its inputs and passes it to a sigmoid to calculate its output.

So, what this means is that we can write the output as the network as we already done many times as this. So, the, we take the output node to have linear activation. So, because there's only one output node the output is the weight connecting into the output node multiplied by the output of the j th hidden node. This is the output at the j th hidden node multiplied by the weight connecting j th hidden node to the output node. How do I get the output of the j th hidden node? It is a weighted sum of the inputs right x_i is multiplied by w_{ij} for the j th hidden node plus b_j is the bias of the j th hidden node.

So, I take a weighted sum of the inputs and pass it pass it through a sigmoidal nonlinearity. So, this is how we have been writing the output of the network that we seen, so far we can bottle up all this is what the hidden node is doing into some function, which is a function of some parameters and the input vector. This is if we if we want to generalize this we are what we are saying is the hidden node does not necessarily have to just find the weighted sum and pass it through a sigmoid.

At similar activation function it actually calculates some non-linear function of the input some non-linear function that is represented by some parameter vector. So, I can bottle up this entire thing into some non-linear function if I do that I can actually the write this alternatively as I have this beta j into something, which is, of course for the multilayer feed forward networks that we considered is specifically a weighted sum of inputs passing through a sigmoid.

(Refer Slide Time: 05:28)

• We can rewrite this as

$$y = \sum_{j=1}^p \beta_j \phi(X, \theta_j)$$

• This also represents the output of a 3-layer network where the output of j^{th} hidden node is given by $\phi(X, \theta_j)$.

• We consider such networks now.

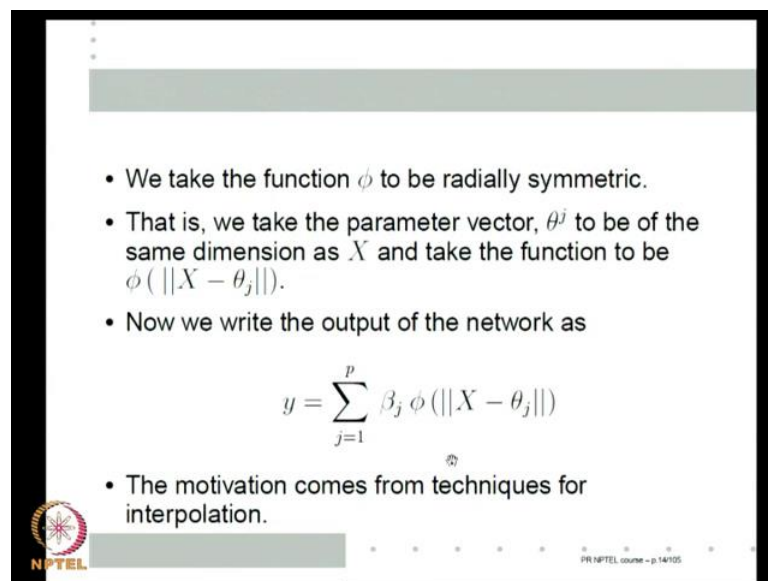
But in general we will write it as beta j of some phi of x comma theta j, so this is some non-linear function of the input as well as some parameters. The theta j at the parameters of phi, now we can think of this itself as some other three layer network. If I take different phi's what it means is if I have a network a three layer network where when the input is the vector x the output of the j th hidden node is given by phi of x comma theta j. Then the output node is linear, so the output of the network will be beta j times phi of x comma theta j.

So, this represents output of a three layered network with 1 output node capital x is the vector of inputs they. There are p hidden nodes and the j th hidden node has output phi of x comma theta j when the input is x and theta j is some parameter vector that we associate with the j th hidden node. Now, this is the kind of structure of networks you are going to consider. Now, where this is an alternate structure of the network where we are

making the output of the hidden nodes to be some non-linear function of x , which has some parameters inside it, those parameters are what we may have to fix or learn.

The kind of non-linear function that we have may not be capturable as a weighted sum of inputs. But it does not matter as long as some non-linear function that we specify this will possibly represent a non-linear function and later on we shall see it also can represent almost any continuous function. It has a similar network structures say this straight forward three layer network structure. Normally in this network not normally if I have the particular class of network that we are considering. We take the function ϕ to be radially symmetric, this function this function ϕ to be radially symmetric. What do you mean by that? That each of these functions for each j this term is such that it will have the same value for all x on a sphere that is what we mean by radially symmetric.

(Refer Slide Time: 07:36)



- We take the function ϕ to be radially symmetric.
- That is, we take the parameter vector, θ^j to be of the same dimension as X and take the function to be $\phi(\|X - \theta_j\|)$.
- Now we write the output of the network as

$$y = \sum_{j=1}^p \beta_j \phi(\|X - \theta_j\|)$$

- The motivation comes from techniques for interpolation.

So, what do we do for that that is we take the parameter vector θ_j to be of the same dimension as x . Take the function to be ϕ of distance of x minus θ_j , so if the j th term there is ϕ of $\|x - \theta_j\|$. Then as long as $\|x - \theta_j\|$ is same the ϕ value will be same and $\|x - \theta_j\|$ will be same for all x in a sphere around θ_j that is what we mean by these are radially symmetric. Now, this is how we can write the network output y is equal to one to p $\beta_j \phi(\|x - \theta_j\|)$, so β_j has the weights connecting into the output node there is β_j is the weight that connects j th hidden node to the output node.

Theta j is a weight or weight vector parameter vector associated with the j th hidden node. We will look at this these class of networks now. The reason for considering that particular class one networks comes from techniques for interpolation.

(Refer Slide Time: 08:38)

- Suppose we have data $\{(X^i, d^i), i = 1, \dots, N\}$ where $X^i \in \mathbb{R}^m, d^i \in \mathbb{R}$.
- We want a smooth function to interpolate this data.
- In our context, the question is: do there exist some β_j, θ_j and a ϕ so that the function

$$h(X) = \sum_{j=1}^p \beta_j \phi(\|X - \theta_j\|)$$

satisfies $h(X^i) = d^i$ for $i = 1, \dots, N$.

What what is interpolation? Suppose you have some data $X^i d^i$ as usual suppose a regression problem. So, X^i belongs to some \mathbb{R}^m it is a is an m dimensional real vector d^i is a single real value so you want to learn a function from \mathbb{R}^m to \mathbb{R} . So, we are given these examples $X^i d^i$. So, far we have been considering we want to learn the function, so in some statistical sense the function it well represents this relationship. So, when we learn a function like this let us say when we learn this function using neural network models there is no guarantee that for any of the training examples if you put X^i at the input we exactly get d^i .

We are essentially minimizing the error between what we get at the output and d^i . By having sufficiently many examples and you know minimizing this error we hope to capture the functional relationship. We make capture the functional relationship quite well, but there is no guarantee that when you put X^i at the input you will get d^i at the output. Matter of fact we may not even want such a guarantee because there may be noise in there here training the samples.

However, when one is talking about interpolation there are many problems where I put some points and ask can you draw a curve that passes through all these points and given

that it passes all these points I want the curve to be sufficiently smooth. Some of you might have used (()) for interpolation in graphics or some such courses where especially in graphics or when I am creating some surfaces I may choose some convenient points and then ask what is a very smooth surface that passes through all these points.

That is the interpolation problem, so in interpolation problem is to find a smooth function such that their function. Suppose this smooth function is g then g of X_i is equal to d_i for i is equal to one to n . So, it should pass through all these points and be smooth, so we want a smooth function to interpolate this data that is what interpolation means. So, in the context of the network structures, we we we are discussing what does the interpolation mean.

Can I can if I use the network as a as the network structure as the function then can I have such a function to interpolate this data my network function is given by $\beta_j \phi_j(X - \theta_j)$ and θ_j are the parameters of this function. So, I am asking the question is do there exists some β_j and θ_j by note that β_j 's are scalars, where θ_j 's are vectors. We are asking is do there exists some β_j 's and θ_j 's and some function ϕ_j , so that if I calculate h of X a function of X as j is equal to 1 to p there are p hidden nodes $\beta_j \phi_j(X - \theta_j)$ for this function I can show that h of X_i is equal to d_i for i is equal to 1 to n because I want interpolation.

So, I want this ϕ_j to be sufficiently smooth, so that h of X is nicely smooth. In addition I want h of X_i is equal to d_i for the n training data, because I want to do interpolation. So, I am asking do there exist, a β_j θ_j and a ϕ_j such that this representation will satisfy my interpolation condition. This is my interpolation condition it turns out that if you take p is equal to n p is the number of hidden nodes you have n is the number of examples.

(Refer Slide Time: 12:01)

• It turns out that if we take $p = N$ and $\theta_j = X^j$, then there are many functions ϕ which can achieve such perfect interpolation.

• That is, we take

$$h(X) = \sum_{j=1}^N w_j \phi(\|X - X^j\|)$$

• We want, $h(X^i) = d^i$ for $i = 1, \dots, N$.

• This gives us N linear equations in the N unknowns, $w_j, j = 1, \dots, N$.

NPTEL

PR NPTEL course - p-21105

So, if I take p is equal to n I take as many hidden nodes at that examples. Take each of the θ_j 's to be X^j 's, so the parameter vector with the j th hidden node is simply the j th training vector then there are many functions ϕ which can achieve such a perfect interpolation. That is what I am saying is that take a network structure like this h of X take a function representation like this h of X is equal to, now I am changing β to w because this is specific function representation. So, h of X is $w_j \phi$ of X minus X^j θ_j is taken to be X^j and there are exactly n hidden nodes, the j th hidden node has parameter vector x_j .

So, if I take j is equal to one to n $w_j \phi$ of ϕ of norm X minus X^j such a function for such a function. Now, the only thing that is unknown is w_j . Now, the the result is that there are many choices for ϕ for which we can find this corresponding w_j . What is finding w_j mean? Given this function representation the only thing unknown is w_j , so we want h of X^i is equal to d^i . So, if I put h of X^1 here it simply means summation j is equal to 1 to N $w_j \phi$ of X^1 minus X^j nom that should be equal to d^1 .

So, in general summation j is equal to 1 to N $w_j \phi$ of nom X^i minus X^j should be equal to d^i . So, essentially I get N equations. What are my unknowns? Unknowns were w 's because X^i and X^j are given ϕ is given suppose if I choose a particular ϕ then it becomes n linear equations because these equations are linear in the unknown variables w_j . We essentially land up with N linear equations in N unknowns w . So, whether or not

we can find such a function which smoothly interpolates with the data is simply a matter of whether those linear equations have a solution or not.

So, essentially my interpolation condition gives me N linear equations in the N unknowns w_j . If I can solve it then I found the interpolation. Interpolation means it perfectly represents the data, so as I said if we take p is equal to N theta j equal to X_j . Now, the problem simply turns out to be whether or not a set of N linear equations have a solution.

(Refer Slide Time: 14:38)

• The N linear equations are

$$\sum_{j=1}^N w_j \phi(\|X^i - X^j\|) = d^i, \quad i = 1, \dots, N$$

• Let Φ be a $N \times N$ matrix whose $(i, j)^{th}$ element is $\phi(\|X^i - X^j\|)$. Let $W = [w_1 \ \dots \ w_N]^T$ and let $\mathbf{d} = [d^1 \ \dots \ d^N]^T$. Then we can write the above as

$$\Phi W = \mathbf{d}$$

NPTEL logo and course ID: P18 NPTEL course - p.24/105

So, what are the linear equations j is equal to 1 to N $w_j \phi$ of norm X_i minus X_j is equal to d_i , i is equal to 1 to N these are the N linear equations. If I take capital Φ to be N by N matrix whose i, j th element is this ϕ of X_i minus X_j then i . this is the i, j th element of a matrix this is the j th element of a vector summed over j . That will give me the if I if I post multiply this matrix with this vector, then this gives me the i th element of the the vector of the multiplication that is equal to i th element of d .

So, if I take a w vector to be $w_1 \ w_N$, all vectors are column vectors d to be the vector consistent d_1 to d_N and capital Φ to be N by N matrix. Whose i, j th element is given by this ϕ of X_i minus X_j then this equation in matrix form simply becomes capital $\Phi \cdot W$ is equal to d capital Φ is the N by N matrix w is a N by N matrix w is an n vector d is an n vector. So, this set of n linear equations become Φw is equal to d and Φ . Now, simply is this little ϕ a capital is simply little ϕ of $X_1 \ X_i$ minus X_j that is

the i, j th element. So, now ultimately is a choice of ϕ such that given any N vectors in \mathbb{R}^m X_1, X_2, \dots, X_N this matrix Φ form like this Φ of norm $\|X_i - X_j\|$ where ϕ of norm $\|X_i - X_j\|$ that matrix should have full rank. Then the linear equations will have a solution correct.

(Refer Slide Time: 16:26)

• If the matrix Φ is full rank (which depends on ϕ) then we get a solution.

• Some possibilities for ϕ are

$$\phi(z) = \exp\left(-\frac{z^2}{2\sigma^2}\right)$$

$$\phi(z) = (z^2 + \sigma^2)^{-\alpha}, \quad \alpha > 0$$

$$\phi(z) = (z^2 + \sigma^2)^\beta, \quad 0 < \beta < 1$$

• Note that $\phi(z) = z$ also gives us nonlinear function for $h(X)$.

NPTEL logo and course information are visible at the bottom of the slide.

So, if the matrix Φ is full rank which depends on the little ϕ function that we then we will get a solution, that is all (()) right.

(Refer Slide Time: 16:40)

• The N linear equations are

$$\sum_{j=1}^N w_j \phi(\|X^i - X^j\|) = d^i, \quad i = 1, \dots, N$$

• Let Φ be a $N \times N$ matrix whose $(i, j)^{th}$ element is $\phi(\|X^i - X^j\|)$. Let $W = [w_1 \dots w_N]^T$ and let $\mathbf{d} = [d^1 \dots d^N]^T$. Then we can write the above as

$$\Phi W = \mathbf{d}$$

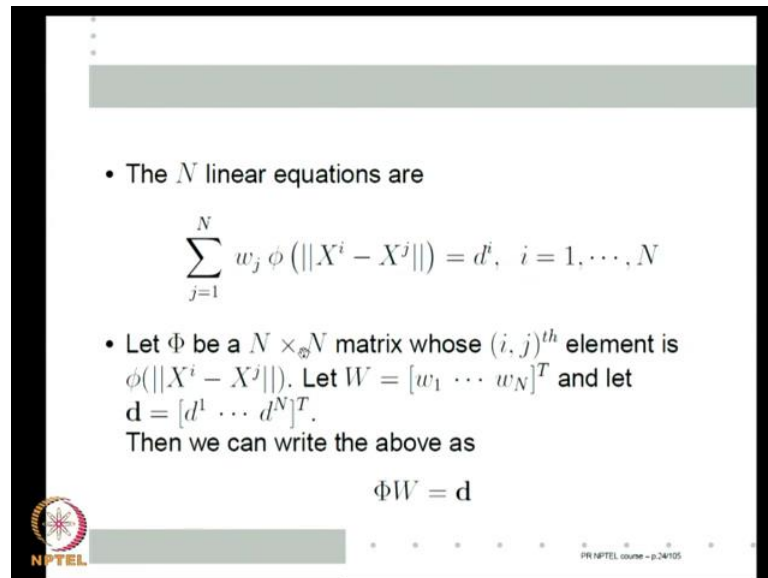
NPTEL logo and a small video inset of the speaker are visible at the bottom of the slide.

So, to be able to find the right w this set of equation should have a solution, this set of equation should have a solution is same as if the vector vector vector mapping. We are saying this, so everything boils down whether this matrix is full rank matrix full rank simply means if I take X_1 to X_N for any N and form a matrix whose i, j th element is $\phi(X_i - X_j)$ will that matrix be full rank. So, one can ask what kind of ϕ functions it will be of full rank? What kind of ϕ functions may not be right?

So, ultimately the question boils down to if I have the matrix Φ to be of full rank. What kind of function ϕ can I take? It turns out there are many many choices for ϕ we are not proving this, but here is one simplest choice ϕ of z is exponential minus z^2 by $2\sigma^2$ σ is some extra parameter for ϕ . So, $\phi(X_i - X_j)$ will be exponential minus $\|X_i - X_j\|^2$ by $2\sigma^2$ and so on.

So, this is one kind of possibility for ϕ these are what we consider later on another possibility. Once again for some parameter σ , which is positive I have forgotten to write that and some parameter α which is positive I can take $\phi(z)$ to be $z^2 + \sigma^2$ to the power minus α or I can take $\phi(z)$ to be $z^2 + \sigma^2$ to the power β where β is between 0 to 1. There are many such choices for ϕ , for all the choices of ϕ it is this matrix Φ will be full rank, which means I would be able to get this perfect interpolation as a matter of fact I just want to know that even if I take $\phi(z)$ is equal to z we still get a non-linear function $h(X)$.

(Refer Slide Time: 18:37)



• The N linear equations are

$$\sum_{j=1}^N w_j \phi(\|X^i - X^j\|) = d^i, \quad i = 1, \dots, N$$

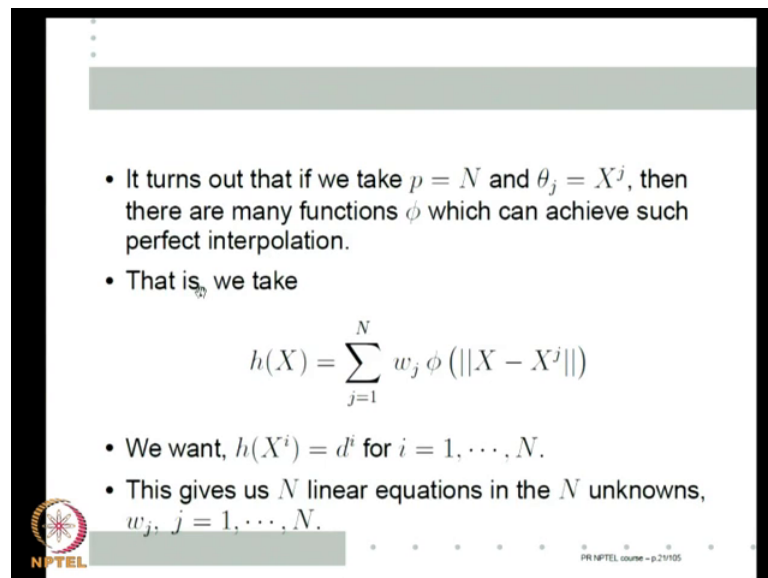
• Let Φ be a $N \times N$ matrix whose $(i, j)^{th}$ element is $\phi(\|X^i - X^j\|)$. Let $W = [w_1 \dots w_N]^T$ and let $\mathbf{d} = [d^1 \dots d^N]^T$. Then we can write the above as

$$\Phi W = \mathbf{d}$$

NPTEL logo and footer text are visible at the bottom of the slide.

Let us just go back to $h(X)$ that, so that you appreciate it, my $h(X)$ is this right otherwise my $h(X)$ is this, so even if $\phi(z)$ is equal to z , so this term becomes a norm X minus X^j .

(Refer Slide Time: 18:40)



• It turns out that if we take $p = N$ and $\theta_j = X^j$, then there are many functions ϕ which can achieve such perfect interpolation.

• That is, we take

$$h(X) = \sum_{j=1}^N w_j \phi(\|X - X^j\|)$$

• We want, $h(X^i) = d^i$ for $i = 1, \dots, N$.

• This gives us N linear equations in the N unknowns, $w_j, j = 1, \dots, N$.

NPTEL logo and footer text are visible at the bottom of the slide.

So, this $h(X)$ is equal to $\sum_{j=1}^N w_j \phi(\|X - X^j\|)$ this is a non-linear function of X because of the norm even if I take this standard nuclear norm it is still a non-linear function. So, even if I take $\phi(z)$ to be z even then I get a non-linear function.

(Refer Slide Time: 19:07)

• If the matrix Φ is full rank (which depends on ϕ) then we get a solution.

• Some possibilities for ϕ are

$$\phi(z) = \exp\left(-\frac{z^2}{2\sigma^2}\right)$$
$$\phi(z) = (z^2 + \sigma^2)^{-\alpha}, \quad \alpha > 0$$
$$\phi(z) = (z^2 + \sigma^2)^\beta, \quad 0 < \beta < 1$$

• Note that $\phi(z) = z$ also gives us nonlinear function for $h(X)$.

NPTEL PR NPTEL course - p.30105

But anyway these are phi z is equal to z also works, so for the interpolation, but these are some of the more often used functions radial basis functions for interpolation.

(Refer Slide Time: 19:25)

• We choose any of these ϕ and define

$$h(X) = \sum_{j=1}^N w_j \phi(\|X - X^j\|)$$

• Then we can find $w_j, j = 1, \dots, N$ to satisfy $h(X^i) = d^i$.

• Called perfect interpolation property

• We can use a similar representation to have an alternate 3-layer network structure.

NPTEL PR NPTEL course - p.34105

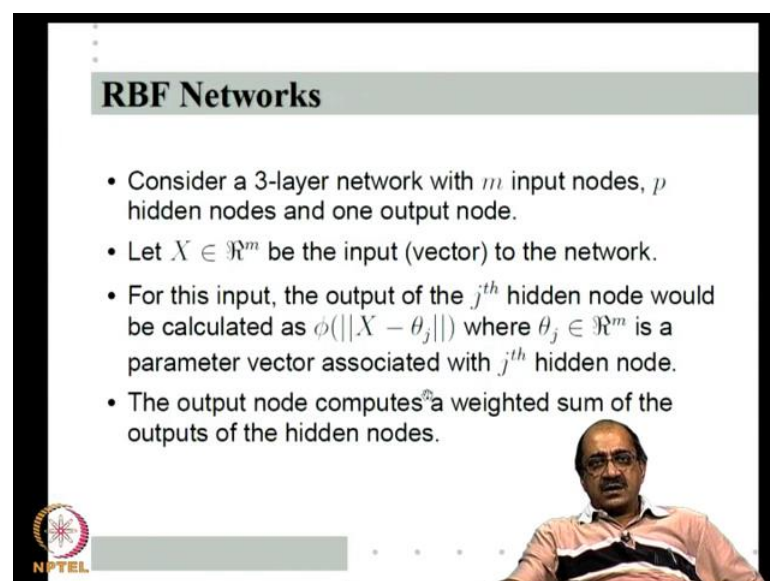
So, if we choose any of these phi and then define my h of X is equal to j is equal to 1 to N w j of phi X minus X j. Then the result is that I can satisfy, I can find w j's such that h of X i is equal to d i i can exactly interpolate the data. Of course, exact interpolation of the data is of not much concern to us. In in learning a classifier or learning a regression function just learning the function value on the training data is not enough. Then I can

just simply remember the training data, but that I can find a smooth function, which exactly represents the training data is always an interesting thing.

So, we will... one can think of a network structure of similar kind, of course I may not want to put as many hidden nodes as their examples as we already know in many of these networks the $v \times c$ dimension is of the order of the number of nodes. So, if I put as many examples as a number of nodes then obviously I am not going to learn very well. So, I may have hidden nodes, which are much less than the number of examples hence I would not get the perfect interpolation, but still I can think of this kind of a structure then I do not take θ_j to be X_j .

But I take θ_j to be some other parameter vector, which is to be learnt, but because if I put N hidden nodes they have perfect interpolation capability. The hope is that this kind of functions if I put some p here and some θ_j here can represent almost any function. The fact that when I put as many hidden nodes as there are examples and take the θ_j 's to be x_j 's. Then I can always find w_j 's, so that $h(X) = d$, this property is called the perfect interpolation property. So, there is a whole lot of ϕ functions, whole lot of radially symmetric functions, which they are called radially symmetric basis functions, which have which have this perfect interpolation property. The idea is that we will use similar representation to have an alternate three layer network structure.

(Refer Slide Time: 21:31)



RBF Networks

- Consider a 3-layer network with m input nodes, p hidden nodes and one output node.
- Let $X \in \mathbb{R}^m$ be the input (vector) to the network.
- For this input, the output of the j^{th} hidden node would be calculated as $\phi(\|X - \theta_j\|)$ where $\theta_j \in \mathbb{R}^m$ is a parameter vector associated with j^{th} hidden node.
- The output node computes a weighted sum of the outputs of the hidden nodes.

NPTEL

The slide also features a video inset of a man in a light-colored shirt speaking, and a small NPTEL logo in the bottom left corner.

So, we will call them RBF networks, so it is like the following we have a three layer network with m input nodes p hidden nodes and one output node as earlier. So, X is the input vector to the network X is a m vector. Now, we no longer want our hidden nodes to be doing some specific weighted sum, they can they compute some non-linear function in the following sense. So, given X as the input network the output of the j th hidden node is calculated as ϕ of X minus θ_j where θ_j is some parameter θ_j is also an $r \times m$ some parameter which is associated with the j th hidden node and ϕ is a radial basis function that we choose.

So, the output of the j th hidden node is calculated as ϕ of X minus ϕ of X norm of X minus θ_j where θ_j is a parameter vector. So, the output node simply computes a weighted sum of the outputs of the hidden nodes. Output node is like a layer it has a linear activation function it simply computes a weighted sum of the output of the hidden nodes.

(Refer Slide Time: 22:40)

• Now, the output of the network is calculated as

$$h(X) = \sum_{j=1}^p w_j \phi(\|X - \theta_j\|)$$

where w_j is the weight connecting the j^{th} hidden node to the output node.

- Such a network is called a Radial Basis Function (RBF) network.
- The hidden nodes here are sometimes called RBF nodes.

NPTEL logo and footer text: PR NPTEL course - p.41/05

So, this gives us exactly the kind of function we are calculated. So, this is so the output of the network can be represented as $\sum_{j=1}^p w_j \phi(\|X - \theta_j\|)$ where w_j and θ_j are the parameters. w_j is the weight connecting the j th hidden node to the output node and θ_j is the parameter vector associate with the j th hidden node. Such a network is called a radial basis function network, so a radial basis function network is a three layer network. Whose output is computed like this a three layer network with p

nodes or p hidden nodes w_j is the weight this is a three layer network with only output node.

So, we are learning real valued function as there is only one output node there is only 1 subscript on w . So, w_j is the weight connecting the j th hidden node to the output node. $\phi_j(X - \theta_j)$ is taken to be the output of the j th hidden node, where θ_j is the parameter vector of the j th hidden node and X is the input to the network. A network that is whose output is represented like this is called a radial basis function network. These hidden nodes whose output is $\phi_j(X - \theta_j)$ are often called the RBF nodes; radial basis function nodes that they implement a function that has that has radial symmetry.

(Refer Slide Time: 24:13)

• Our representation is

$$h(X) = \sum_{j=1}^p w_j \phi(\|X - \theta_j\|)$$

• We are expressing the desired function using basis functions $\phi_j(X) = \phi(\|X - \theta_j\|)$.

• For all X on a hypersphere centered around θ_j we get the same value for $\phi_j(X)$.

• Hence these are called Radial Basis functions and thus the name RBF networks.

NPTEL

PR: NPTEL course - p45/105

We can see this like this, see the representation of that we are using in this class of models if j is equal to 1 to p w_j of $\phi_j(X - \theta_j)$. So, we can actually think of this as w_j into some basis function the basis function itself has some parameters. But anyways w_j into some basis function, so we are expressing the desired function using sum basis functions. It is just a linear sum sum of basis functions, so I if I think of this as the j th basis function. Now, it is called the j th basis function ϕ_j tilde.

So, our representation is that we want to express the desired function as a linear sum of basis functions. The J th basis function is given by $\phi_j(x - \theta_j)$. Now, if I view it like this my j th basis function has radial symmetry because for all X . A hyper sphere

centred around θ_j ϕ_j \tilde{X} is same because ϕ_j \tilde{X} is ϕ of norm X minus θ_j ϕ is a function. So, as long as norm X minus θ_j is the same value ϕ of norm X minus θ_j will be same value norm X minus θ_j will be the same value for all X , which are in a hyper sphere centred around θ_j .

Hence, for all X on a hyper sphere centred around θ_j centred on θ_j we get this same value of ϕ_j \tilde{X} . So, all my basis functions have radial symmetry and hence I call this network radial basis function networks. Because essentially I am representing the desired function in as the basis function expansion, where the basis functions themselves are radially symmetric. So, that is why we call them radial basis functions a radial basis function networks.

(Refer Slide Time: 26:03)

Gaussian RBF Networks

- Suppose we take $\phi(z) = \exp\left(-\frac{z^2}{2\sigma^2}\right)$.
- Then the network is called Gaussian RBF network.
- For a Gaussian RBF network, the output is

$$y = \sum_{j=1}^p w_j \exp\left(-\frac{\|X - \theta_j\|^2}{2\sigma^2}\right)$$

where $w_j, \theta_j, j = 1, \dots, P$ (and σ) are parameters of the network.

NPTEL PRE NPTEL course - p 49/105

Now, if we take our radially symmetric basis function to be the Gaussian function ϕz is exponential minus z square by 2 sigma square, then the resulting network structure is called a Gaussian RBF network. These are the most often used RBF networks in classification and regression context. So, this is the, this is the only kind of the most of the time this is the only kind of radial basis function one uses.

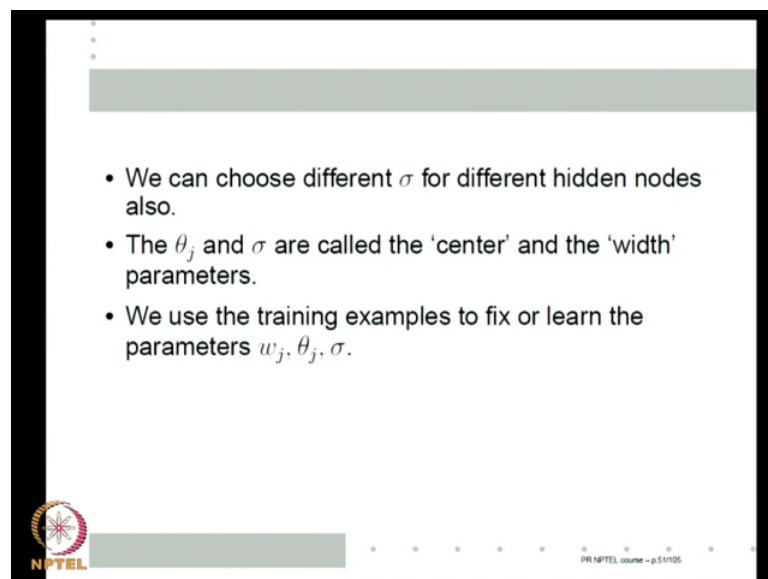
So, such networks are called Gaussian RBF networks, so in a Gaussian RBF network what will be the output of the network is w_j exponential X minus θ_j whole square by 2 sigma square w_j and θ_j are parameters. Of course, sigma is also a parameter of

the network, so to actually learn the network I have to somehow fix w_j , θ_j and σ .

But for a Gaussian RBF network the output can be represented like this as a weighted sum of exponentials. So, if j is equal to 1 to p $w_j \exp(-\frac{\|X - \theta_j\|^2}{2\sigma^2})$, this is the structure of the Gaussian RBF network. Of course, in general the σ can also be a function of j . So, when we write $\phi(\|X - \theta_j\|)$ ϕ has to be a function of that $\|X - \theta_j\|$ there can be other parameters in that function. That parameter could be σ and because we are inside a j summation we can even make it make that a function of j .

So, different hidden nodes can have different σ 's, but often it is very difficult to know on what basis to learn those σ 's. So, very often you take this same σ we will come back to that later. θ_j 's are course are parameters to be learnt and w_j 's are also parameter term. So, w and θ_j are essentially parameters of the network.

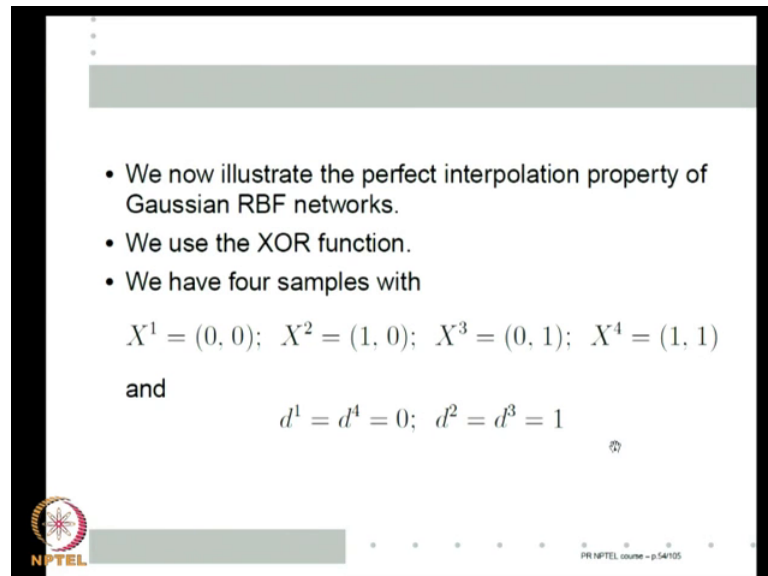
(Refer Slide Time: 28:02)



We can choose different σ for different hidden nodes also often we call θ_j as the centre of the j th hidden node and σ is the width of the j th hidden node. See the j th basis function is a Gaussian centred at θ_j and with a variance of σ^2 . Hence we say θ_j is the centre of the j th hidden node or centre of the j th basis function. σ or σ_j is the width of the j th basis function or j th hidden node.

So, θ_j and σ_j are normally referred to as the centre and width parameters of the j -th hidden node. So, essentially we have to use the training examples to fix or learn or whatever you call the parameter w_j , θ_j , so ultimately that we will see may not be in this class we will see the learning algorithm in next class. But essentially to represent a specific function we have to fix w_j 's and θ_j 's.

(Refer Slide Time: 29:11)



• We now illustrate the perfect interpolation property of Gaussian RBF networks.

• We use the XOR function.

• We have four samples with

$$X^1 = (0, 0); X^2 = (1, 0); X^3 = (0, 1); X^4 = (1, 1)$$

and

$$d^1 = d^4 = 0; d^2 = d^3 = 1$$

NPTEL

PR NPTEL course - p.54/105

Now, how we will it from training data we will see later on. Now, because we said that we have approached the RBF networks from the point of view of perfect interpolation. So, let us look take the Gaussian RBF networks, which is said they are are the most often used RBF networks in pattern recognition regression. So, we will take the Gaussian RBF networks and illustrate what we mean by the perfect interpolation property these are very interesting thing to go through this exercise at least once.

To do that we will use the same non-linear function we used for the feed forward networks, to show how feed forward networks can represent non-linear function we looked at XOR function because we know that XOR as a classifier or a function cannot be represented by a linear model. So, we need a non-linear model for it, so we use the same XOR function to illustrate the perfect interpolation property of the Gaussian RBF networks.

Now, for XOR function, so what is the perfect interpolation property you give me samples X_i and I have to create a smooth function such that 1 input is X_i output will

be d_i exactly not approximately. It has to be exactly equal to d_i what is the XOR function it has four sample points, so our training sample has four data X_1, X_2, X_3, X_4 and these are the vectors $(0, 0, 1, 0), (0, 1, 1, 1)$. Because an XOR function on X_1 and X_4 it takes value 0 on X_2 and X_3 it takes value 1.

So, I have d_1 and d_4 equal to 0 d_2 and d_3 equal to 1. So, I have four examples $X_1, d_1, X_2, d_2, X_3, d_3, X_4, d_4$ where X_1, X_2, X_3, X_4 are given by this and d_1, d_2, d_3, d_4 are given by this. This is the sample given to me, now I have to find a Gaussian RBF network which exactly reproduces this function that is the perfect interpolation property.

(Refer Slide Time: 31:14)

- For perfect interpolation, we need four hidden nodes.
- Let us take Gaussian RBF net with $\sigma^2 = 0.5$.
- Then the output of network is

$$y = h(X) = \sum_{j=1}^4 w_j \exp\left(-\frac{\|X - X^j\|^2}{2\sigma^2}\right)$$

- Now we need to find $w_j, j = 1, 2, 3, 4$, to satisfy

$$h(X^i) = d^i, \quad i = 1, 2, 3, 4.$$

So, for perfect interpolation we already know we need four hidden nodes because there are four samples that this capital N is equal to 4, so I need four hidden nodes. So, let us take all the Gaussian we will let us take a Gaussian RBF network with sigma square is equal to half. Why sigma square is equal to half? In the expression I get 2 sigma square in the denominator inside the exponent. If I take sigma square is equal to half writing my equation becomes simple because in the denominator becomes small, almost any sigma square will work.

So, we will take sigma square is equal to half if I take sigma square is equal to half my output of the Gaussian RBF network simply becomes $w_j \exp(-\|X - X_j\|^2)$ whole square. Normally I have $\|X - X_j\|^2$ by $2\sigma^2$ I made $2\sigma^2$ is equal to 1 by my choice of sigma square. So, this is my output of the

network. Now X_j 's are given to me, so if I put X_i equal to X_i the output should be d_i . So, I have to find w_j 's, so that h of X_i is equal to d_i where X_i and d_i are given by this.

So, we need to find w_j there are 4 w_j 's j is equal to 1 2 3 4 to satisfy X_i is equal to d_i . What is...? Suppose I have put X_1 , what is my X_1 ? X_1 is both components 0 and d_1 is equal to 0. So, for example, what will I get? Now, if I put X is equal to X_1 I will be calculating norm of X_1 minus X_1 X_1 minus X_2 X_1 minus X_3 X_1 minus X_4 . Essentially because all the vectors are 0 1 vectors this norm will simply tell me how many components they are same. So, if I take norm between norm square between 0 0 and 1 1 I get $1 - 0$ whole square plus $1 - 0$ whole square I will get 2. If I take 0 0 and 0 1 1 component is same only the other component will give me 1.

So, essentially if I take X_1 with X_1 the norm will be 0 with X_2 and X_3 in the norm will be one with X_4 the norm will be two whereas, the norm square will be 1. That is how I will get this. So, essentially my equations will be like this I hope one can see. So, with w_1 I get exponential minus X_1 minus X_1 norm that is 0. So, this is to emphasize that instead writing 1 I wrote e power minus 0 into W_1 . If I put the second term will be w_2 exponential minus X_2 X_1 . I am taking X equal to X_1 X_1 minus X_2 because j is equal to 2 X_2 is equal to 1 0. So, X_1 is 0 0. So, the norm square will be 1, so I get e power minus 1.

(Refer Slide Time: 34:09)

• This gives us the following set of linear equations:

$$e^{-0} w_1 + e^{-1} w_2 + e^{-1} w_3 + e^{-2} w_4 = 0$$

$$e^{-1} w_1 + e^{-0} w_2 + e^{-2} w_3 + e^{-1} w_4 = 1$$

$$e^{-1} w_1 + e^{-2} w_2 + e^{-0} w_3 + e^{-1} w_4 = 1$$

$$e^{-2} w_1 + e^{-1} w_2 + e^{-1} w_3 + e^{-0} w_4 = 0$$

NPTEL PR NPTEL course - p.59/105

Similarly, with w_3 also I will get e^{-1} because w_4 is 1 and I am choosing X_1 , which is 0 I get e^{-2} and the output should be 0 similarly, for everything else.

(Refer Slide Time: 34:28)

• For perfect interpolation, we need four hidden nodes.

• Let us take Gaussian RBF net with $\sigma^2 = 0.5$.

• Then the output of network is

$$y = h(X) = \sum_{j=1}^4 w_j \exp(-\|X - X^j\|^2)$$

• Now we need to find w_j , $j = 1, 2, 3, 4$, to satisfy

$$h(X^i) = d^i, \quad i = 1, 2, 3, 4.$$

NPTEL

PRE NPTEL course - p.59/105

For example if I have taken this h of X_2 , so this 1 will become X_2 X_2 is 1 0 . So, if this was 0 0 there norm square will be one if it is 1 0 the norm square will be 0 if it is 0 1 norm square will be 2 1 1 norm square norm square will be 1 . So, once again I get one $e^{-0.2}$ e^{-1} e^{-2} and so on. So, I get e^{-1} e^{-0} e^{-2} e^{-1} and so on.

(Refer Slide Time: 34:52)

• This gives us the following set of linear equations:

$$\begin{aligned}e^{-0} w_1 + e^{-1} w_2 + e^{-1} w_3 + e^{-2} w_4 &= 0 \\e^{-1} w_1 + e^{-0} w_2 + e^{-2} w_3 + e^{-1} w_4 &= 1 \\e^{-1} w_1 + e^{-2} w_2 + e^{-0} w_3 + e^{-1} w_4 &= 1 \\e^{-2} w_1 + e^{-1} w_2 + e^{-1} w_3 + e^{-0} w_4 &= 0\end{aligned}$$

• Subtracting first eqn from the fourth one

$$(w_1 - w_4) - (w_1 - w_4) e^{-2} = 0 \Rightarrow w_1 = w_4$$

• Similarly from the other two eqns, $w_2 = w_3$.

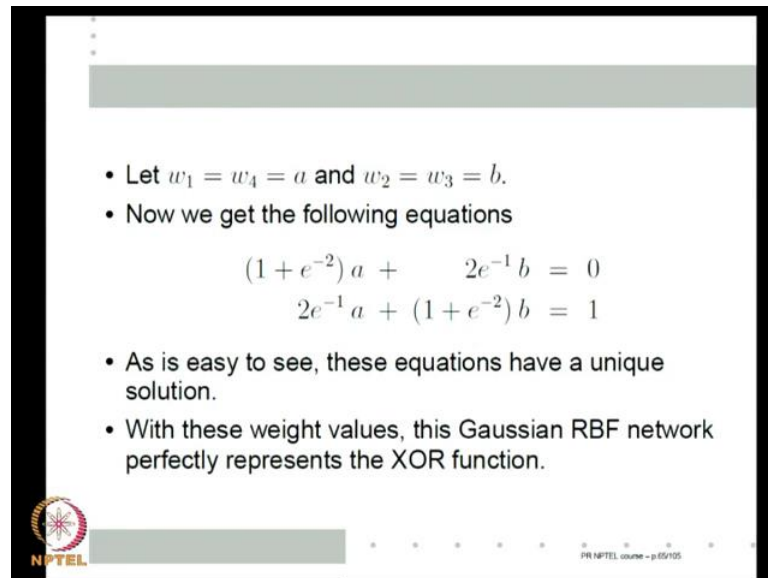
NPTEL

PR NPTEL course - p-EE105

So, a little thought will tell you that these will be the four equations. Now, we have to solve them, yeah actually we do not have to solve if a four equations system. If we look at this look at the first and the last equations e^0 this is 1 this is $w_1 e^{\text{power minus } 2} w_4$ and it has $e^{\text{power minus } 2} w_4$ and $e^{\text{power minus } 0} w_4$. The rest of the 2 terms are same $e^{\text{power minus } 1} w_2$ $e^{\text{power minus } 1} w_3$ $e^{\text{power minus } 1} w_2$ $e^{\text{power minus } 1} w_4$.

So, if I subtract one equation from other I get w_1 minus w_4 these two are and $e^{\text{power minus } 2}$ into w_1 minus w_4 with a minus $(())$. So, this essentially shows me that w_1 is equal to w_4 this is the symmetry similarly, these inner two equations will show me that w_2 is equal to w_3 . So, essentially even though there is a 4 4 unknown system we already know w_1 is equal to w_4 w_2 is equal to w_3 . So, we can reduce it to that two equations in two unknowns.

(Refer Slide Time: 36:10)



• Let $w_1 = w_4 = a$ and $w_2 = w_3 = b$.

• Now we get the following equations

$$(1 + e^{-2})a + 2e^{-1}b = 0$$
$$2e^{-1}a + (1 + e^{-2})b = 1$$

• As is easy to see, these equations have a unique solution.

• With these weight values, this Gaussian RBF network perfectly represents the XOR function.

NPTEL

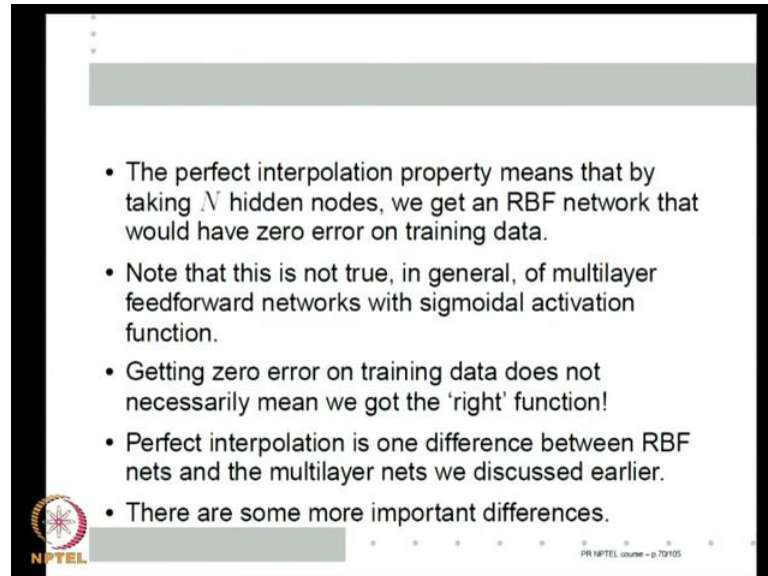
PR NPTEL course - p.05/105

If we do that reduction take w_1 is equal to w_4 is equal to a and w_2 is equal to w_3 is equal to b and substitute in these four equations then we get this, that is also easy enough to see w_1 is equal to w_4 is equal to a . So, it will be one plus e power minus 2 into a these 2 are same 2 be e power minus 1 is equal to 0 and so on. Now, 1 power e power minus 2 into a plus 2 e power minus 1 to be is equal to 0 similarly the other one. Now, it is very easy to see that this system has unique solution.

Because if you take the on the system 1 plus e power minus 2 whole square minus 4 e power minus 1, which is 1 minus e power minus 2 whole square, which is always positive. Hence this will this will have a a solution as it is easy to see these equations have a unique solution these equations have unique solution means I can find a and b to satisfy this. If I can find a and b to satisfy this and then I take w_1 is equal to w_4 is equal to a and w_2 is equal to w_3 is equal to b . Then I can satisfy these four equations, satisfying these four equations means I can satisfy these four equations that is my perfect interpolation property.

So, in this set of representation I can find values w_1 w_2 w_3 w_4 , so that for the given data XOR data X_1 X_2 X_3 X_4 with the only $(())$ 4 I can ensure that h of X_i is equal to d_i . That is my perfect interpolation property, so with these weight values this Gaussian RBF network perfectly represents the XOR function perfectly in the sense for those four points.

(Refer Slide Time: 38:01)



The slide contains a list of five bullet points. The first point states that the perfect interpolation property of an RBF network with N hidden nodes results in zero error on training data. The second point notes that this property is not generally true for multilayer feedforward networks with sigmoidal activation functions. The third point clarifies that zero error on training data does not guarantee the 'right' function. The fourth point identifies perfect interpolation as a key difference between RBF networks and other multilayer networks. The fifth point mentions that there are other important differences. The slide also features the NPTEL logo in the bottom left corner and the text 'PR NPTEL course - p.70105' in the bottom right corner.

- The perfect interpolation property means that by taking N hidden nodes, we get an RBF network that would have zero error on training data.
- Note that this is not true, in general, of multilayer feedforward networks with sigmoidal activation function.
- Getting zero error on training data does not necessarily mean we got the 'right' function!
- Perfect interpolation is one difference between RBF nets and the multilayer nets we discussed earlier.
- There are some more important differences.

The output is exactly equal to either 0 or 1 as needed. So, the perfect interpolation property means that by taking n hidden nodes we get an RBF network that would have 0 errors on the training data. If we think of this as learning with those four training patterns we are saying on the training patterns I get 0 errors. So, first let us understand that this is in general not true of multilayer feed forward networks. When we try to represent XOR, we are not talking about learning here just representation.

We chose the weights here right by whatever means we want. We are not saying whether a particular learning algorithm will learn the right weights. If I just say can I find weights, so that if a network will correctly represent this function. In general this property is not there for multilayer feed forward networks. If you remember as we as when we constructed the XOR representation on the multilayer feed forward network to represent XOR we are only looking for a good approximation.

In the sense we say that there are these weights, which are such that if the output should be 1 the output would be close to 1 in as much as the input of the sigmoid is greater than half or greater than 3 by 4 or whatever. Similarly, when the output should be 0, it would be close to 0, so we will get good approximation, but we will not get exact function value. As a matter of fact if I use sigmoidal activation, I will never if I use sigmoidal activation for the output.

Even otherwise I cannot always manage to get the perfect interpolation value. On the other hand the radial basis function networks can always no matter what the input you give if I take as many hidden nodes as their training examples I can get perfect interpolation as I already mentioned getting 0 error on the training data does not necessarily mean we go to the right function. There are many we have seen in when we discussed the statistical learning theory that we can get trivial functions which get 0 error on the training data, but otherwise they are useless. So, but all the same this perfect interpolation property is interesting and the multilayer feed forward networks do not have this.

So, in that sense there are a class of networks say distinctly different from the multilayer feed forward networks. The perfect interpolation property is one difference between the between the RBF networks and the multilayer feed forward networks we discussed earlier because there are many other differences. Let us look at these two networks, now in contrast. So, there are many other important differences between what we considered earlier whom we will call multilayer feed forward networks.

Whenever we say multilayer feed forward networks, we mean the earlier networks were each the output of each node is obtained by taking a weighted sum of inputs. Passing through it an activation function such as a tan hyperbolic or sigmoid and the RBF networks is what we are seeing today. So, one difference between these two class of networks is this perfect interpolation property.

(Refer Slide Time: 41:18)

• The multilayer feedforward networks we saw use the representation

$$y = \sum_{j=1}^p \beta_j f \left(\sum_{i=1}^m w_{ij} x_i + b_j \right)$$

• The Gaussian RBF networks use the representation

$$y = \sum_{j=1}^p w_j \exp \left(- \frac{\|X - \theta_j\|^2}{2\sigma^2} \right)$$

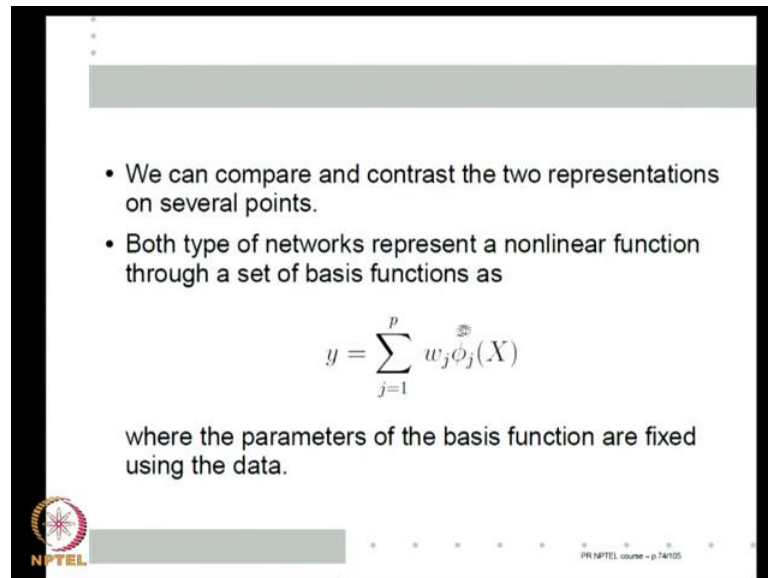
NPTEL

PR NPTEL course - p. 72105

There are other differences we will see both the, I mean what way they are same in what way they are different. Now, we can write where a three layer network using linear output node the output of the, is given by this where f is the sigmoid which is the activation function of the hidden nodes. We will take a linear output node because in RBF also we are taking a linear output node. Now, the RBF has the representation, which is w_j and instead of f of this, let us say we take Gaussian RBF this is this is what the RBF does.

So, I can think of this as the j th basis function some ϕ_j of X and this is the j th basis function once again some ϕ_j of X . So, both of them have a very similar kind of representation, so we can compare in contrast the two representations on several points. So, keep these two representations in mind, both of them have a syntactically similar thing, this is also a weighted sum of some non-linear function this is also a weighted sum of some non-linear functions.

(Refer Slide Time: 42:26)



• We can compare and contrast the two representations on several points.

• Both type of networks represent a nonlinear function through a set of basis functions as

$$y = \sum_{j=1}^p w_j \phi_j(X)$$

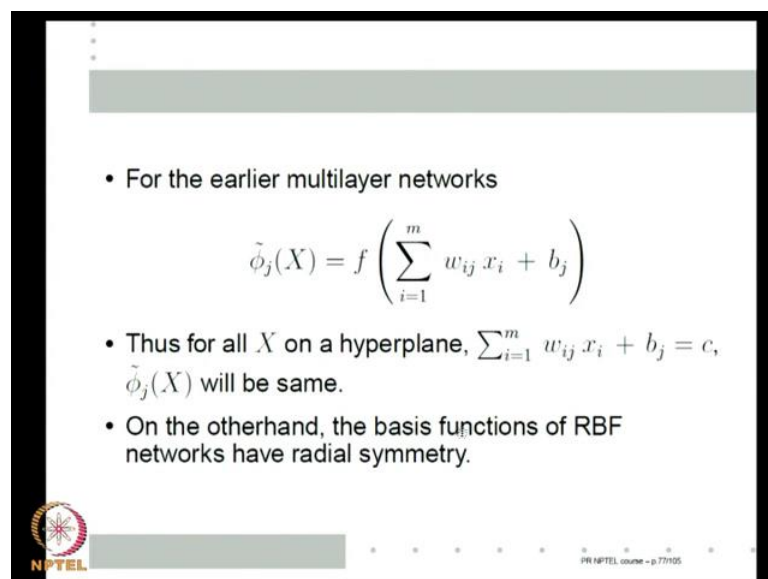
where the parameters of the basis function are fixed using the data.

NPTEL

PR NPTEL course - p.74105

Both of them are essentially doing basis function expansions. Both type of networks represent a non-linear function through a set of basis functions. Both of them are essentially writing the output as summation j is equal to 1 to p $w_j \phi_j(X)$. Also compared to the earlier fixed basis function expansions, which are what we've used at the linear models these $\phi_j(X)$'s have parameters, which themselves are learnt using the data. So, actually the basis function themselves are fixed using the data it is not a fixed basis function expansion unlike the linear models both the networks.

(Refer Slide Time: 43:00)



• For the earlier multilayer networks

$$\tilde{\phi}_j(X) = f\left(\sum_{i=1}^m w_{ij} x_i + b_j\right)$$

• Thus for all X on a hyperplane, $\sum_{i=1}^m w_{ij} x_i + b_j = c$, $\tilde{\phi}_j(X)$ will be same.

• On the otherhand, the basis functions of RBF networks have radial symmetry.

NPTEL

PR NPTEL course - p.77105

If this is the $\phi_j(X)$ for this model this is the $\phi_j(X)$ for this model both of them have parameters inside these w 's and b 's here and the θ 's and θ 's and and σ here and these parameters can, of course be fixed based on the data. So, both of them use a basis function expansion where unlike the linear model that use a fixed basis both of them use basis functions that are themselves determined based on the data. However the nature of the basis functions is different, the multilayer feed forward networks this is the basis function.

So, what does this mean if I take X on a hyperplane that is if I take all vectors X that satisfy this equation for a given constant c $w_j X_i + b_j$ is equal to c . For a particular j th node I take all its w_j , so and I look for all X 's that are $w_j X_i + b_j$ is equal to c . So, if I call this w_j 's components are some vector called w_j . Then this is $w_j^T X$ is equal to some constant that is nothing but a hyperplane. So, essentially the j th basis function have the same value for all X lying on a hyperplane.

Whereas, the for the RBF networks the base j th basis function has the same for all X lying on sphere. So, the symmetry structures of the basis functions are different in one case. The basis functions will have this for any given constant the basis function will have value that constant for all X lying on a hyperplane here, right this is for all x lying on a sphere. So the the the the two kinds of basis functions have different types of symmetries.

(Refer Slide Time: 44:56)

• The output of the Gaussian RBF network is

$$y = \sum_{j=1}^p w_j \exp\left(-\frac{\|X - \theta_j\|^2}{2\sigma^2}\right)$$

• If we change a specific w_j then for X far apart from that θ_j , the function value is not affected much.

• Thus, we can say different weights essentially take care of different regions in the feature space.

NPTEL PRE NPTEL course - p.00105

Next the output of the Gaussian RBF network is written like this, now because of the exponential essentially of X is too far from θ_j for a particular X , which is too far from θ_j far or near is in terms of in the units of σ right. Too far in the terms of in terms of σ X minus θ_j is large then this exponential we have very very small value. Hence that particular j th term would not contribute much to the function. That is which means suppose I change a particular w_j we change one particular w_j , then for all X which are far away from this particular θ_j the function value will not change.

The function value will change appreciably only for X 's, which are near to this θ_j . So, we can really think of suppose σ 's are very very small then each term essentially contributes to X 's close to its own θ_j 's. So, it is as if in this expansion each particular weight is essentially taking care of the function value near to its own θ_j . So, if we take θ_j points in \mathbb{R}^n and draw nice circles around these θ_j we can say the corresponding w_j is essentially affecting the function value mostly only in that circle that is the region around the θ_j .

So, we can say different weights essentially take care of different regions of the feature space. The weight w_j is essentially taking care of value of the function for all X , which are close to θ_j that is where that w_j plays maximum role. This particular is true if σ is small this is quite different from the multilayer feed forward network given the multilayer feed forward networks structure.

(Refer Slide Time: 46:55)

- For the earlier multilayer networks

$$\tilde{\phi}_j(X) = f\left(\sum_{i=1}^m w_{ij} x_i + b_j\right)$$

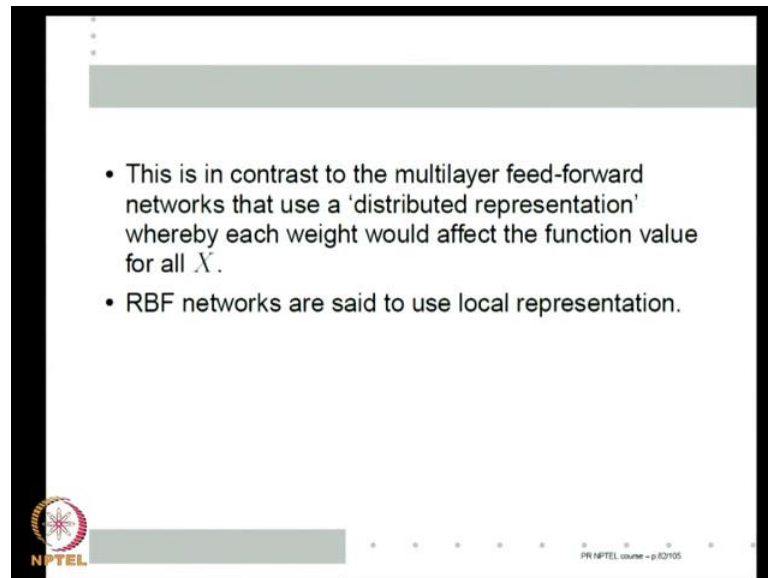
- Thus for all X on a hyperplane, $\sum_{i=1}^m w_{ij} x_i + b_j = c$, $\tilde{\phi}_j(X)$ will be same.
- On the otherhand, the basis functions of RBF networks have radial symmetry.

NPTEL

PRI NPTEL course - p.77105

Because it is this kind of structure the w_{ij} 's are all mixed up. Each w_{ij} can affect the function value of many X 's. So, I cannot say any w is contributing to represent the function on a particular X or a particular set of X 's. So this is in contrast to multilayer feed forward networks, which can be thought of using as the distributed representation.

(Refer Slide Time: 47:23)

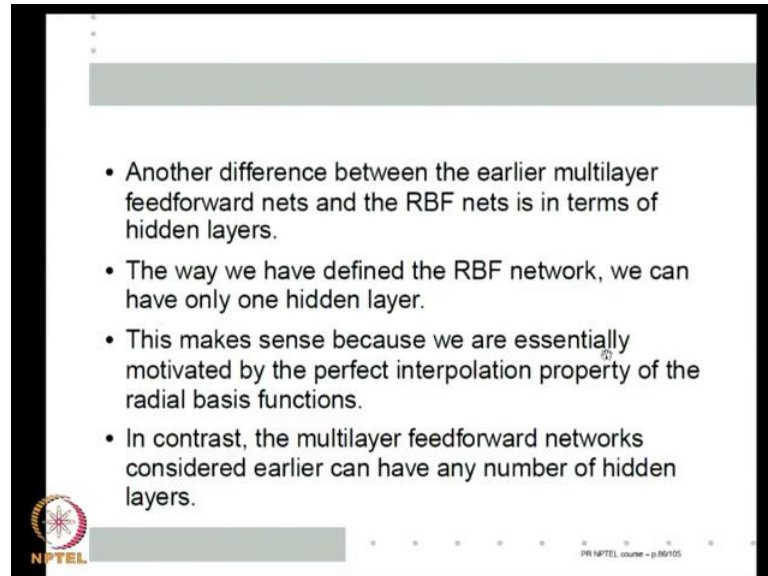


Whereby, each weight would affect the function value for all X , whereas in the RBF networks, it is kind of a local representation. Because each weight is essentially affecting the function in a certain range around its own θ_j , so that is a very interesting thing. Because which essentially means if I have sufficiently many hidden nodes. So, I can actually tweak these σ 's and the θ_j 's, so I can put θ_j 's at some representative points. So, around these θ_j by tweaking w_j well I can get fairly good accuracy for the function. So, in this sense ultimately become a little easier to understand the function represented by an RBF network because it uses local representations.

So, this is another difference right, so the reference we have seen, so far are one is this perfect interpolation property only RBF networks have it. The RBF networks and feed forward networks well both use basis function expansions. The basis functions are different in the sense they have different kinds of symmetries. Third is that we can think

of the multilayer feed forward networks as using distributed representation, whereas RBF networks use local representations.

(Refer Slide Time: 48:53)



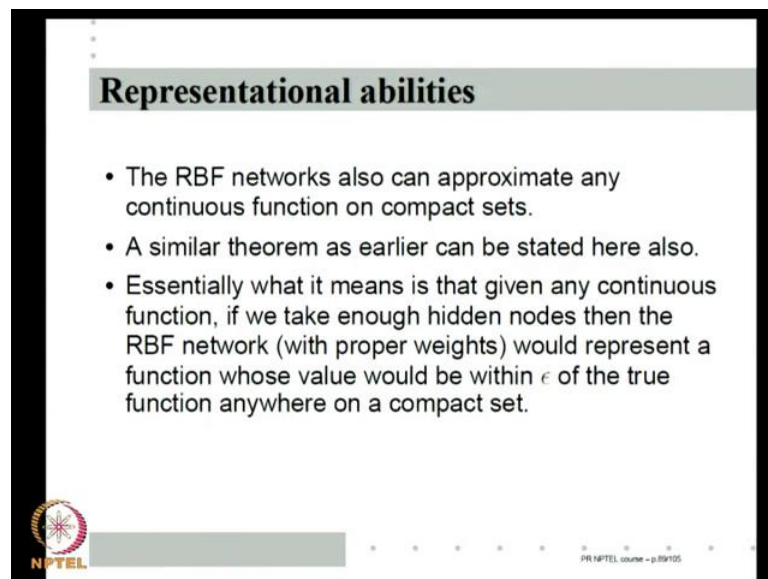
Another difference between the earlier multilayer feed forward networks and the RBF networks is in terms of hidden layers. The way we have written the RBF network it makes no sense to have more than one hidden layer, we do not even know how to have more than one hidden layer. Because we are saying each hidden node has some kind of a ϕ of norm X minus θ_j value as its output. Then we are just combining the output of the hidden layer.

So, by its very definition RBF network can have only one hidden layer. Whereas, as we saw in the multilayer feed forward network the outputs of a node can be viewed as input of some other node and to any such any number of levels like that. So, in a in a in a multilayer feed forward network any number of hidden layers are possible. In the RBF network essentially we have only one hidden layer. Because we have started with the special kind of interpolation property of radial basis function networks, which which comes for your expansion, which is essentially a single hidden layer network.

As you can, as you may remember even in the multilayer feed forward networks, we saw that theoretically one hidden layer is enough; given one hidden layer we can represent any continuous function on compact sets. In that sense even feed forward networks having one hidden layer are good enough, so having only one hidden layer does not

mean that we are compromising on the representational abilities. But in these multilayer networks we can have multiple hidden layers and last class we have discussed some reasons why we did not want to have multiple hidden layers but, the RBF networks do not make sense to have multiple hidden layer. We can have only one hidden layer in RBF networks.

(Refer Slide Time: 50:46)



The slide is titled "Representational abilities" and contains the following text:

- The RBF networks also can approximate any continuous function on compact sets.
- A similar theorem as earlier can be stated here also.
- Essentially what it means is that given any continuous function, if we take enough hidden nodes then the RBF network (with proper weights) would represent a function whose value would be within ϵ of the true function anywhere on a compact set.

The slide also features the NPTEL logo in the bottom left corner and the text "© NPTEL course - p. RBF105" in the bottom right corner.

This is the kind of comparison between the two networks. Now, another comparison we can say is that both networks are as good for representing function, both of them represent in their own way good classes of parameterized non-linear functions. So, which means just like what we showed for the multilayer feed forward networks RBF networks also approximate any continuous function on compact sets. What does that mean? We can write a similar theorem for RBF networks I would not write it again because it will exactly be the same, what it means is give me any function on a compact set.

Then I can find θ_j 's and w_j 's such that a function represent as an RBF network will be no more than epsilon away from the true value of the function. That is all the theorem will state, so in intuitively given any continuous function, if we take n of hidden nodes then the RBF network with proper weights. That is proper w_j 's and the θ_j 's would represent a function whose value would be within epsilon of the true function value anywhere on a compact set.

This is essentially the result that we have proved for multilayer feed forward networks also. Of course, we have not proved we only stated. Here also we are only stating, but because RBF networks use local representation here at least we can intuitively in a hand waving manner see why it may happen if you take Gaussian RBF networks essentially this is the output.

(Refer Slide Time: 52:21)

The slide contains the following text:

- The output of the Gaussian RBF network is

$$y = \sum_{j=1}^p w_j \exp\left(-\frac{\|X - \theta_j\|^2}{2\sigma^2}\right)$$

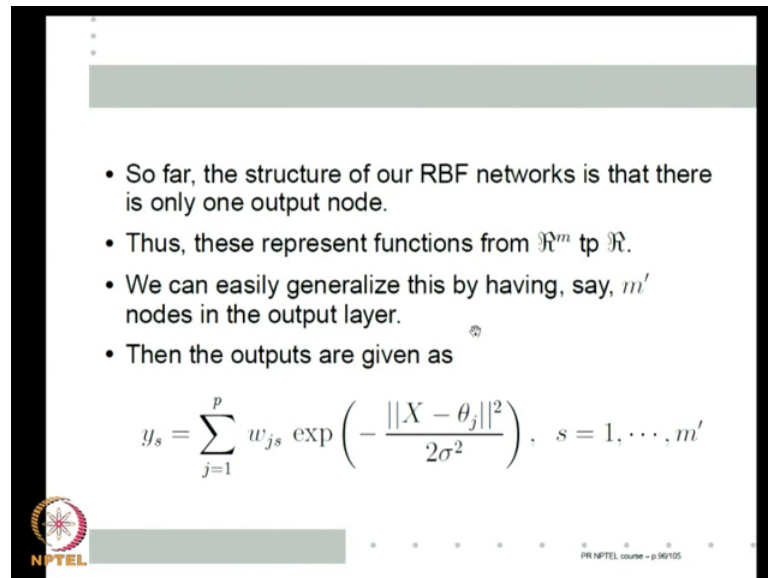
- If the different θ_j are far apart (in relation to σ) then at any X there is only one dominant term in the summation.
- Thus by choosing enough θ_j and taking σ to be appropriately small, we can approximate any continuous function on a compact set.

The slide also features the NPTEL logo in the bottom left corner and the text '© NPTEL course - p102105' in the bottom right corner.

Now, we can think of suppose you are in a two dimensional plane suppose your m is 2, then what it means is that you can plant many θ_j 's. So, to say and (()) let us say on a grid. So, around it θ_j you are at a small exponential and you take this some of these exponentials for the function. Now, the function is continuous and σ we take to be very small then at each point I can my tweak my w_j . So, that at θ_j the function value is w_j , but that it has to be and because I take σ small no other term will contribute contribute more than ϵ .

So, to say and in a small neighbourhood around θ_j it is because the continuous function it cannot... its value cannot differ too much from its value on θ_j . So, the same w_j can easily work for that function. So, this is roughly how the how your representation becomes. If the different θ_j are far apart in relation to σ then at any X there is only one dominant term in the summation. Hence we are placing enough of θ_j 's we can approximate any continuous function on any compact set this the basic idea of sets representations so far.

(Refer Slide Time: 54:00)



• So far, the structure of our RBF networks is that there is only one output node.

• Thus, these represent functions from \mathbb{R}^m to \mathbb{R} .

• We can easily generalize this by having, say, m' nodes in the output layer.

• Then the outputs are given as

$$y_s = \sum_{j=1}^p w_{js} \exp\left(-\frac{\|X - \theta_j\|^2}{2\sigma^2}\right), \quad s = 1, \dots, m'$$

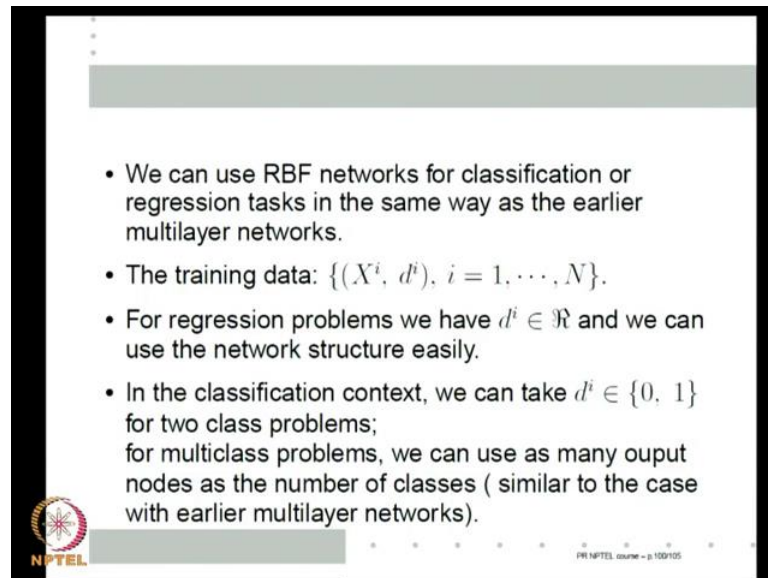
NPTEL

PR NPTEL course - p.06/105

We were only looking at RBF networks where there is only output node, but that is simply a notational convenience. So, when we have only one output node the it represents functions from \mathbb{R}^m to \mathbb{R} , but suppose let us say we want to represent functions from \mathbb{R}^m to $\mathbb{R}^{m'}$. It is very easy to generalize what will, what will it mean instead of one output node I will have m' output nodes then the weights connecting j th hidden node to the output node will, now have two subscripts. Earlier j th output node to hidden node because hidden node in 1 I call it w_j otherwise. Now, j th hidden node to the s th output node will be w_{js} .

So, now the output of the s th output node will be a summation w_{js} the output of the j th hidden node is still the same X minus θ_j whole square sigma square and s runs from one to m' . So, while we so far in this structure we have considered only RBF networks with one output node. Hence the representation is only for real valued functions representing vector value function is nothing vector value functions are simply you know a stat representation of real valued functions. So, you can just have an extra output node that is all, so because this does not add anything in tomorrow's class. When you look at the next class when we will look at learning algorithms, we will consider only one output node.

(Refer Slide Time: 55:25)

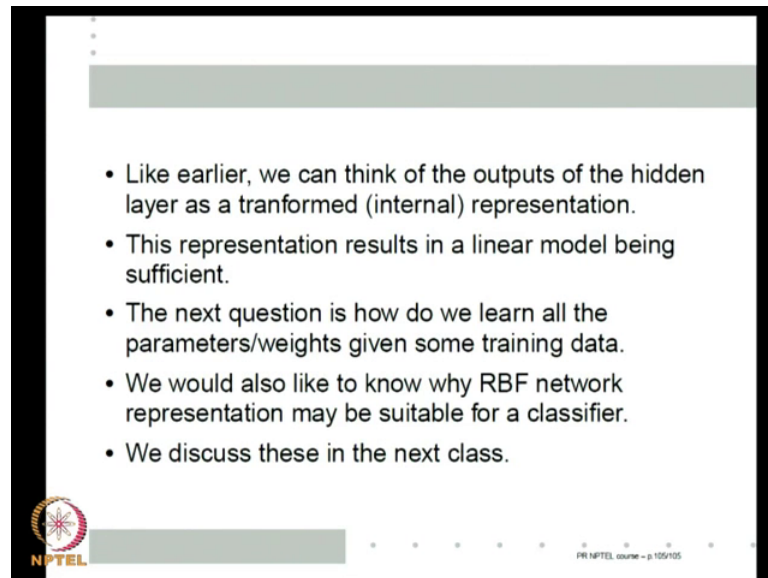


- We can use RBF networks for classification or regression tasks in the same way as the earlier multilayer networks.
- The training data: $\{(X^i, d^i), i = 1, \dots, N\}$.
- For regression problems we have $d^i \in \mathcal{R}$ and we can use the network structure easily.
- In the classification context, we can take $d^i \in \{0, 1\}$ for two class problems; for multiclass problems, we can use as many output nodes as the number of classes (similar to the case with earlier multilayer networks).

We can use RBF networks for classification regression tasks in the same way that the earlier multilayer networks are used what does that mean i am given training data $X^i d^i$ right. Now, I have to learn all the weights for regression problems. How do I choose the training data regression problem? No problem d^i belongs to \mathcal{R} , so we can directly we know what the network structure is I count for I get output as the network. The output of the network is a real number, so d^i is a real number it is fine for classification of course, I can have, I can have a choice for 2 class case.

We can take to be d^i to be 0 1 like earlier like the way we saw in the XOR problem for multiclass problems we can use the same representation that we used with multilayer networks that is we use as many output nodes as the number of classes. The desired output is given as a vector with k th component 1 and all others 0 if it is a k th class pattern.

(Refer Slide Time: 56:37)



The slide contains a list of five bullet points. At the bottom left is the NPTEL logo, and at the bottom right is the text 'PR NPTEL course - p 105/105'.

- Like earlier, we can think of the outputs of the hidden layer as a transformed (internal) representation.
- This representation results in a linear model being sufficient.
- The next question is how do we learn all the parameters/weights given some training data.
- We would also like to know why RBF network representation may be suitable for a classifier.
- We discuss these in the next class.

So, much like what we did with the earlier networks, we can represent any classifier or any regression function using these networks. Essentially, like earlier, we can think of this also as the output of the hidden layer is transformed representations, there is some internal representations of X . Because once given the outputs of the hidden layer the rest is just a linear interpolation $\sum_j w_j$ into output of the j th hidden node summed over j is the output of the network. That is just a linear function, so if I get the right representation of the hidden nodes I am done. So, this representation results in a linear model being sufficient from then onwards just like in the feed forward networks. So you have to learn the internal representation.

So, next question is how do we learn all the parameters and weights given in the training data. Essentially you have learn all the w_j 's and also all these centres and widths of the hidden nodes. Also we can ask, we are saying there is some differences RBF networks use local representation and so on. So, we can ask is there something we can say where RBF networks are more suitable than as a classifier than others or in what way can we think of RBF network as a classifier. So, both these issues, we will address in the next class, and then we will close the discussion on RBF networks.

Thank you.