<https://www.analyticsvidhya.com/blog/2017/12/fundamentals-of-deep-learning-introduction-to-lstm/>

<https://www.analyticsvidhya.com/blog/2019/01/sequence-models-deeplearning/>

<https://www.analyticsvidhya.com/blog/2019/01/introduction-time-series-classification/>

<https://towardsdatascience.com/illustrated-guide-to-lstms-and-gru-s-a-step-by-step-explanation-44e9eb85bf21>

LSTM’s and GRU’s were created as the solution to short-term memory.

They have internal mechanisms called gates that can regulate the flow of information.

These gates can learn which data in a sequence is important to keep or throw away.

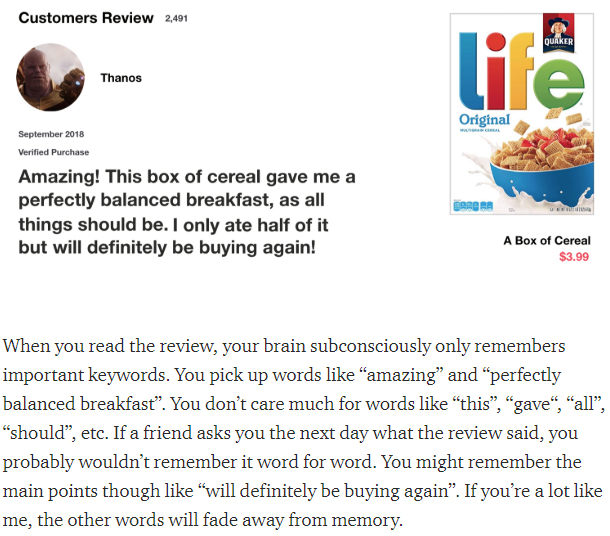
By doing that, it can pass relevant information down the long chain of sequences to make predictions

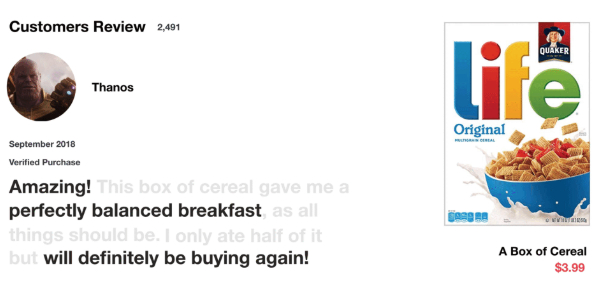
**Applications**

Speech recognition, speech synthesis, and text generation

You can even use them to generate captions for videos

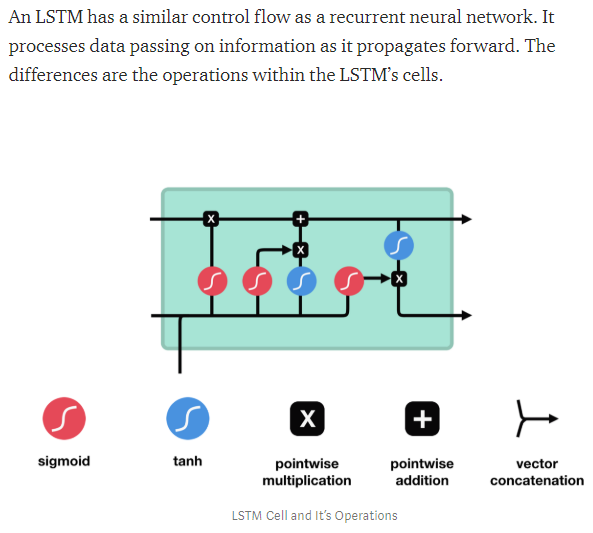
And that is essentially what an LSTM or GRU does. It can learn to keep only relevant information to make predictions, and forget non relevant data.





And that is essentially what an LSTM or GRU does. It can learn to keep only relevant information to make predictions, and forget non relevant data.

**LSTM**



**Core Concept**

The core concept of LSTM’s are the cell state, and its various gates

The cell state act as a transport highway that transfers relative information all the way down the sequence chain

You can think of it as the “memory” of the network.

The cell state, in theory, can carry relevant information throughout the processing of the sequence.

 So even information from the earlier time steps can make its way to later time steps, reducing the effects of short-term memory

As the cell state goes on its journey, information get’s added or removed to the cell state via gates.

The gates are different neural networks that decide which information is allowed on the cell state.

The gates can learn what information is relevant to keep or forget during training.

## Sigmoid

Gates contains sigmoid activations.

A sigmoid activation is similar to the tanh activation.

Instead of squishing values between -1 and 1, it squishes values between 0 and 1.

That is helpful to update or forget data because any number getting multiplied by 0 is 0, causing values to disappears or be “forgotten.”

Any number multiplied by 1 is the same value therefore that value stay’s the same or is “kept.”

The network can learn which data is not important therefore can be forgotten or which data is important to keep.

Let’s dig a little deeper into what the various gates are doing, shall we?

So we have three different gates that regulate information flow in an LSTM cell.

A forget gate, input gate, and output gate.

## Forget gate

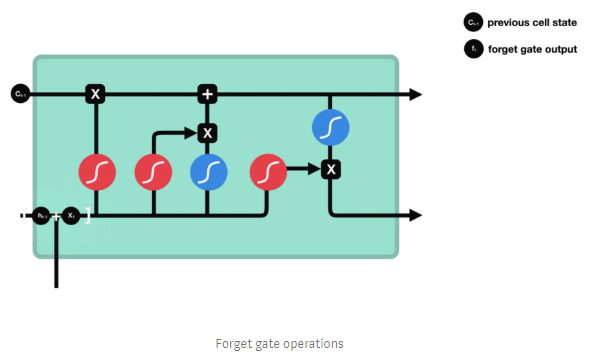
First, we have the forget gate.

This gate decides what information should be thrown away or kept.

Information from the previous hidden state and information from the current input 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



## Input Gate

To update the cell state, we have the input gate.

First, we pass the previous hidden state and current input into a sigmoid function.

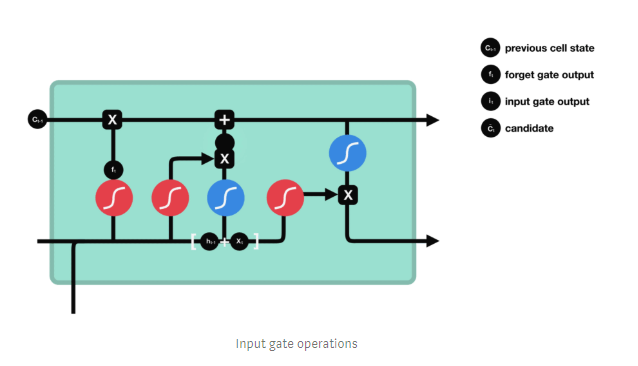
That decides which values will be updated by transforming the values to be between 0 and 1.

0 means not important, and 1 means important.

You also pass the hidden state and current input 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.



## Cell State

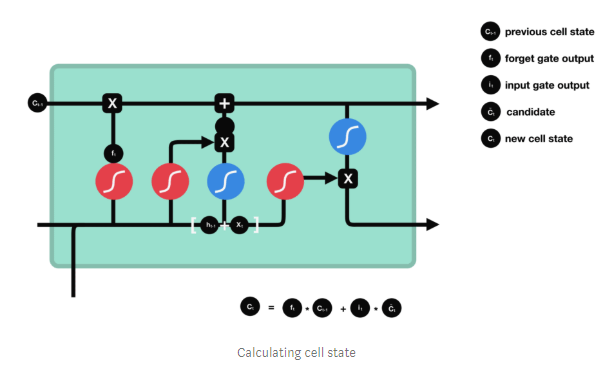
Now we should have enough information to calculate the cell state

 First, the cell state gets point-wise 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.



## Output Gate

The output gate decides what the next hidden state should be.

Remember that the hidden state contains information on previous inputs.

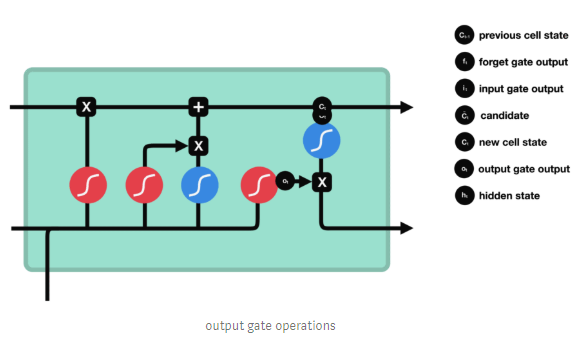
The hidden state is also used for predictions.

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. + current input

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.