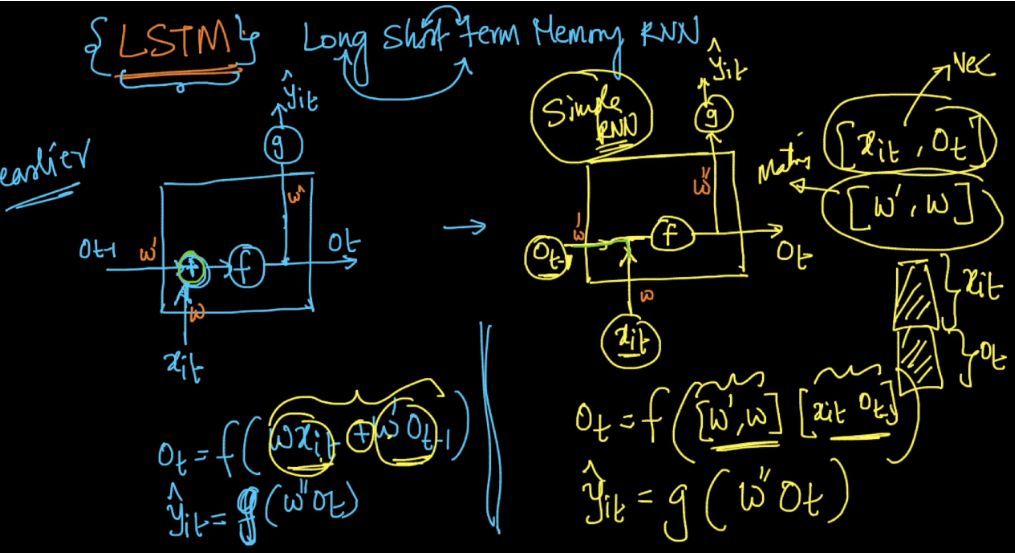
**LSTM:**

Earlier we’ve seen that we were adding ot-1 and xit, now we’ll merge them. By merge here means is they are concatenate,

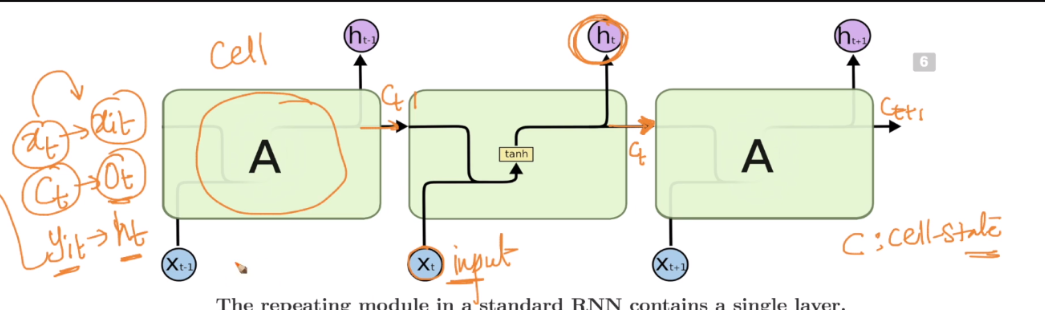
weights concatenation: [w’, w], i/p concatenation: [xit, ot-1]



Notations:

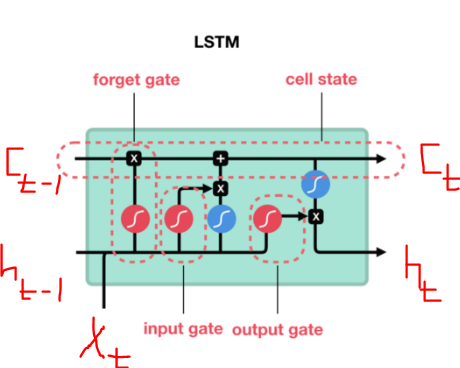
Earlier we were using ot as state ouput now we use ct which is also called cell state.

Yt will be use as ht



LSTM is shown below basically there are 3 units/gate:

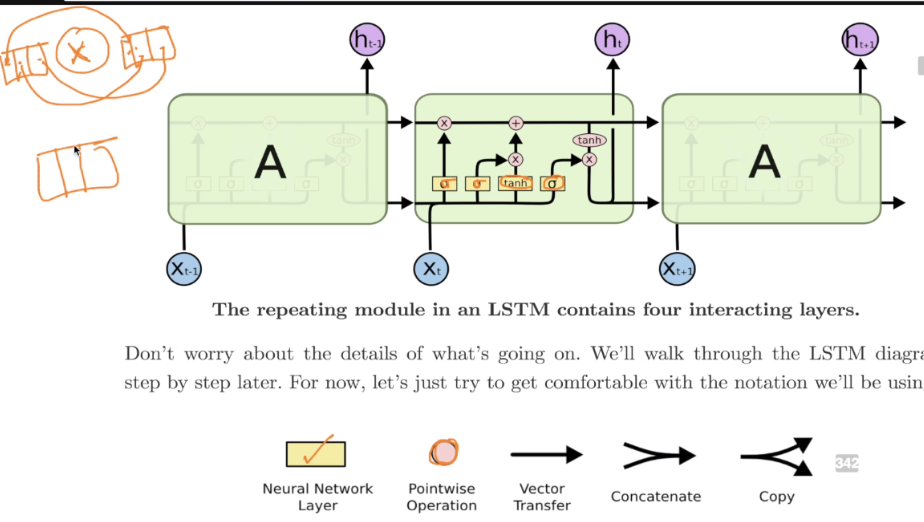
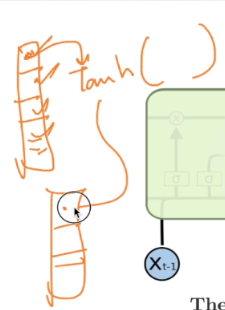
1. Forget Gate
2. Input Gate
3. Output Gate



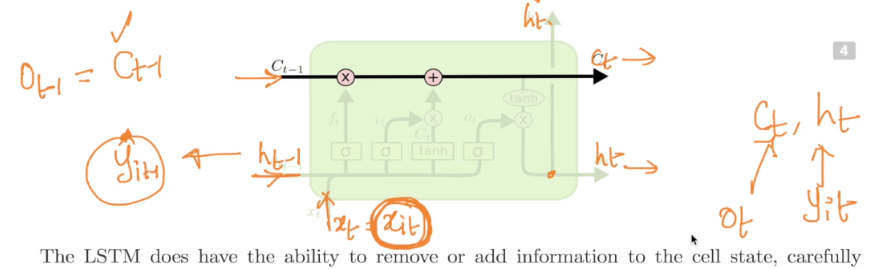
We’ll see them one by one.

Before looking them let’s see notation in LST, all are give in below image

1. Neural Network layer: whenever a rectangle appears it means that it’s a NN layer
2. Pointwise operation: Pointwise operation means suppose pointwise addition of two vectors, in this all elements are added one by one. Or let’s say tanh() of a vector it will give tanh of each value present in the vector.

 \

Let’s see the first line in LSTM, it’s taking the cell state of previous cell(c\_t-1) to current cell output(c\_t) And it’s linked with two gates input and output.

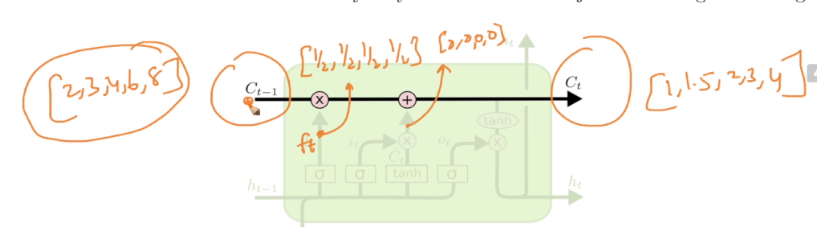
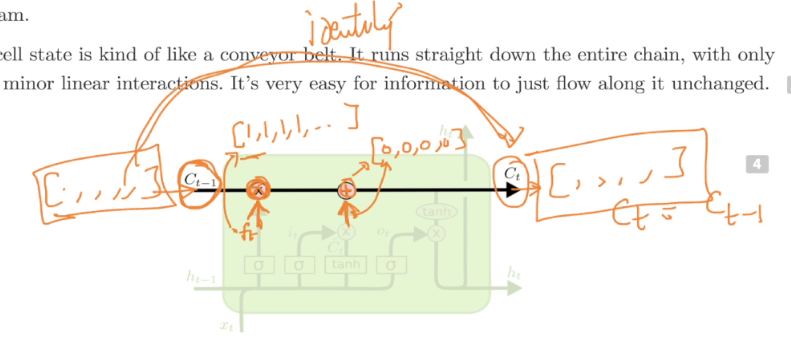


Here first link which is point wise multiplication says how much information to forget that is coming from previous cell,

let’s say [2,3,4,5] is coming from previous cell and we want to forget 50% of previous info, so forget gate will produce [1/2, ½, ½, ½]. So now on pointwise multiplication output will be [1, 0.5, 2, 2.5] that means we are forgetting 50% of info coming from previous cell, And this forgetting information is done by forget gate.

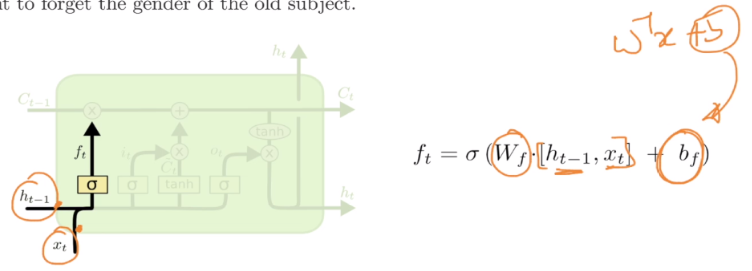
And second link is point wise addition which says how much new information to be added on info coming from previous cell(after preforming forget operation). And this operation is done by input gate.

Let’s see **forget gate**.



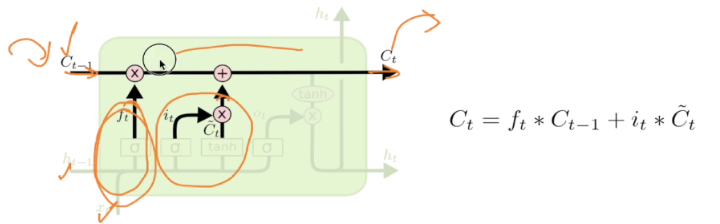
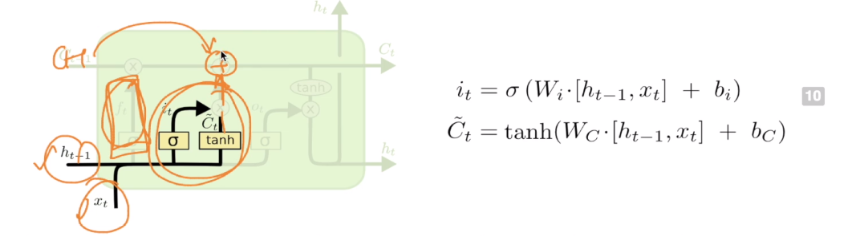
**Forget Gate:** It’s shown in below image. It’s taking output of previous cell(h\_t-1 or y\_t-1) and current input and merging them, then performing sigmoid over them.

Equation for forget gate is given below, it also contains bf which is a bias term. And this is responsible for forget operation.

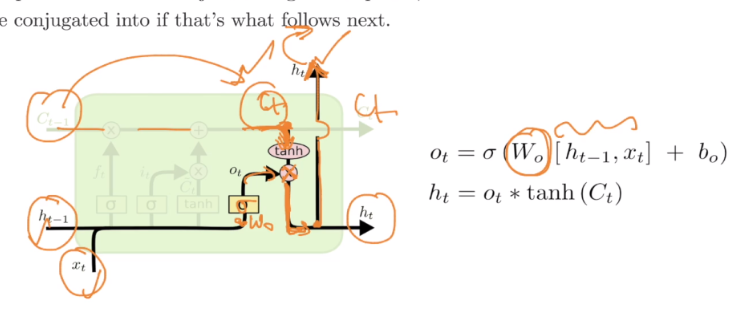


**Input Gate:** It’s shown in below image. It’s taking output of previous cell(h\_t-1 or y\_t-1) and current input and merging them. Then it applies the multiplication of weights and merged input via two functions sigmoid and tanh generating o/p it  and Ct respectively.

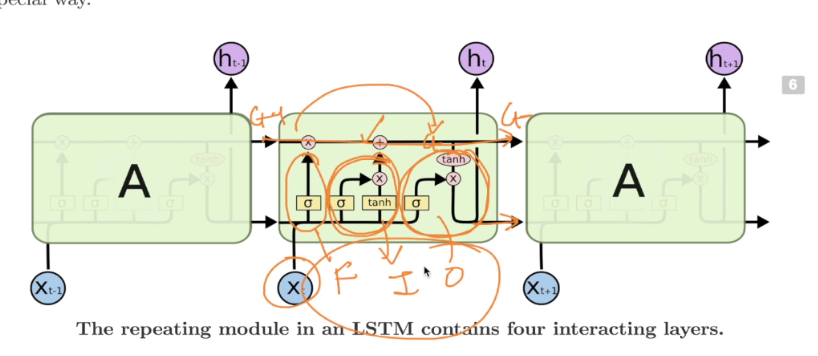
Now this point wise multiplication is performed on these two outputs it  and Ct, and this point wise multiplication will give the amount of new information to be added on previous cell state.



**Output Gate:** It’s shown in below image. It’s taking output of previous cell(h\_t-1 or y\_t-1) and current input and merging them. Then it applies the multiplication of weights and merged input to sigmoid function the output of this sigmoid function(o\_t) is point wise multiplied by point wise tanh of c\_t, and output of this point wise multiplication will generate output h\_t or y\_t

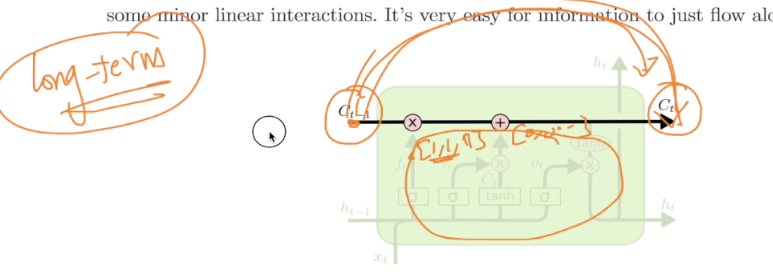


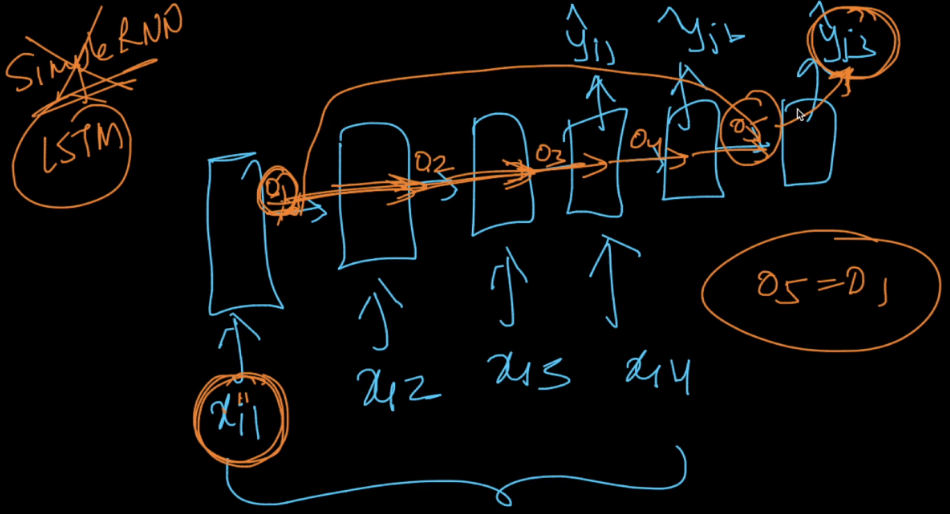
Complete LSTM is shown below.



**So how does it helps in preserving long term info:**

If forget percent is 0, and input layer is not adding any new information then no matter how many time steps are there in between old information will be retained.





**Overview:**

1. Forget Gate(weights corresponding to W\_f):  
First, we have the forget gate(W\_f). This gate decides what information should be thrown away or kept. Information from the previous hidden state(h(t-1)) and information from the current input(x(t)) is passed through the sigmoid function. Values come out between 0 and 1. The closer to 0 means to forget, and the closer to 1 means to keep.

2. Input Gate(weights corresponding to W\_i):  
To update the cell state, we have the input gate. First, we pass the previous hidden state(h(t-1)) and current input(x(t)) into a sigmoid function. That decides which values will be updated by transforming the values to be between 0 and 1. 0 means not important, and 1 means important. You also pass the hidden state(h(t-1)) and current input(x(t)) into the tanh function to squish values between -1 and 1 to help regulate the network. Then you multiply the tanh output with the sigmoid output. The sigmoid output will decide which information is important to keep from the tanh output.

3. Cell State  
Now we should have enough information to calculate the cell state. First, the cell state gets pointwise multiplied by the forget vector. This has a possibility of dropping values in the cell state if it gets multiplied by values near 0. Then we take the output from the input gate and do a pointwise addition which updates the cell state to new values that the neural network finds relevant. That gives us our new cell state(c(t)).

4. Output Gate  
Last we have the output gate. The output gate decides what the next hidden state should be. Remember that the hidden state contains information on previous inputs. The hidden state is also used for predictions. First, we pass the previous hidden state and the current input into a sigmoid function. Then we pass the newly modified cell state to the tanh function. We multiply the tanh output with the sigmoid output to decide what information the hidden state should carry. The output is the hidden state. The new cell state and the new hidden is then carried over to the next time step.  
  
To review, the Forget gate decides what is relevant to keep from prior steps. The input gate decides what information is relevant to add from the current step. The output gate determines what the next hidden state should be.  
  
This is a good video explanation of internals of LSTM: <https://www.youtube.com/watch?v=8HyCNIVRbSU>

Nice tutorial:

<http://colah.github.io/posts/2015-08-Understanding-LSTMs/>

<https://www.youtube.com/watch?v=yCC09vCHzF8>