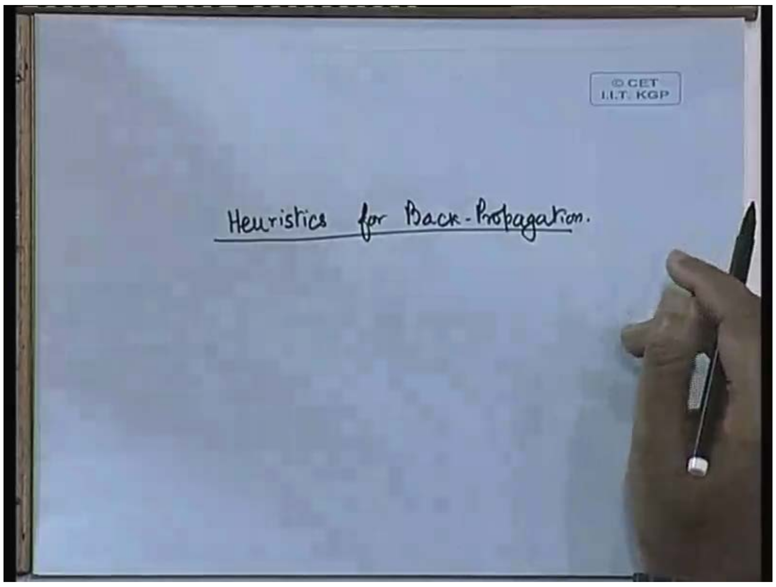
Neural Network and Applications

Prof. S. Sengupta
Department of Electronics & Electrical Communication Engineering
Indian Institute of Technology, Kharagpur

Lecture No - 22
Heuristics for Back-Propagation

The title of today's talk is Heuristics for Back Propagation.

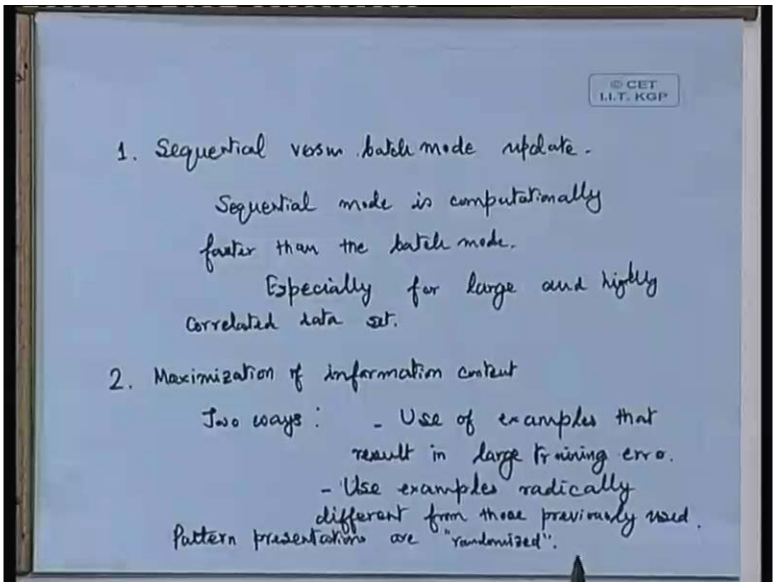
(Refer Slide Time: 01:00)


Heuristics for Back-Propagation.

We have gone through the back propagation algorithm with sufficient details. We now know that, how to adjust the weights of the outer as well as the hidden layers dependent upon the local induced field and the local gradients. And then we also saw, some of the other aspects like the stopping criteria, then whether we prefer the batch mode or the sequential mode.

Now, having done all these things, it is required that we know about some of the heuristics that one can apply to the back propagation in algorithm. In order to ensure that, the back propagation algorithm functions more efficiently. So, in order to ensure it is more efficient functioning, there are some heuristics, which are commonly used and these are the things that we are going to discuss in today's class. Now, some of the heuristics that we use for the back propagation is….

(Refer Slide Time: 02:26)



© CET
I.I.T. KGP

1. Sequential versus batch mode update.

Sequential mode is computationally faster than the batch mode.

Especially for large and highly correlated data set.

2. Maximization of information content

Two ways: - Use of examples that result in large training error.

- Use examples radically different from those previously used.

Pattern presentations are "randomized".

Firstly, that we had initiated already the debate related to the sequential verses batch mode of processing. So, sequential verses batch mode update and there one can note that the sequential mode is computationally faster, as compare to the batch mode. This is the very interesting observation, in fact sequential mode as you know sequential mode means, that it is a pattern by pattern training that we do. Whereas, batch mode means that where the weights are up dated only at the end of one epoch.

Only at the end of every epoch the weights are adjusted. Now, sequential mode is computationally often faster than the batch mode, because if we go into the equations of the batch mode update which we did not. We had consider the equations, for the sequential mode of update only, but if we see the batch mode update, we will see that that involves the computation of a Hessian matrix, which is highly done time consuming.

Whereas, sequential in pattern by pattern that means to say we do not require that a competition analyses the competitions are quite simple. And in fact, the fact that the sequential mode is computationally faster than batch mode is true, especially when the training data set is large and highly co-related, so we can say that this is true especially for large and highly correlated data set.

Now, thus second heuristics that one has to consider is regarding the maximization of the information content, which means to say that, if the training patterns contain good amount of information in it then the learning should be better, the learning should be faster. So, that means to say that we have to go in for some kind of a diversity in the training set. If the training set consists of a diverse set of patterns, that would really lead to more information content and a better learning and there are two ways, where by this can be achieved.

So, the two ways where by the maximization of information content can be achieved is first is the use of examples that result in large training error, this may appear to be a puzzling. Because why are we really preferring those examples which are having large training error, but definitely what happens, is that when there is a large training error. The large training error definitely leads to a larger value of the local gradient then the large value of local gradient makes the learning process faster.

So, one of the ways is to have to pick up examples having large training error and the other is to use examples, which are radically different from those that is previously used. In fact, in the pattern classification problem, we often achieve this by randomizing the presentation of patterns that means to say that whenever, we are going in for a sequential mode of leaning then what happens is that we complete one epoch.

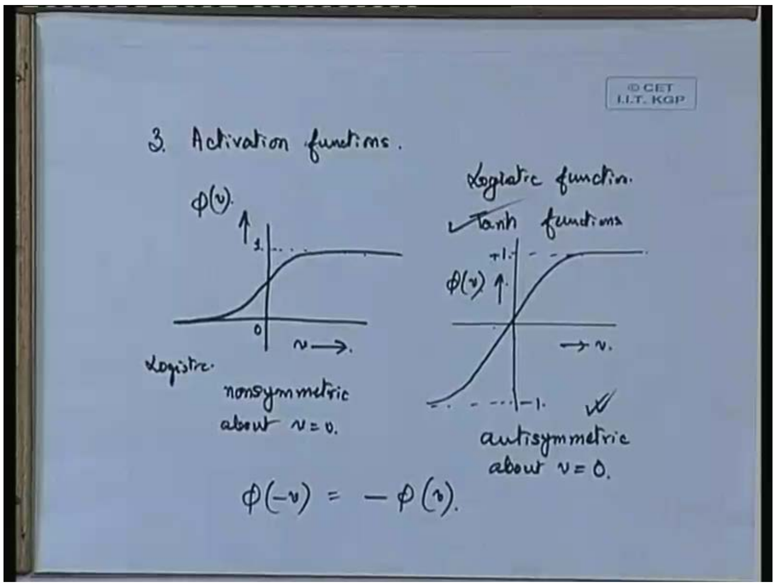
Let us say that there is some order in which, the patterns are presented and then when a next epoch is carried out that means to say, when the training is done for the next epoch that time you just randomize the order in which the patterns are presented. Now, this could appear to be somewhat unnecessary by some people because after all, your job is to present the set of patterns, which constitute the training set does not matter. Whether they are presented in one particular order or without they are presented in a different order, but one thing is very important that if you keep the same order.

Let us say, that from pattern to pattern you are keeping the same ordering from epoch to epoch you keep the same ordering in that case, there is a risk that the network will learn the ordering also. Learning the ordering means that as in to say that, when you are feeling the second pattern it expects that pattern to be classified into some class, when you present the third it expects that to be classified into the other categories.

So, in order to avoid that and to have a better learning capability, one very often randomizes the order in which the patterns are presented. So, we can say that the pattern presentations pattern presentations are randomized, in fact this has got more meaning only if you are having sequential mode. Because in batch mode, it really does not matter that, if you are altering the order of pattern presentations on or not because in the batch mode, any way, the weight adjustments are done at the end of the epoch.

So, this is the second heuristics and the third important heuristics that one considers is the activation function.

(Refer Slide Time: 10:15)



Now, we have so for explode different type of activation functions and one of the characteristics or one of the requirements that we have already specified, for the back propagation networks is that the activation functions are must necessarily be differentiable. And more than that, we did not specify any other limitation for the activation functions, but it is seen that one of the some typical activation functions, which people use are the logistic functions, which involves the sigmoids which are in the range of 0 to 1.

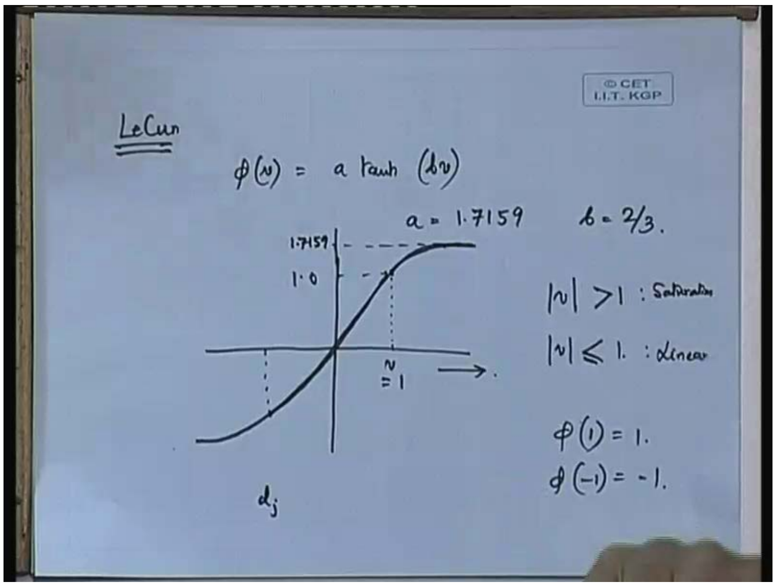
That is of the form of 1 by 1 plus exponential to the power minus av functions of that form or the other possibility is to use tan hyperbolic functions. Now, if you plot the logistic function you will see that, you will get a characteristic like this, if this is the axis in which v is plotted and if this is the axis in which the phi v is plotted, then 1 will be the final value this is 0. So, you will get a function of this nature and in the case of tan hyperbolic function you will be getting a characteristic of this nature, because there it will vary from minus 1 to plus 1.

So, if this is the v and if this is the phi of v, which is of the form of tan hyperbolic then this will be the nature, now here you can see that this function is not symmetric about 0. So, this is a non-symmetric function, this is non-symmetric about v is equal to 0 whereas, this function is anti-symmetric about v is equal to 0. And if you compare the learning capabilities of non-symmetric sigmoidal functions and anti-symmetric sigmoidal functions.

Then, it is seen that the sigmoidal functions, with anti-symmetric activation functions of the tan hyperbolic for example, we will have a better learning capability or a faster learning capability. So, any multilayered perceptron that is, trained with anti-symmetric activations, where anti-symmetric property will be simply defined as phi of minus v equal to minus of phi v. Now, this condition is not satisfied by logistic whereas, this condition is definitely satisfied by tan hyperbolic.

In fact, for the tan hyperbolic logistic for the tan hyperbolic sigmoidal functions, the researches have come up with different type of standard implementation and one of the implementations of the tan hyperbolic functions was proposed by Lecun.

(Refer Slide Time: 14:14)



And they have specified the function of the form phi v equal to a tan hyperbolic and the argument of the tan hyperbolic is b into v where, a and b are constants. And for their implementation that is the one that was proposed by Lecun, it was suggested that a, has got a value of 1.7159 and b was taken to be 2 3rd, the significance of this is that we are going to have a characteristic of this nature. If we are taking a to be of this value in that case, definitely the final value that this phi v function will achieve at the saturation is going to be 1.7159 because, it is going to approach a in the limit.

It is going to approach plus a in the positive limit and minus a in the negative limit, so it will vary between minus 1.7159 up to plus 1.7159, this will be the nature of the tan hyperbolic function. And for a function of this type the interesting characteristic is that, whenever you keep v this is the axis of the v, if you keep v is equal to 1 that means to say that if you are computing phi of 1 then phi of 1 is equal to 1, although the final value is 1.7159.

At v is equal to 1 you are going to get phi v equal to 1 and likewise is the function is anti-symmetric, you are going to have phi of minus 1 to be equal to minus 1. In fact, in this characteristic one thing which you can notice that, there are two portions of the curve one is the linear part of it and the other is the saturation part of it. So, I can describe this part the part beyond v equal to 1, that means to say that when mod of v is greater than 1.

We can say that this belongs to the saturation part of the characteristic whereas, as long as mod of v is less than or equal to 1 this is a linear characteristic. So, this is linear whereas, this one is the saturation and we can say roughly that as if to say that, v is equal 1 is lying at the boundary of the linear to saturation zone and we will use this very shortly. So, the third heuristic that we were describing is again, related to the activation function and we have seen that anti-symmetric functions are better as compared to the non-symmetric functions.

The fourth heuristic is pertaining to the target values, now target values means that definitely in the training set we are going to have the input set, we are going to have the input set, we are going to have input vector as also the output values. Now, the output values will be again in the range that is specified by the activation function is in it, so definitely if we are using a characteristics of this nature, let say again a tan hyperbolic characteristics of this nature we take.

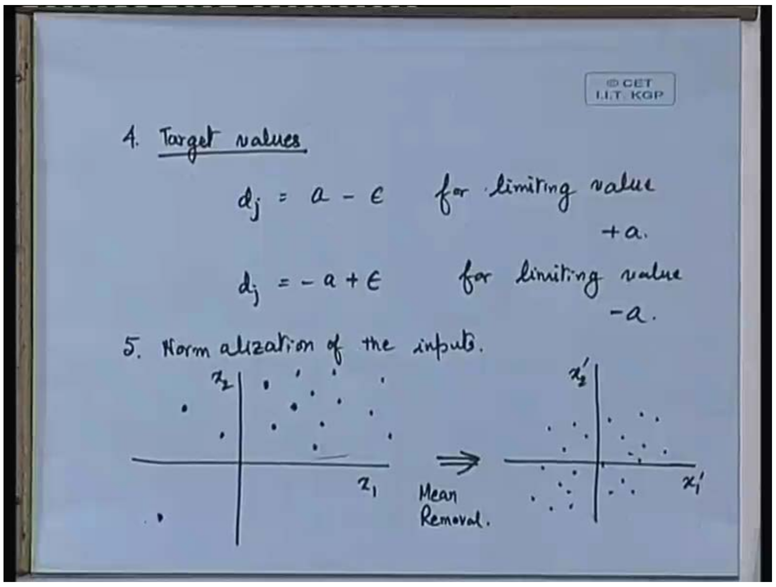
Then, the activation function, we cannot definitely exceed an activation function d, an expected output d of the neuron j, if we are taking dj, dj certainly cannot exceed 1.7159 at this end or cannot be less than minus 1.7159 at the same. So, definitely it is implied that the activation function need to follow this, but again the question is that where should we keep the expected, how should be select the patterns, so that the activation functions, they could be lying close to the saturation region or they could be lying in the linear region.

Now, which one is better, are we going to choose those patterns which are having there expected outputs close to the saturation region of the characteristic or close to the linear range of the characteristic. Now, it is seen that the linear range of characteristics, if we are choosing the dj's closure to the linear range of characteristic, it is better from a training point of view, but certainly this is not the thing, that you can always guarantee because dj is after all what your expected outputs.

Which are drawn from the examples, now it could be that for some of the patterns, which are available as examples this dj could be very close to 1.7159, in fact, there is no harm if it is exactly equal to 1.7159. So, what is done is that in order to ensure a better learning, as a heuristic what one does is that, one applies and offset to the desired output before the training is performed that means to say that we are not really training it with exact dj. If dj close to the limits if it is close to plus a or if it is close to minus a then we are not taking the exact value of dj rather we are taking some offsetted value of dj.

So, that we shift the dj points more towards the linear range for example, in this case if I get a dj equal to 1.7159, in that case the nice thing will be is that I apply an offset of minus 0.7159. So, that my dj is equal to 1 because this 1 I know is definitely closer to the linear range, so that is why what we have to do is to modify the target value.

(Refer Slide Time: 21:18)



© CET I.I.T. KGP

4. Target values

$d_j = a - \epsilon$ for limiting value $+a$.

$d_j = -a + \epsilon$ for limiting value $-a$.

5. Normalization of the inputs.

x_2
x_1
Mean Removal.
x_2'
x_1'

So, we must keep dj equal to a minus epsilon, some positive quantity epsilon for the limiting value, when the limiting value is plus a, for the limiting value of plus a we must make dj equal to a minus epsilon and when the limiting value is minus a then what we should do yes. So, for a limiting value of minus a, we should have dj equal to minus a plus epsilon, so this is the heuristic on the target values and the next point that we are going to the deal with, is regarding the normalization of the inputs.

Now, normally we are not putting any restriction on the set of patterns with which we are going to train, in general it will be a set of training patterns in the m dimensional space. Now, since we again cannot visualize and m dimensional space let us, consider this simplest case of a two dimensional space for our ease of visualization. Supposing we have got an x 1 x 2 space where, we have got two inputs basically x 1 and x 2 and the set of patterns, that could be very widely distributed.

Like, we could have let us say the set of patterns like this, supposing all these points that I have marked in this two dimensional space, they are the set of points with which we are going to train those are the input patterns that are available. Now, what is the characteristic that you are observing, you see that the way I have drawn the axis both x 1 and x 2 could vary in the positive as well as the negative range.

But, we have chosen the patterns where, we are always having the patterns selected out of baring except one or two patterns which are lying in this quadrant, but most of the patterns in this particular example, are lying in the first quadrant with x 1 and x 2 both as positive. Now, is it very good surely not because, you are training the back propagation network for a good generalization capability, so that means to say that it should encounter the patterns, which are varying in values equally between positive and negative values.

So, ideally we should take the patterns where, x 1 varies in the range of say minus a to plus a and not this a of course, I am saying some limits of x 1 in the positive and negative and similarly x 2 also in some minus and plus range, that can vary. And if we can get the patterns like that, that is the best, but again as I told you that obtaining the patterns may not be at our hands because, we are drawing the examples from some real life data's which are already available as examples.

I could be that the examples themselves are such that, only these many examples are available, but that does not mean that whenever we are testing the network with unknown patterns, the unknown patterns cannot be like this unknown patterns can be, who says that everything should be available or very similar pattern should be available as examples only. We can have unknown patterns here and in this case there is no good reason to believe that the network will have a good generalization capability.

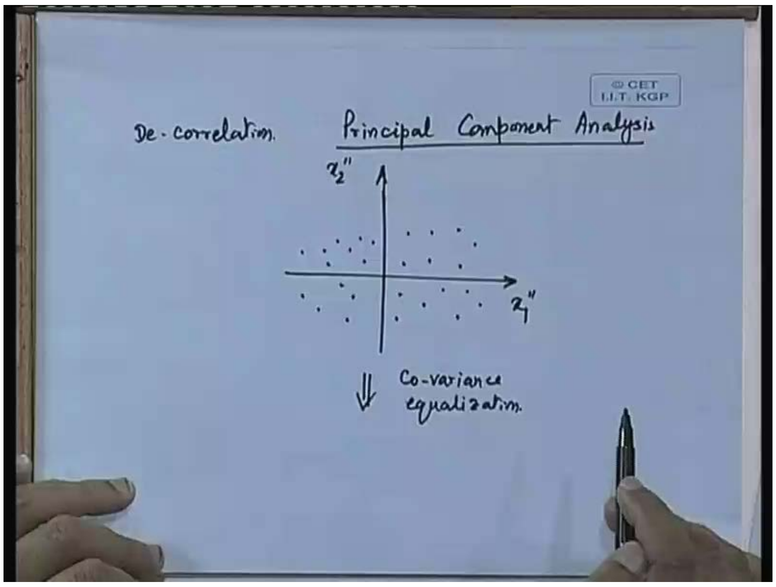
Because it has seen the patterns in this region, but it has not seen the pattern in this region during the training, but during the testing it has in kind of pattern like this. So, that is why we have to do something with the inputs, some manipulation, some manipulation is needed with the input data before we actually train the network with the patterns, so what is the kind of normalization that we do.

Now, in this example I have shown you that x 1 and x 2 are both having some positive bias in this example there is a positive bias to both x 1 and x 2. So, what if we obtain the mean of the x 1 and x 2 values and if we subtract the mean value from all these observations, from each one of these observations in that case, we should get a distribution that should be equally balance between the positive and negative.

So, what we can have is that the picture in this case may look something like this, whatever was on like this restricted to the first quadrant alone may now be looking like this, but. So, this is the process of mean removal, so this is the process of mean removal and in the process of mean removal actually, we are altering the inputs space, we cannot call the input space as x 1 x 2 space any more. We have to call it as a new space x 1 prime x 2 prime.

So, that is one of the steps that we do, that is the mean removal and in order to accelerate the back propagation algorithm, we need to have some more steps to be performed after that and immediately after the mean removal, the step that is normally performed is a de correlation of inputs, so those who are having difficulties in understanding the term de correlation.

(Refer Slide Time: 28:25)



Let me give you a simple example, let us say that you have got a set of time domain samples, supposing you have got some voltage data or any analog data you have got and you have taken time samples of that. And each one of those time samples, you are representing with some digitized value, now you will be finding that the time samples are highly correlated from each other, by correlation what we mean that the time sample that you are taking at n, will not be remarkably different from the time sample that you are going to take at n plus 1 or n minus 1.

So, the data is highly correlated, now supposing you have gotten sinusoidal wave and the time samples, if you are taking many samples you will be finding that the time sample are highly correlated. But, now if you are converting the temporal data, that time domain data into a frequency domain data then what will you get I said the example of a sinusoidal wave, sinusoidal wave means it will have a frequency component and also if you are observing the sinusoid only through a small window, if it is not a continuous sinusoidal wave, which will be in the case of the discrete thing.

Then, we will be having because of the truncation effect of the sign wave, you are going to have some more components also, but other than that the frequency components will be highly uncorrelated. So, the reason why we often transform from the time domain to the frequency domain is because, we want to de correlate the data and de correlation is very often useful for thinks like data compression because, if we de correlate the individual components then, it is possible for us to represent it by a smaller number of samples, as compare to the original domain with which may ever working.

In fact, de correlation there are different methodologies for performing de correlation and one of the effective ways, where by the de correlation is performed is principal component analysis. And we are going to study principal component analysis shortly, not immediately may be after a few lectures, we will be a coming to the topic of principal component analysis, which are very often used in order to de correlate the data, now, what happens is that.

So, the steps that, we are doing is that the first step is mean removal and the next step is going to be de correlation, so whatever you have as x 1 prime x 2 prime will now be de correlated, de correlated means this data set could now be getting more spreaded like, if you are having a de correlated set of patterns then the patterns may be looking like this. In fact, what happens is that this is the x 1 axis not x 1 anymore because already because of the mean removal we were calling it as x 1 prime.

So, now with the de correlation, we are going to call it as x 1 double prime and this axis will be x 2 double prime. In fact, the scaling on x 1 w prime and x 2 double prime are quite different as you can see from here, so what happens is that after the de correlation you have to do one more step and that is called as the covariance equalization. What is meant by covariance equalization is that, the covariances in the x 1 direction and x 2 direction in this case they are different.

So, you scale it appropriately, so that the covariance‘s in x 1 direction and x 2 direction are equal and after the covariance equalization you will get the patterns somewhat in this range, if they are equalized, if the covariance are equalized we are going to get the pattern. So, now we are going to call this as a new space x 1 triple prime and x 2 triple prime.

So, in effect what we are doing is the normalization of the inputs, so now that we have transformed the input space into a new space x 1 triple prime x 2 triple prime where, the datas are having 0 mean number one, number two the datas are highly de correlated and number three where, there covariance's are equalize. Now, on this data, on this space if we apply the back propagation algorithm the learning is seen to be better.

So, this is the process of the normalization of the input, which we had listed as the fifth heuristic in our case. The next heuristic and which is perhaps the most crucial and important of this is the initialization, by initialization we mean to say that the question, which should very often arise is that we begin with the training of a network, let say a multi layer perceptron network, which we are going to train with the set of examples that we have got, but the billion dollar question is that how to choose the initial weights.

Is it absolutely arbitrary, that we choose any initial weight and the system showed adapt itself, should adjust the synaptic connections in such a manner that, ultimately all the training pattern examples are learned and then when we feed the test pattern, then it can correctly generalize the test patterns also. If it can do that it is good, if it is capable of doing that, but the question is that yes, I mean given the convergence criteria's of the multilayer perceptron.

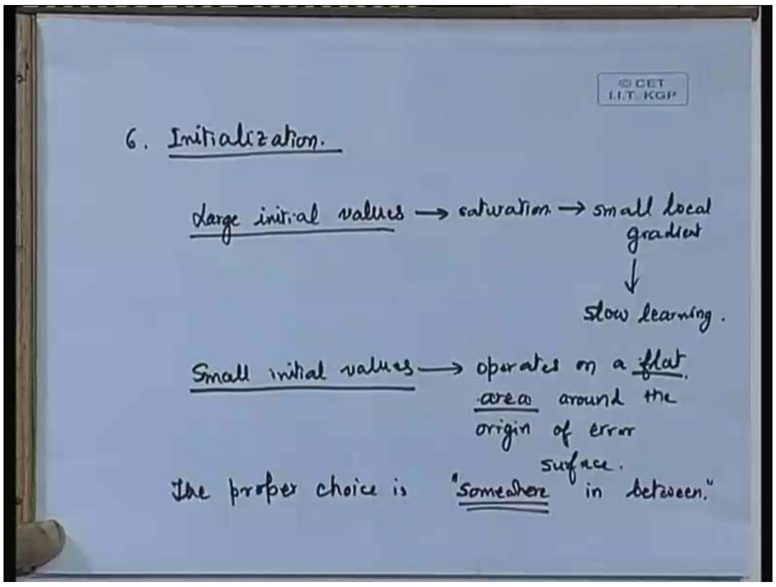
It is going to happen, but surely the question is that how quickly is the convergence going to be how quickly will the network learn. If we start with any arbitrary combination of weights and then the question that should come is that, if you are having let us say you are having an activation function of this nature, now you start with some initial weights, let us say that this is the activation function of a neuron, which is connected to several inputs.

One neuron connected to several inputs say it is like this, now the question is that I can choose the initial weights such that, the total activation function, the activation value v or rather the local induced field what we refer to. So, we can choose the wji's such that the local induced field is here, in some other occasion we could choose somewhere over here, in some other occasion we could choose the initial weights, such that the initial local induced field is lying over here and the question is that which one is better.

Whether we want again the initial local induced field in this region or whether we want the initial local induced field is in this region, whether it is close to the linear region or close to the saturation region. So, this is another decision that we have to make and let us see the two extreme cases, one of the extreme cases is that we can choose the initial weights, such that the local induced field is very small that means to say we chose small initial values, if we chose small values of wji's because, ultimately our induced local field that is vj is going to be what, summation of wji yi.

I summed up from 1 to m that is part we are doing where, m is the number of inputs, now the question is that whether we want, now if we have the wji is to be small in that case naturally that will lead to a small value of vj. So, we can have small w ij is to start with or we can have large wij is to start with, now which one is having what kind of effect. So, that is what we are going to study in the sixth heuristic, which is under consideration that is the initialization.

(Refer Slide Time: 38:51)



© CET I.I.T. KGP

6. Initialization.

large initial values $\rightarrow$ saturation $\rightarrow$ small local gradient $\downarrow$ slow learning.

Small initial values $\rightarrow$ operates on a flat area around the origin of error surface.

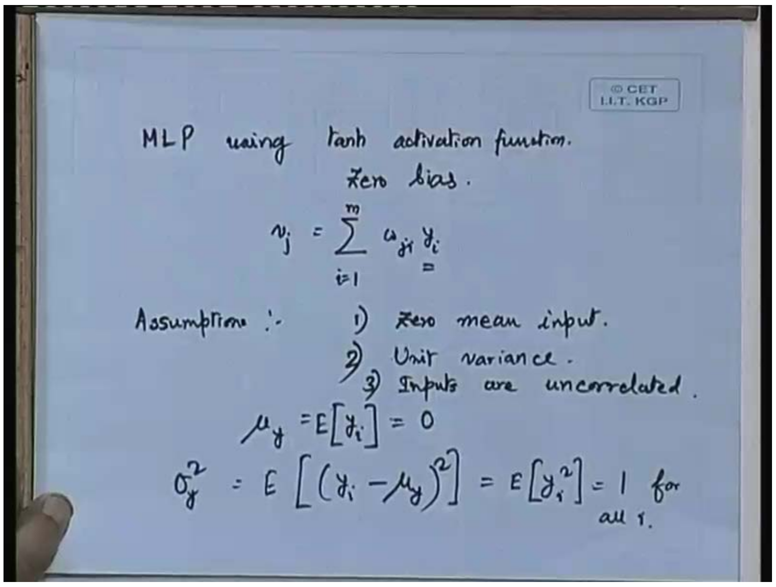
The proper choice is "somewhere in between."

Now, we can began with large initial values, now large initial values means obviously, our operating point in this characteristic would be somewhere close to the saturation, so large initial values could lead to saturation because, the initial values could be very close to the saturation region and in the saturation region you can see that the gradient is 0. The gradient of the activation function is 0, so if we choose large initial values it leads to saturation and saturation; obviously, leads to small local gradient and small local gradient means that it leads to slow learning.

So, naturally very large initial values are not good for us and the other extreme is that, we can chose small initial values and small initial values means that our operating point will be close to here, close to the origin. But, in this case the drawback that one encounters is that, the network operates on a flat area around the origin of error surface, so this is also not, if it is on a flat area then definitely it takes time or it is not a guaranteed convergence that one gets.

So, even small initial values are also not welcome, so what we really require is something in between these two, so the proper choice is somewhere in between. So, this is what we are going to study, that what exactly is this somewhere in between, whether we should have here or here or here, is it possible for us to decide, so let us do a bit of statistical analysis in order to decide that. So, we take the example of a multilayered perceptron using tan hyperbolic function.

(Refer Slide Time: 42:17)



MLP using tanh activation function.

Zero bias.

$$v_j = \sum_{i=1}^{m} w_{ji}\, \underline{\underline{y_i}}$$

Assumptions :-
1) Zero mean input.
2) Unit variance.
3) Inputs are uncorrelated.

$$\mu_y = E[y_i] = 0$$

$$\sigma_y^2 = E\left[(y_i - \mu_y)^2\right] = E[y_i^2] = 1 \text{ for all } i.$$

So, we take a multilayer perceptron in short I am writing it as MLP, using tan hyperbolic activation function and also we are considering the neurons to have zero bias. So, for the time being we are neglecting the bias or considering it to very zero bias, so that the induced local field of neuron j, vj you can understand that the induced local field of neuron j will be given by what wjiyi, where, yi is the input and wji is the strength of the connection between the neuron j and the neuron i.

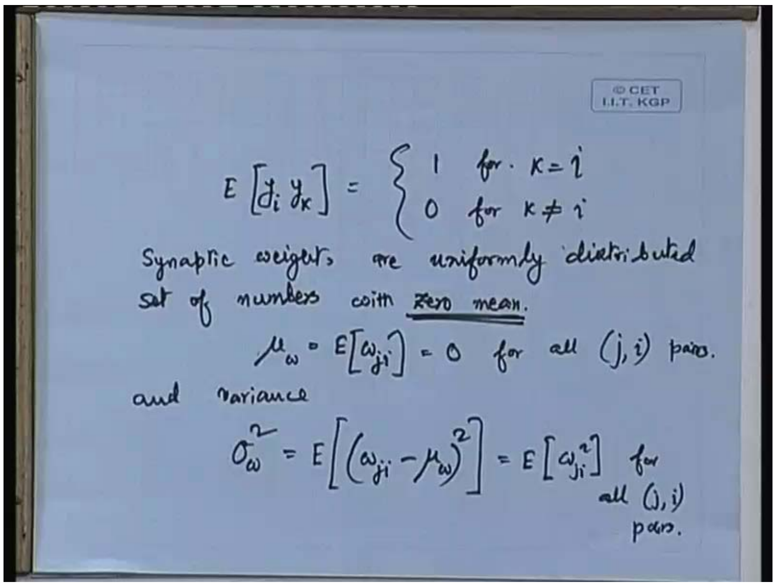
I mean j is the neuron under consideration, so it is wji and this will be summed up for what for i is equal to 1 to m. Now, we are assuming we are making some assumptions on this input and the assumptions that we are making are one that it is zero mean input and it is assumed to have unit variance, although this unit variance is not mandatory, it is only for the ease of our analysis that we are choosing the unit variance

So, what we can do is that from this equation, it is possible for us to obtain a mu y what is meant by mu y, mu y will mean the expectation of the yi's, so we are taking all the yi's; that means to say i is equal to 1, 2 etc, etc up to m. So, we are taking the expectation over this yi's and this is since, it is assumed to have a zero mean input we should have expectation of yi to be equal to 0.

And the variance because we are assuming unit variance, we can write sigma square y to be equal to the expectation of yi minus mu y this squared, this is the definition of the variance and this in fact, because mu y is going to be equal to 0. We can simply write it as expectation of yi square and what is expectation of yi square that is equal to unity, so this is equal to unity for all i because, we have assumed unit variance and another assumption that we are making is that the inputs are uncorrelated.

So, the third assumption is that the inputs are uncorrelated, so if we take that assumption into consideration, then it is possible for us to write that expectation of yi yk will be equal to what, 0 for k not equal to i.

(Refer Slide Time: 46:20)



$$E[y_i y_k] = \begin{cases} 1 & \text{for } k = i \\ 0 & \text{for } k \neq i \end{cases}$$

Synaptic weights are uniformly distributed set of numbers with <u>zero mean.</u>

$$\mu_\omega = E[\omega_{ji}] = 0 \quad \text{for all } (j, i) \text{ pairs.}$$

and variance

$$\sigma_\omega^2 = E[(\omega_{ji} - \mu_\omega)^2] = E[\omega_{ji}^2] \quad \text{for all } (j, i) \text{ pairs.}$$

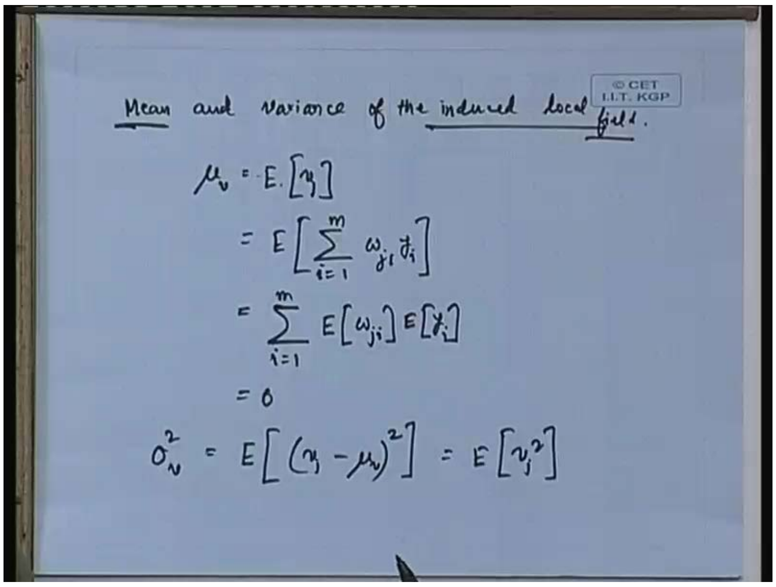
And what happens if k is equal to i 1, so this is the uncorrelated thing that only when i and k are the same, then we are going to get the expectation to be equal to unity otherwise it is equal to 0 and also we are making some assumptions, about the synaptic weights. So, what we are expecting is that the synaptic weights that we are choosing here are uniformly distributed set of numbers with zero mean, so for which we can now write that mu w, so that is what we are getting as the expectation of wji, the expectation of this weights.

So, since we are choosing the weights also from a zero mean variable, we should have expectation of wji to be equal to 0 for all ji pairs and for the variance we can write and for the variance of the synaptic weights, we can write sigma square w which is equal to the expectation of wji minus mu w, mu w is the mean weight and mean weight is equal to 0 as I said. So, here we can simply write it as, this is equal to expectation of wji square.

Again since this is valid for all ji pairs, we are considering I mean, we are now drawing note, this is valid for not only for initial this is valid everywhere, but we are using this result in order to choose the initial weight. So, all this analysis is being done in order to help us in choosing the initial weight followed, so, this is our sigma square w and now we can write that accordingly, the mean of the induced local field that is mu of v.

That we have not really expressed, we have obtained the expression of vj, we have use the expression vj, but we are now interested in finding out the variance of the induced local field. So, we are first of all going to find out the mean of the induced local field and then the variance of the induced local fields.

(Refer Slide Time: 49:56)



Mean and variance of the induced local field.

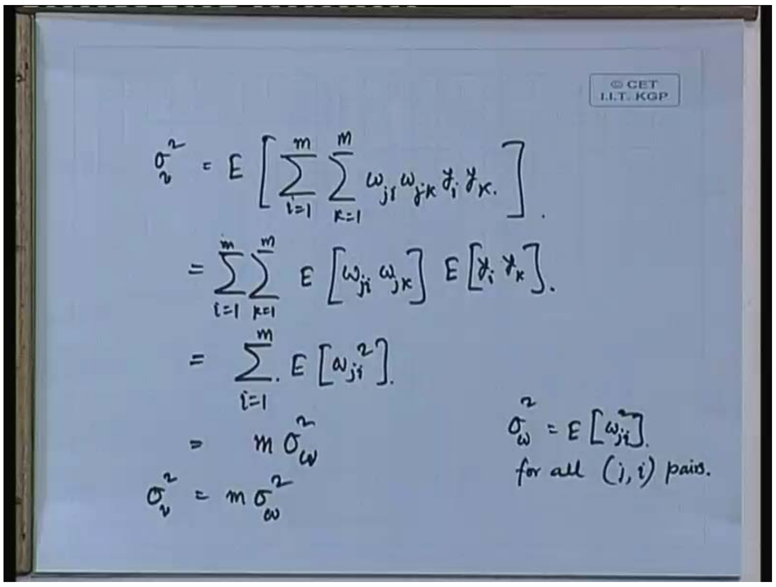
$$\mu_v = E[v_j]$$
$$= E\left[\sum_{i=1}^{m} \omega_{ji} y_i\right]$$
$$= \sum_{i=1}^{m} E[\omega_{ji}]\, E[y_i]$$
$$= 0$$
$$\sigma_v^2 = E\left[(v_j - \mu_v)^2\right] = E\left[v_j^2\right]$$

So, mean and variance of induced local field this is what we are calculating, so the mean is expectation of vj, which is equal to expectation of summation i is equal to 1 to m wji yi, have an simply substituting the expression for the vj. And this is equal to I can take the summation outside also, that is its one on the same as writing, summation i is equal to 1 to m and writing it as a product of expectation wji expectation of yi. In fact, there is no need to expand it any further because, this value is going to be equal to v.

So, the mean of the induced local field is definitely going to be 0, I mean that is there and then the question is that what is going to be the variance of the induced local field. So, the variance we can express similarly, so we are interested in finding out sigma square v, so sigma square v can now be expressed as the expectation of what vj minus mu v, vj bar or mu v, so which is vj minus mu v square and mu v in this case is equal to 0, so it is expectation of vj square simply.

So, in the next slide I am going to just write down the expression for this vj, I am going to substitute the expression for this vj, in fact, because the expression involves vj square I can. In fact, express it as the product of two vj’s or rather I can express it in this manner vj square, because it is expectation of vj square, I can write it as….

(Refer Slide Time: 52:31)



$$\sigma_v^2 = E\left[\sum_{i=1}^{m}\sum_{k=1}^{m} \omega_{ji}\,\omega_{jk}\,y_i\,y_k\right].$$

$$= \sum_{i=1}^{m}\sum_{k=1}^{m} E\left[\omega_{ji}\,\omega_{jk}\right] E\left[y_i\,y_k\right].$$

$$= \sum_{i=1}^{m} E\left[\omega_{ji}^2\right].$$

$$= m\,\sigma_\omega^2$$

$$\sigma_v^2 = m\,\sigma_\omega^2$$

$\sigma_\omega^2 = E[\omega_{ji}^2]$. for all (j, i) pairs.

Sigma square v equal to expectation of summation, I can express it as a double summation i is equal to 1 to m and k is equal to 1 to m wji wjk yi yk and this could be re written as taking the summation out, i is equal to 1 to m, k is equal 1 to m expectation of wji wjk expectation of yi yk which is equal to summation of i is equal to 1 to m expectation of wji square because, we definitely making use of the property of the correlation, we have already defined that.

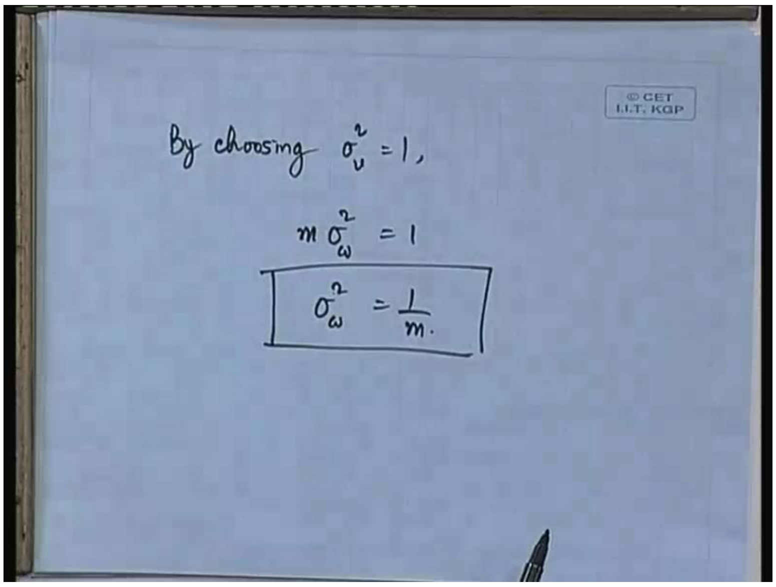
The datas are uncorrelated because of which, we are going to have expectation of yi yk to be equal to 1 for k is equal to i, so only for k is equal to y terms we are going to have this equal to 1. So, this definitely leads to this expectation of wji square and since, the expectation of wji square is what, that is equal to sigma square w for all ji pairs, so what will be the final value this is summed up for i is equal to 1 to m. So, it will be m times sigma square w, because sigma square w is equal to expectation of wji square for all ji pairs.

Now, this is an expression that is of interest to us, that is sigma square v that is to say the variance of the induced local field is equal to m times sigma square w, now, what is going to be the best value of sigma square v. Again we were mind you, considering the tan hyperbolic characteristic, tan hyperbolic activation function, now in the tan hyperbolic activation function, if we choose the values which were proposed by Lecun and others.

We choose a is equal to this value and b is equal to two third, so that for v is equal to 1, we get the phi v to be equal to 1, in that case this one is becoming our linear region and this is the saturation region. So, if the variance is chosen, so that it is up to the borderline of the saturation and the linear region that means to say what, that your activation function could vary its variance should be close to 1

If you are choosing that, then we can say that the patterns are equally, almost equally distributed between the linear and the saturation region. So, if we choose the sigma v to be equal to unity from this characteristic in that case, so by choosing sigma square v equal to 1.

(Refer Slide Time: 56:41)



By choosing $\sigma_v^2 = 1$,

$$m\sigma_\omega^2 = 1$$

$$\boxed{\sigma_\omega^2 = \frac{1}{m}.}$$

Out of this characteristics, what we get is, we are getting m into sigma square w equal to 1 which means sigma square w is equal to 1 by m and this is a relation that we can make use of; that means, to say that the variance the initial variance that you are going to have for the weights is equal to 1 by m where, m is the total number of inputs. So, now you are going to know about this m, a priory that is how many number of inputs you are having, so the best thing will be to design the synaptic weights, the initial synaptic weights.

Such that, you have the initial synaptic weights with zero being that is number one and the variance of the synaptic weight should be equal to 1 upon m. So, this is up to the heuristic point number six where, initialization is concern and we have got one or two points more to be covered on the heuristics, which we are going to cover in the next class till then.

Thank you.