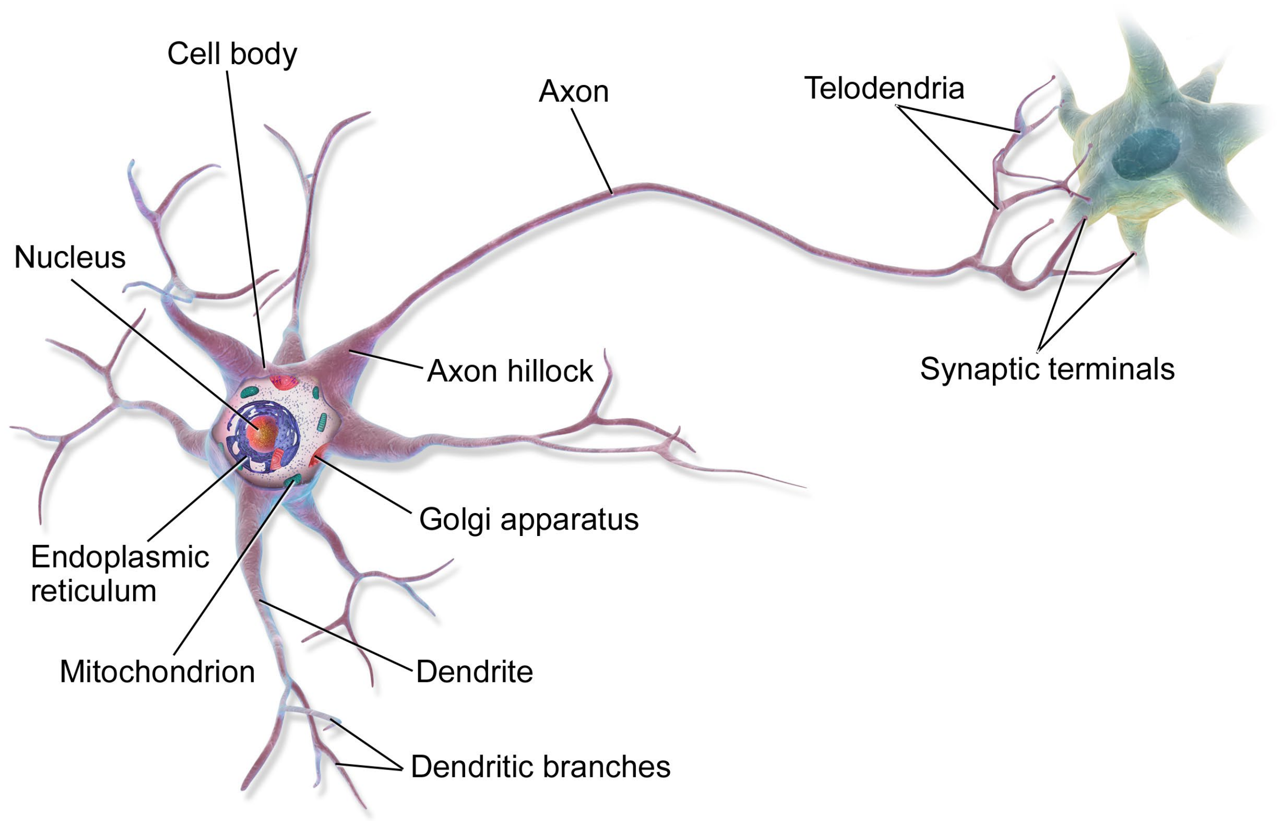
Machine Learning and its Applications

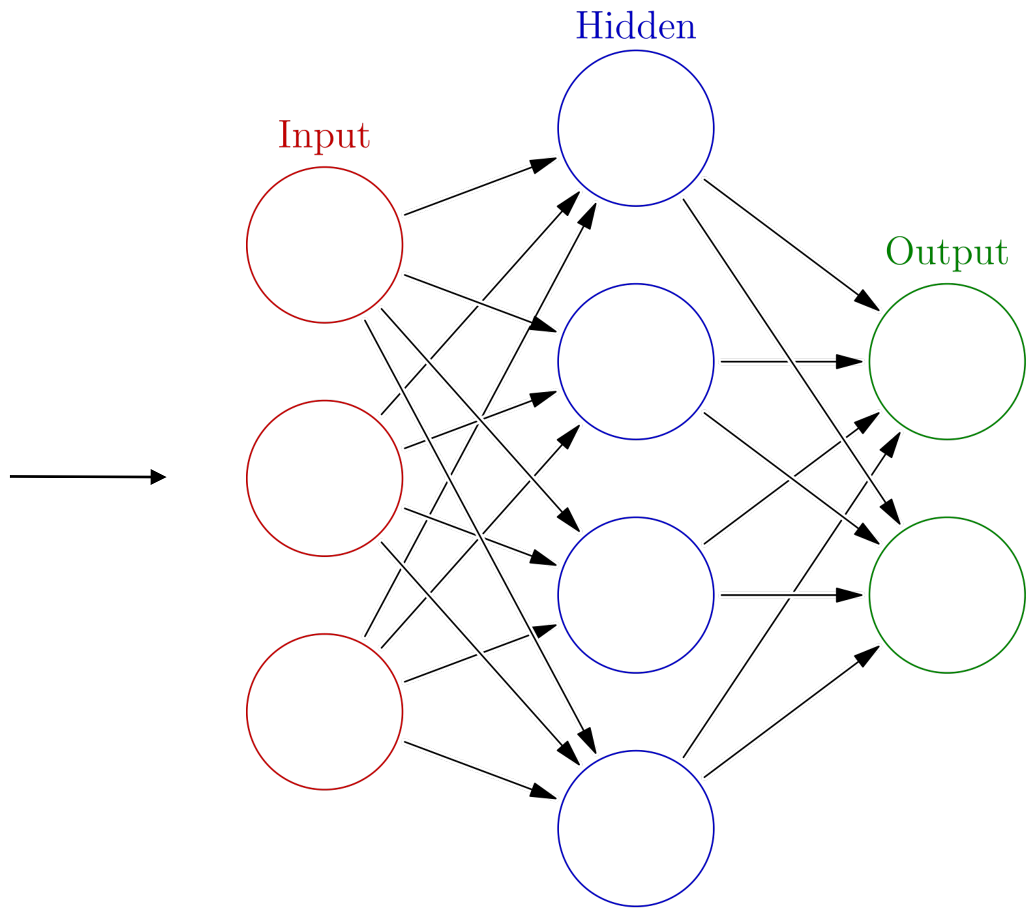
Recurrent Neural Network (RNN) & LSTM & GRU

Urban Information Lab

Introduction: Neurons? Neural Network?

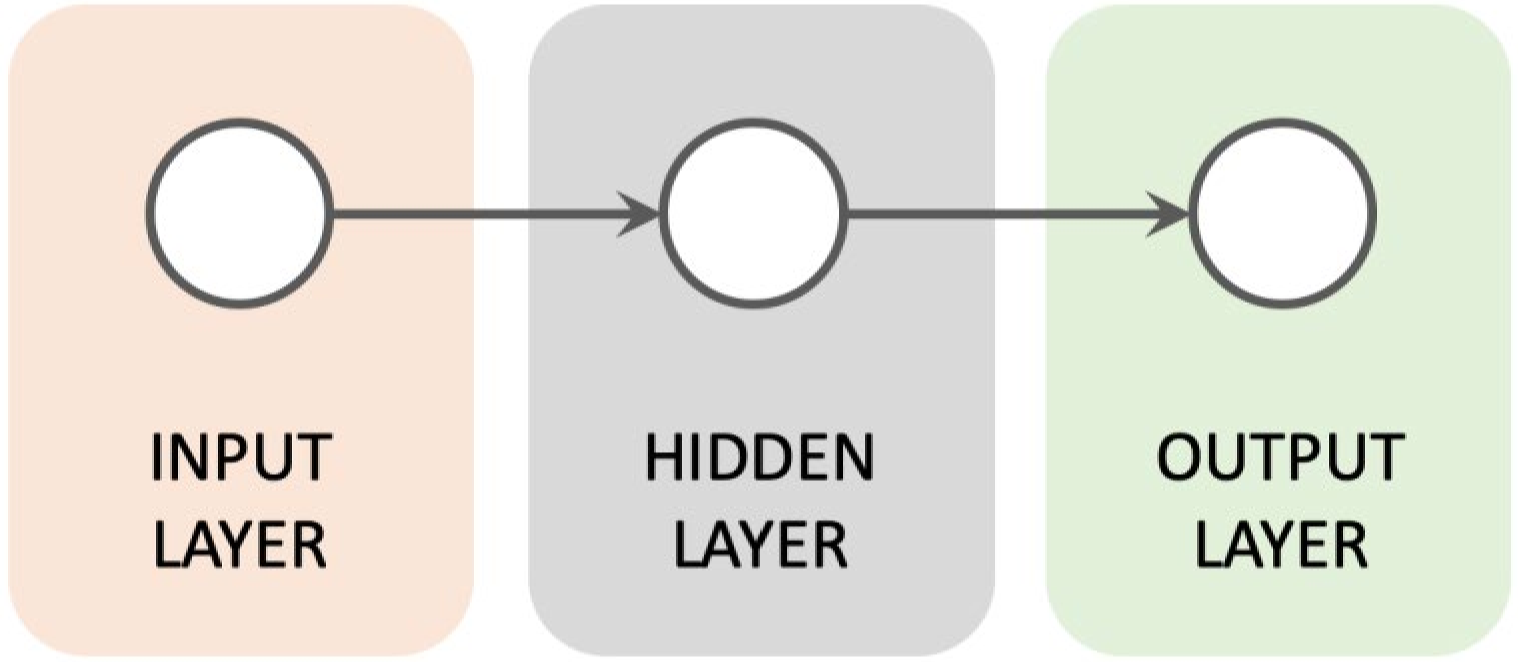


Neurons



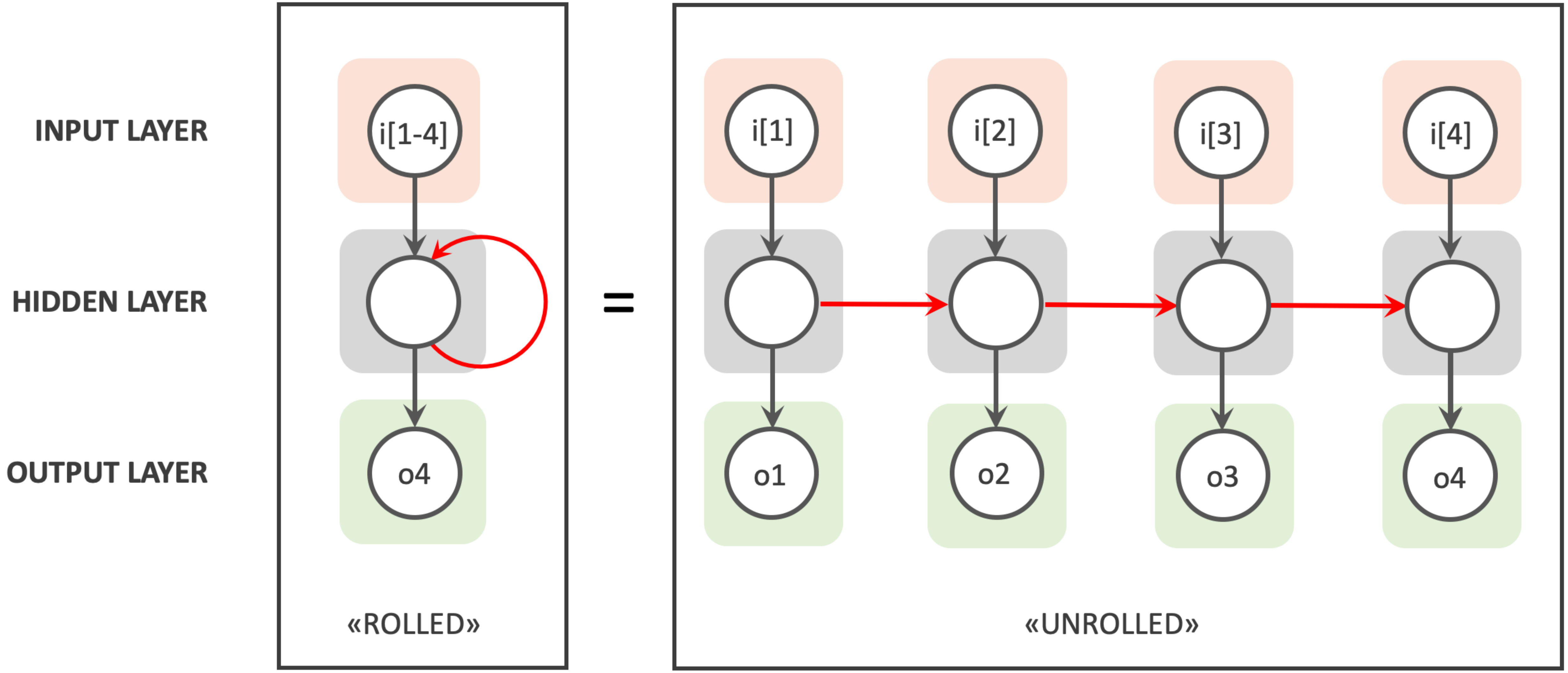
Neural Network

Conventional Feed Forward Network

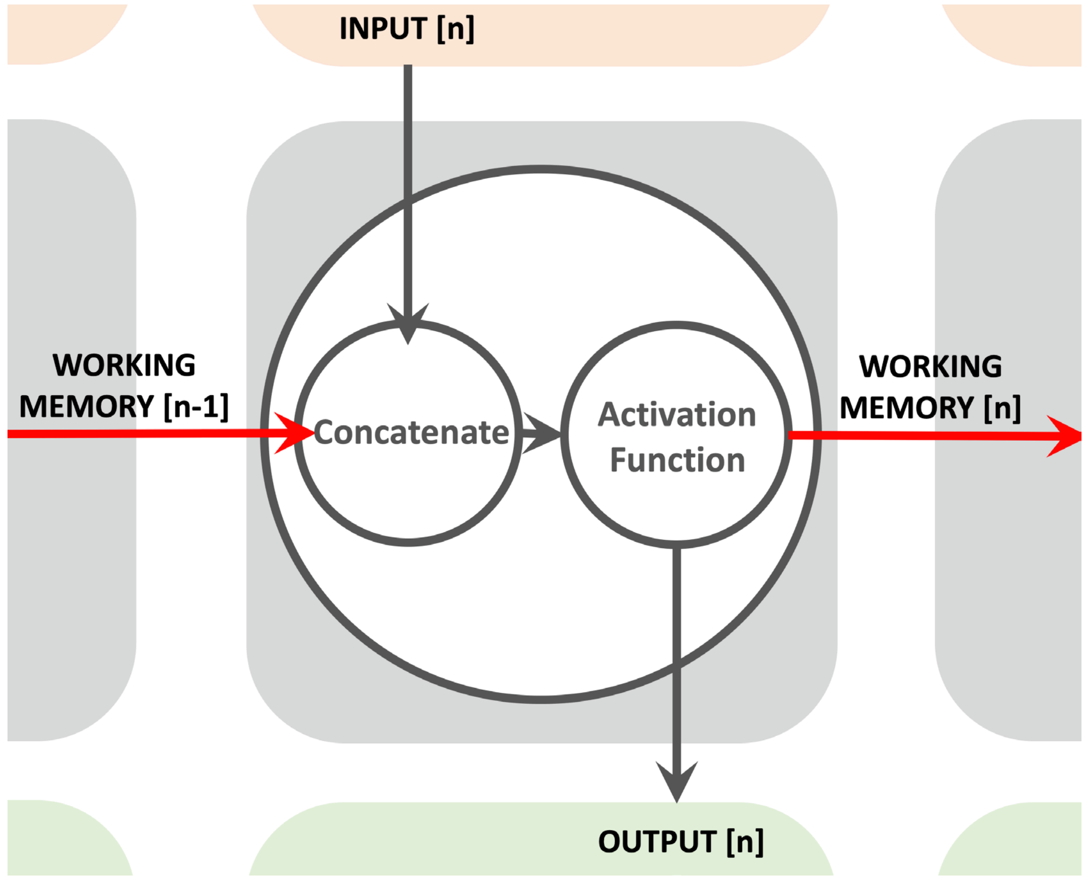


Recurrent Neural Network (RNN)

-RNNs support processing of sequential data by the addition of a **loop**

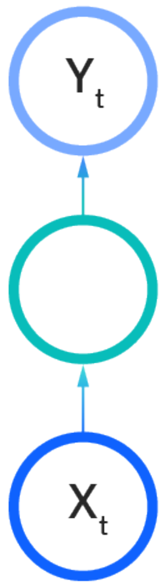


Internal Operation of RNN (Inside the Hidden Node)

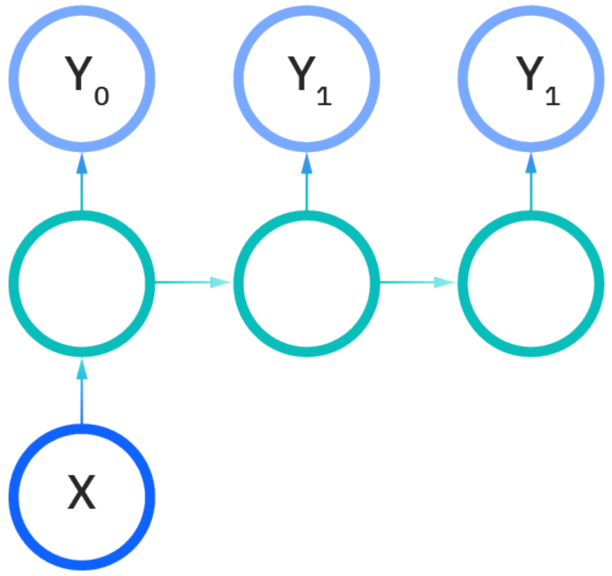


Shows how it concatenates it's current input and working memory from the previous iteration, before passing the result on to an activation function.

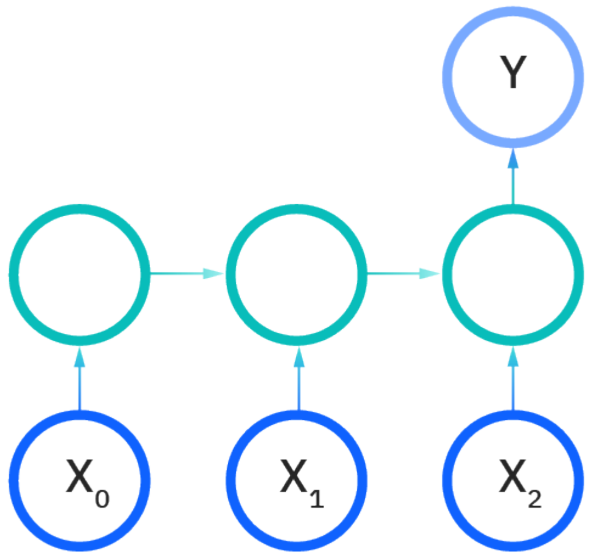
Types of RNN



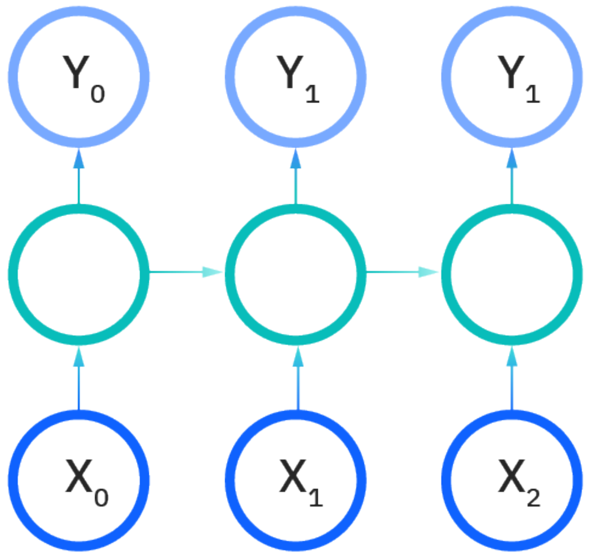
One to One



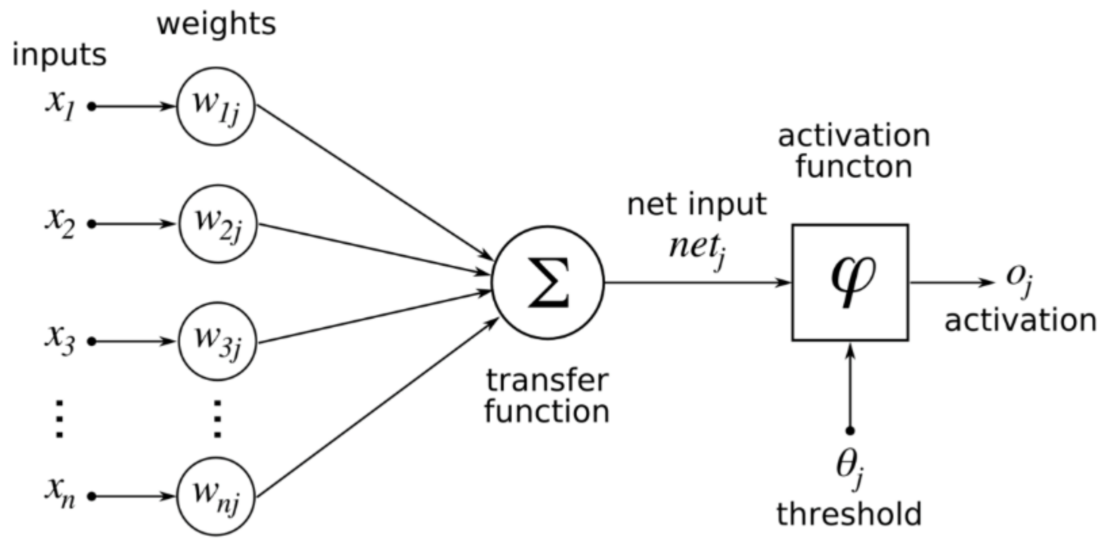
One to Many



Many to One



Many to Many

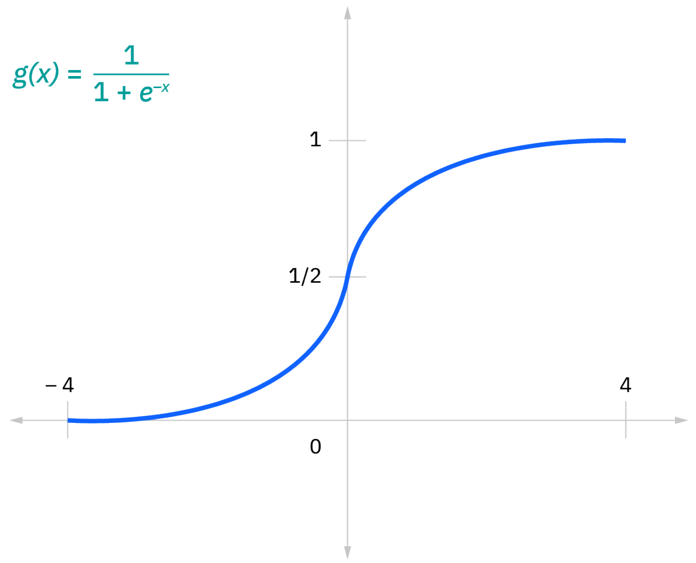


Activation Functions: Just like others

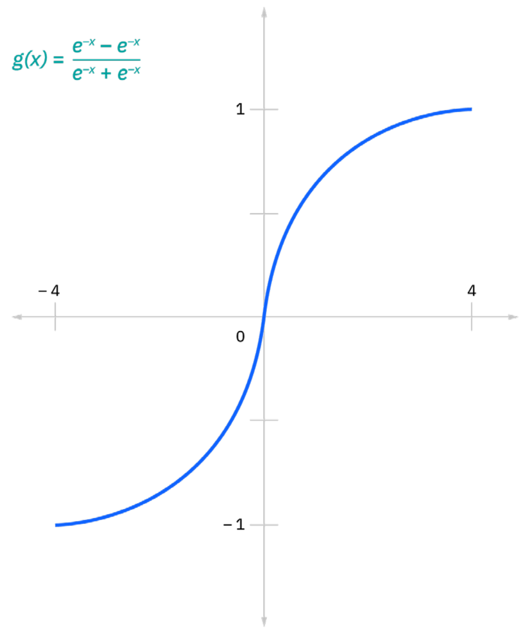
An activation function determines whether a neuron should be activated. The nonlinear functions typically convert the output of a given neuron to a value between **0 and 1 or -1 and 1**. Some of the most commonly used functions are defined as follows:

Activation Functions: Just like others

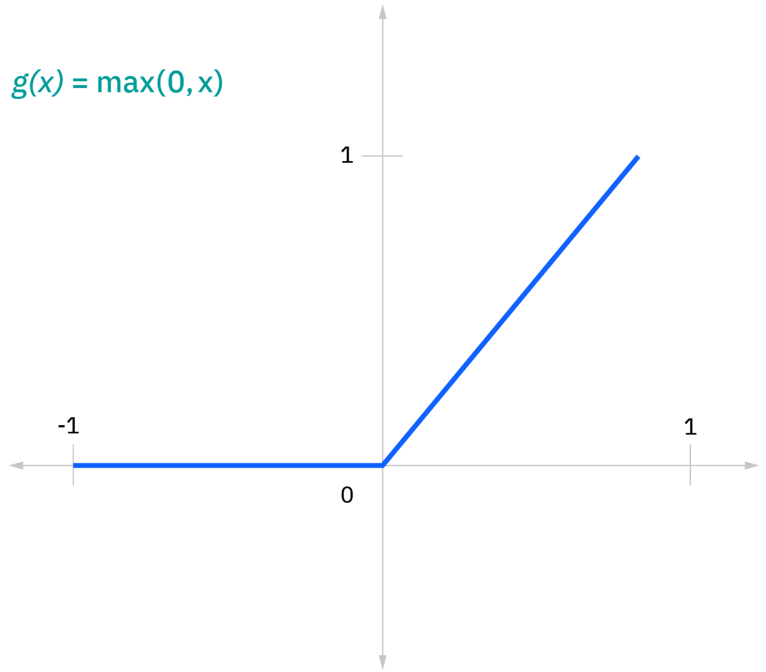
Sigmoid



Tanh



Relu



Sigmoid: This is represented with the formula g(x) = 1/(1 + e^-x)
Tanh: This is represented with the formula g(x) = (e^-x - e^-x)/(e^-x + e^-x).
Relu: This is represented with the formula g(x) = max(0 , x)

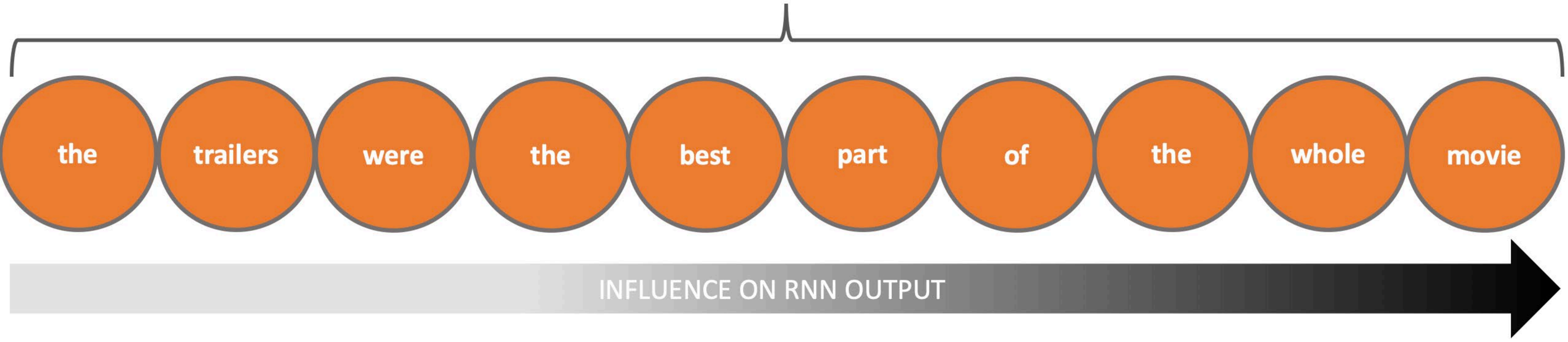
Where to use it?

1. Prediction problems
2. Language Modelling and Generating Text
3. Machine Translation
4. Speech Recognition
5. Generating Image Descriptions
6. Video Tagging
7. Text Summarization
8. Call Center Analysis
9. Face detection, OCR Applications as Image Recognition
10. Other applications like Music composition
11. **COVID-19 Outbreak Prediction**

Limitations

The "working memory" of standard RNNs struggles to retain longer term dependencies. The below image illustrates this problem:

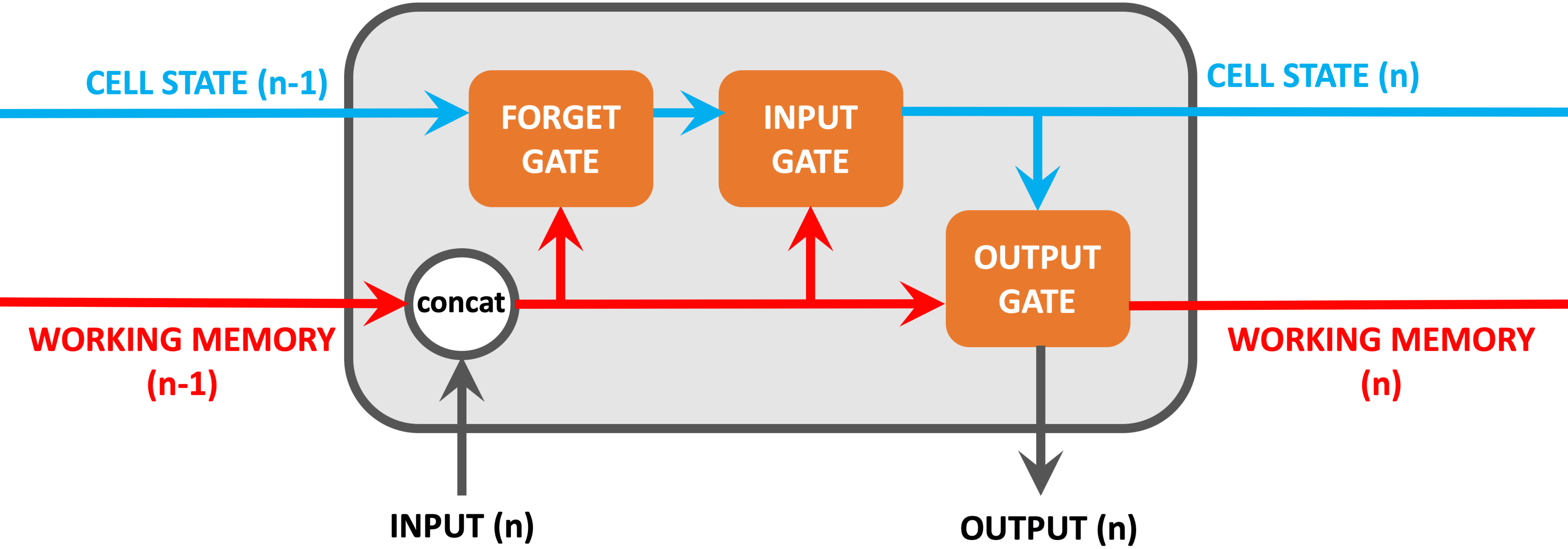
“the trailers were the best part of the whole movie.”



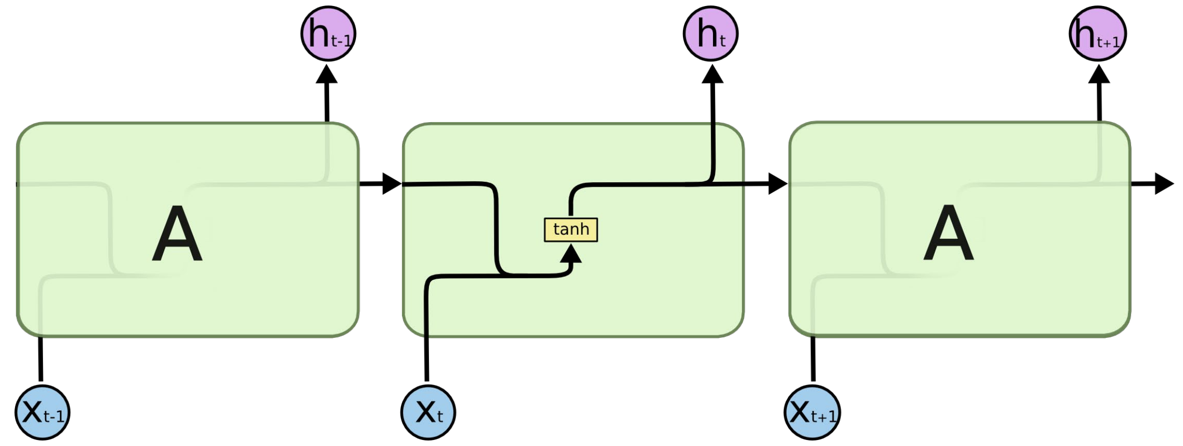
This behavior is due to the ***Vanishing Gradient* problem**, and can cause problems when early parts of the input sequence contain important contextual information.

To resolve vanishing gradient: LSTM (Long Short-term Memory) Network

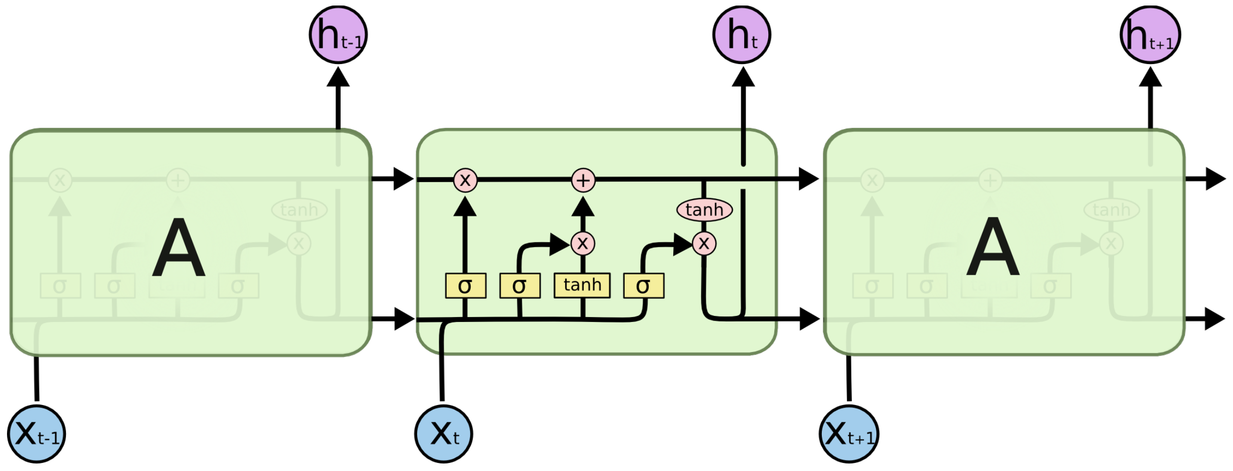
- LSTM uses RNN structure
- However, has several 'new' aspects

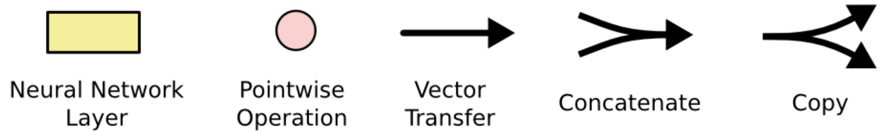


Structure of RNN

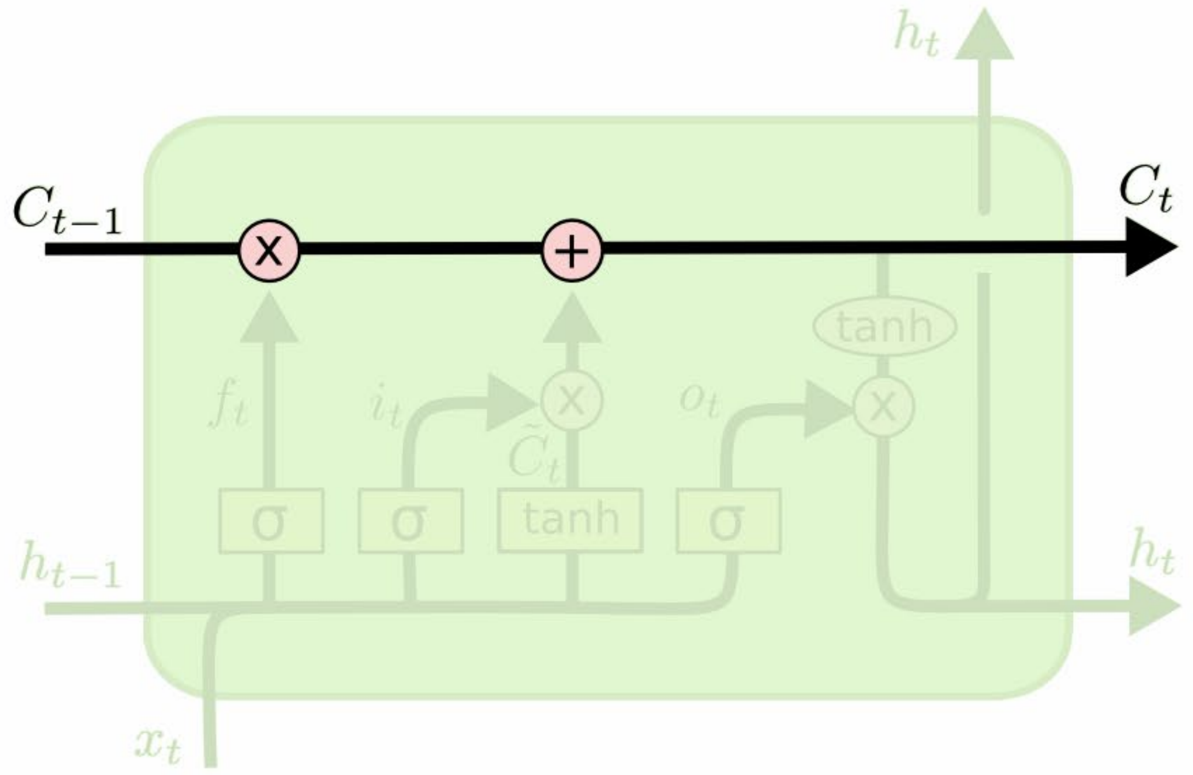


Structure of LSTM



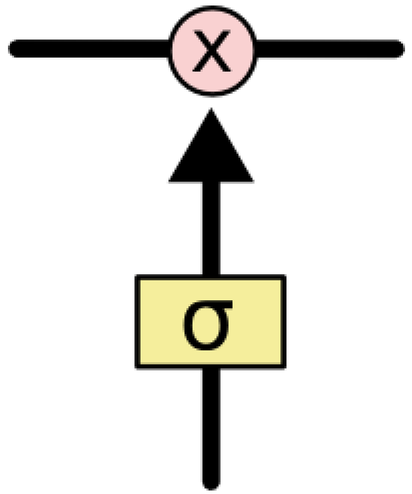


The key to LSTMs is the cell state, the horizontal line running through the top of the diagram.
The cell state is kind of like a conveyor belt. It runs straight down the entire chain, with only some minor linear interactions. It's very easy for information to just flow along it unchanged.



The LSTM does have the ability to remove or add information to the cell state, carefully regulated by structures called gates.

Gates are a way to optionally let information through. They are composed out of a sigmoid neural net layer and a pointwise multiplication operation.

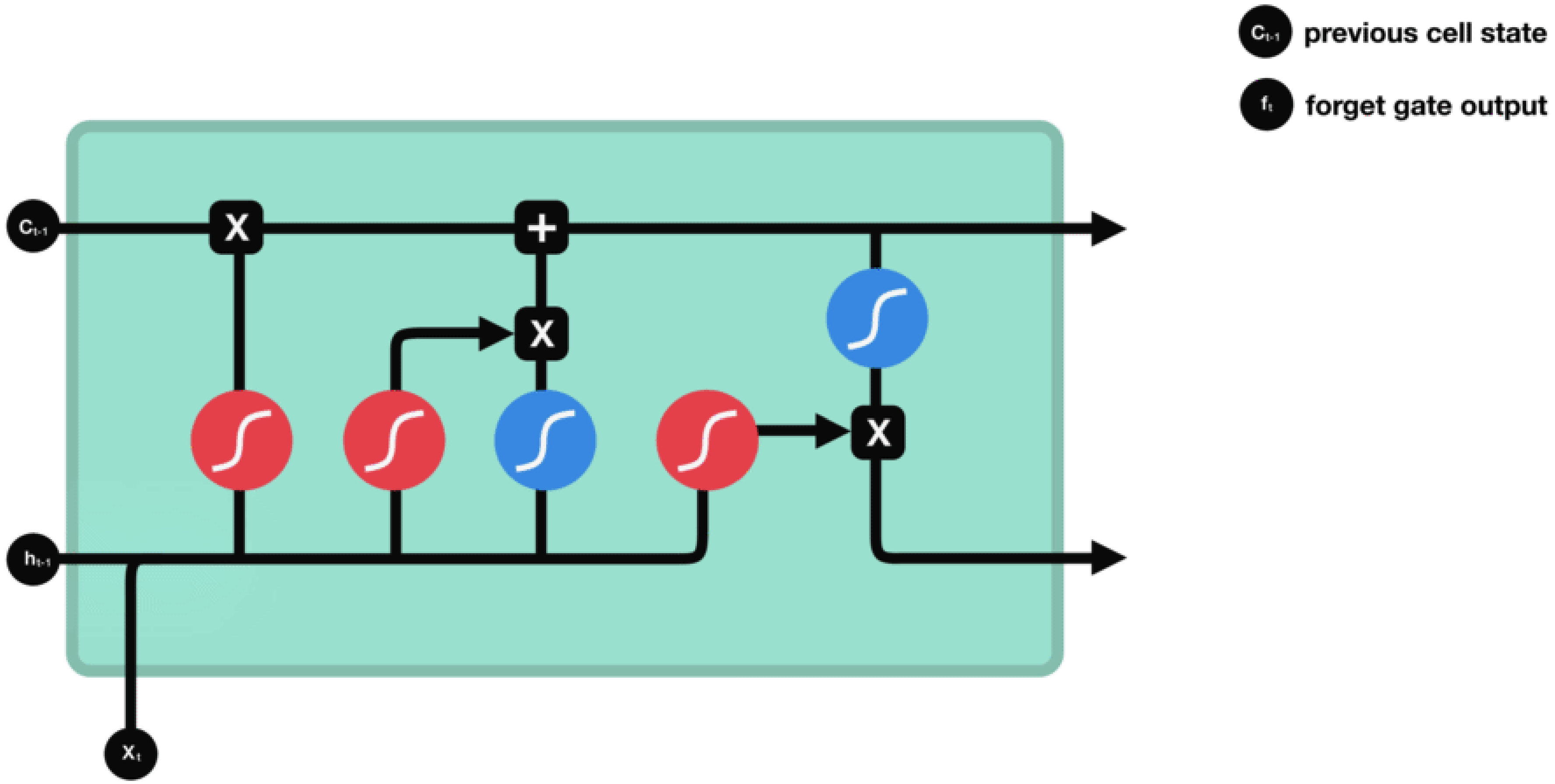


The sigmoid layer outputs numbers between zero and one, describing how much of each component should be let through. A value of zero means “let nothing through,” while a value of one means **“let everything through!”**

An LSTM has three of these gates, to protect and control the cell state.

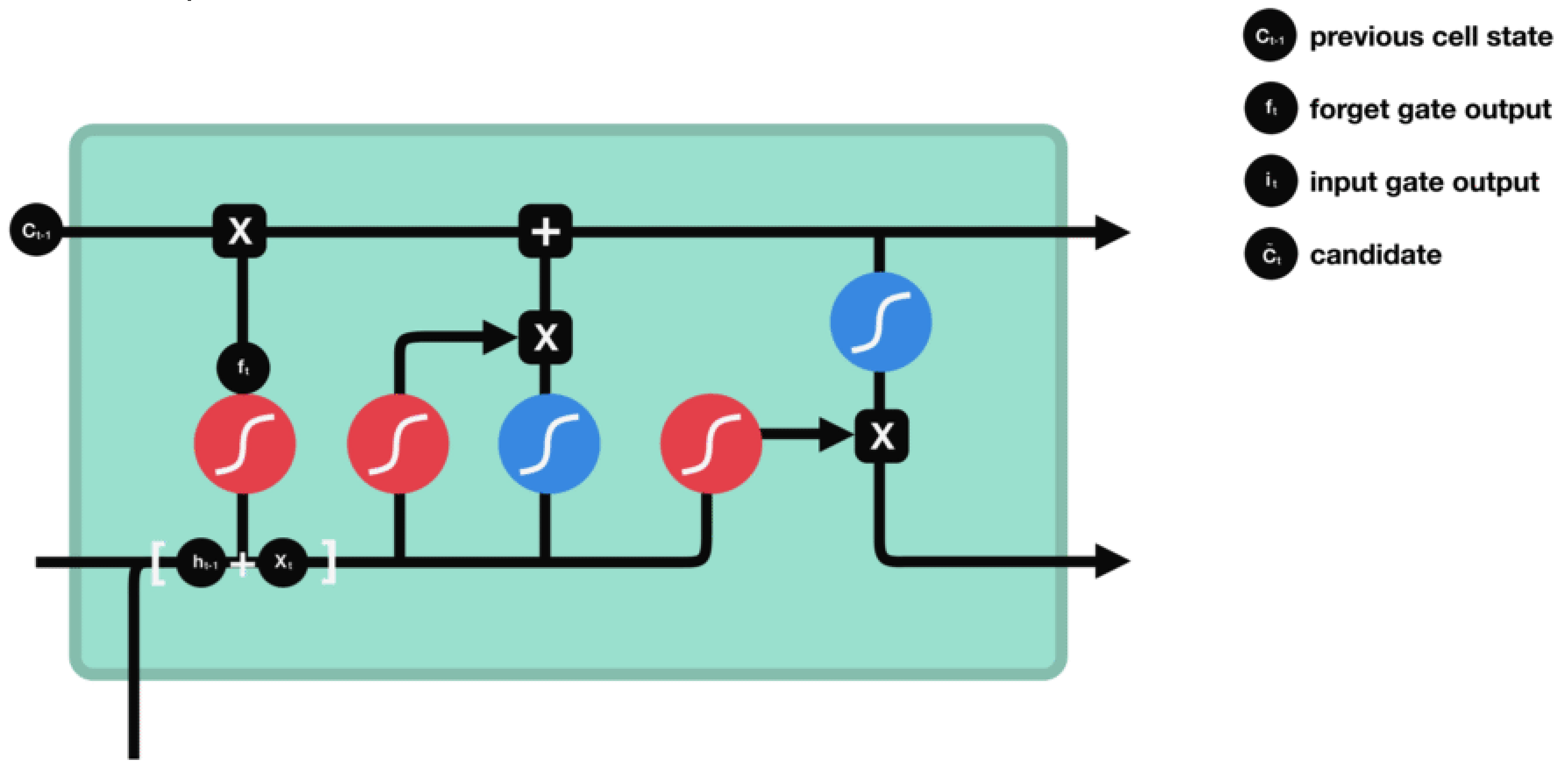
Forget gate

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

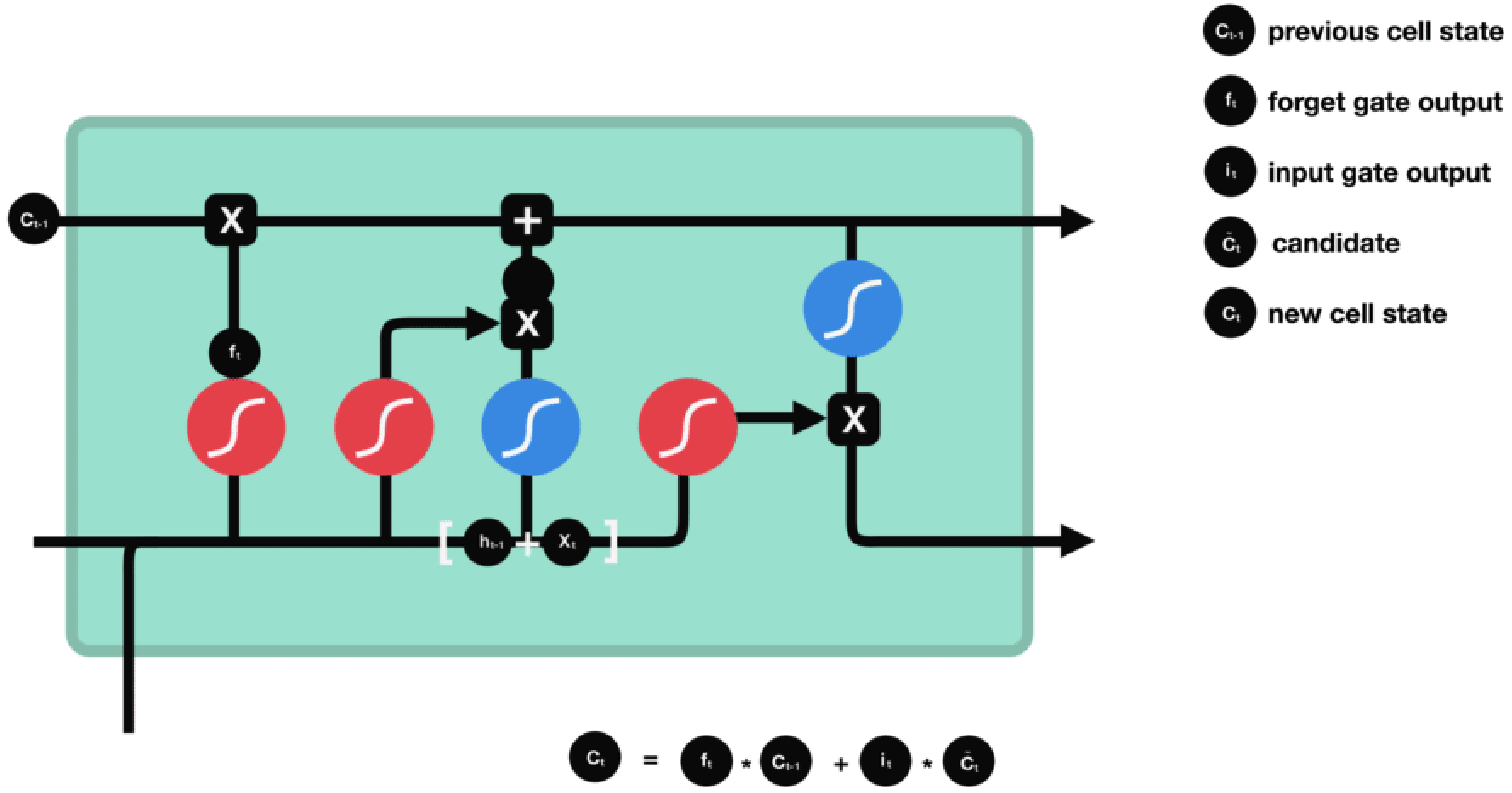
To update the cell state, we have the input gate. First, we pass the previous hidden state and current input into a sigmoid function. That decides which values will be updated by transforming the values to be between 0 and 1. 0 means not important, and 1 means important. You also pass the hidden state 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.



c_{t-1} previous cell state
f_t forget gate output
i_t input gate output
c̃_t candidate
c_{t-1}
X
+
X
f_t
X
h_{t-1}
x_t

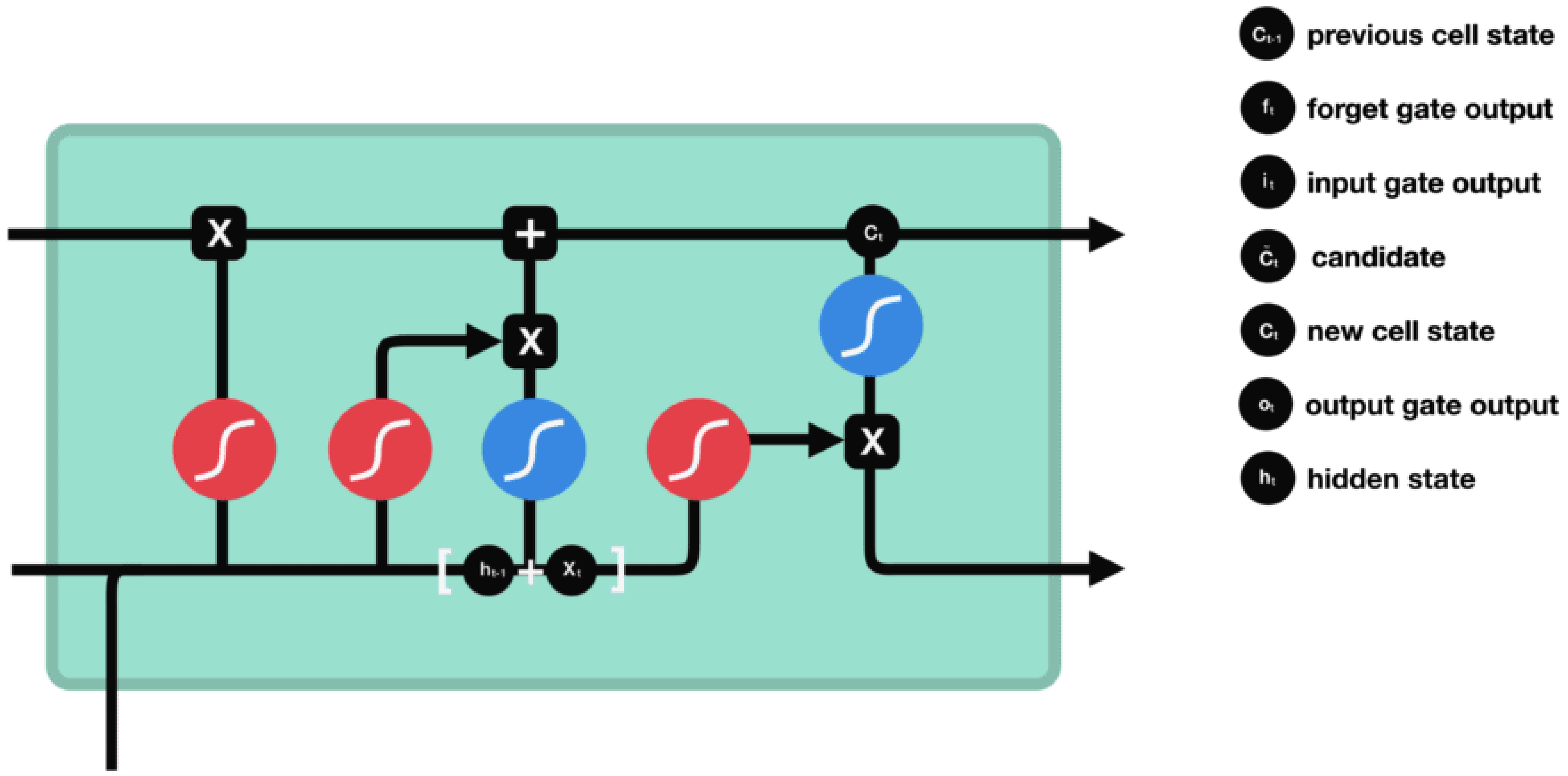
Input gate

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.



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.

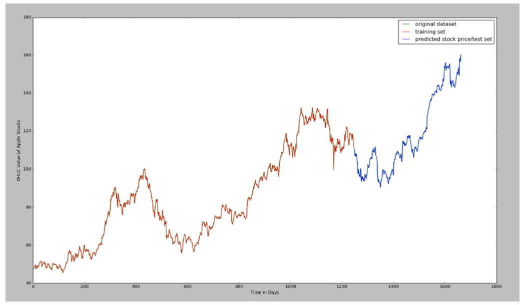


To review, the Forget gate decides what is relevant to keep from prior steps. The input gate decides what information is relevant to add from the current step. The output gate determines what the next hidden state should be.

1. **Forget Gate**

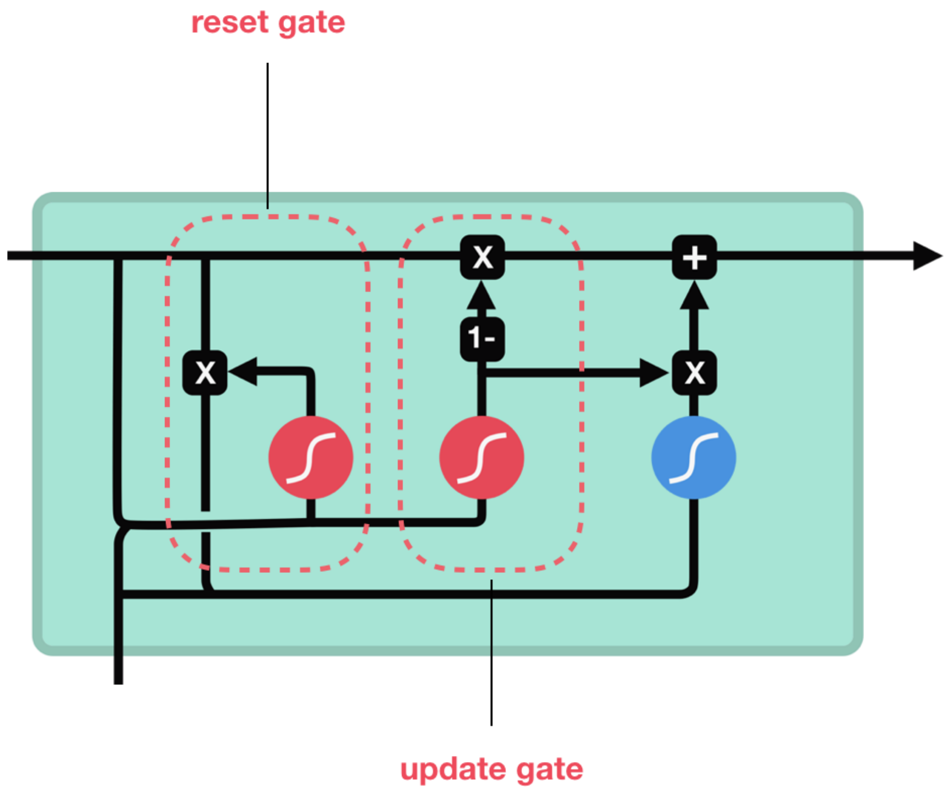
2. **Input Gate**

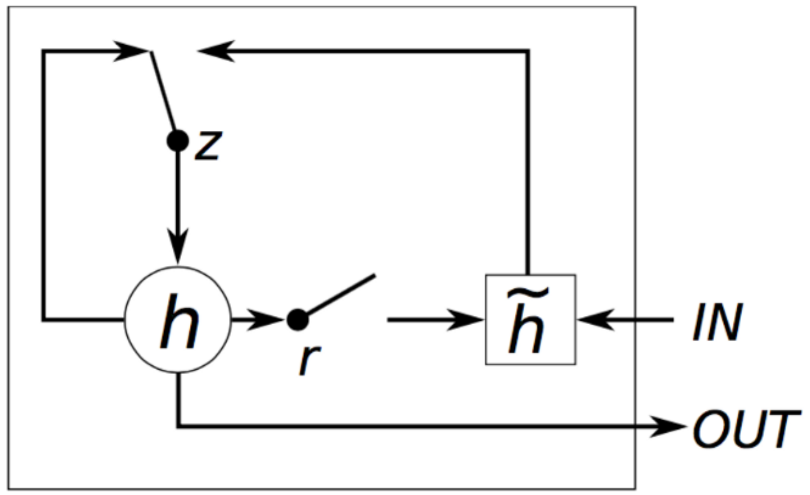
3. **Output Gate**



GRU (Newer one)

So now we know how an LSTM work, let's briefly look at the GRU. The GRU is the **newer generation of Recurrent Neural networks** and is pretty similar to an LSTM. GRU's got rid of the cell state and used the hidden state to transfer information. It also only has two gates, a reset gate and update gate.





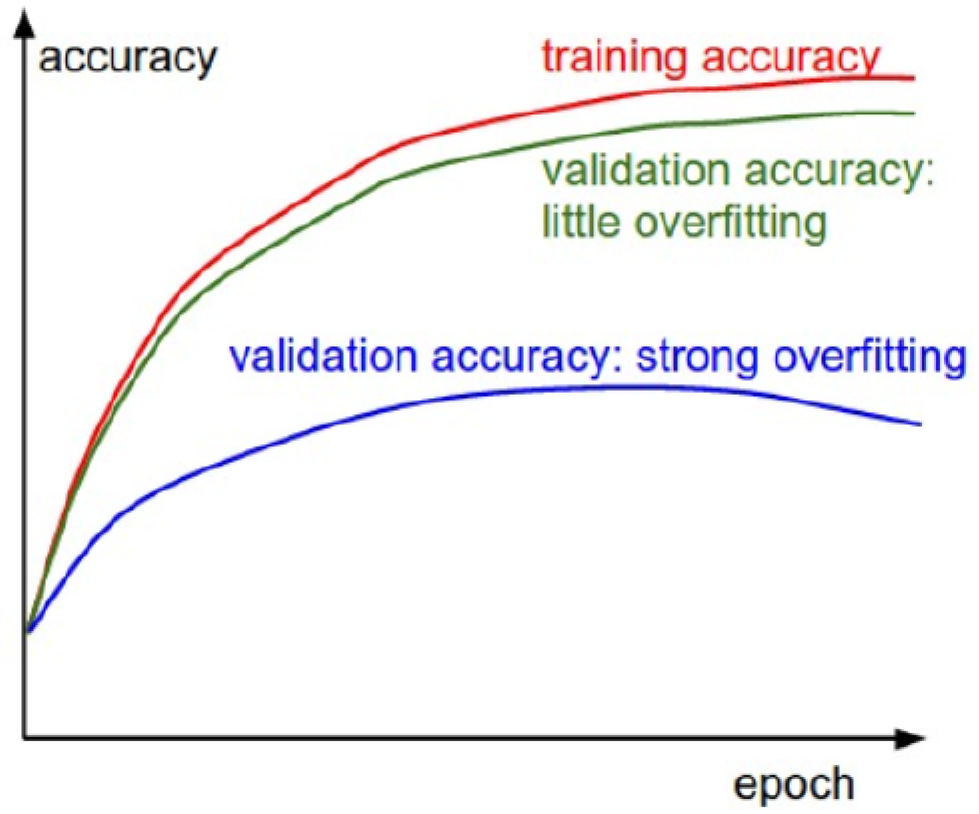
Update Gate
The update gate acts similar to the forget and input gate of an LSTM. It decides what information to throw away and what new information to add. (It is also a variation of LSTM)

Reset Gate
The reset gate is another gate is used to decide how much past information to forget.

And that’s a GRU. GRU’s has fewer tensor operations; therefore, they are a little speedier to train then LSTM’s. There isn’t a clear winner which one is better. Researchers and engineers usually try both to determine which one works better for their use case.

Validation: Monitoring Accuracy (Classification)

- Check how your desired performance metrics behaves during training



Preventing Overfitting

Standard ways to limit the capacity of a neural net:

- Limit the number of hidden units.
- Limit the size of the weights. Weight decay
- Stop the learning before it has time to overfit. Early stop

Austin Housing Price Forecast will be introduced during the lab session

+ Others

Thank You!