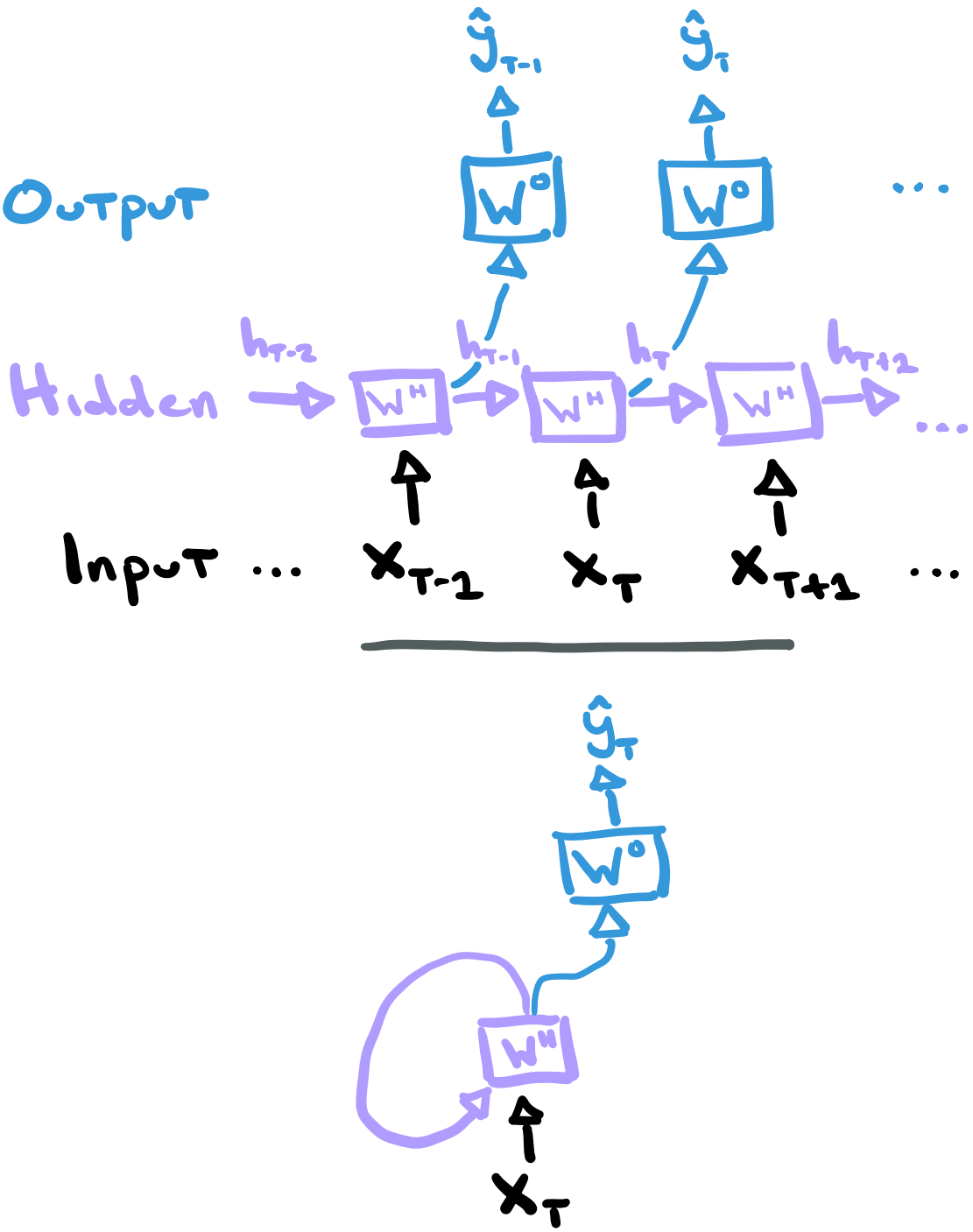
DS 4440

Recurrent Neural Networks 2

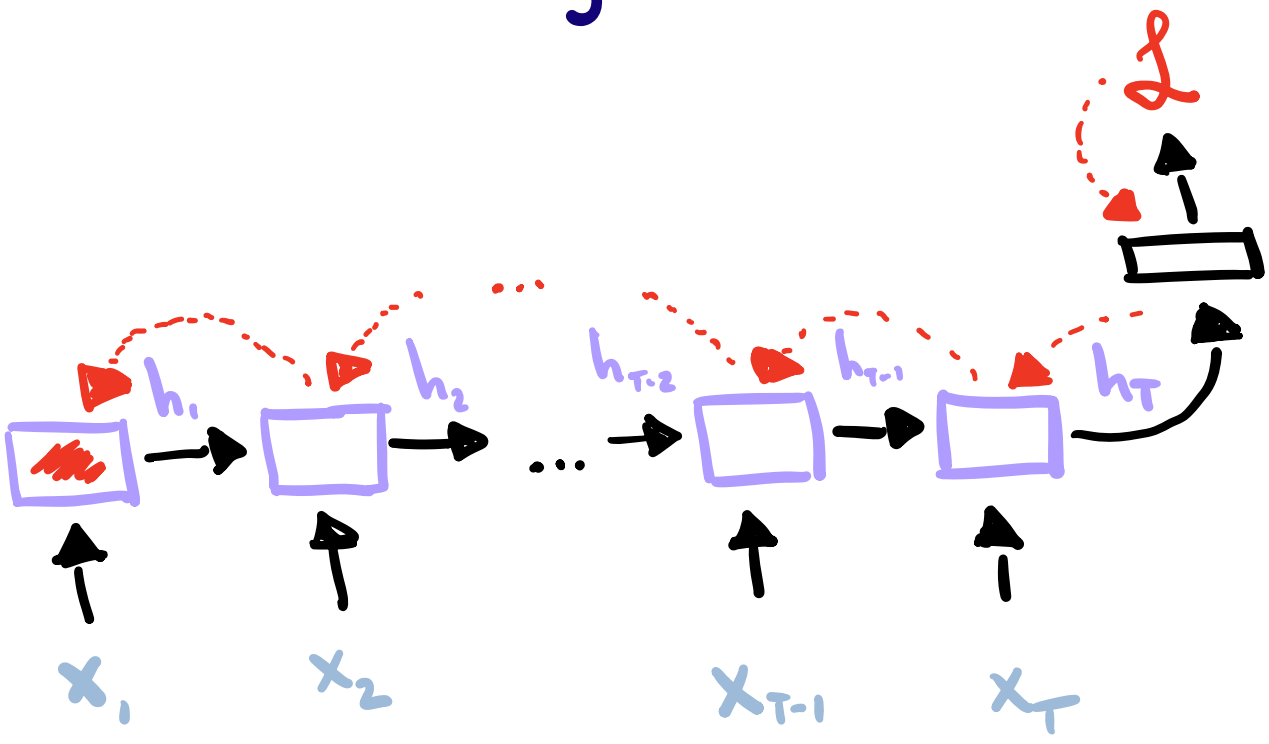
Last Time: RNNs for Sequences



Problem long term dependencies

(0 0 0 0 0)	0
(1 0 0 0 0)	1
...	
(1 0 0 0 0)	1
x	y

Signal must flow through entire sequence!

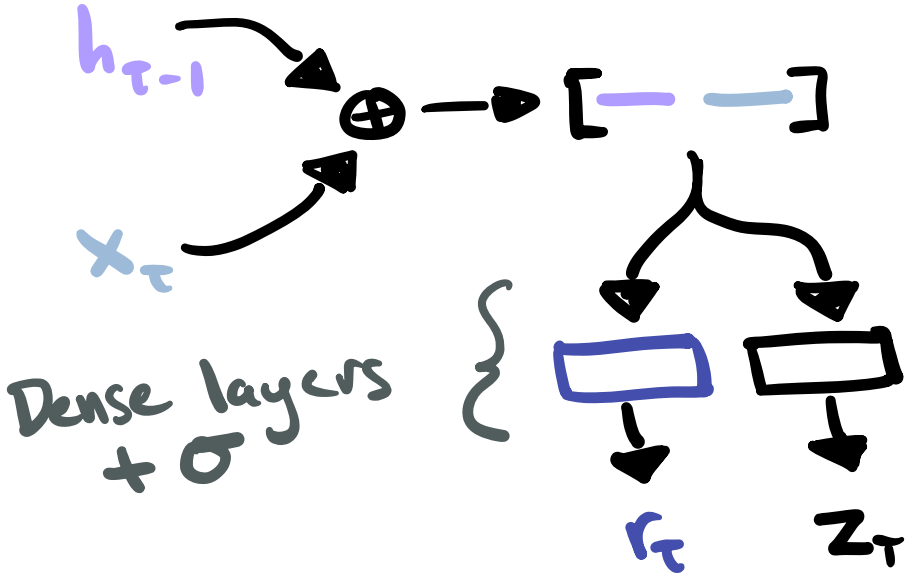


Gated Recurrent Units (GRUs)

Similar to LSTMs (next) but simpler.

IDEA Introduce gating mechanisms

To allow Model to update h
or opt not to (Skip)



r_t = Reset

z_t = update

$$r_t = \sigma(x_t W_r^x + h_{t-1} W_r^h + b_r)$$

$$z_t = \sigma(x_t W_z^x + h_{t-1} W_z^h + b_z)$$

Reset

$$\tilde{h}_t \leftarrow \tanh(x_t W_h^x + (\underbrace{r_t \odot h_{t-1}}_{\text{Permits forgetting}}) W_h^h + b_h)$$

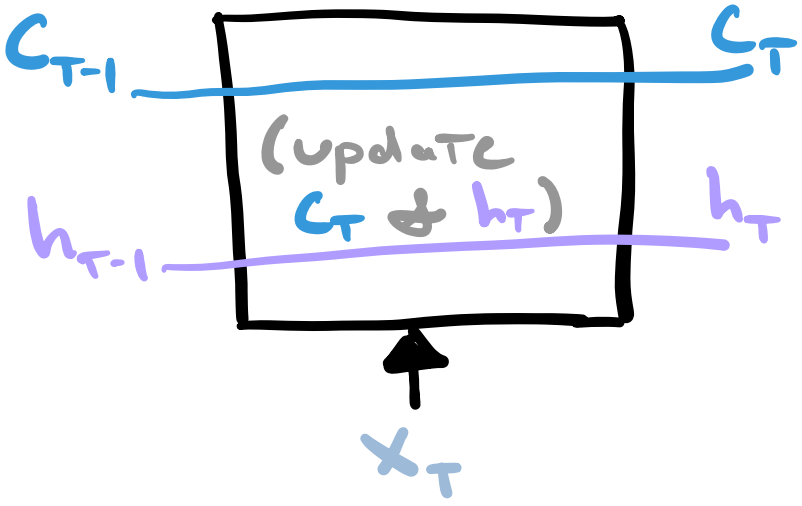
Update

$$h_t \leftarrow \underbrace{z_t \odot h_{t-1}}_{\text{Copy}} + \underbrace{(1 - z_t) \odot \tilde{h}_t}_{\text{Update}}$$

See Notebook

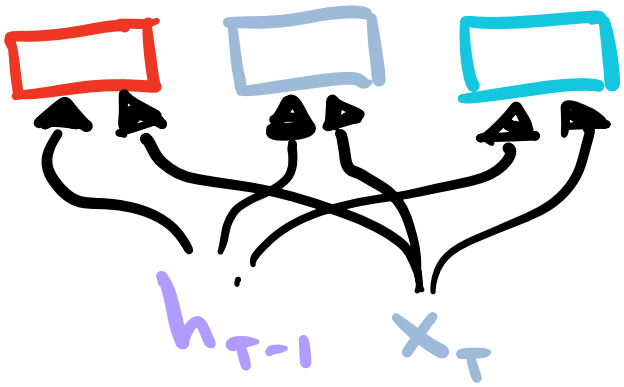
Long Short Term Memory (LSTM)

Like GRUs, use gating for long term dependencies.



Three gates:

forget Input Output



Assume batch size b, d hidden dims.

$$X_t \in \mathbb{R}^{b \times d_{in}} \qquad h_{t-1} \in \mathbb{R}^{b \times d}$$

$$f_t = \sigma(X_t W_f^x + h_{t-1} W_f^h + b_f)$$

$$i_t = \sigma(X_t W_i^x + h_{t-1} W_i^h + b_i)$$

$$o_t = \sigma(X_t W_o^x + h_{t-1} W_o^h + b_o)$$

o_t: $b \times d$; W_o^x: $d_{in} \times d$; W_o^h: $d \times d$; b_o: $1 \times d$

Apply these gates to a Candidate "Memory Cell" $\tilde{C}_t$.

$$\tilde{c}_t = \tanh(X_t W_c^x + h_{t-1} W_c^h + b_c)$$

$\tilde{c}_t$: $b \times d$

Next update the Memory

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

Finally, the Output layer, which

is a modified version of C_T.

$$h_T \leftarrow o_T \circ \tanh(C_T)$$

Info can be stored in C_T w/o affecting h_T if o_T near 0.