LSTM networks explained easily

Valerio Velardo

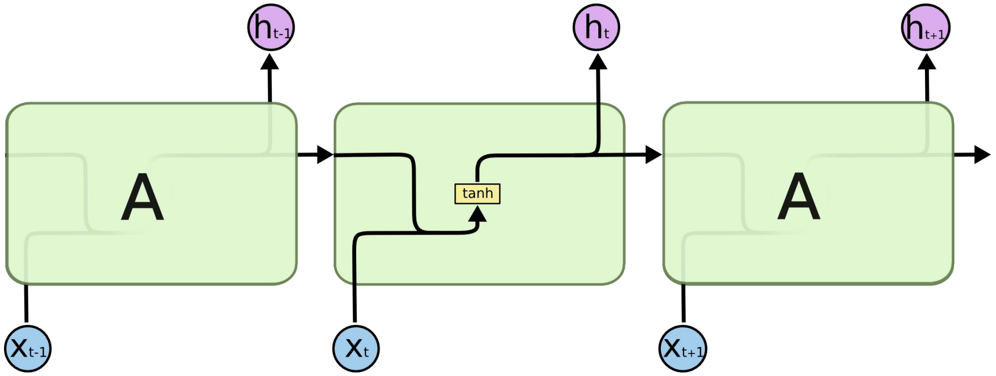
Issues with simple RNNs

- No long-term memory
- Network can’t use info from the distant past
- Can’t learn patterns with long dependencies

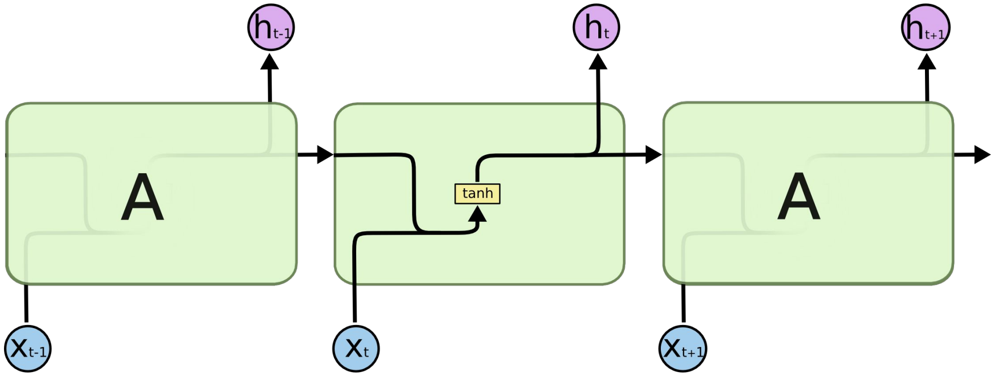
Long Short Term Memory (LSTM)

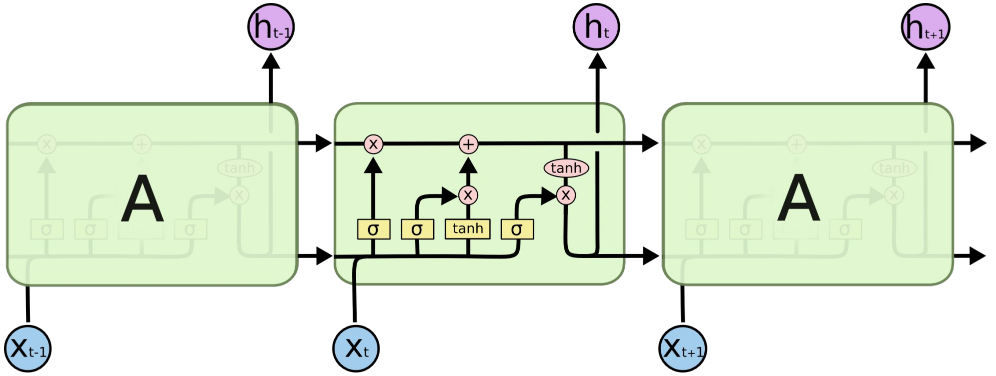
- Special type of RNN
- Can learn long-term patterns
- Detects patterns with 100 steps
- Struggles with 100s/1000s of steps

Simple RNN vs LSTM



Simple RNN vs LSTM

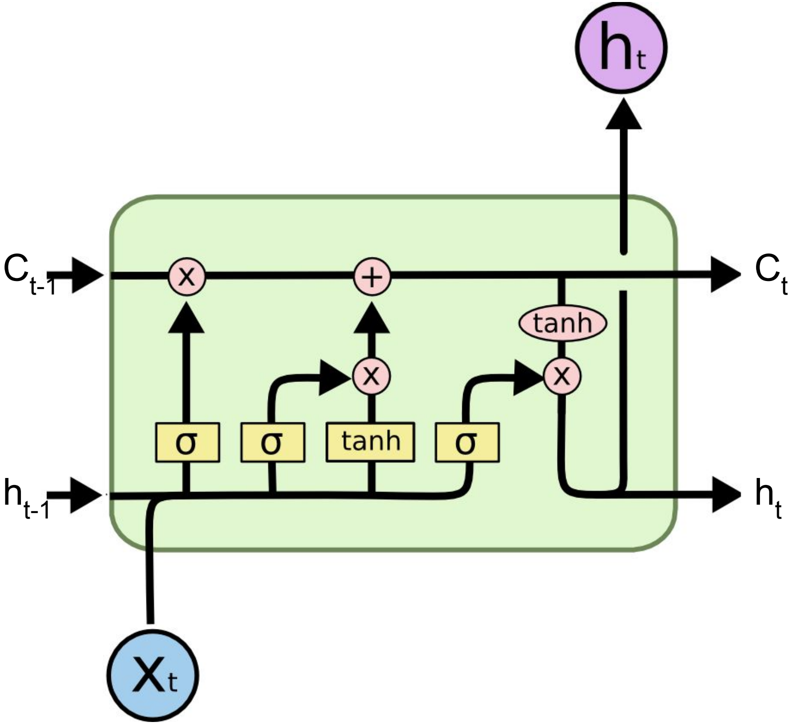




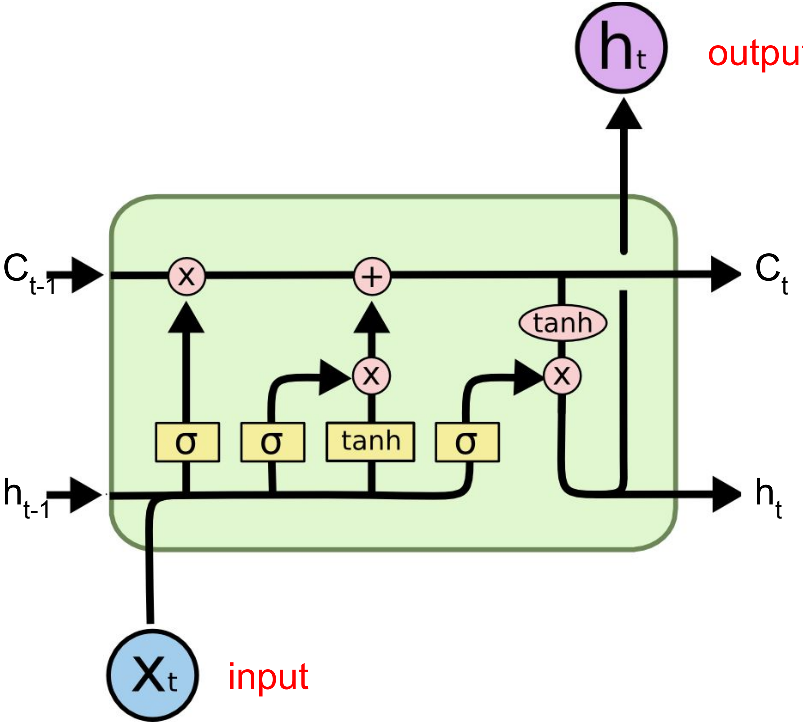
LSTM cell

- Contains a simple RNN cell
- Second state vector = cell state = long-term memory
- Forget gate
- Input gate
- Output gate
- Gates work as filters

LSTM cell



LSTM cell



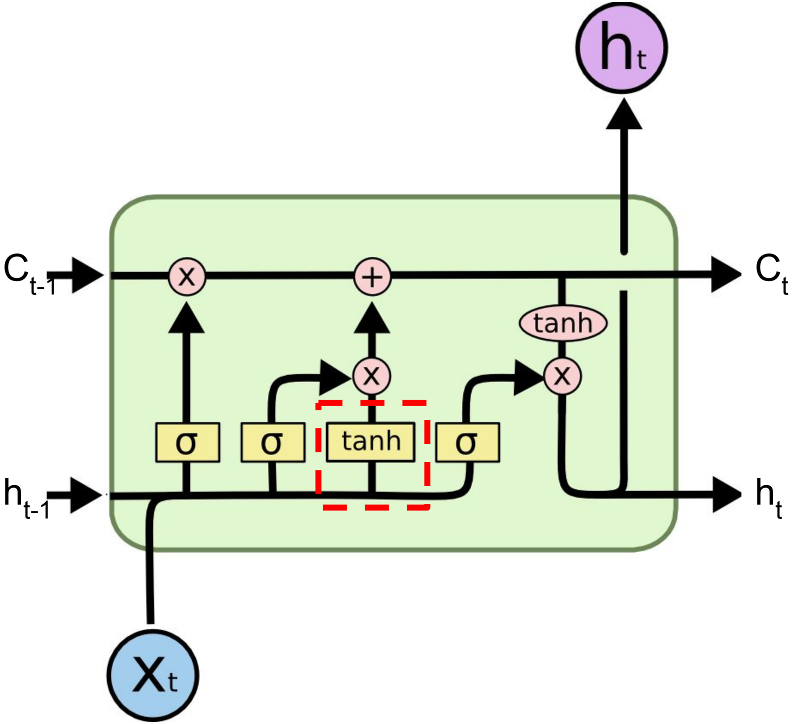
output

cell state

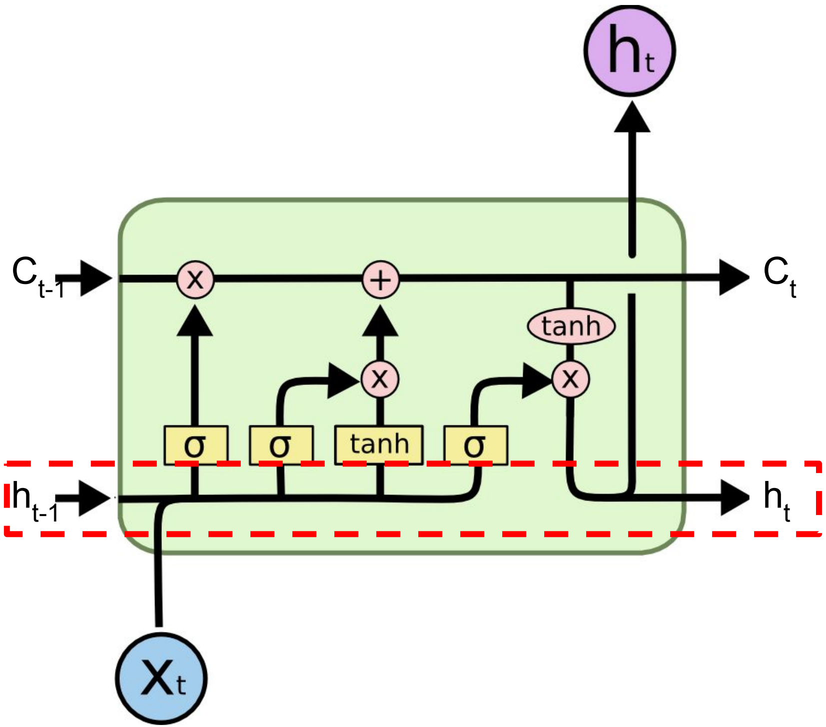
hidden state = output

input

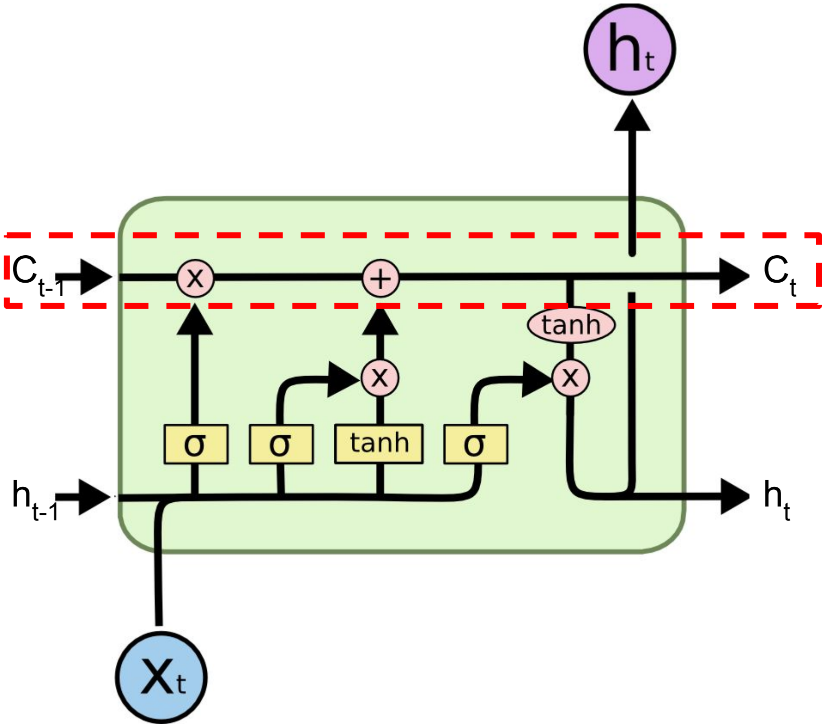
Simple RNN cell



Short-term memory/hidden state

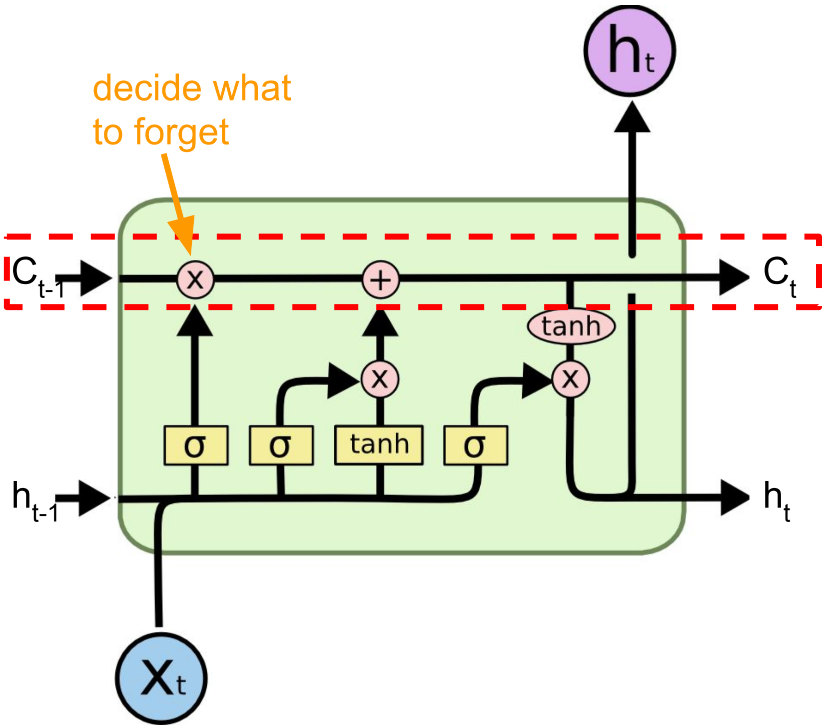


Cell state



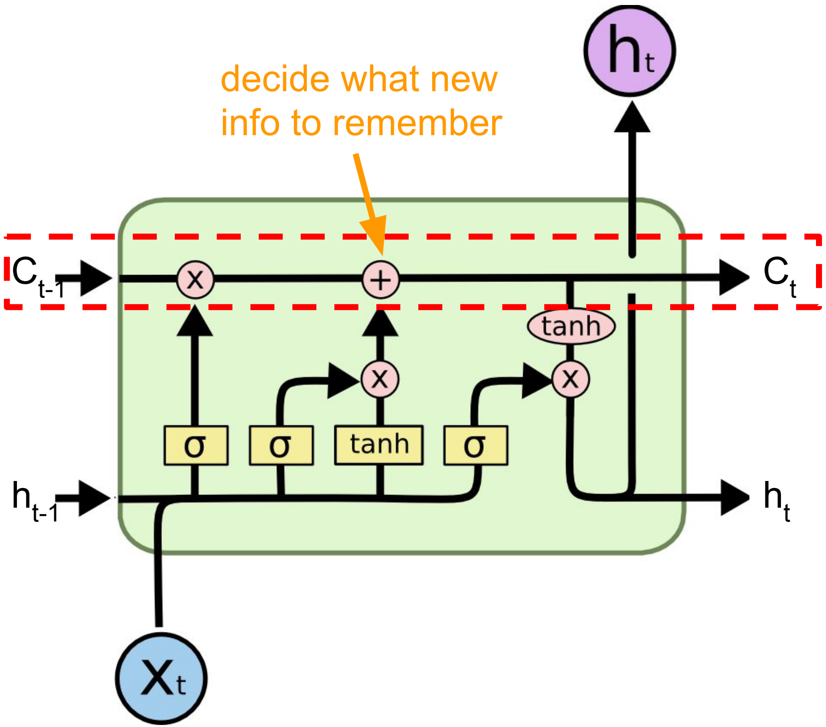
- Cell state updated twice
- Few computations -> stabilise gradients

Cell state



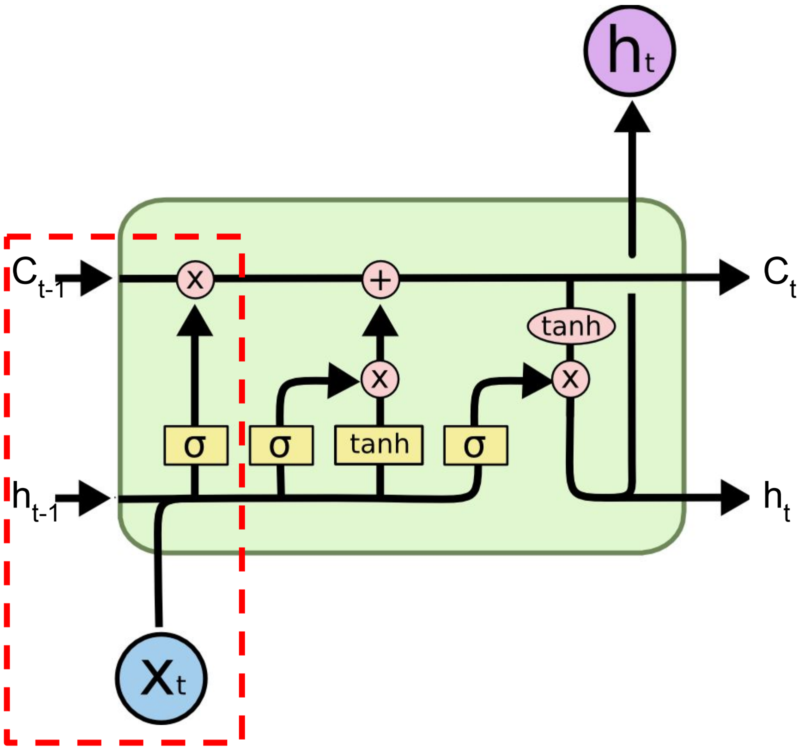
- Cell state updated twice
- Few computations -> stabilise gradients

Cell state

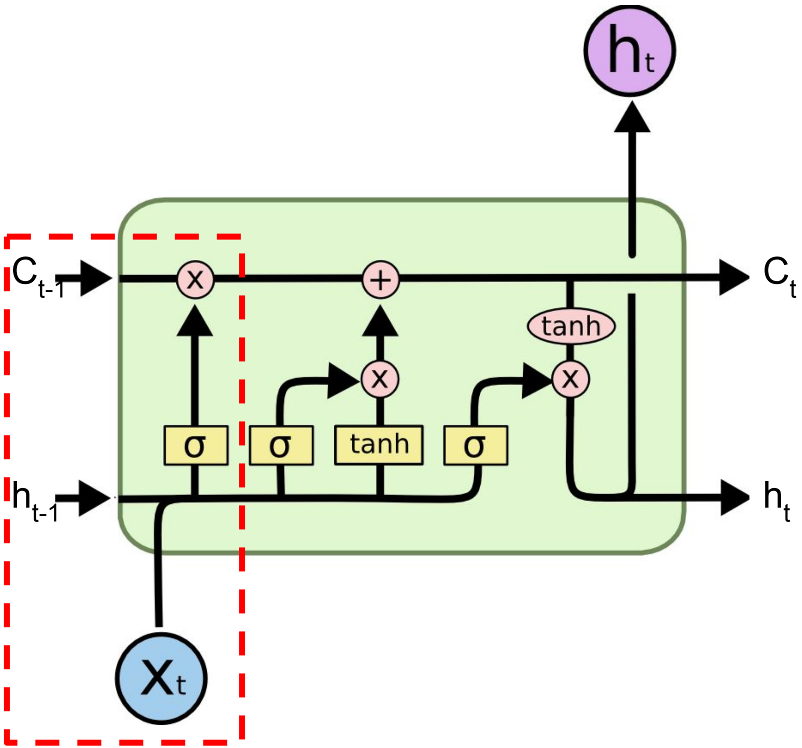


- Cell state updated twice
- Few computations -> stabilise gradients

Forget

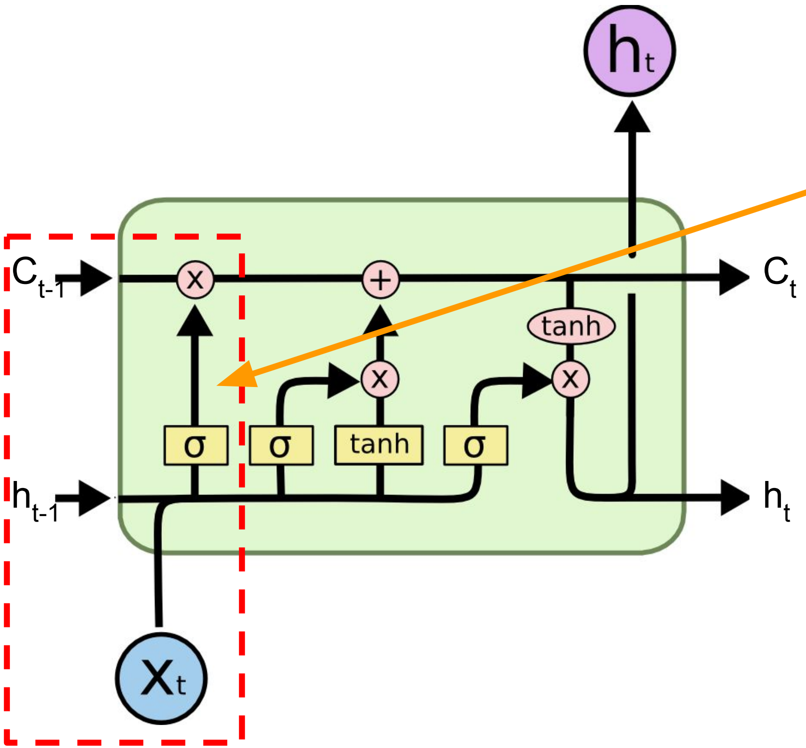


Forget



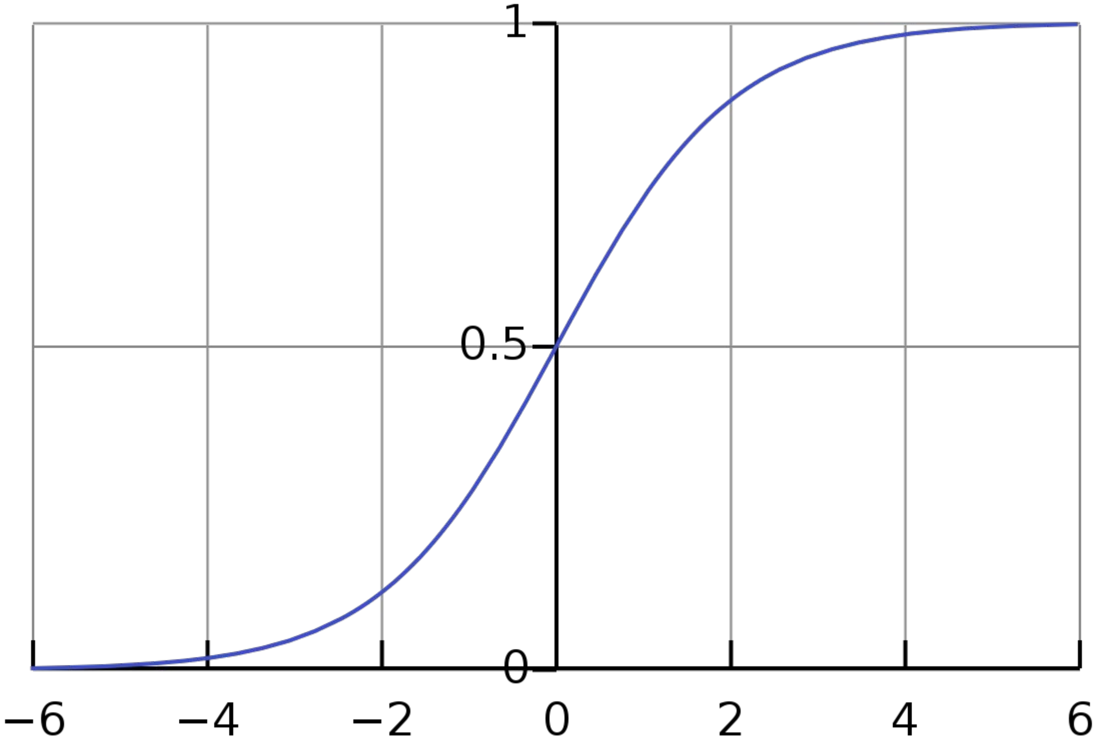
$$f_t = \sigma(W_f[h_{t-1}, x_t] + b_f)$$

Forget

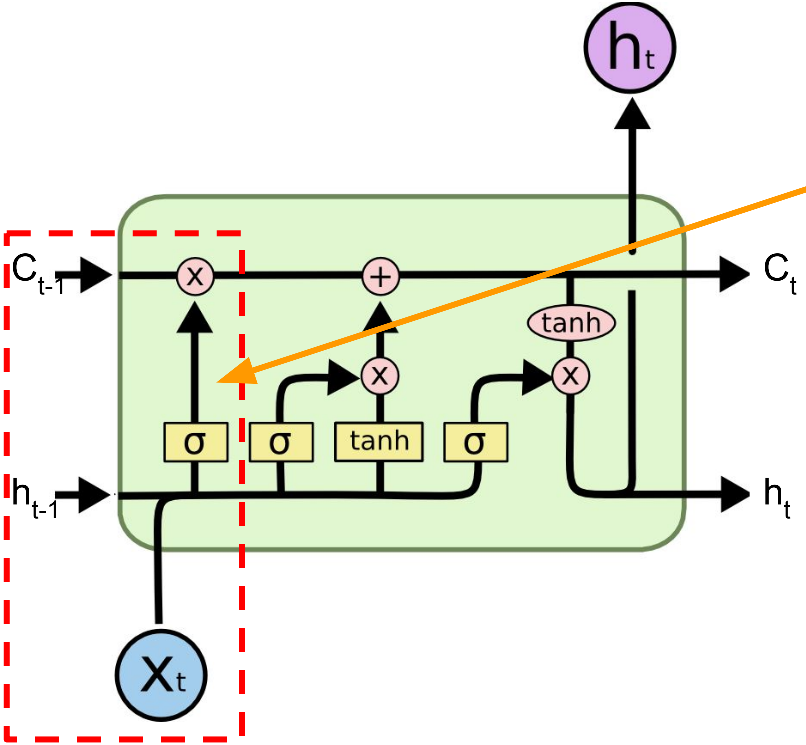


$$f_t = \sigma(W_f[h_{t-1}, x_t] + b_f)$$

Sigmoid

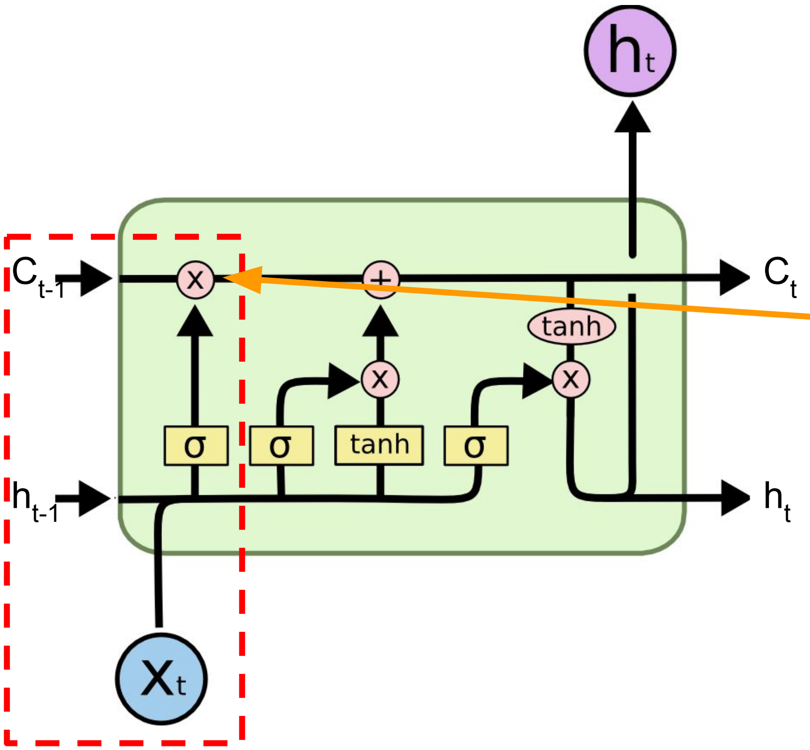


Forget



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

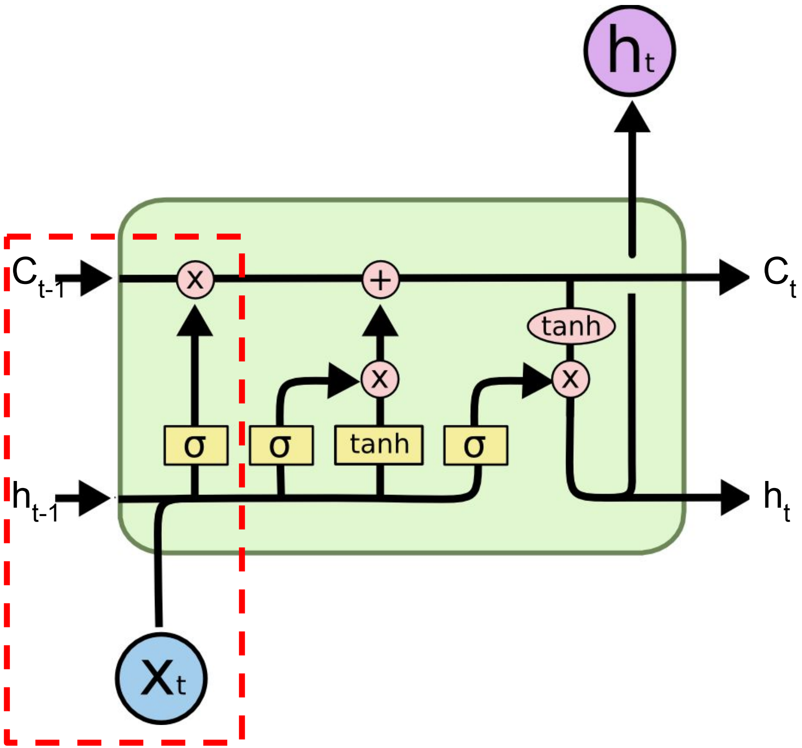
Forget



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

$$C_t^f = C_{t-1} * f_t$$

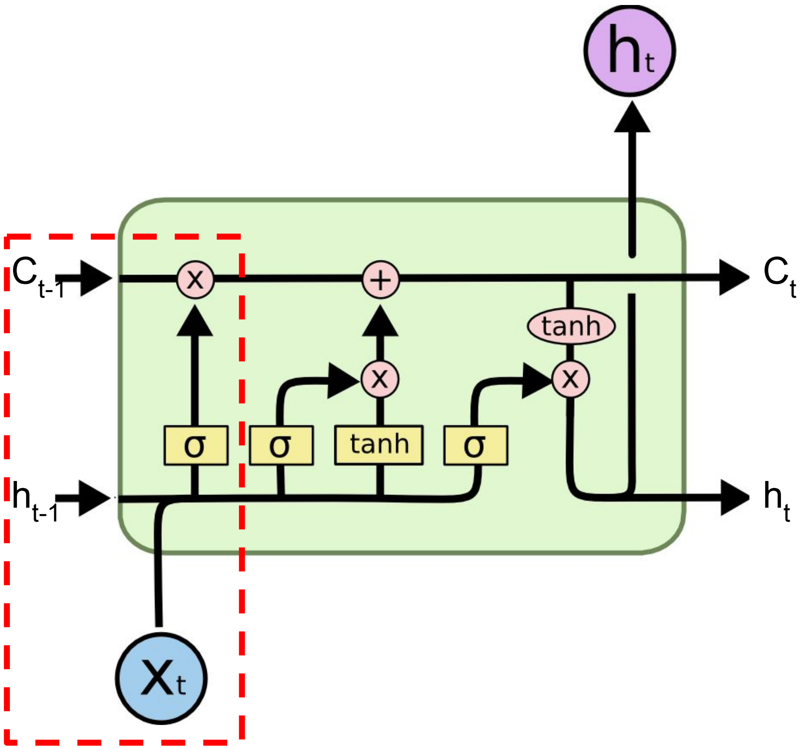
Forget



$$C_t^f = C_{t-1} * f_t$$

$$C_{t-1} = [1, 2, 4]$$
$$f_t = [1, 0, 1]$$

Forget

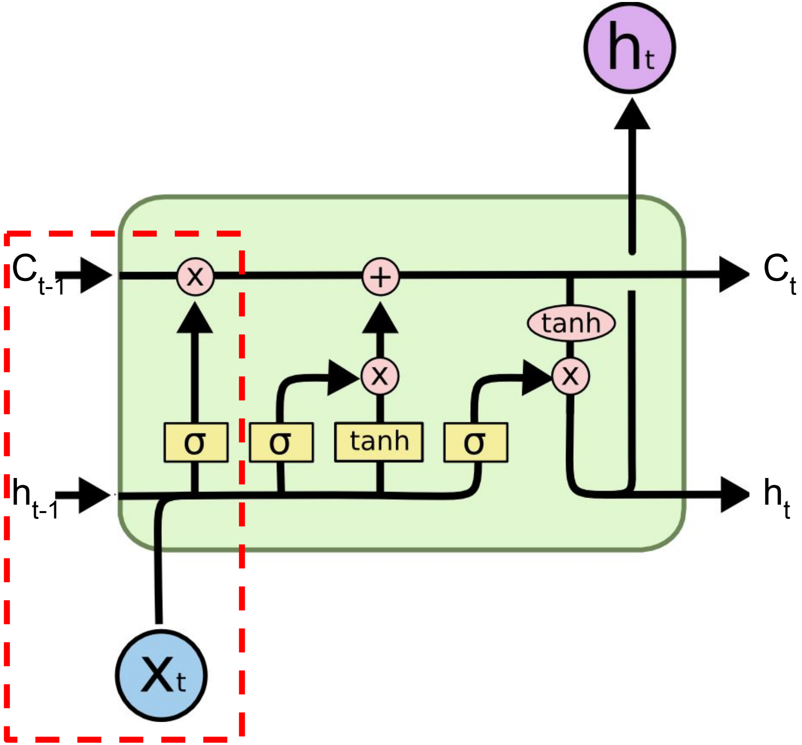


$$C_t^f = C_{t-1} * f_t$$

$$C_{t-1} = [1, 2, 4]$$
$$f_t = [1, 0, 1]$$

$$\boxed{C_t^f = [1, 2, 4] * [1, 0, 1] = [1, 0, 4]}$$

Forget

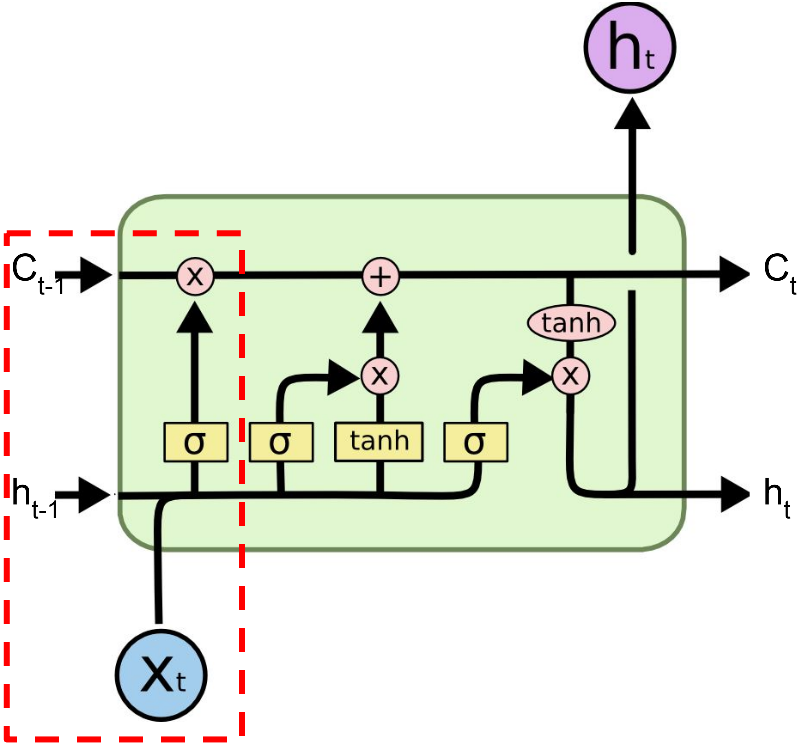


$$C_t^f = C_{t-1} * f_t$$

$$C_{t-1} = [1, 2, 4]$$
$$f_t = [1, 0, 1]$$

$$C_t^f = [\boxed{1}, 2, \boxed{4}] * [1, 0, 1] = [\boxed{1}, 0, \boxed{4}]$$

Forget

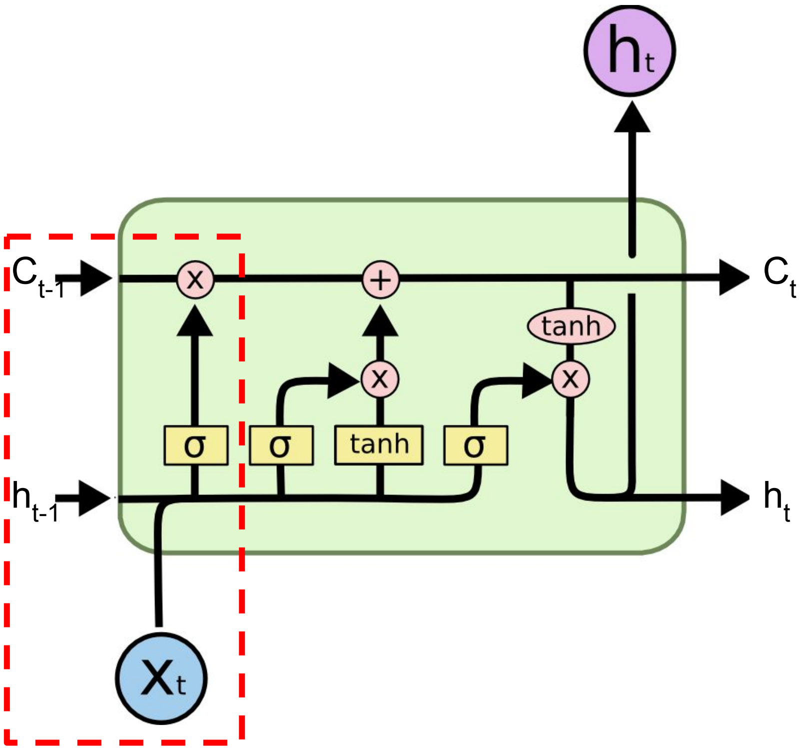


$$C_t^f = C_{t-1} * f_t$$

$$C_{t-1} = [1, 2, 4]$$
$$f_t = [1, 0, 1]$$

$$C_t^f = [\boxed{1}, 2, \boxed{4}] * [\boxed{1}, 0, \boxed{1}] = [\boxed{1}, 0, \boxed{4}]$$

Forget

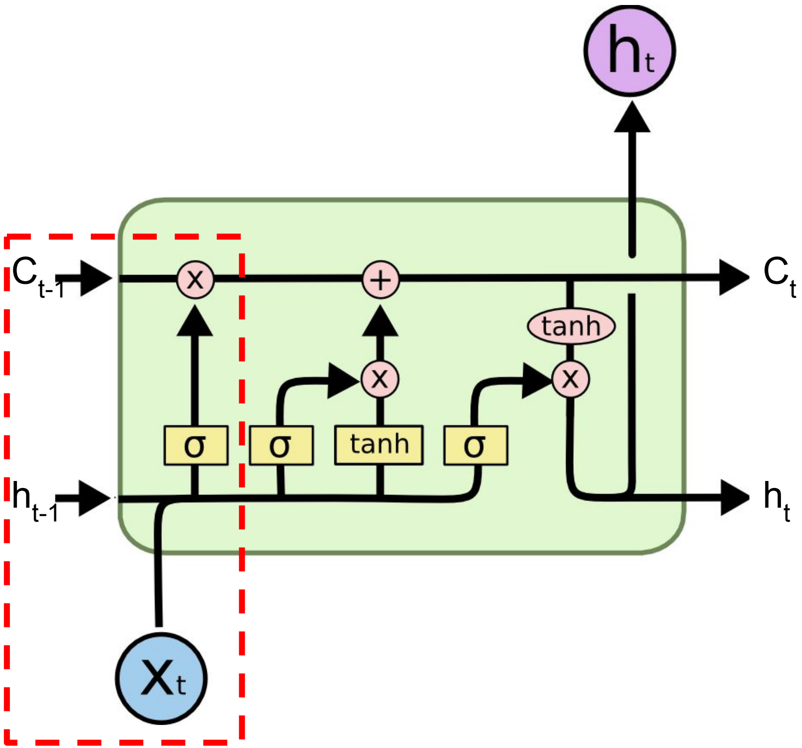


$$C_t^f = C_{t-1} * f_t$$

$$C_{t-1} = [1, 2, 4]$$
$$f_t = [1, 0, 1]$$

$$C_t^f = [1, \boxed{2}, 4] * [1, 0, 1] = [1, \boxed{0}, 4]$$

Forget

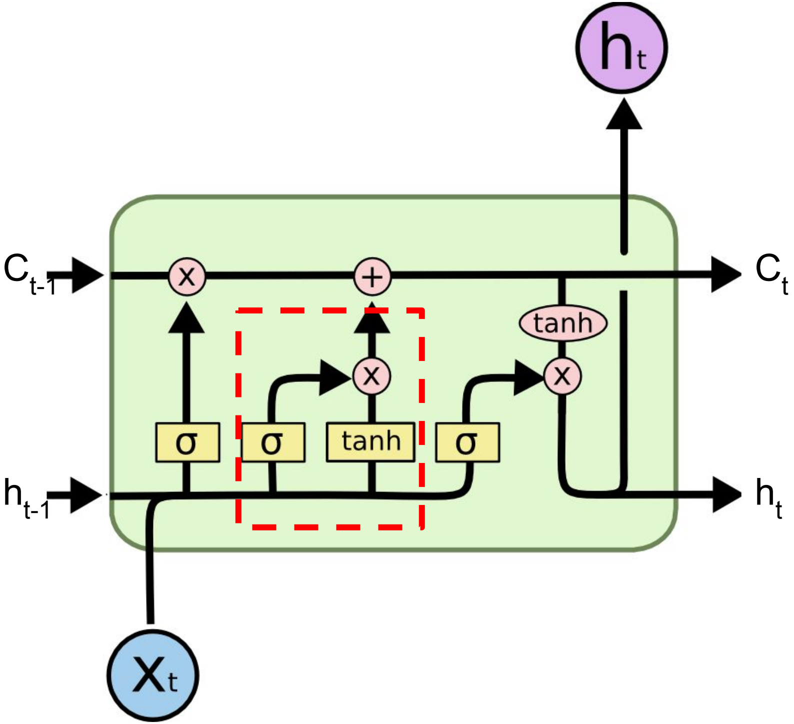


$$C_t^f = C_{t-1} * f_t$$

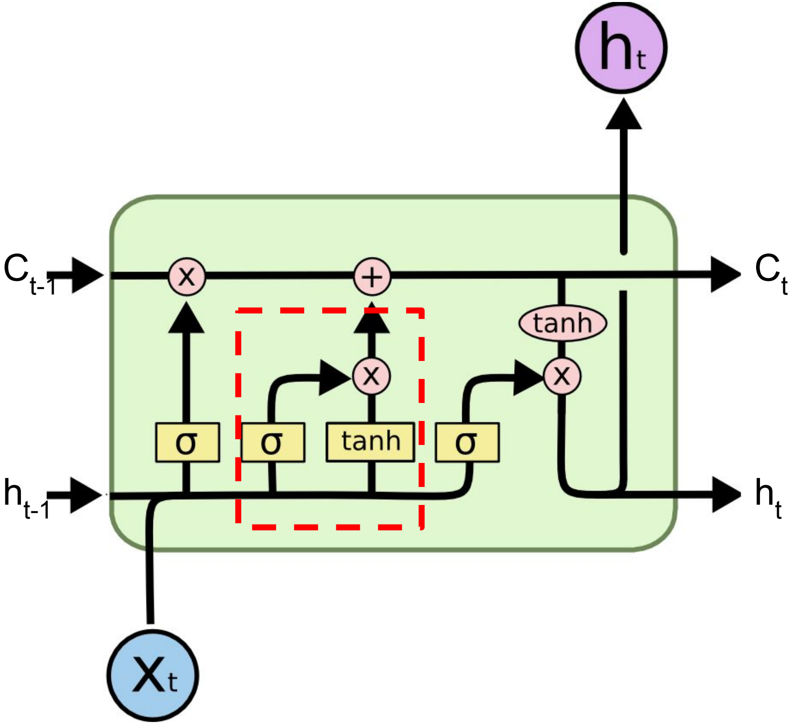
$$C_{t-1} = [1, 2, 4]$$
$$f_t = [1, 0, 1]$$

$$C_t^f = [1, \boxed{2}, 4] * [1, \boxed{0}, 1] = [1, \boxed{0}, 4]$$

Input

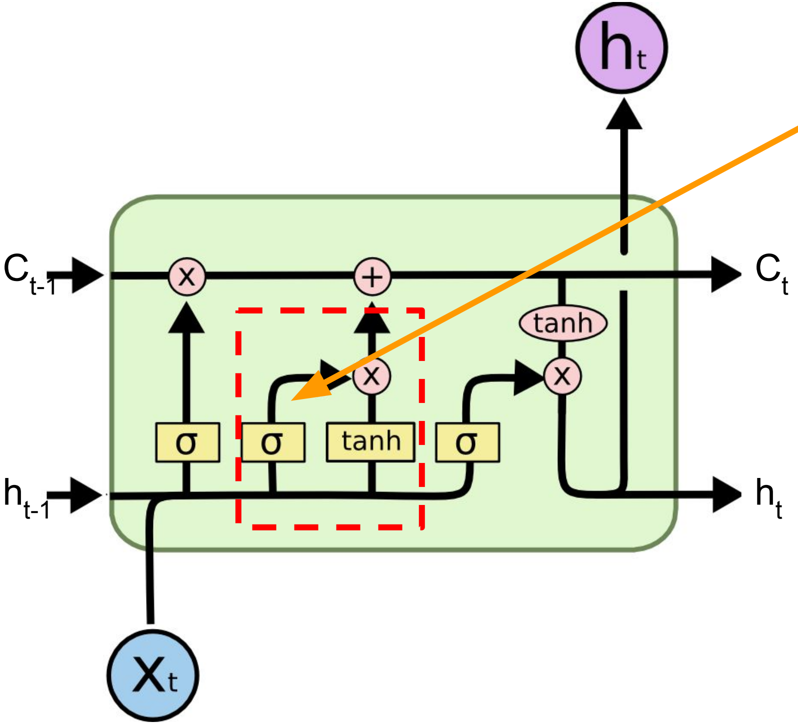


Input



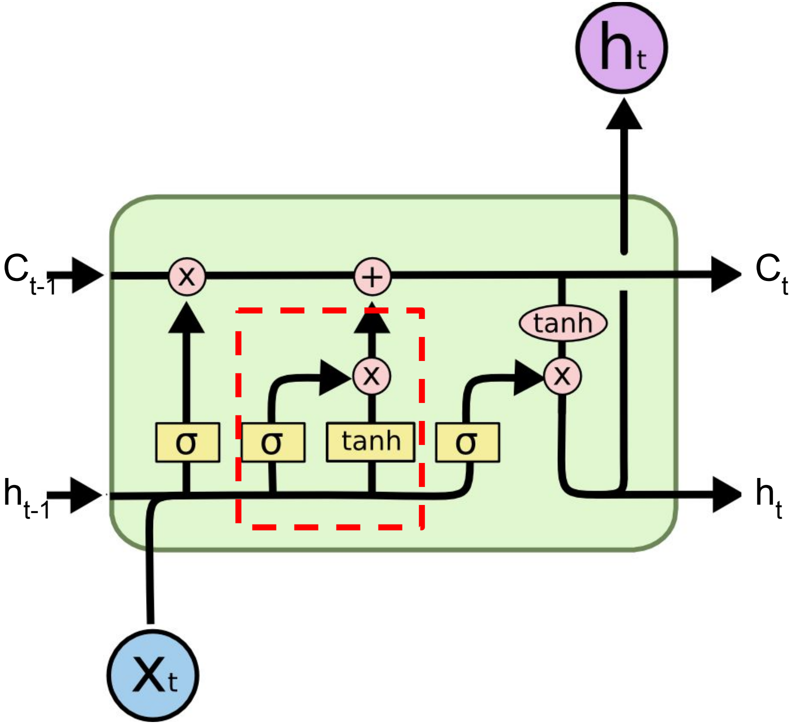
$$i_t = \sigma(W_i[h_{t-1}, x_t] + b_i)$$

Input



$$i_t = \sigma(W_i[h_{t-1}, x_t] + b_i)$$

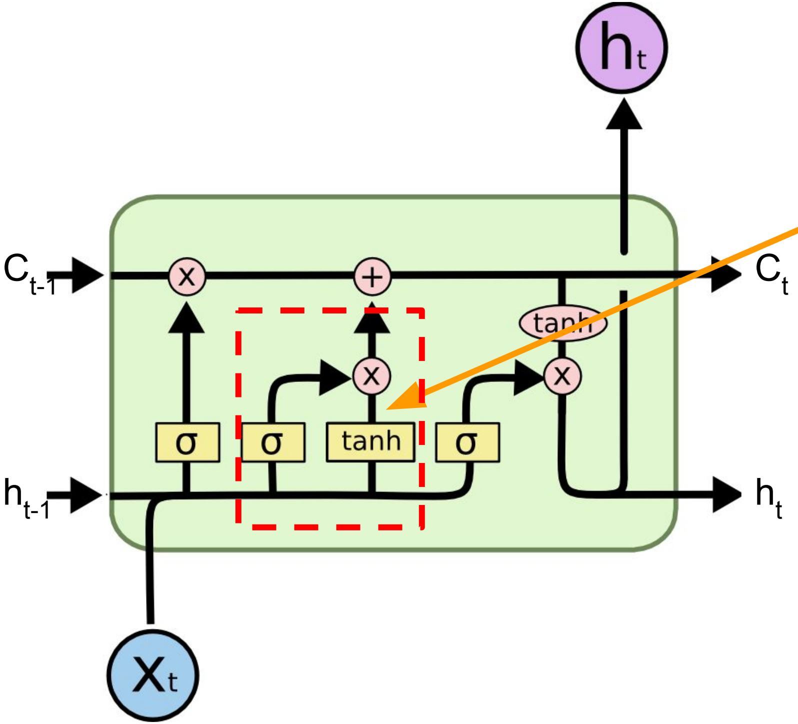
Input



$$i_t = \sigma(W_i[h_{t-1}, x_t] + b_i)$$

$$C_t^{'} = tanh(W_c[h_{t-1}, x_t] + b_C)$$

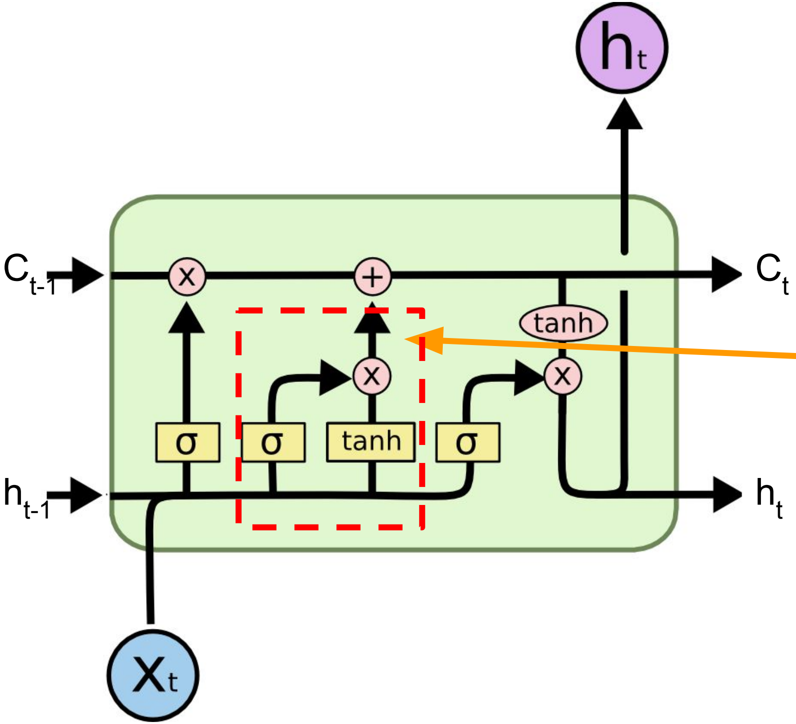
Input



$$i_t = \sigma(W_i[h_{t-1}, x_t] + b_i)$$

$$C_t^{'} = tanh(W_c[h_{t-1}, x_t] + b_C)$$

Input

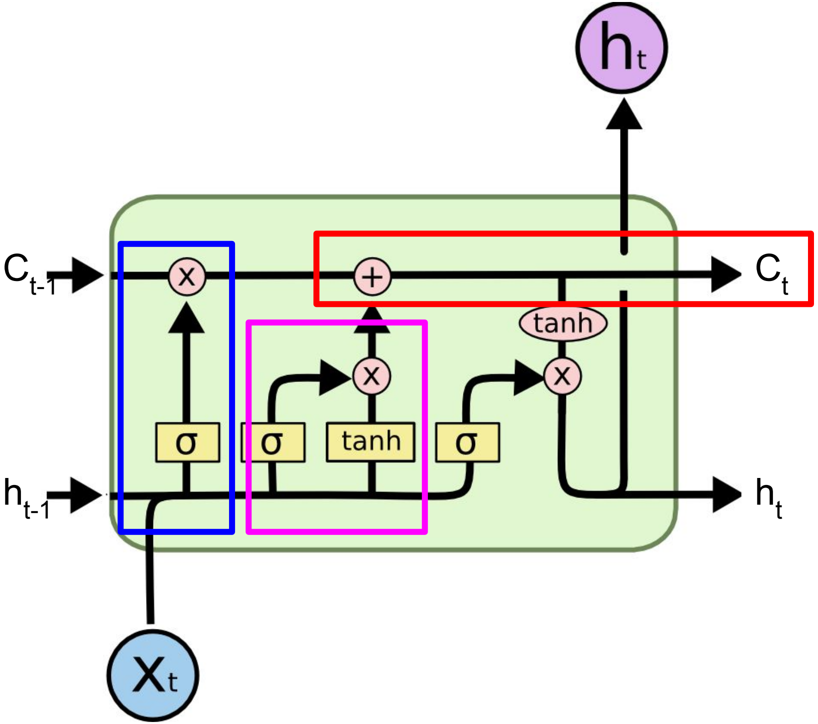


$$i_t = \sigma(W_i[h_{t-1}, x_t] + b_i)$$

$$C_t^{'} = tanh(W_c[h_{t-1}, x_t] + b_C)$$

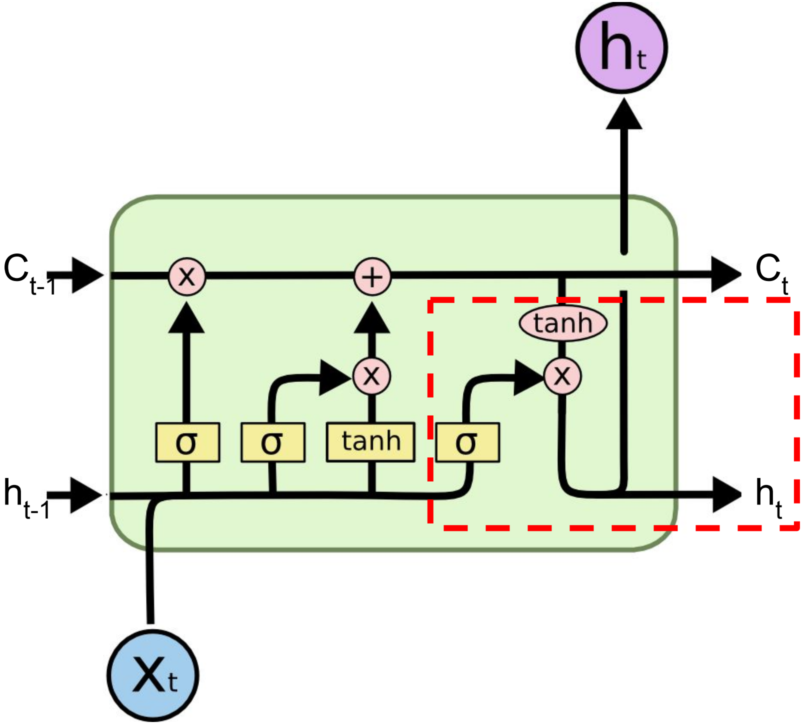
$$C_t^i = C_t^{'} * i_t$$

Cell state

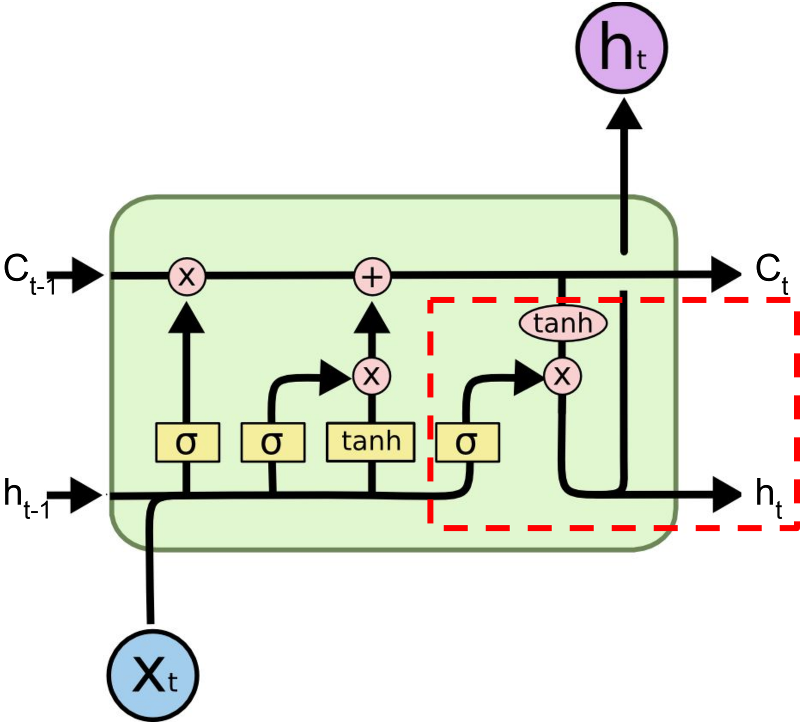


$$\boxed{C_t} = \boxed{C_t^f} + \boxed{C_t^i}$$

Output

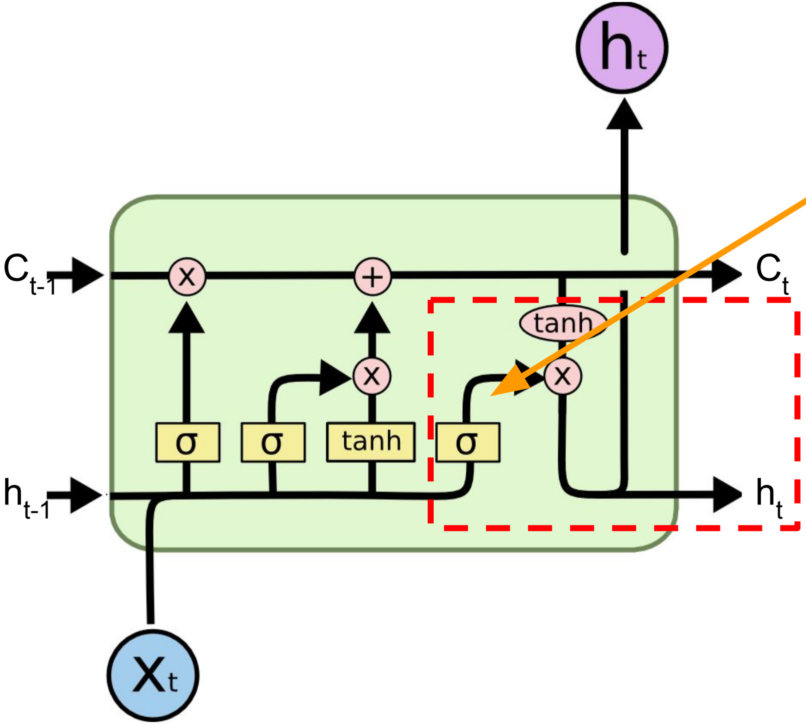


Output



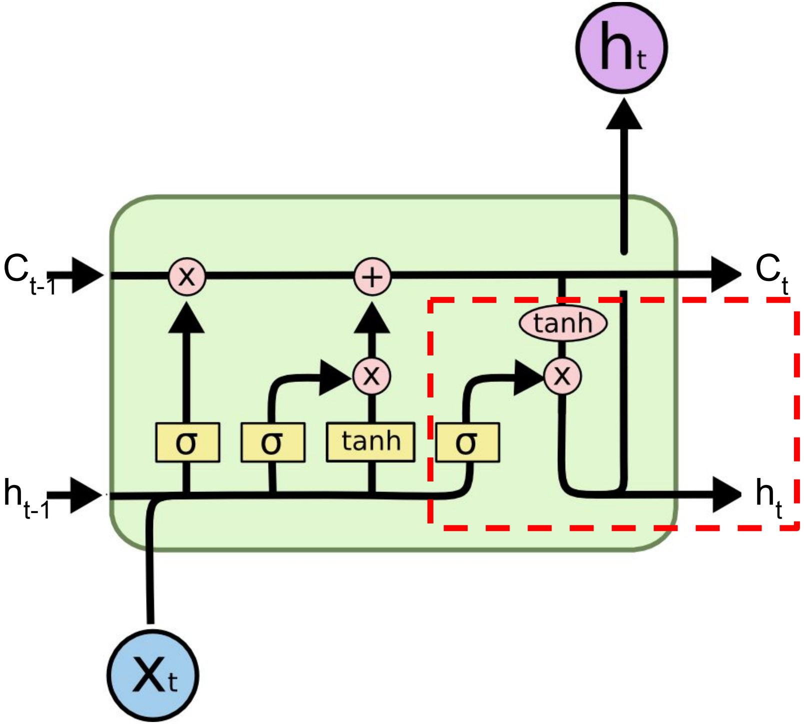
$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o)$$

Output



$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o)$$

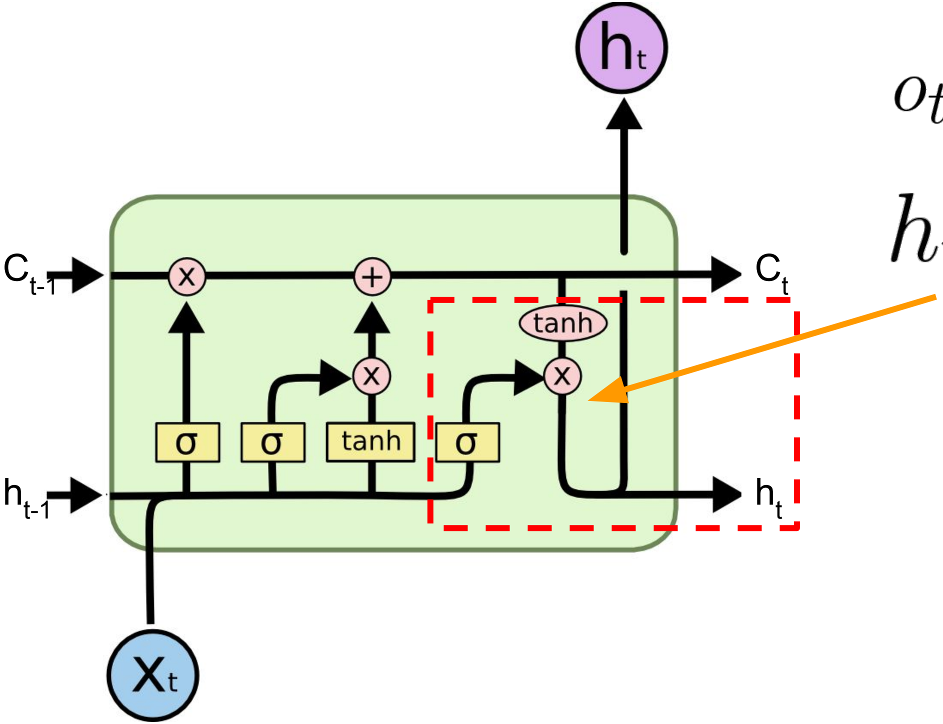
Output



$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o)$$

$$h_t = o_t * tanh(C_t)$$

Output

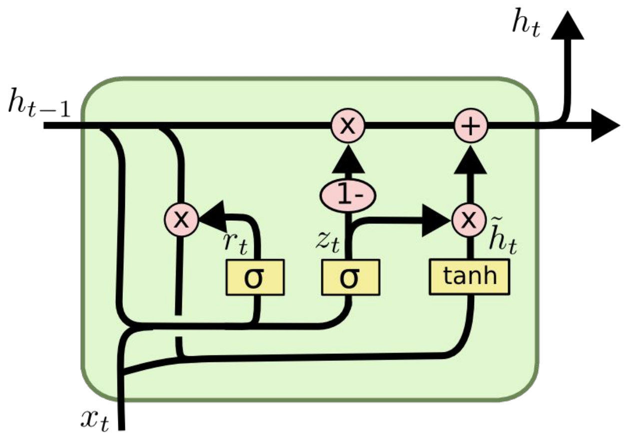


$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o)$$

$$h_t = o_t * tanh(C_t)$$

LSTM variants

- Gated Recurrent Unit - GRU



What’s up next?

- Preprocess data for RNN