LSTM

July 31, 2025 1:10 PM

L S T M

Long Short Term Memory

3 Gate

- Forget
- Input
- Output

1 Cell

RNN (Recurrent Neural Net)

Remembers sequences over time

Trained to remember important things & forget unimportant things

Works like a series of logic gates

Used for time Series prediction task

e.g. ⇒ $y_t = [1, 0.9, 1.1]$ (timeseries)

A single LSTM Cell has
- Input $x_t = y_t$
- Cell State C_t , h_t
- Output h_t

C_t → Cell State (Long Term memory)

h_t → Hidden State (Short term output)

Lets walk through our simple Cell LSTM

(1) Initialize

$C_0 = 0$

$h_0 = 0$

$t = 1 \Rightarrow x_1 = 1.0$

(2) Forget Gate

Weight Matrix for forget gate (Learnt)

Bias vector for forget gate

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

Prev. hidden state (t-1)

Input at t

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

$$\sigma(1(0) + 1(1.0) + 1) = 2\sigma$$

KEEP

i.e. 88% of previous memory

In our case previous memory = 0

(3) Input Gate

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

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

ie. ADD 88% of input

④ Candidate Memory

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

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

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

⑤ Cell State update

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

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

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

⑥ Output

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

$t=1 \Rightarrow \quad o_1 = \sigma(\omega_o [h_0, x_1] + b_o)$

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

⑦ Hidden State

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

$t=1 \Rightarrow \quad h_1 = o_1 \tanh(C_1)$

$= 0.88 \tanh(0.848) = 0.607$

Repeat this for every element

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

Repeat this for every element

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

$x_2 = 0.9$

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

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

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

$C_2 = f_2 C_1 + i_2 \tilde{C}_2 = (0.925)(0.848) + (0.925)(0.987) = 1.697$

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

$h_2 = o_2 \tanh(C_2) = 0.925 \tanh(1.697) = 0.865$

The above process basically describes a forward pass

This is followed by standard NN training procedure

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

Back Propagation

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

Loss function

Weight matrix

Learning rate