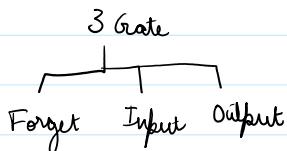


LSTM

Long Short Term Memory



RNN (Recurrent Neural Net)

Remembers sequences over time

Trained to remember important things & forget unimportant things

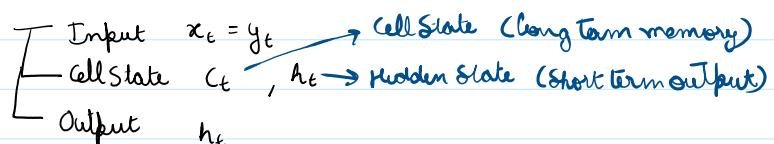
1 Cell

Works like a series of logic gates

Used for time Series prediction task

$$\text{e.g. } y_t = [1, 0.9, 1.1] \quad (\text{timeseries})$$

A single LSTM Cell has



Let's walk through our sample 1 cell LSTM

①

Initialize

$$c_0 = 0$$

$$h_0 = 0$$

$$t = 1 \Rightarrow x_1 = 1.0$$

②

Forget Gate

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

Annotations: 'weight Matrix for forget gate' points to w_f ; 'bias vector for forget gate' points to b_f ; 'Prev. hidden state ($t-1$)' points to h_{t-1} ; 'Input at t ' points to x_t .

$$t=1 \Rightarrow f_1 = \sigma (w_f [h_0, x_1] + b_f)$$

$$\sigma (1(0) + 1(1.0) + 1) = 0.25$$

KEEP
i.e. 88% of previous memory

In our case previous memory = 0

③

Input Gate

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

$$t=1 \Rightarrow i_1 = \sigma(1(0) + 1(1) + 1) = 2\sigma$$

ie. ^{ADD} 88% of input

④ Candidate Memory

$$\tilde{C}_t = \tanh (W_c [h_{t-1}, x_t] + b_c)$$

$$t=1 \Rightarrow \tilde{C}_1 = \tanh (W_c [h_0, x_1] + b_c)$$

$$= \tanh (1(0) + 1(1) + 1) = \tanh(2) = 0.964$$

⑤ Cell State update

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

$$t=1 \Rightarrow C_1 = f_1 C_0 + i_1 \tilde{C}_1$$

$$= 0.88(0) + (0.88)(0.964) = 0.848$$

⑥ Output

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

$$t=1 \Rightarrow O_1 = \sigma (W_o [h_0, x_1] + b_o)$$

$$= \sigma (1(0) + 1(1) + 1) = 2\sigma = 0.88$$

⑦ Hidden State

$$h_t = O_t \tanh(C_t)$$

$$t=1 \Rightarrow h_1 = O_1 \tanh(C_1)$$

$$= 0.88 \tanh(0.848) = 0.607$$

Repeat this for every element

$$\underline{t=2} \Rightarrow x_2 = 0.9$$

Repeat this for every element

$$\underline{t=2} \Rightarrow$$

$$x_2 = 0.9$$

$$f_2 = \sigma(w_f [h_1, x_2] + b_f) = \sigma(1(0.607) + 1(0.9) + 1) = 2.507\sigma = 0.925$$

$$i_2 = \sigma(w_i [h_1, x_2] + b_i) = \sigma(1(0.607) + 1(0.9) + 1) = 2.507\sigma = 0.925$$

$$\tilde{c}_2 = \tanh(w_c [h_1, x_2] + b_c) = \tanh(1(0.607) + 1(0.9) + 1) = \tanh(2.507) = 0.987$$

$$c_2 = f_2 i_2 + \tilde{c}_2 \tilde{c}_2 = (0.925)(0.848) + (0.925)(0.987) = 1.697$$

$$o_2 = \sigma(w_o [h_1, x_2] + b_o) = \sigma(1(0.607) + 1(0.9) + 1) = 2.507\sigma = 0.925$$

$$h_2 = o_2 \tanh(c_2) = 0.925 \tanh(1.697) = 0.865$$

The above process basically describes a forward pass

This is followed by standard NN training procedure

$$\text{Loss Function} = L = \frac{1}{N} \sum (\hat{y}_{t+1} - y_{t+1})^2$$

Back Propagation

$$w \leftarrow w - \eta \frac{\partial L}{\partial w}$$

Annotations: $\frac{\partial L}{\partial w}$ is labeled "loss function", η is labeled "learning rate", and w is labeled "weight matrix".