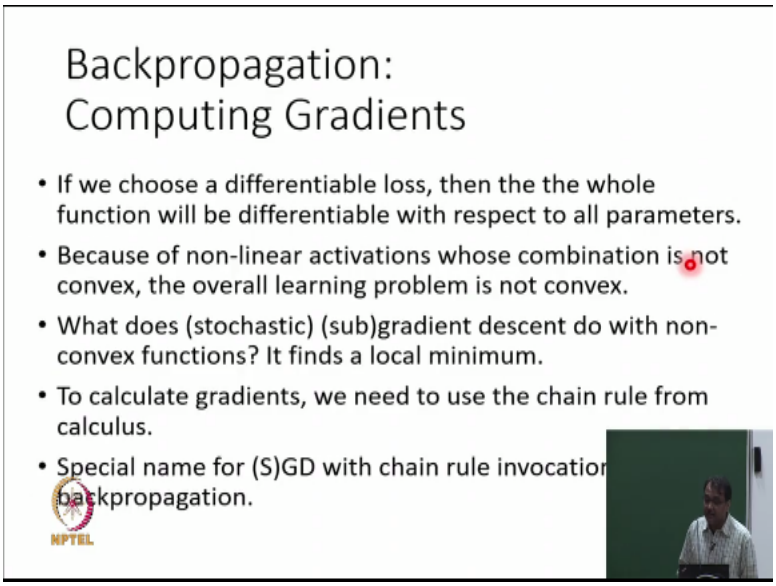
Artificial Intelligence
Prof. Mausam
Department of Computer Science and Engineering
Indian Institute of Technology-Delhi

Lecture - 88
Deep Learning: Backpropagation through a Computational Graph

For the next 12, 13 minutes, I want everybody to put on your thinking caps, your computation caps, your math caps because this is where you are looking at a final exam question or at least learning the most important part of neural network, okay. So the main goal is I have chosen a differentiable loss, right. That means technically my whole loss is differentiable with respect to each parameter of my neural network.

I may have lots of w i's for every layer. Lots of b i's. I need to somehow update them, but w l depends on output of l – 1 which depends on output of l – 2. So the parameter at w l depends on the output of the parameters w l - 1 1 and so on so forth. All these parameters are intertwined and interdependent in complicated ways. How do I compute the gradients.

(Refer Slide Time: 01:22)

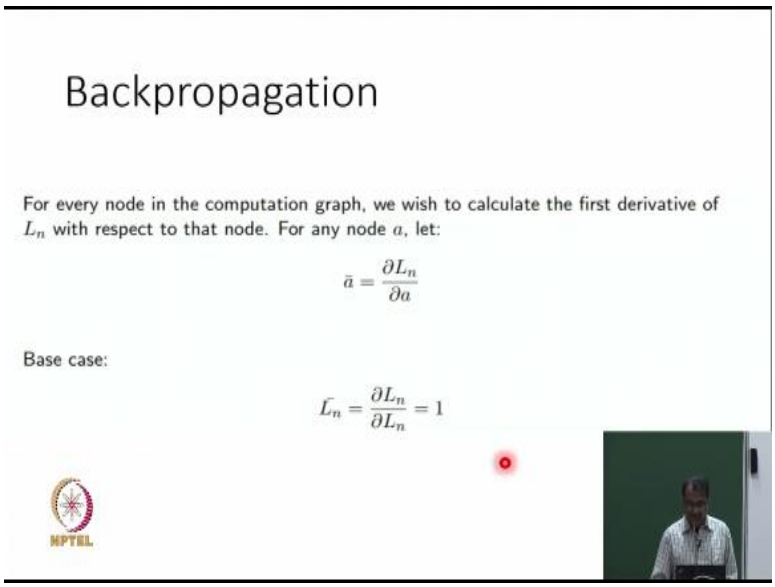


And the algorithm that allows us to compute these gradients is called the back propagation algorithm attributed to Geoff Hinton came from the I think 80s or 90s that time era has not changed till late. It is the exact same algorithm, okay because it is mathematically appropriate, logical, sound. Okay, so let me quickly go through the introduction and then we will do one example.

Because of nonlinear activations the combination is not convex. The learning problem is not convex. That is okay, who cares. So we will basically do stochastic gradient descent with non-convex functions. What would it do? It will find a local minimum, good. That is okay. To calculate gradients we will use the chain rule, okay. And the chain rule for calculus is derivative of loss with respect to a if a computes b.

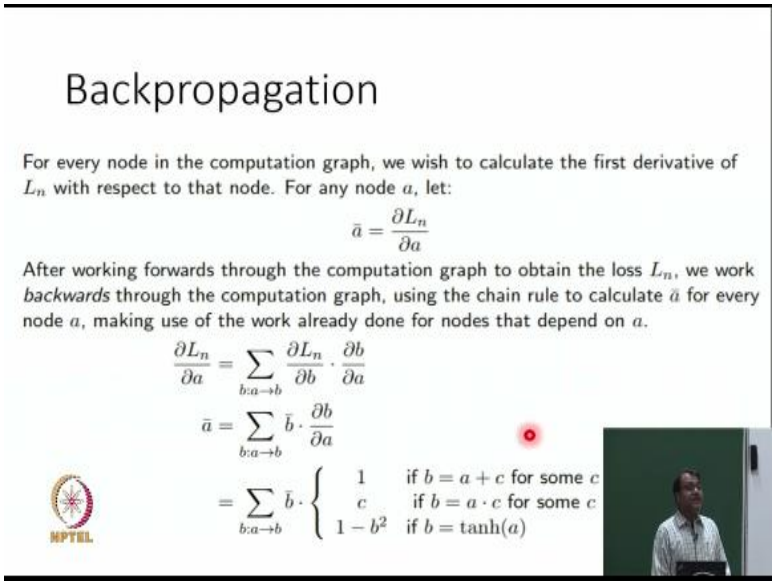
And b computes loss would be derivative of loss with respect to b times derivative of b with respect to a okay. This is the general principle that we are going to use. And special name backpropagation is nothing but a special name for gradient descent with chain rule in the context of a neural network and computed efficiently using dynamic programming. But we will not talk about the dynamic programming part of it. But you can see, right.

(Refer Slide Time: 02:47)



So for every node, so what we are going to do is we will convert the neural network into a computation graph where each point one weight or few weights is computing one particular new symbol and I will show you and with an example and so the base case would be derivative of the loss with respect to the loss is 1, of course. And we are going to use this notation a bar which stands for derivative of the loss with respect to a okay.

(Refer Slide Time: 03:23)



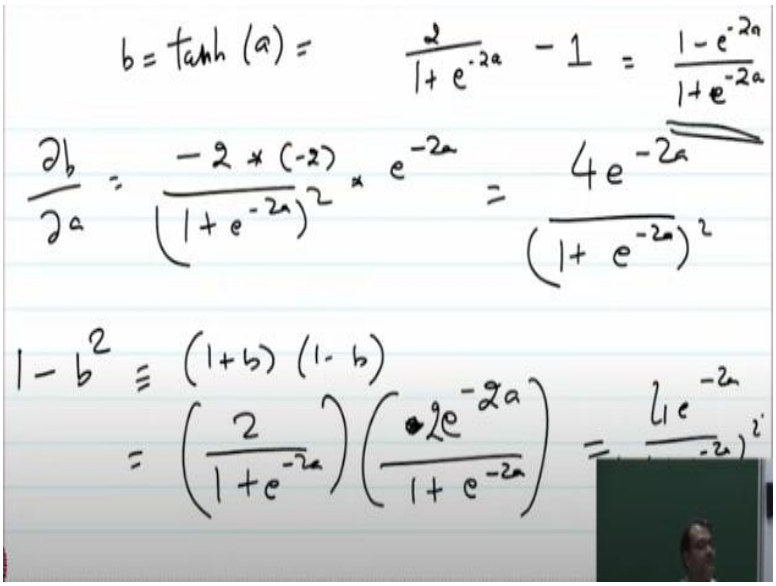
Now for every node in the computation graph, so computation graph think about forward computation, okay. For every node in the computation graph we wish to calculate the first derivative of loss with respect to that node. That node has only one symbol. For any node a we want a bar to be del L over del a. After working forward through the computation graph we will finally obtain the loss L.

We will work backwards through the computation graph using the chain rule to calculate a bar for every node a making use of the work already done for nodes that depend on a. As an example if I have an edge from a to b, which basically means if I compute a then I have some function that computes b and I have already computed the loss with respect to b, which is b bar.

Then I am going to simply multiply this with the loss with respect of b with respect to a so we will basically have b bar times del b by a, del b by del a. But multiple such b's may be computed from a. So a is a node which informs multiple nodes in the future all the b's. So all those b's I will sum over, okay. And slide gives me one example and let us look at this example. It says that suppose b is equal to a + c.

Then what is del b by del a? What is del b by del a? 1. C is not dependent on a right. On the other hand if b is equal to a times c. What is del b by del a? c. So therefore in such situations, I will do b bar times 1 or b bar times c. Now it also gives you an example of what if b is equal to tanh (a). So I thought we will just take two minutes and quickly do this computation just so that you know how to get these done.

(Refer Slide Time: 05:39)



So what if B is equal to tanh (a) and the answer we are looking for is 1 - b square. So I have quickly written down the equation for tanh (a). So tanh (a) is nothing but 2 upon 1 + e to the power -2a – 1. And if you take the -1 in the numerator, then it becomes equivalent to 1 - e to the power -2 upon 1 + e to the power -2a. So let us quickly compute del b by del a. So 2 over x the derivative would be minus 2 by x square, right?

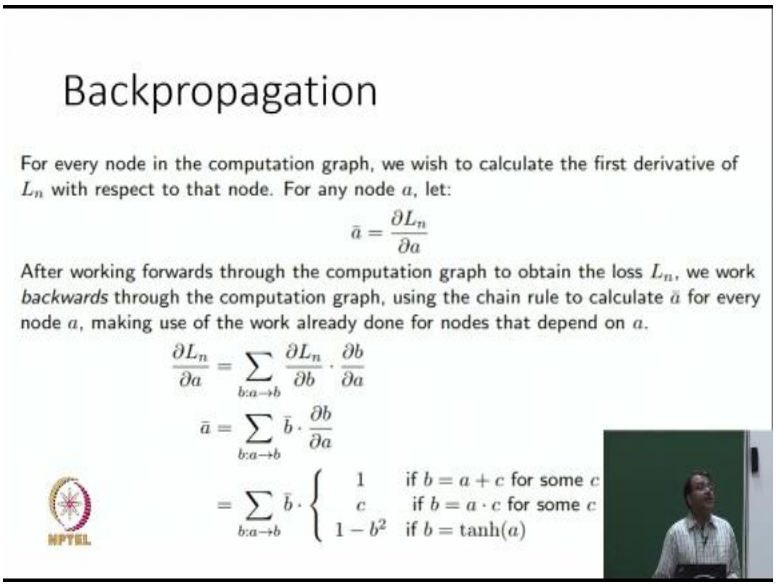
So this will become -2 upon 1 + e to the power -2a whole square. But now you have to take the derivative of x. So that is e to the power y. So that is e to the power - 2a. But now you have to take derivative of y which is -2a. So that becomes -2. So therefore del b by del a is nothing but -1 of course when you take derivative as nothing. So you get four times e to the power -2a divided by 1 + e to the power -2a whole square.

Now you do not want to remember this. This is a complicated formula. So the book the slide also says that this is nothing but 1 - b square. So let us check, okay. What is 1 - b square? Let us use the, if it is 1 - b square it should be same as 1 + b into 1 - B. And what is b here? Let us use the second definition of b, it will make our life easier. So what is 1 + b? 2 upon 1 + e to the power -2 a and what is 1 - b?

-2 e to the power -2a upon 1 + e to the power -2a. No there is no minus here because I have already consumed the minus inside the denominator. And so if you do this, then

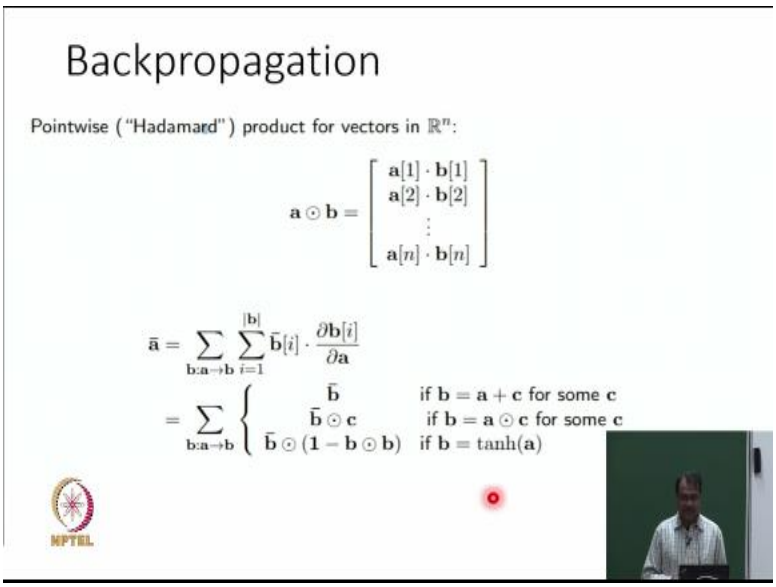
this is nothing but 4e to the power -2a divided by 1 + e to the power -2a whole square, right?

(Refer Slide Time: 07:52)



So basically what have we done? We have taken the tanh function and if you say okay suppose B was tanh (a). Suppose b was a non-linearity here and with respect to a and then I want to backpropagate the derivative of b to derivative of a the derivative of a would be nothing but b bar times 1 - b square. And of course in life, we will be working with matrices and vectors.

(Refer Slide Time: 08:20)



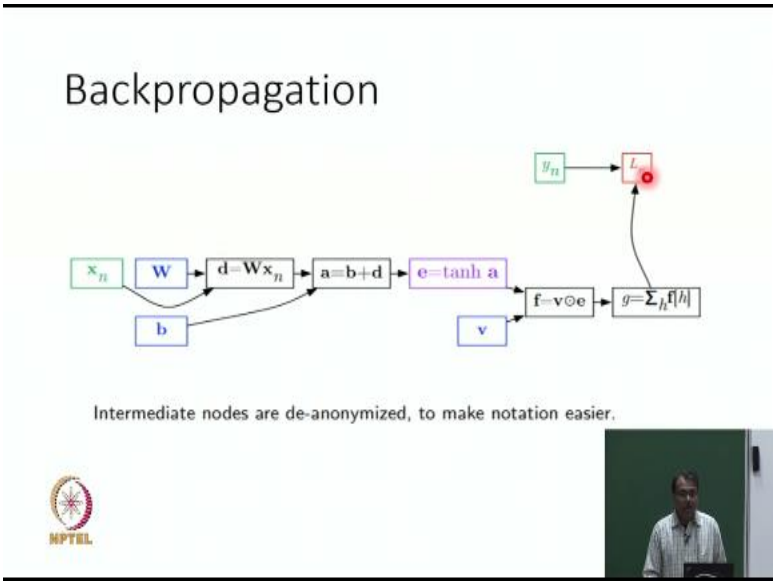
So all of these things would have an equivalent linear algebra notation where everything is a vector and we are trying to we are interested in a particular i at term of that vector, but then we will use it in the vector notation, okay. Now I do not know if

you have done calculus in linear algebra. Whether you have done it or whether you have not done it you can verify it either which way so here essentially what is going to happen is that if b vector is equal to a + c vector then it is still be the 1 vector.

If b vector is equal to a dot c vector, now this dot, this special symbol is what is called the Hadamard product or the inner product where the a vector and the b vector are multiplied element wise. They are also called element-wise product, okay. So then there it will become the b bar element-wise product with c.

And if it is 1 - b square, then it will become 1 - b Hadamard product with b which is and then b bar will be also Hadamard product. This you can take from me on face value, but you can verify it in the slides.

(Refer Slide Time: 09:23)



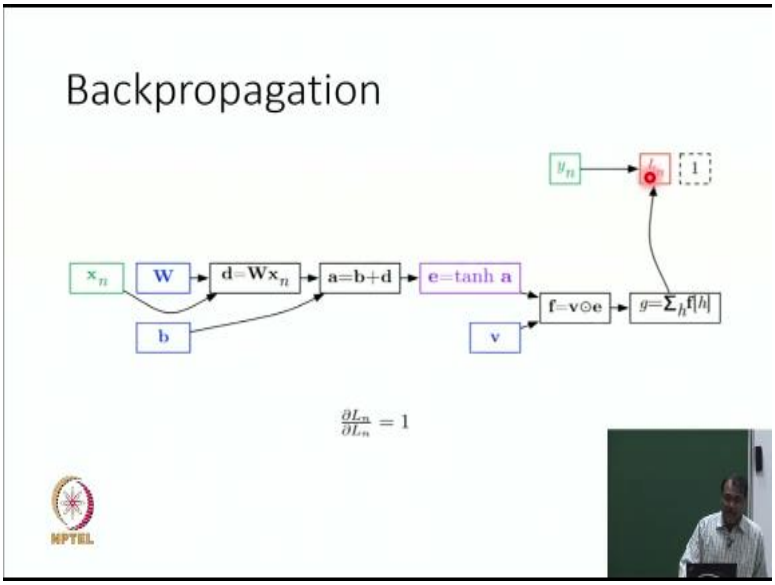
So now let us look at this particular example. okay. Now in this particular example, let us look at what is it computing? Let us say x n is the input. And you compute W x n plus b and tanh on top of it. So tanh of w x n plus b is what e is. But we have split it into in unit steps. So first x n came, W came and that created W x n. Let us call it d. Now b came and that we added b + d. So that created W x n + b, let us call it a.

Then we computed e equal to tanh (a). We call it e. Then let us say something else happen. Let us say we did a Hadamard product between v and e. So let us say f is equal to v Hadamard product with e. Then let us say g is nothing but sum over all

elements of this vector f to compute summation h f h. Let us say let that computed the loss function L n.

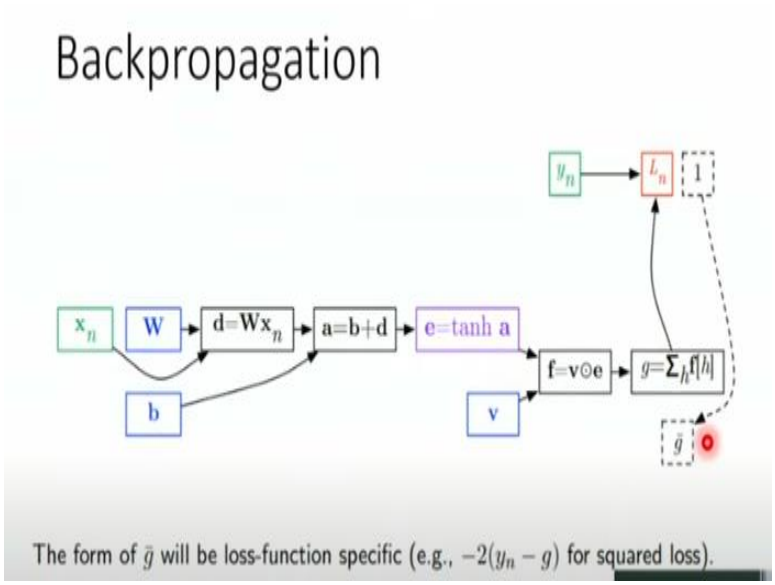
And that loss function would also have the gold output and the intermediate output. Let us say this intermediate output is a number and we want to make this number as close to y n as possible. Then the loss function would be y n - g whole square, right.

(Refer Slide Time: 10:57)



And now remember del n over del n is equal to 1, del n by del n, okay.

(Refer Slide Time: 11:17)

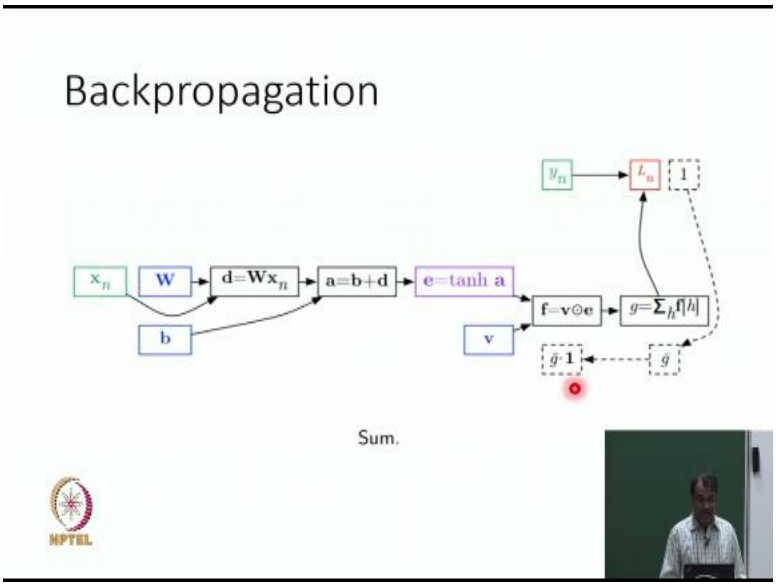


And the form of g will be function specific the loss function specific. For example, if the loss function is y n - g whole square then the derivative would be -2 times y n minus g. This would be g bar. So therefore the g bar computation we cannot do, that

depends on the loss function. If it is cross entropy, it will be the derivative of the cross entropy function. If it is the squared loss, it will be the derivative of the squared loss function and so on so forth.

But let us say we computed g bar. Now our goal is to backpropagate it all the way. This is the example that we have to work on. And let us start one step at a time. Now if I know g bar, which bar will I compute next? f bar. Why because g depends on f. And in particular g depends such that it is a sum of all the f i's or f h's; f i's let us just say f i's. So derivative of g with respect to f i would be because it is just a summation, it will be 1.

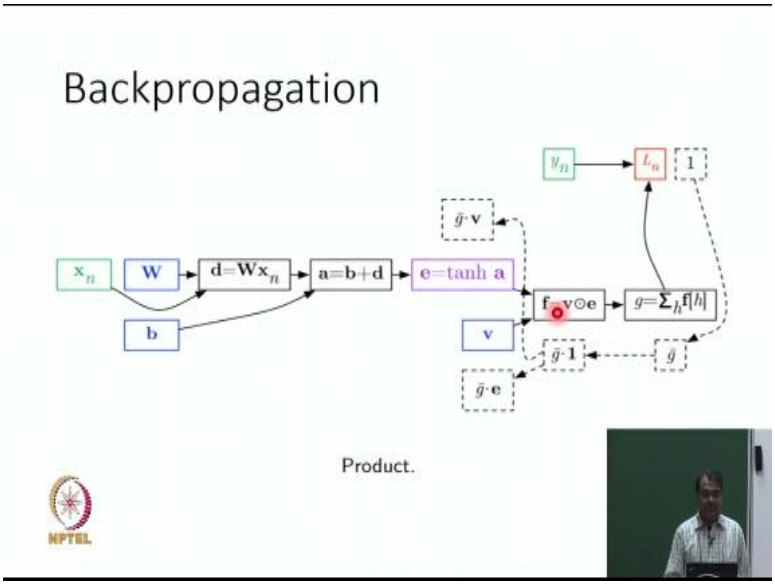
(Refer Slide Time: 12:43)



So therefore f bar would be same as g bar. Please try to work with me on this. This is where all your concentration is and focus is needed. Let us look at the segment. So what is f bar? f bar is nothing but g bar times the dot with 1 which is same as g bar, okay. Now f is computed based on v and e; f is computed based on v and e. So now we will next compute the v bar function. Not here we will use what we just studied.

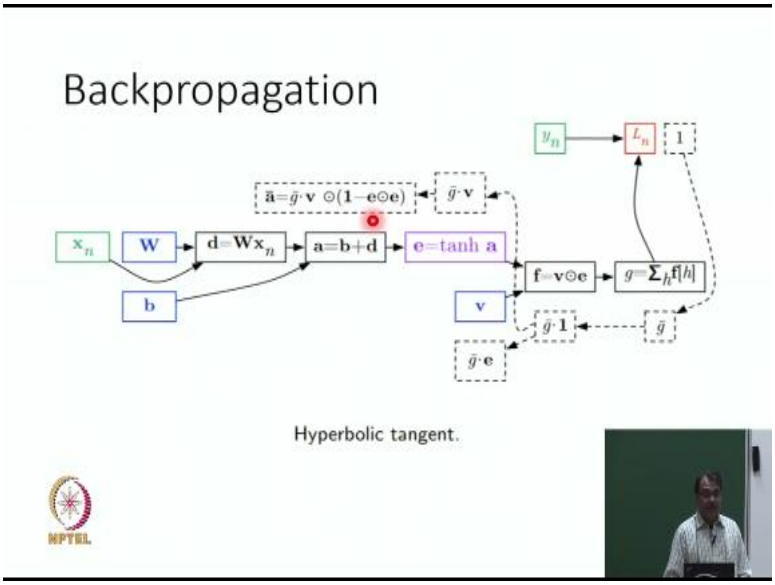
That if b is equal to a Hadamard product with c, then derivative of b with respect to a is c, right and we have to do b bar Hadamard product with c for the actual backpropagation. So therefore v bar will be equal to right. So let us see if we can get somebody to answer this question. So what is v bar? g bar Hadamard product with e. Very good.

(Refer Slide Time: 14:02)



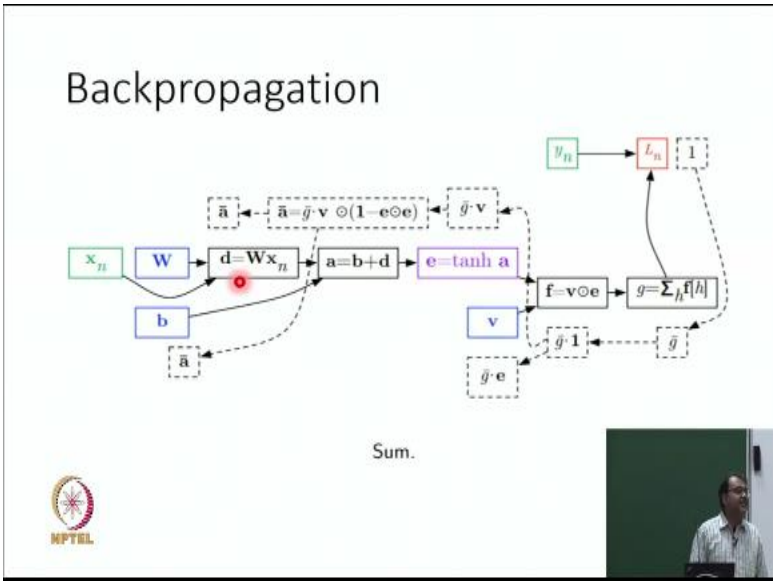
Similarly for the other one because they are symmetric. So v bar became g bar product with e and e bar became g bar product with v. These are vector products. Now this is equal to e bar. Now that allows us to compute a bar and because it is a tanh function we can use what we just studied and do 1 - b square or b dot d as our formula.

(Refer Slide Time: 14:37)



So therefore we are able to compute a bar as nothing but g bar dot v Hadamard product with 1 – e, Hadamard product with e. Now I have computed a bar. a is equal to b + d. So derivative of a with respect to b will be 1 and derivative of a with respect to d would be 1.

(Refer Slide Time: 15:11)



So therefore the d bar will be a bar, b bar will also be a bar. Now we have computed, now if you think about it what is going on, what were the parameters that we had? w, b, v. These are my 3 parameters and I have computed derivative of loss with respect to v. I have computed derivative of loss with respect to b, and I am now left with derivative of loss with respect to w, okay. Now this d bar is a vector.

This x n is also a vector. But this w i is a matrix. And the matrix and the vector x n are getting computed to compute the vector d. Now I have to differentiate d with respect to w ij and then do it in the matrix form. Suppose I have computed the derivative of d. So derivative of loss with respect to d. That is d Bar and let us say that comes out to be a bar because a was equal to b + d.

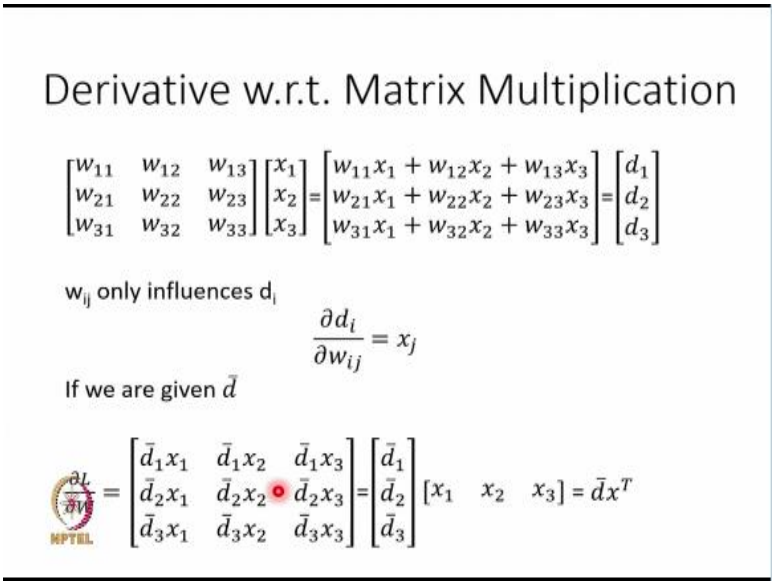
So derivative of d is same as derivative of a right. I mean d bar is same as a bar not derivative of d. So now the last question. So now I am given that d is equal to w times x. Now w is a weight matrix, d is a vector that I sent out, x is also a vector that I get in. And I have computed d bar. And now with this d bar, I need to compute w bar, right? So what would be the expression? Okay. So what did you get? No, we do not want x bar. Wait.

What do we want? Do we want x bar or w bar? Come on, guys. This is not very hard. x is the input to the whole problem. So we cannot take derivative with respect to the input. The input is what the input is. Now, what is very interesting is the sometimes

you have the input created by another process. In that case you can take derivative with respect to the input. But that is extremely advanced. For now x is the input.

What are the parameters? w. So we need the derivative of loss with respect to w, which is w bar. So we need to compute w bar. So what is w bar? Yes, Jay. Okay, very good. So he has finally figured out he has figured out the final expression for the derivative and that expression is correct, but for everybody else, let us look at a specific example.

(Refer Slide Time: 18:07)



So if you are like me who gets confused with linear algebra, it is slower, but it is always a good idea to expand it out and check what is happening, okay. It is always it gives you the clarity. So actually what is happening is let us say I have a 3 cross 3 w matrix. Let us say I have 3 cross 1 vector of x's and when I multiply them I will compute again a vector of d's which is again 3 dimensional, right?

So what is going to happen is this is going to be the expression w 1 1 x 1 plus w 1 2 x 2 plus w 1 3 x 3 will be d 1 and so on and so on and so on. Now the main point to realize is that w ij only influences, you can see this, w ij only influences d i, okay. So that is an important aspect right? So when we take derivative with respect to w ij only d i derivative is going to flow back. The d i bar is going to flow back, okay.

So and moreover the derivative of d i with respect to w ij would be the derivative of d i with respect to w ij would be equal to x j, right because it is a simple addition and a

product and the product is with respect to x j. And now let us say the backpropagation d bar is flowing back. So let us say I am given d bar. So what I need to compute? I need to compute derivative of loss with respect to that w.

Now this is going to be what? The d l the partial derivative of l by w is going to be a vector a matrix a number how many w's do I have? 3 cross 3. So therefore this output because it can be with respect to any one of these is also going to be a matrix, right? So let us look at any one of them because if you look at any one of them life would be you know easy. So let us look at w 2 1 for example and look at this expression and let us say I am given d bar.

And remember we multiply these things. So what would be derivative of loss with respect to w 2 1? Which derivative will flow back? Which bar which d bar? d 1 bar, d 2 bar, d 3 bar? d 2 bar because look w 2 1 is only influencing d 2 right? So only d 2 bar will flow back and it would be multiplied with x 1 because that is the derivative that we have just computed which is the local derivative.

So basically I will get an expression like this. Everybody with me, right you have to get practice with this if you want to do deep planning, Specially if you do not want to use black box deep learning and you want to come up with new architectures. So at least for people now these things are becoming easier and easier, deep learning is getting democratized, you know faster than you believe.

Now you can just say okay run a CNN as a API call and then the whole CNN computation will happen and you will get some answer. If you want to run it at that level of abstraction that is a different story. But if you want to understand what is going on, allow yourself to debug, come up with new architectures. If you come up with new architecture I can assure you it will not work in the first instance because you would have made one or the other mistake.

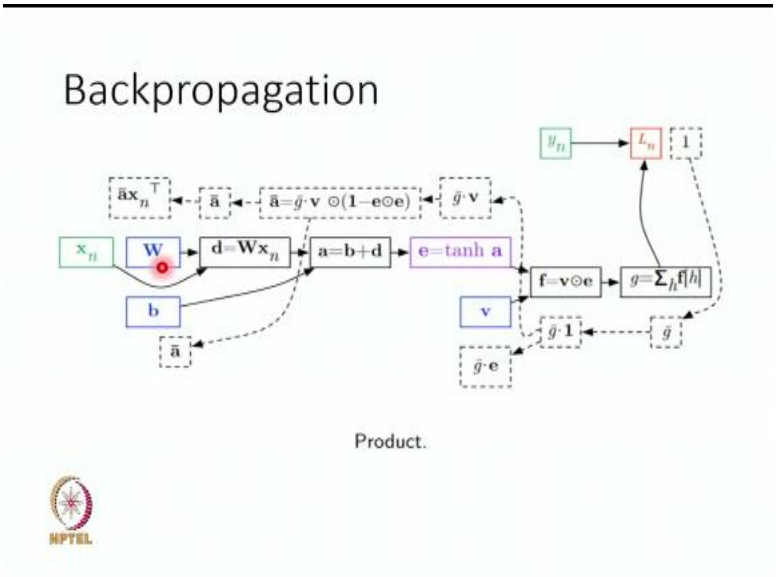
You would have to literally go inside, figure carefully what is happening, is the program computing the correct derivatives, are the derivatives flowing right, are there other in out of bound. So this that and the other. So you will have to be really you will have to really understand the math in order to be able to debug these things. So try to

get some practice with this. Now we want to convert this into a linear algebra notation.

So now I have to come up with 3 cross 3 matrix. Look, I have to somehow take one aspect, which is d bar, which is a 3 cross 1 matrix. Another aspect which is x vector, which is also 3 cross 1 matrix. How do I convert these two to result in this 3 cross 3 matrix? I multiply. But I multiply them how? I multiply them 3 cross 1 and 1 cross 3, right. So basically that means that I can think of it as a multiplication between the vector d bar and a transpose of the vector x.

This is sometimes also called the outer product. And when I do this, that is the final term. So d bar x transpose is the derivative of loss with respect to w.

(Refer Slide Time: 23:13)



And d bar was a bar. So therefore derivative with respect to w and a backpropagate would be a bar x n transpose, okay? All right, so that sort of wraps up our understanding of backpropagation, again at a highest level. Each computation that the neural network is doing can be divided into a computation graph where at one node only one computation got done, either addition or a non-linearity or a multiplication or you know whatever dot product whatever it is, right.

Only one computation at a time. So when I say w non-linearity of w x + v we convert it into three steps; w goes in, x goes in, one multiplication; b goes in one addition, non-linearity goes in or right one non-linearity. So three steps. So similarly each

neural network can be converted into a slightly larger computation graph and then we back propagate the gradients at each step multiplying the gradient that is flowing back times the gradient of the edge and adding for all outgoing edges.

So I get from all outgoing edges gradients and multiply with all the local gradients and that and add them and that gives me the gradient of my node and then I sent it back, okay. This backpropagation algorithm as I think I already said became the basis for training these deep networks. Before backpropagation algorithm it was not clear how can we train such large network. Because if it is one layer network life is easy.

You know the input, you know the output you can figure out the weights. But in two layer network also we do not know the output of the intermediate layer, we do not know the input of the next layer. So things are very hard to train and if it is a very large network, we do not know exactly what the hidden layers are supposed to do. What should be the output at every step. We had no idea whatsoever.

So how do we train these things? Well this kind of a chain rule over derivatives became the basis for doing that. But even then it did not work and it finally got to work in the late 2000's, 2010's at a level where we wanted it to. And then there were many reasons for it. Some reasons were algorithmic which we will not get into. If you are interested check out something called drop out.

Drop out was one of the main reasons why initially a deep network started to do very well. They were many other ways to regularize these networks because these networks can have many local optima and not all local optima are very good. Then there were so regularization is a very important branch of machine learning which we are not going to talk about.

And then the hardware was not very strong to be able to do all these computations at a large scale and then data sets were also limited. So all these factors played a role. The three-way street of large data sets, strong GPUs and good algorithmic practices came together in training these deep neural networks that we have now. Okay.