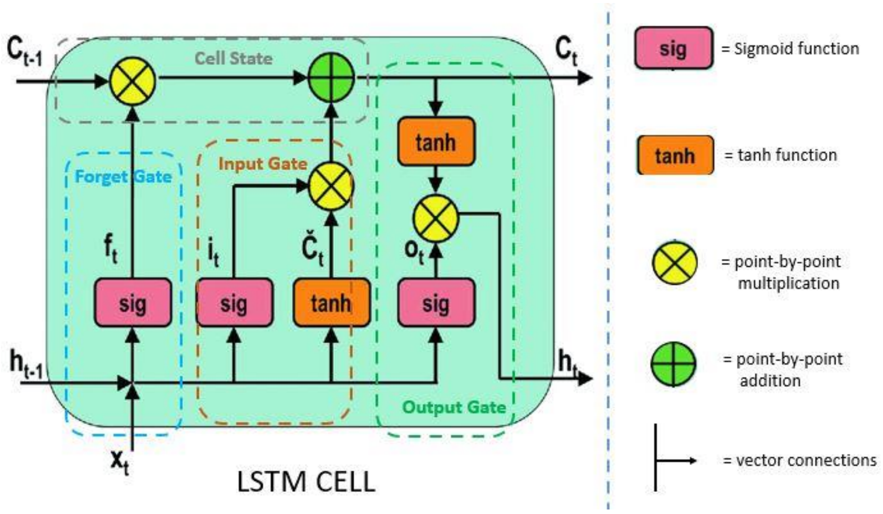
BIDIRECTIONAL LSTM

(Implementation from scratch)

Forward Propagation



The forget gate: Decides which information needs attention and which can be ignored. The information from the current input x(t) and the previous hidden state h(t-1) are passed through the sigmoid function. Sigmoid generates values between 0 and 1. If the value is closer to 1 that means that part necessary.

$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$$

The input gate: First, the current state x(t) and previously hidden state h(t-1) are passed into the second sigmoid function. The values are transformed between 0 and 1. Next, the same information of h(t-1) and x(t) will be passed through the tanh function. It will create a vector (C~(t)) with all the values between -1 and 1. Now the output that we get from sigmoid and tanh functions will have point to point multiplication.

$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \; + \; b_i\right)$$
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \; + \; b_C)$$

Cell State: After passing through forget and input gates, now we have to decide and store the information for the current state in the cell state. The previous cell state C(t-1) gets multiplied with forget vector f(t). If the outcome is 0, then values will be ignored. Next, the network takes the output of the input vector i(t) and performs point-by-point addition, which updates the cell state giving the network a new cell state C(t).

$$\boxed{C_t = f_t * C_{t-1} + i_t * \tilde{C}_t}$$

Output gate: Determines the value of the next hidden state.
The values of the h(t-1) and x(t) are passed into the third sigmoid function. Then the new cell state C(t) is passed through the tanh function. Both these outputs are multiplied point-by-point. Based upon the final value, the network decides which information the hidden state h(t) should carry.

$$\boxed{\begin{aligned} o_t &= \sigma\left(W_o\left[h_{t-1}, x_t\right] + b_o\right) \\ h_t &= o_t * \tanh\left(C_t\right) \end{aligned}}$$

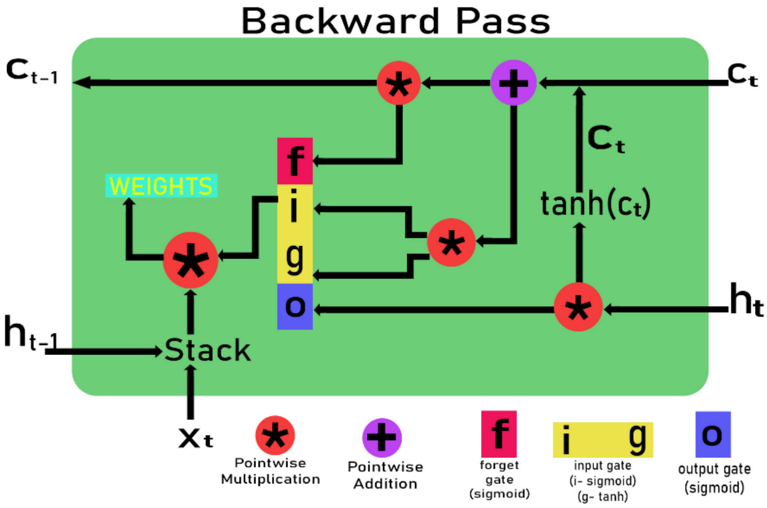
Activation Functions:

Sigmoid: $\dfrac{1}{1+e^{-x}}$

Tanh: $\dfrac{\left(e^{x}-e^{-x}\right)}{\left(e^{x}+e^{-x}\right)}$

Softmax: $s\left(x_i\right) = \dfrac{e^{x_i}}{\sum_{j=1}^{n} e^{x_j}}$

Backward Propagation



gate derivatives

$$d\Gamma_o^{\langle t\rangle} = da_{next} * \tanh(c_{next}) * \Gamma_o^{\langle t\rangle} * (1 - \Gamma_o^{\langle t\rangle})$$

$$d\tilde{c}^{\langle t\rangle} = dc_{next} * \Gamma_i^{\langle t\rangle} + \Gamma_o^{\langle t\rangle}(1 - \tanh(c_{next})^2) * i_t * da_{next} * \tilde{c}^{\langle t\rangle} * (1 - \tanh(\tilde{c})^2)$$

$$d\Gamma_u^{\langle t\rangle} = dc_{next} * \tilde{c}^{\langle t\rangle} + \Gamma_o^{\langle t\rangle}(1 - \tanh(c_{next})^2) * \tilde{c}^{\langle t\rangle} * da_{next} * \Gamma_u^{\langle t\rangle} * (1 - \Gamma_u^{\langle t\rangle})$$

$$d\Gamma_f^{\langle t\rangle} = dc_{next} * \tilde{c}_{prev} + \Gamma_o^{\langle t\rangle}(1 - \tanh(c_{next})^2) * c_{prev} * da_{next} * \Gamma_f^{\langle t\rangle} * (1 - \Gamma_f^{\langle t\rangle})$$

parameter derivatives

$$dW_f = d\Gamma_f^{\langle t\rangle} * \begin{pmatrix} a_{prev} \\ x_t \end{pmatrix}^T$$

$$dW_u = d\Gamma_u^{\langle t\rangle} * \begin{pmatrix} a_{prev} \\ x_t \end{pmatrix}^T$$

$$dW_c = d\tilde{c}^{\langle t\rangle} * \begin{pmatrix} a_{prev} \\ x_t \end{pmatrix}^T$$

$$dW_o = d\Gamma_o^{\langle t\rangle} * \begin{pmatrix} a_{prev} \\ x_t \end{pmatrix}^T$$

$$da_{prev} = W_f^T * d\Gamma_f^{\langle t\rangle} + W_u^T * d\Gamma_u^{\langle t\rangle} + W_c^T * d\tilde{c}^{\langle t\rangle} + W_o^T * d\Gamma_o^{\langle t\rangle}$$

$$dc_{prev} = dc_{next}\Gamma_f^{\langle t\rangle} + \Gamma_o^{\langle t\rangle} * (1 - \tanh(c_{next})^2) * \Gamma_f^{\langle t\rangle} * da_{next}$$

$$dx^{\langle t\rangle} = W_f^T * d\Gamma_f^{\langle t\rangle} + W_u^T * d\Gamma_u^{\langle t\rangle} + W_c^T * d\tilde{c}_t + W_o^T * d\Gamma_o^{\langle t\rangle}$$