LSTM

Long short-term memory networks = type of recurrent neural network

Capable of learning order dependence in sequence prediction problems

1. Neural networks (traditional feed-forward)
2. Recurrent neural network (short term memory)

If a sequence is long enough, difficult to carry information from earlier time steps to later ones

Vanishing gradient problem during back propagation.

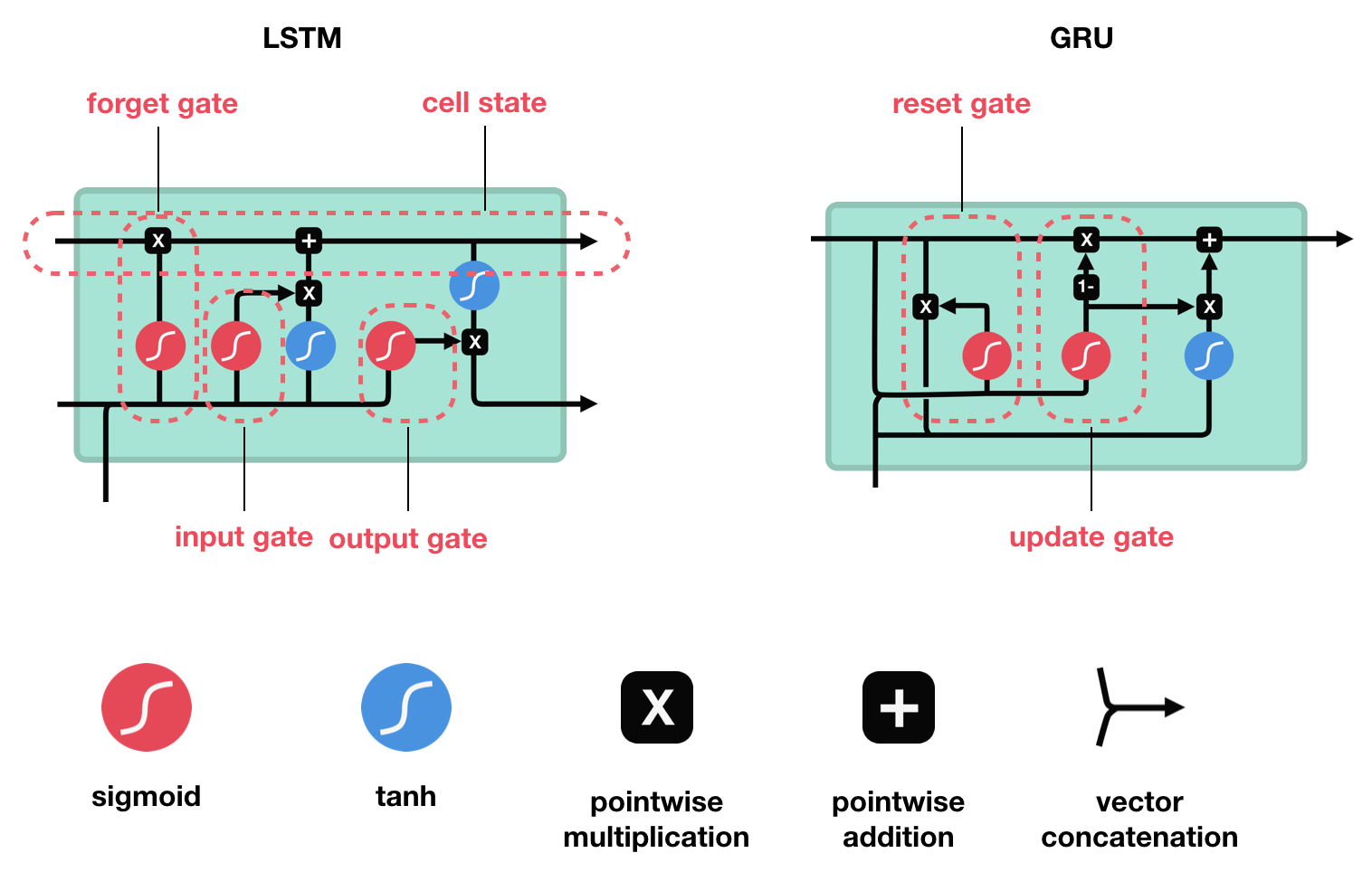
Gradients are values used to update a neural networks weight.

Vanishing gradient problem = gradient shrinks as it back propagates through time

Gradient becomes very small and does not contribute to learning

LSTM

* System is able to store information for an arbitrary duration
* System is resistance to noise (fluctuations of the inputs that are random or irrelevant to predicting a correct output)
* System parameters be trainable



Gates: learn which data in a sequence is important to keep or throw

Pass relevant info down the long chain of sequences to make predictions

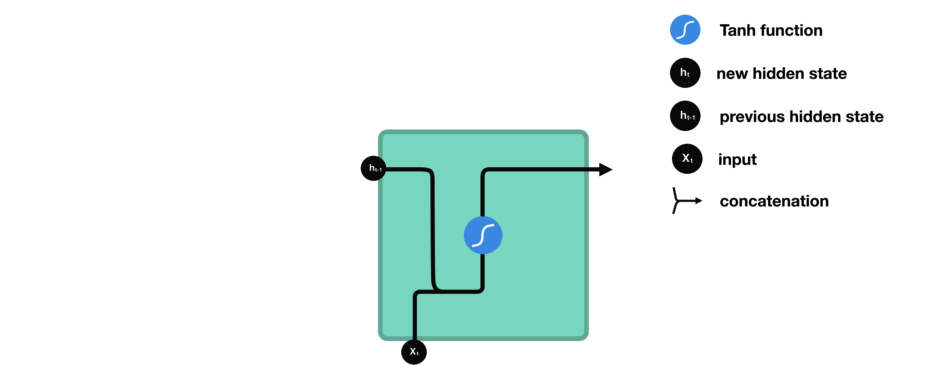
LSTM learns to keep only relevant info to make predictions, forget non relevant data

RNN

1. Data is preprocessed into machine-readable vectors
2. RNN processes the sequence of vectors one by one
3. While processing, passes previous hidden state to the next step of the sequence
4. Hidden state = neural network memory, holds info on previous data network has seen before
5. Input and previous hidden state combined to form a factor
6. Vector goes through tanh activation and output is the new hidden state
7. Tanh activation regulates the values between -1 and 1 (normalization)

Feedback loops in the recurrent layer. Maintains information in memory over time.

Difficult to train standard RNNs to solve problems that require learning long-term temporal dependencies.



LSTM

**Cell state** and **various gates**

Cell state = transport highway that transfers relative information all the way down the sequence chain. Carry relevant information throughout the processing of the sequence. Allowing information at earlier timesteps to make its way to later time steps. Reduce effects of short-term memory

Information is added or removed to the cell state via gates

Gates are different neural networks that decide which info is allowed on the cell state

Gates learn what info is relevant to keep or forget

Gates contains sigmoid activations (0-1)

0 is needed to cause values to be forgotten (0 times 0 is 0, any number multiplied by 1 is the same and the value is kept)

3 gates: forget, input and output

1. Forget gate:

Decide what info is kept or forgotten (from previous steps)

Info from previous hidden state and info from the current input is passed through the sigmoid function (0 – 1) closer to 0 means forget, 1 means keep

1. Input gate:

To update the cell state (which info is relevant to add from the current step)

First pass previous hidden state and current input into a sigmoid function (0-1)

0 not important, 1 means important.

Then pass previous hidden state and current input into a Tanh function (-1 and 1), used to regulate the network

Lastly multiple tanh output with sigmoid output. Sigmoid output will decide which info is important to keep from the tanh output

Calculate cell state:

Multiply cell state with forget vector (drop values if near 0)

Output from input gate is added to previous. Get new cell state

1. Output gate: decides next hidden state

Hidden state contains info on previous inputs, used for predictions

1. Pass previous hidden state and current input into sigmoid function (0,1)
2. Pass new cell state through tanh function
3. Multiply tanh output with sigmoid output = new hidden state
4. New cell state and new hidden state is carried over to next step