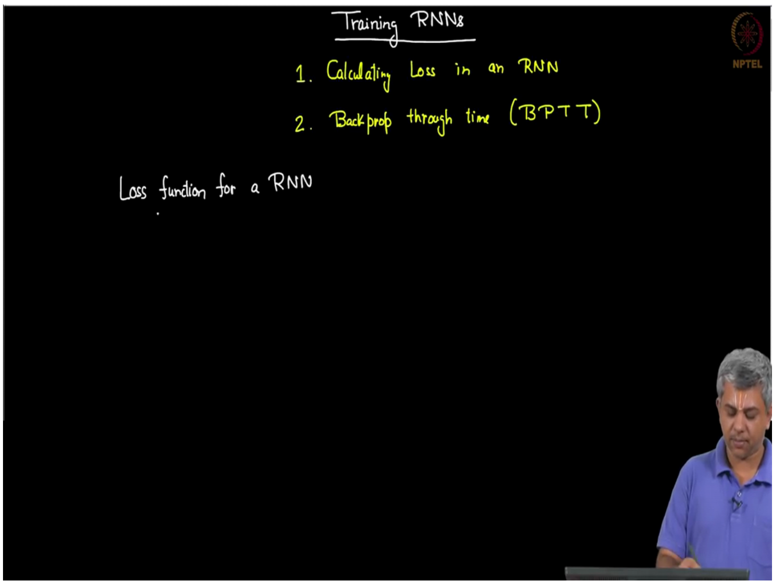
Machine Learning for Engineering and Science Applications
Professor Dr Balaji Srinivasan
Department of Mechanical Engineering
Indian Institute of Technology, Madras
Training RNNs

Welcome back. In this video you will be seeing how RNNs are trained. You will see that there are several commonalities between RNNs, and let us say CNNs and even ANNs, as far as training them is concerned. Of course, as usual you should have a training set, a validation set and a testing set. But that apart, given the particular structure of the RNN's, there are few certain things that you need to be aware of in terms of training. So we will just go through those in this video. There are several even deeper ideas that need to be conveyed, that we will not do, okay.

(Refer Slide Time: 1:19)



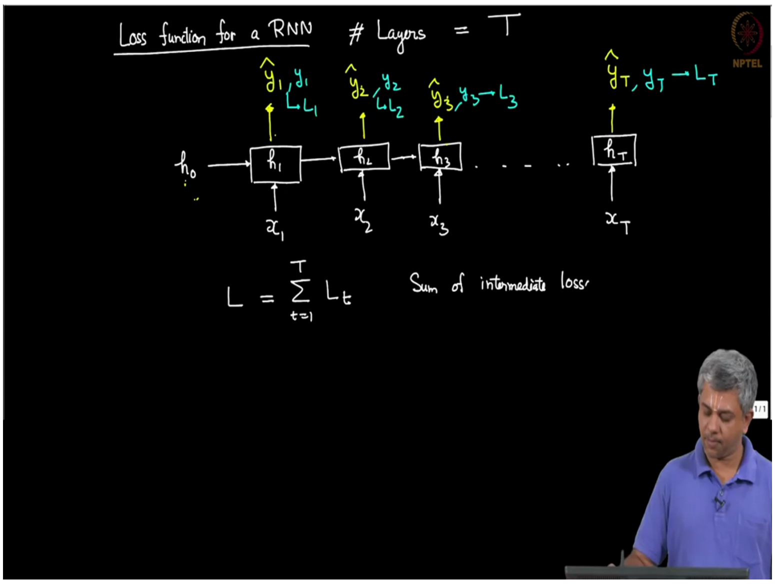
Training RNNs

1. Calculating Loss in an RNN
2. Backprop through time (BPTT)

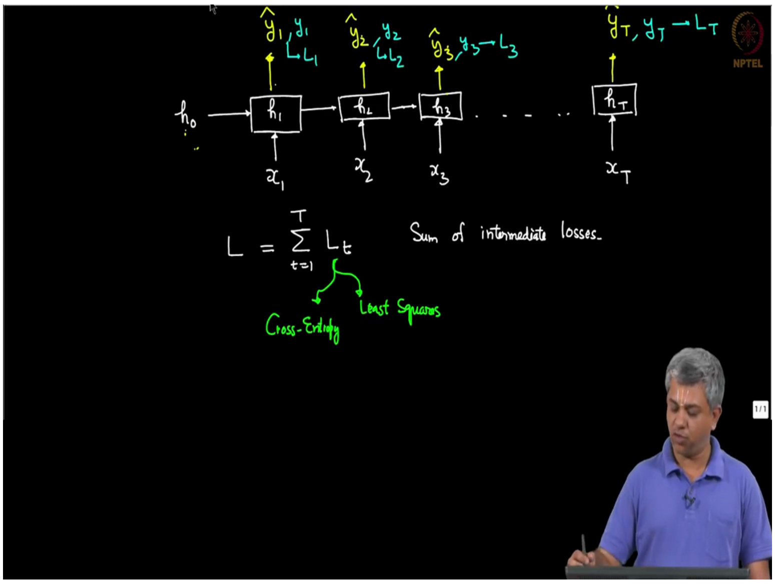
Loss function for a RNN

So in terms of implementation, luckily all the training has been abstracted into the various packages whether it is MATLAB or whether it is tensor flow or whether it is pie torch, etc. But there are a few ideas that will help you later on when you will try to train RNN's yourself. So the 2 issues that we will be concerned with in this video is 1st calculating loss in an RNN. There is a mild difference between what happens in an RNN and what happens in a CNN or let us say an ANN. The 2nd issue is what is called back propagation through time. Sometimes simply called BPTT.

(Refer Slide Time: 2:31)


Loss function for a RNN # Layers = T

$$L = \sum_{t=1}^{T} L_t$$ Sum of intermediate loss


$$L = \sum_{t=1}^{T} L_t$$ Sum of intermediate losses

So we will be looking at this from the overall view, I will not be doing too deep of mathematics, this little bit of mathematics, we will be doing this. And hopefully this will give you some insight into what actually goes them into the code when it tries and trains RNNs. So, 1st let us look at the loss function for an RNN. So let us see a simple structure. So, as usual you have some X0 going in, let us say we have unrolled an RNN through many many layers. So let us say the total number of layers is equal to capital T.

Why capital T? Because we are thinking of RNN as something that goes through time, so let us say this is T1, it is the 1st instant, T2 the 2nd instant, T3 3rd instant, so on and so forth and let say we are going to H capital T, okay. Now a question with an RNN usually is, where are we going to take out the outputs? And as we saw in the introductory videos, you have several choices, it depends on really what you want. In some cases you will be taking out an output only here but there are several possibilities where or several cases where you might be interested in let us say finding out outputs at all intermediate layers also.

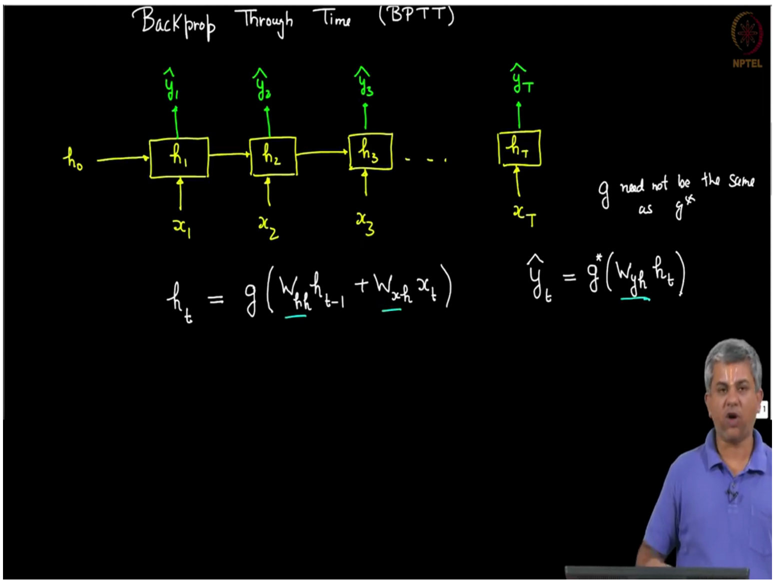
So just for consistency, I will call it Y hat, because that is what we had been calling our model or predicted values so far. Okay. Now when you have multiple predicted values, so let us take the example of, let us say having 10 days before is the weather of X0 or temperature of X0 in some city, let us say Chennai, okay. So suppose you have that input, you would have the next day's temperature, let us say that is Y1 hat, the next day's temperature Y2 hat, next day's temperature Y3 hat till let us say today's temperature which is YT hat.

Now for each one of them, you also have a corresponding ground truth, which should be Y1, Y2, Y3, YT. Okay, so this is the ground truth. And whenever you have a ground truth and a prediction and these 2 differ, you will have a loss function. Okay. So not only do you have losses right at the end, like we do add, let us say with ANN's or CNN's or at least usual architectures of them, you can have possibly, I mean this is not necessary, as I discussed earlier it could be optional. But suppose you do take out an output, you do have a corresponding loss function.

So the total loss is actually summation of all the intermediate losses through the layers. Okay. Now in terms of LT itself, or the local loss function, you again have many choices but we having seen only 2, you can either use cross entropy or you can use least-squares, depending on what sort of problem matters. Typically what we have done so far in this course is we have used least-squares, whenever it was a regression or a numerical output. For example let us say temperature today. And we have been using cross entropy in case it was a classification issue.

For example you could ask will it rain or not? And in such a case, you would probably use something like cross entropy is a loss function. In either case, will simply say that L is Sigma of LT from T equal to 1 to T. So this is the issue of calculating loss function, it is a simple deviation over or simple correction over all the previous loss functions that we have seen so far. Now more important idea or a subtler idea really is that of back propagation.

(Refer Slide Time: 6:48)



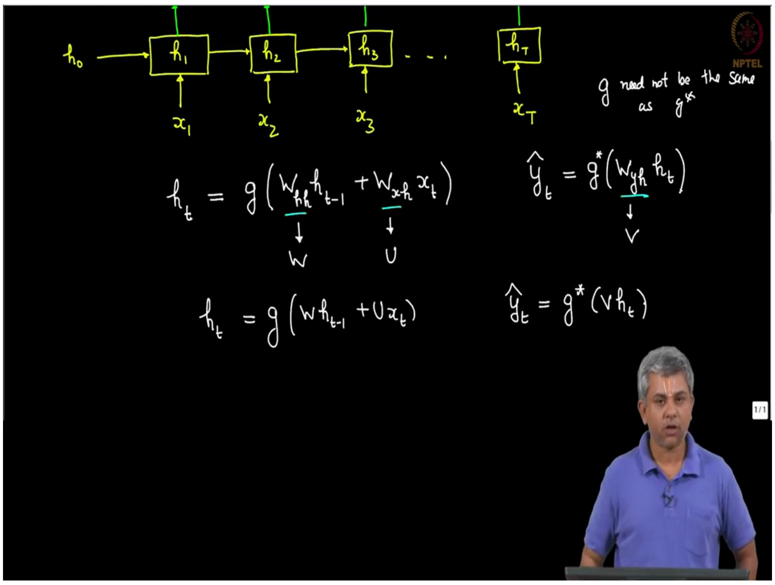
Backprop Through Time (BPTT)

g need not be the same as g^*

$$h_t = g\left(\underline{W_{hh}} h_{t-1} + \underline{W_{xh}} x_t\right) \qquad \hat{y}_t = g^*\left(\underline{W_{yh}} h_t\right)$$

So let us look at back prop through time, we will call it B PTT for short. Now let us say what the subtle issues involved here are. Okay. So, as usual, we will assume this, there is H1, we start with an H0, the input here is X1, H2, X2, H3, X3, so on and so forth up until HT, XT. And for now we will assume that we are taking out an output at every single time instant. Okay. Now let us write the expression, for any of this H, we have this HT is some G, usually tan H of some weight matrix multiplying HT -1 which we called WHH + some weight matrix multiplying XT which we called WX H and we also had YT hat was some nonlinearity, remember I will call this let us say G star.

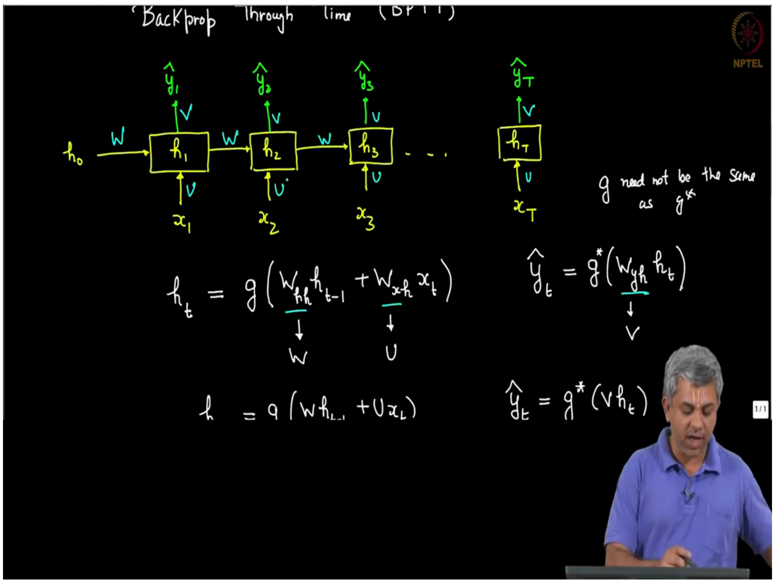
(Refer Slide Time: 9:30)



g need not be the same as g^*

$$h_t = g\left(\underset{\substack{\downarrow \\ W}}{\underline{W_{hh}}} h_{t-1} + \underset{\substack{\downarrow \\ U}}{\underline{W_{xh}}} x_t\right) \qquad \hat{y}_t = g^*\left(\underset{\substack{\downarrow \\ V}}{\underline{W_{yh}}} h_t\right)$$

$$h_t = g\left(W h_{t-1} + U x_t\right) \qquad \hat{y}_t = g^*\left(V h_t\right)$$

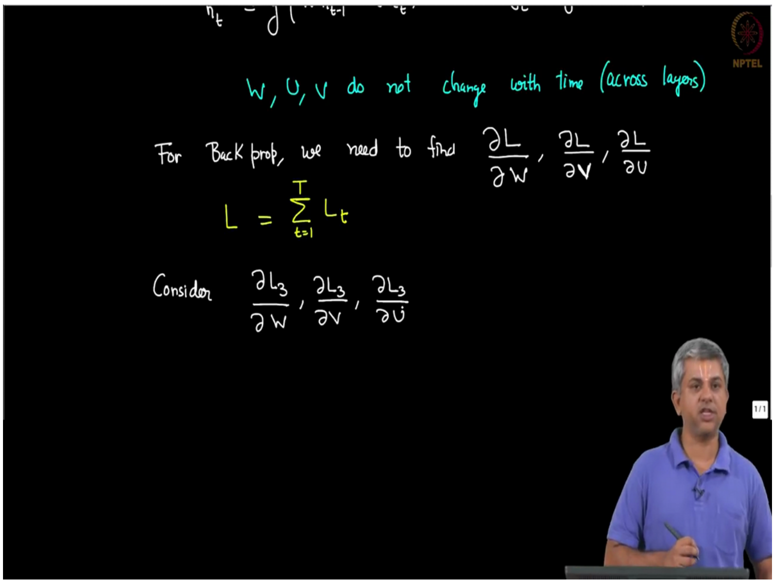

Backprop Through time (BPTT)
$\hat{y}_1$ $\hat{y}_2$ $\hat{y}_3$ $\hat{y}_T$
V V V V
h_0 W h_1 W h_2 W h_3 . . . h_T
U U U U
x_1 x_2 x_3 x_T
g need not be the same as g*
$h_t = g(W_{hh} h_{t-1} + W_{xh} x_t)$
W U
$\hat{y}_t = g^*(W_{yh} h_t)$
V
$h = g(W h_{t-1} + U x_t)$
$\hat{y}_t = g^*(V h_t)$
NPTEL

G need not be the same as G star, that is we can use a different nonlinearity here and a different nonlinearity here, in fact in several cases we simply use a linear prediction here, okay. So we call this WYH times HT. So, these are the 3 sets of matrices. Now for this video and for the few that follow, just for simplicity we will use a different rotation. We will call this matrix W, we will call this matrix U and we will call this matrix V, so that I am going to write HT is g of W HT -1 + U HT YT hat is some G star of V HT, okay.

Now the most important thing about an RNN is W, U and V do not change with time. That is another way of saying this across layers. So, this is what it actually makes us, it is possible to train RNNs at least with a reasonable amount of time. So just to clarify, I have this H1 which was this multiplied by W and this multiplied by U and this H1 multiplied by V with a nonlinearity of course in all cases, gave me a Y1 hat. Now the W here, U here and V here are exactly the same, okay. And you will see how that plays out in, when we do back propagation.

So unlike an ANN, where at each player, these Ws, Us and Vs actually change, in an RNN, they are exactly the same. So, we use the same W, U, V for each layer. Now how does, it helps us of course, now we have fewer parameters to train. And while doing back propagation we have to be a little bit careful.

(Refer Slide Time: 11:23)



W, U, V do not change with time (across layers)

For Back prop, we need to find $\frac{\partial L}{\partial W}, \frac{\partial L}{\partial V}, \frac{\partial L}{\partial U}$

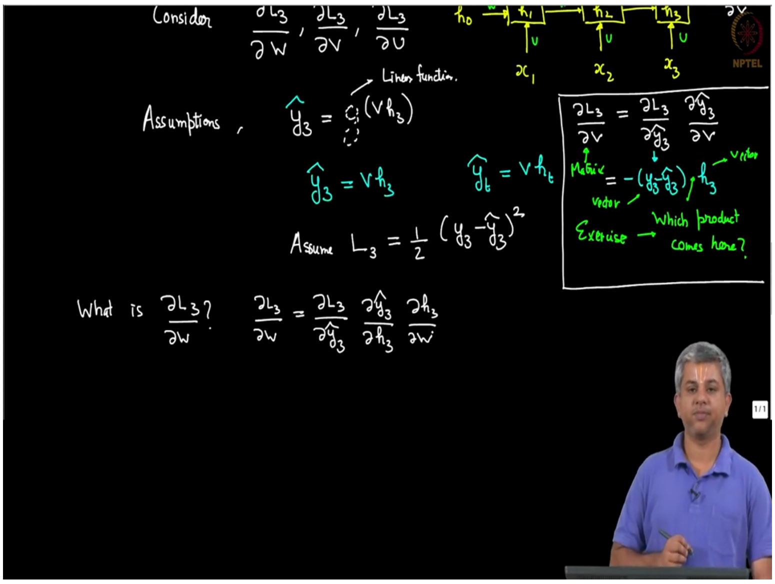
$$L = \sum_{t=1}^{T} L_t$$

Consider $\frac{\partial L_3}{\partial W}, \frac{\partial L_3}{\partial V}, \frac{\partial L_3}{\partial U}$

So let us take a specific case. So remember I want to find out for back prop, we need to find Del of the lost function, I am calling it L here, you can use J or L, depending on what you are comfortable with, for now I am using L, with all the matrixes. Now, this means we need to find out Del L del W, Del L Del V and Del L del U. Because we have 3 matrices as far as RNN's are concerned. So, this sort of RNN that I showed you has 3 matrices, U, V and W and I willing to find out Del L with respect to each, the gradient of L with respect to each of these weights. Okay.

Now remember that L itself is a summation of LT, that is we find out the loss at each of these layers, let us say I will call it L1, L2 as I did just a little bit before and I need to find out Del L1 these 3, Del L2 these 3 and then sum these up and that will give you, that will give me Del L with respect to any of these 3. So let us consider just the sum of one of these. So let us consider Del L3, del W, just to show you what happens, okay. So we will consider just one of these local losses and see how we can apply back propagation in using this, okay.

(Refer Slide Time: 13:27)



Consider $\frac{\partial L_3}{\partial W}, \frac{\partial L_3}{\partial V}, \frac{\partial L_3}{\partial U}$

Assumptions, $\hat{y}_3 = g(Vh_3)$ → Linear function.

$\hat{y}_3 = Vh_3$ $\qquad \hat{y}_t = Vh_t$

Assume $L_3 = \frac{1}{2}(y_3 - \hat{y}_3)^2$

What is $\frac{\partial L_3}{\partial W}$? $\quad \frac{\partial L_3}{\partial W} = \frac{\partial L_3}{\partial \hat{y}_3} \frac{\partial \hat{y}_3}{\partial h_3} \frac{\partial h_3}{\partial W}$

$\frac{\partial L_3}{\partial V} = \frac{\partial L_3}{\partial \hat{y}_3} \frac{\partial \hat{y}_3}{\partial V}$

Matrix

$= -(y_3 - \hat{y}_3) , h_3$ → vector

vector

Exercise → Which product comes here?

Okay, I will draw a small figure here just to repeat what we had before for clarity. For H1, H0, X1, Y1 hat, H2, X2, Y2 hat, H3, X3, Y3 hat and Y3 hat leads to L3 because Y3 hat in general will be different from Y3. So the matrix here involved is V, here U, W and V, okay. No before we proceed, I am going to make some assumptions on the structure, okay. I am going to assume that the nonlinearity here, remember Y3 is some non-linear function which I said was optional, of V times H3, we will assume that the nonlinearity here is simply just the linear function, or it is a linear activation function.

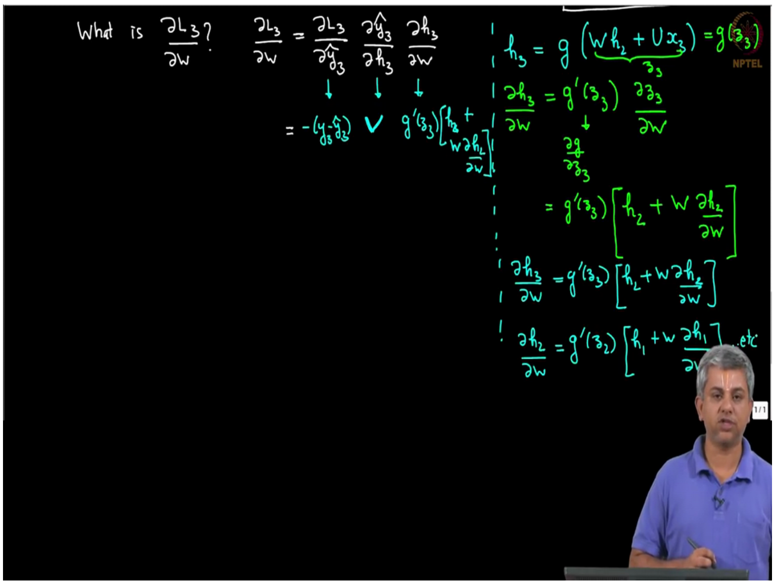
That is just to make some of our derivation a little bit simpler. You can do it for any case. So I will assume that Y3 is V times H3 and in general YT is V times HT, you know that is not going to matter as far as this video is concerned. Okay, next thing is we will assume that the loss function is a least square function. So, let me make it simpler. So, I will call it, let us just deal with L3, this is half of Y3 - Y3 hat square. That is just to make up a differentiation original but easy. Once again you can do this kind of derivation for any loss function.

Now given these 2, what do you need to find out? We want to find out, let me do it here while the figure exists. We want to find out Del L3 del V. Let us say, let us start with that. So let us say I want to Del L3 Del V. How would I do it? Like we did before, Del L3 del V is simply Del L3 del Y3 hat, del Y3 hat Del V. Now Del L3 del Y3 hat, you can see here is simply - Y3 - Y3 hat, okay. So that is fairly straightforward, okay. You can just simply differentiate this, as we did before even while doing ANN's, okay. Now what about del Y3 hat, I should really call it Y3 hat here.

Del Y3 hat Del V is now H3, there is a small catch here, I will mention that here now. I leave this as an exercise, we will be asking this within this week's exercise. Find out whether the matrix sizes. Remember L3 is a scalar, V is a matrix, so this whole thing is actually a matrix, okay. Y3 - Y3 hat is a vector, H3 is also a vector, so please think about which sort of product should come here or how should you are before that you get a matrix out of this order. So please think about this, we will be giving this as one of the exercise questions.

Regardless, what I will write a is simply that Del L3 del B is equal to Y - Y hat times H3. The simply give you how much will the loss change suppose I were to change V. That is fairly straightforward, how much will this loss change suppose I were to change V. Now, there is a subtler question here or a harder question here, which is, what is Del L3 del W? Why is this a harder question? So let us 1st start doing a similar exercise to the one that we have just done. So, suppose I want Del L3 del W, mathematically Del L3 del W will be Del L3 del Y3 hat as we had before, differentiate this with respect to this, multiplied by del Y3 hat del H3, because Y3 had depends on H3.

(Refer Slide Time: 19:22)



What is $\frac{\partial L_3}{\partial W}$?

$$\frac{\partial L_3}{\partial W} = \underset{\downarrow}{\frac{\partial L_3}{\partial \hat{y}_3}}\ \underset{\downarrow}{\frac{\partial \hat{y}_3}{\partial h_3}}\ \underset{\downarrow}{\frac{\partial h_3}{\partial W}}$$

$$= -(y_3-\hat{y}_3)\ V\ g'(z_3)\left[h_2 + W\frac{\partial h_2}{\partial W}\right]$$

$$h_3 = g(\underbrace{Wh_2 + Ux_3}_{z_3}) = g(z_3)$$

$$\frac{\partial h_3}{\partial W} = \underset{\underset{\frac{\partial g}{\partial z_3}}{\downarrow}}{g'(z_3)}\ \frac{\partial z_3}{\partial W}$$

$$= g'(z_3)\left[h_2 + W\frac{\partial h_2}{\partial W}\right]$$

$$\frac{\partial h_3}{\partial W} = g'(z_3)\left[h_2 + W\frac{\partial h_2}{\partial W}\right]$$

$$\frac{\partial h_2}{\partial W} = g'(z_2)\left[h_1 + W\frac{\partial h_1}{\partial W}\right] \text{ etc}$$

Multiplied by del H3 del W, I hope this is clear. We just saw that Del L3 del Y3 hat is simply - Y3 - Y3 hat, that should be straightforward. Similarly if you come down here, whatever del Y3 hat, del H3, please notice this, this is simply V. Now what about Del H3 del W? Del H3 del W, for that we need to know what the expression for H3 is. H3, recall was G of some nonlinearity, usually tan H of, let us see this here. H3 is W times H2 + U times X3, just to recall you can see this. Okay.

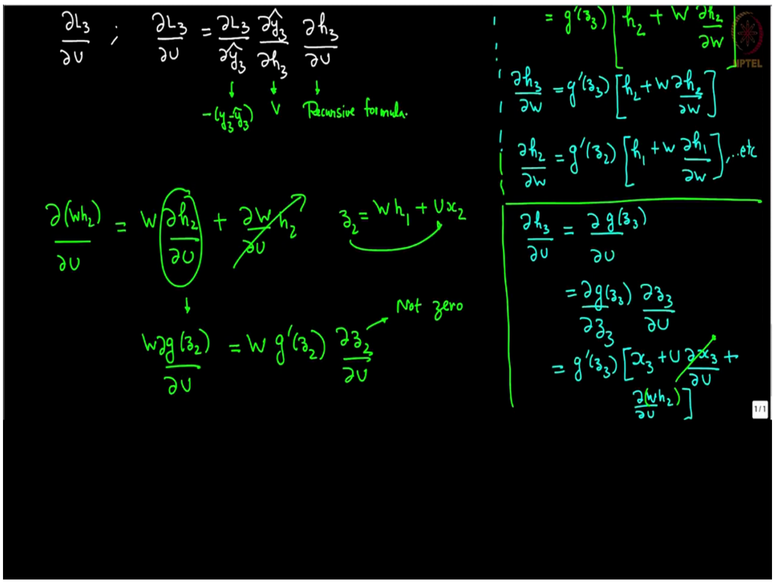
We want Del H3 del W. So for simplification let us call this Z3, okay. So this is therefore g of Z3, very similar to what we had for ANN's. H is G times, G of Z, or activation is G of Z, it is very similar. So, if you have that, then this becomes G Prime Z3, that is Del H3 Del Z multiplied by del Z3 del W, okay. Okay. Let us continue in the same vein. So this gives us G Prime of Z3 multiplied by Del Z3 DLW. So now let us look at this. What is del Z3 del W? Obviously none of these terms depend on del W, so this derivative goes to 0.

You have this derivative, this should give us H2 which is straightforward, + there is one more term which is W times Del H2 del W. Now, why does this tom exist, okay, is this 0 or is this nonzero? This is nonzero because notice that H2 itself depends on W, okay. Just like H3 is dependent on W, H2 also dependent on W, depends on W because it is the same W Throughout. This is the catch with back propagation through time. Unlike ANN where you have a W1 here and a W-2 there, in the RNN, it is the same W Everywhere.

So that you cannot find this out independent of Del H2 del W, okay. So, in summary you have Del H3 del W equal to G Prime Z3 H2 + W Del H2 del W. Now suppose you want a list to del W, you have to go back again. I this is why it is called back propagation through time. So you have G Prime of Z2 H1 + W times Del H1 del W, etc., okay. So when you, whenever you want to find out Del L3 with respect to del W, gradient of L3 with respect to W, you will 1st come down here, okay.

You have these 2 terms sitting there, but this is actually a more complex term, it is G Prime Z3 times H2 + W times Del H2 del W. And in order to calculate Del H2 del W, you will have to go back, okay and so on and so forth. So similarly you will, whenever you find to find out let us say Del LT del W, it will involve all the gradients before. So you will have these repeated sort of recursive additions sitting there and there are very clever ways of writing these codes, as people have already done within tensor flow, etc.

(Refer Slide Time: 24:49)



$$\frac{\partial L_3}{\partial U} \; ; \quad \frac{\partial L_3}{\partial U} = \underbrace{\frac{\partial L_3}{\partial \hat{y}_3}}_{-(y_3 - \hat{y}_3)} \underbrace{\frac{\partial \hat{y}_3}{\partial h_3}}_{V} \underbrace{\frac{\partial h_3}{\partial U}}_{\text{Recursive formula.}}$$

$$\frac{\partial (Wh_2)}{\partial U} = W\frac{\partial h_2}{\partial U} + \cancel{\frac{\partial W}{\partial U}} h_2 \qquad z_2 = Wh_1 + Ux_2$$

$$W\frac{\partial g(z_2)}{\partial U} = W g'(z_2) \frac{\partial z_2}{\partial U}$$

Not zero

$$= g'(z_3)\left[h_2 + W\frac{\partial h_2}{\partial W}\right]$$

$$\frac{\partial h_3}{\partial W} = g'(z_3)\left[h_2 + W\frac{\partial h_2}{\partial W}\right]$$

$$\frac{\partial h_2}{\partial W} = g'(z_2)\left[h_1 + W\frac{\partial h_1}{\partial W}\right], \ldots \text{etc}$$

$$\frac{\partial h_3}{\partial U} = \frac{\partial g(z_3)}{\partial U}$$

$$= \frac{\partial g(z_3)}{\partial z_3}\frac{\partial z_3}{\partial U}$$

$$= g'(z_3)\left[x_3 + U\cancel{\frac{\partial x_3}{\partial U}} + \frac{\partial (Wh_2)}{\partial U}\right]$$

So we have now seen 2 terms, we have seen Del L3 del W, we have also seen Del L3 Del V, which was straightforward. Finally let us look at Del L3 del U. So, once again the same thing, the 1st 2 terms will be the same, Del L3 del U is equal to Del L3 del Y3 hat multiplied by del Y3 hat del H3 multiplied by del H3 del U. Okay. Now this is anything a is mildly trickier than the previous one, this is V. When I was doing Del L3 del W, you could see that this depends on this, this depends on this, this depends on W, okay. And through the W it comes through here.

Now it looks like this should go through when we take a credit with respect to U, that it should go through straightforward but there is some subtlety thereto. So let us calculate this term. So, suppose I want Del H3 del U, remember this is Del by del U of H3 is G of Z3. So this is Del g Z3 with respect to Z3 and del Z3 del U, okay. This is G Prime Z3 times Z3, we have it here right here in front of us. So what is del del U of that? What remains is X3 + U times, you have Del del U of this term here, it W times H2, okay.

Now it might seem like, this tom is of course 0 because H3, X3 does not depend on U at all. Now what about this term? This term is like I said a little bit subtle. So if I look at the term Del of WH 2 with respect to del U, this can be written as W Del H2 del U + this term is 0. This term is not 0 because this is W times G of Z2 Del T of Z2 del U, which in turn is W times G Prime Z2 times Del Z2 del U and this is not 0. Why is that, because Z2 is equal to W times H1 + U times X2 and it depends on U.

So this is very similar to what we did with this, there is a recursion there, okay. So, in all 3 cases, well in both the cases, in the case of W as well as U, you have a dependency which is actually sitting there which will make you back proper get through time. You cannot simply find out the gradient of L3 with respect to W or U without finding the gradient of L3 with respect to this throughout time. And this is why sophisticated expressions exist. When we look at, in the next couple of videos when we look at deep RNN's, you will see that this issue can actually become a little bit more complicated.

Which is why tensor flow for example has a full graph. So all these dependencies are resolved in terms of graphs by using automatic differentiation. And back propagation uses automatic differentiation, one sort of the other. So, that is it for back propagation through time. The basic idea for you was to see that training can be a little bit more complex. In practice, unless you are writing some new architectures by yourself or entirely new architectures, something that nobody has ever thought of, you will not really be doing this by hand and ever. But this is slightly important for you to see or at least get an intuition of what is happening in and as well as to see the next video where we will be dealing with vanishing or exploding variance, thank you.