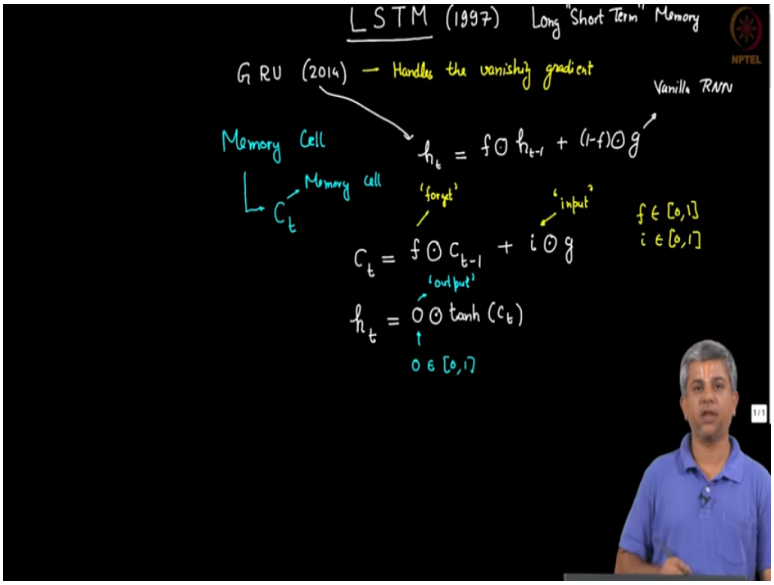
Machine learning for Engineering and Science Application
Professor Balaji Srinivasan
Department of Mechanical Engineering
Indian Institute of Technology Madras
Long Short Term Memory

(Refer Slide Time: 00:13)



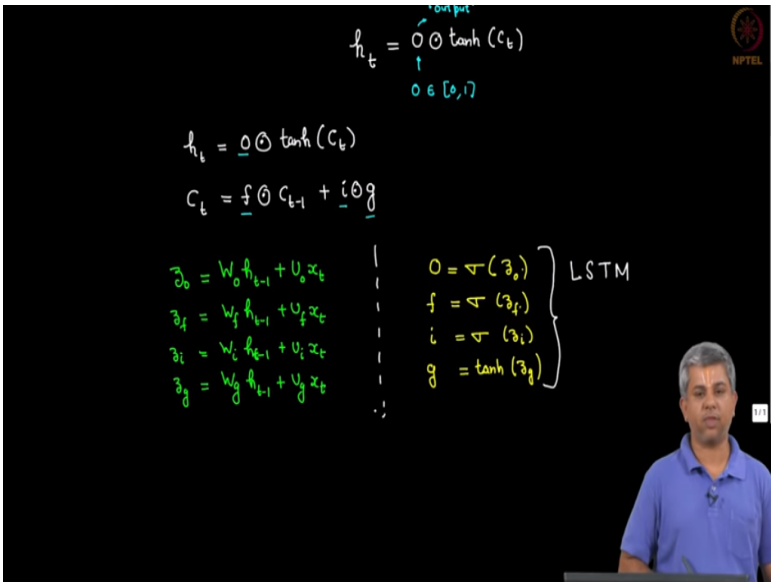
Welcome back in the previous video we saw some variations of the RNN structure using GRU gated recurrent unit GRU of course I a recent version 2014and we saw that by introducing new weights and a slightly more complicated structure we could probably handle the vanishing gradient issue, so GRU of course was not the first architecture to handle this the oldest architecture to do that was LSTM in 1997 and this is still the industry standard I many ways though as you will see the structure is slightly more a bit more complicated than GRU but not by very much if you go the ideas in the previous video the LSTM idea should also be fairly clear.

So what is LSTM, LSTM is stand for as I had said in last video long short-term video memory, so remember the short term sits together and LSTM was the first architecture to use the idea of a separate memory cell so when we were dealing with GRU or the simplified GRU we had something like HT I F times HT minus one plus lets us say one minus F times G, where G was the output of vanilla RNN and the idea there was to retain some portion of you old calculations into other new ones in the case of LSTM we will be actually using a separate cell all together this is a memory cell, so heuristically this is sort of like having some numbers you retain it in a

separate memory back and while you are calculating with some other numbers, so I will just show you the formulation.

So we write CT notice the analogy with what we had before is F times this is once again the Hadamard product or the element wise multiplication product F times CT minus one plus I time G where I is now called the input gate FS like last time it is called the forget gate, so the ideas are very-very similar once again we want F to belong to zero one similarly I should also belong to zero one but this gives you only see T what happens to H, H is the output that we are actually interested in H is now given as O times tan H of CT and O once again is another valve or another gate which also belongs to zero one and it I called the output gate.

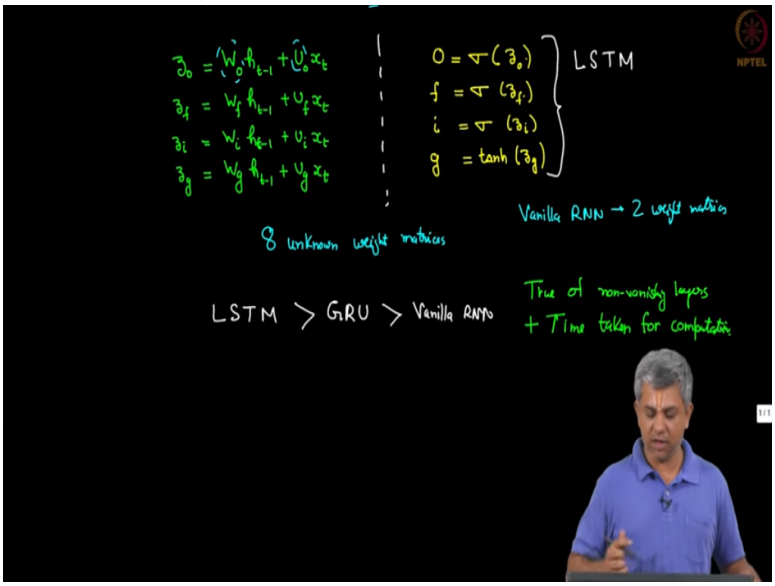
(Refer Slide Time: 04:08)



So now we have three gates FIO and we also have to predict G so just to write a summary of how we calculate LSTM, LSTM is calculated as HT id O times tan H of or some other non-linear of CT where C is the memory and CT itself is calculated as F times CT minus one plus I times G and now all this parameters need some definitions and these definition you could probably write down intuitively eve before I do and I would recommend that maybe you pause the video and try it once just to make sure that you have understood things but I will quickly write them down in a second so these ideas are very similar to the once we had used in simplified GRU and even in GRUR so the idea is simple since O has to belong to zero one you say O is sigmoid of some ZO.

Where Z will be a linear combination of what came in similarly F will be sigmoid of ZF similarly I the input gate will be sigmoid of some ZI and G being the output of a vanilla RNN is simply tan H of ZG , now what are these ZO ZF ZI ZG we can write them down pretty easily ZO will be WO HT minus one plus UO XT, similarly ZF will be WF HT minus one plus UF XD, ZI is WI HT minus one plus UG XT and finally ZG is WG HT minus one plus UG XT all these put together give you LSTM.

(Refer Slide Time: 06:51)



Now if you see LSTM it has how many unknowns you know what ever be the size of the W matrices you have 8 unknown weight matrices just for comparison plain vanilla RNN and just has two weight matrices.

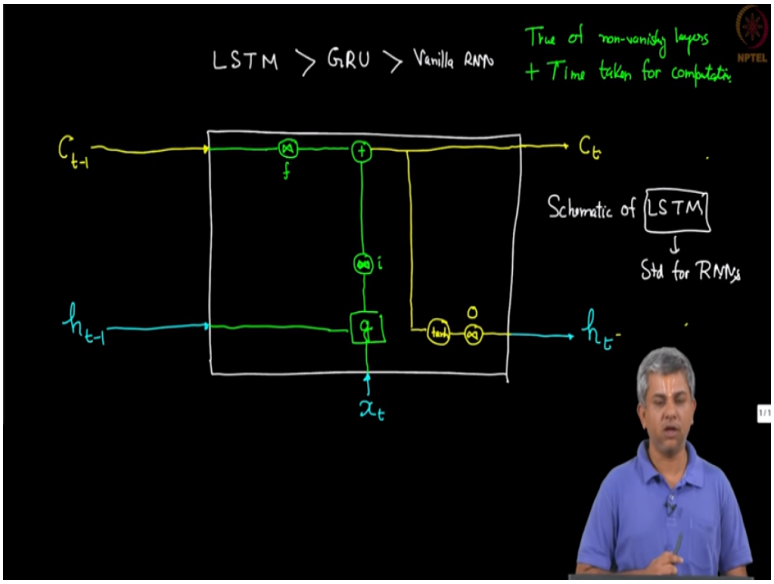
Now if you recall when we were doing back propagation through time we had to find out both this weight matrices including of course the output matrices I have not talked about that here but if you have output you have to find out that propagation for that though the output matrix back prop as you had seen in the back propagation through time was straight forward now if you use all these eight you will have to do back propagation for all eight of these matrices so that what will change and if you simplify GRU we saw that there were six matrices etcetera and sorry simplified GRU had four and GRU had six matrices.

So it is just a question of how much expenditure you are willing to bear , now LSTM typically can trained or can retain non vanishing gradient for greater number of layers compared to GRU

and GRU typically can retain greater number of layers compared to vanilla RNN and so this is remember when I say LSTM greater what it means is the number of layers that you can train with LSTM the depth of the architecture can be greater with LSTM compared to GRU and that compared to vanilla RNN and that you have to balance against typically the number of weights that you have to train, so of course this is also true of non-vanishing layers that you can train plus time taken for computation.

So LSTM will typically be more difficult to train in term of computation time it will take greater time for you to train and it will also take you slightly more time to run because it has more matrices in there, so everything is grater about LSTM now typically a rule of thumb that is suggested at least in modern days meaning just as or four years is that you try vanilla RNN task and if it is a small number of layers if that works well enough good otherwise try GRU on a task and if that work at well enough good if not then try LSTM of course depending on what peoples priority are many people typically tend to use LSTM of the bat.

(Refer Slide Time: 09:59)



So that is certainly a possibility now just to repeat the exercise that we did with GRU and simplified GRU I will also draw sort of the diagram for LSTM remember now that we have not only HT and XT coming in into these box which is finally going to spit out sorry there should be HT minus one, HT comes out but not only that you also have your memory cell or memory computation so we also have CT minus one coming in and CT going got and so on and so forth.

So CT progresses HT progresses and there is some processing that happens inside which was given by our formulation above now what was that HT minus one and XT combine as usual to give our vanilla RNN output now CT minus one if you remember there is valve here it looks like infinity but it is actually evolved so I gets multiplied by G the forget gate gets multiplied by CT the two combine and this is what gives us CT as output, at the same time the same CT comes down here you run it through a tan H, run it through the output gate and what you get is HT.

So this is simple schematic of LSTM now this can be shown an very different complex ways but I like this because this kind of tells you what the mathematics is doing sort of in a simple way you can also see several version of this online each person has their own diagram of LSTM GRY etcetera, I prefer this you do not really have to learn it is for those people who prefer visualizations to arithmetic or algebraic formulae, so LSTM is the industry standard for RNN's you can blindly use LSTM more or less today for any RNN task that you see a warning is that it takes a long time to train for many-many most of the language task that we are interested in thank you.