Deep Learning Basics

Lecture 7: Recurrent Neural Networks

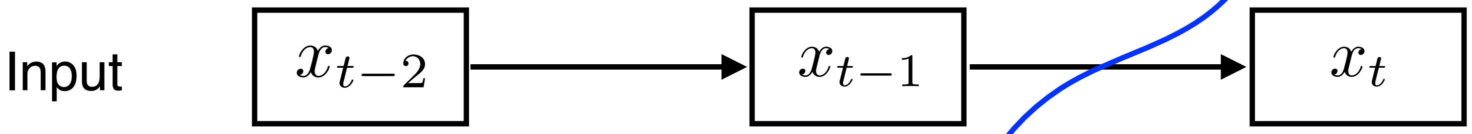
최성준 (고려대학교 인공지능학과)

Sequential Model

Sequential Model

- Naive sequence model

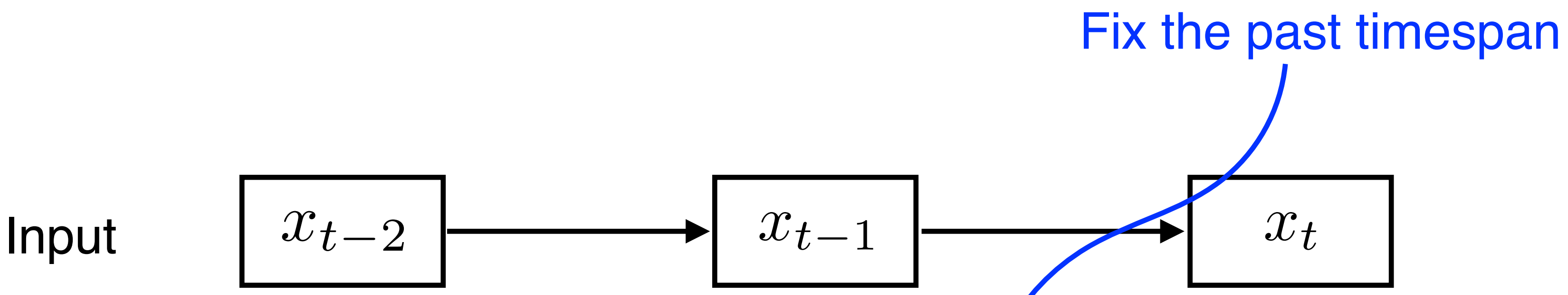
The number of inputs varies.



$$p(x_t | x_{t-1}, x_{t-2}, \ldots)$$

Sequential Model

- Autoregressive model



$$p(x_t | x_{t-1}, \ldots, \boxed{x_{t-\tau}})$$

Sequential Model

- Markov model (first-order autoregressive model)

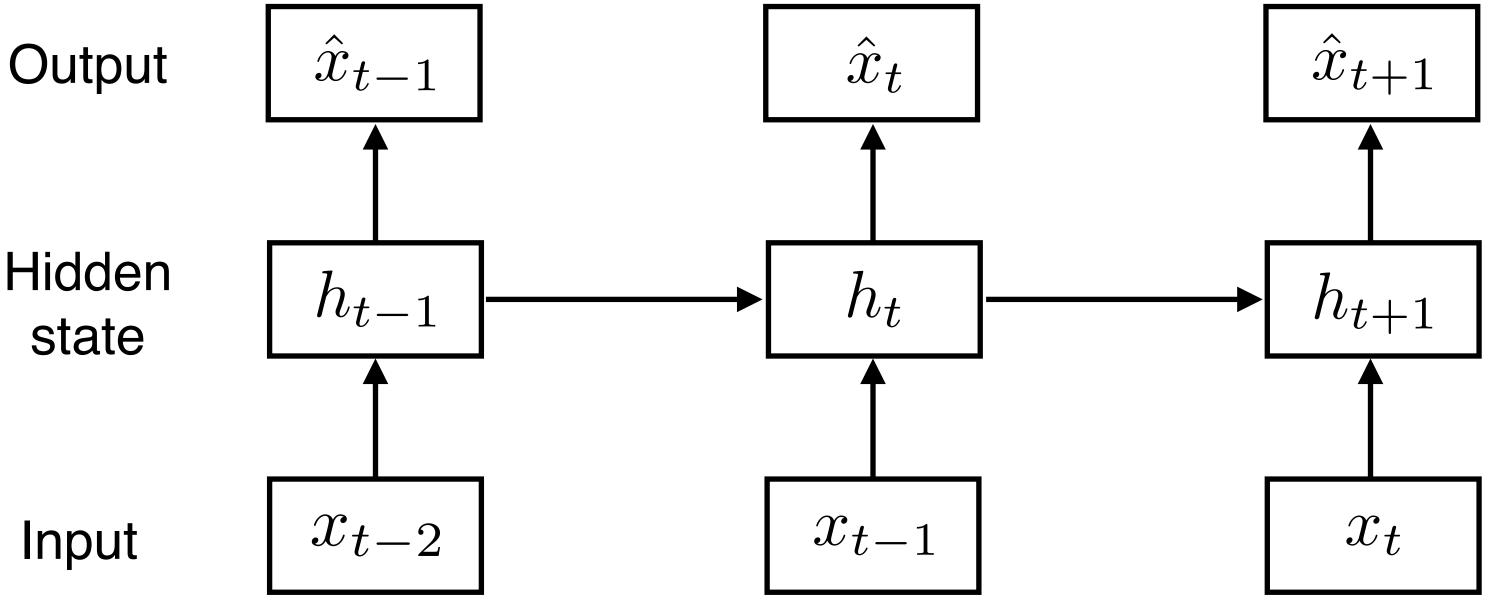
Input $\boxed{x_{t-2}} \rightarrow \boxed{x_{t-1}} \rightarrow \boxed{x_t}$

Easy to express the joint distribution !

$$p(x_1, \ldots, x_T) = p(x_T | x_{T-1}) p(x_{T-1} | x_{T-2}) \cdots p(x_2 | x_1) p(x_1) = \boxed{\prod_{t=1}^{T} p(x_t | x_{t-1})}$$

Sequential Model

- Latent autoregressive model

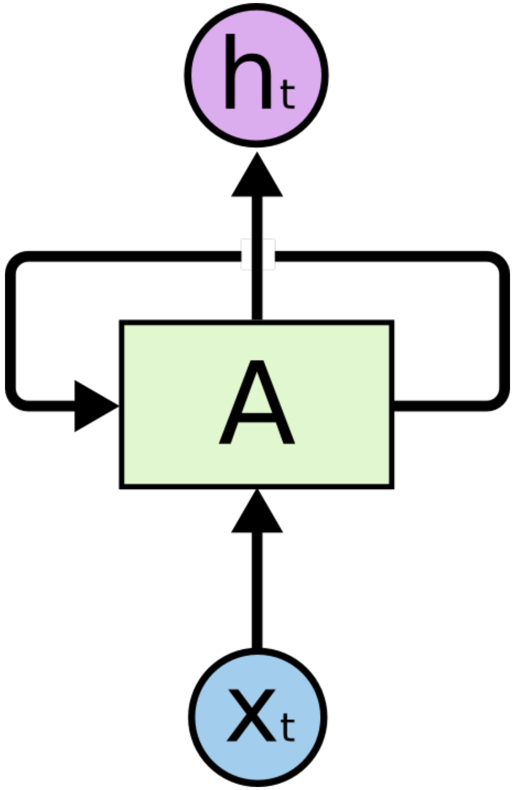


$$\hat{x} = p(x_t | h_t)$$
$$h_t = g(h_{t-1}, x_{t-1})$$

summary of the past

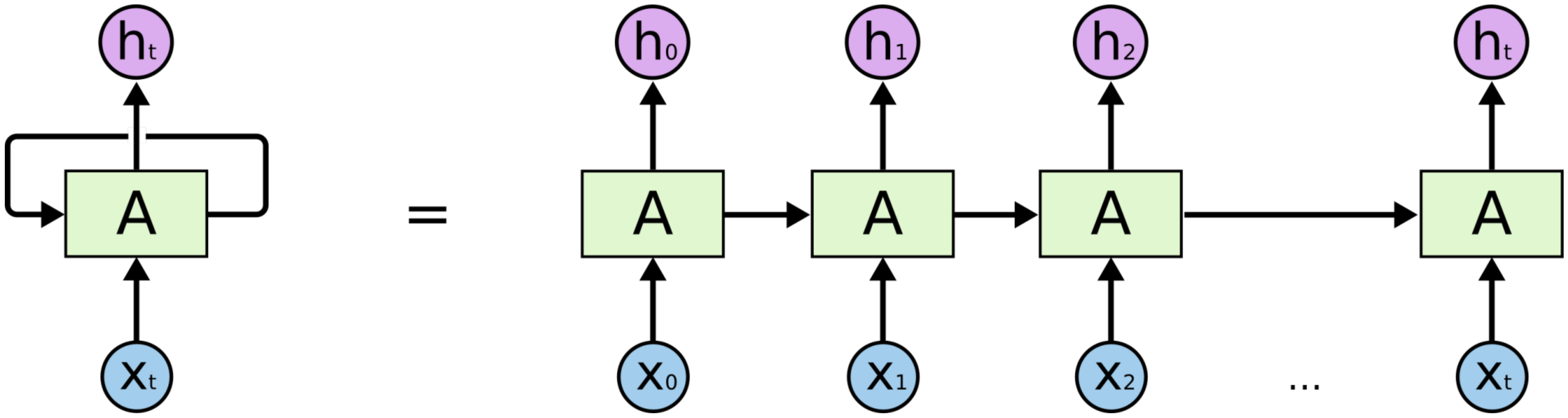
Recurrent Neural Network

Recurrent Neural Network



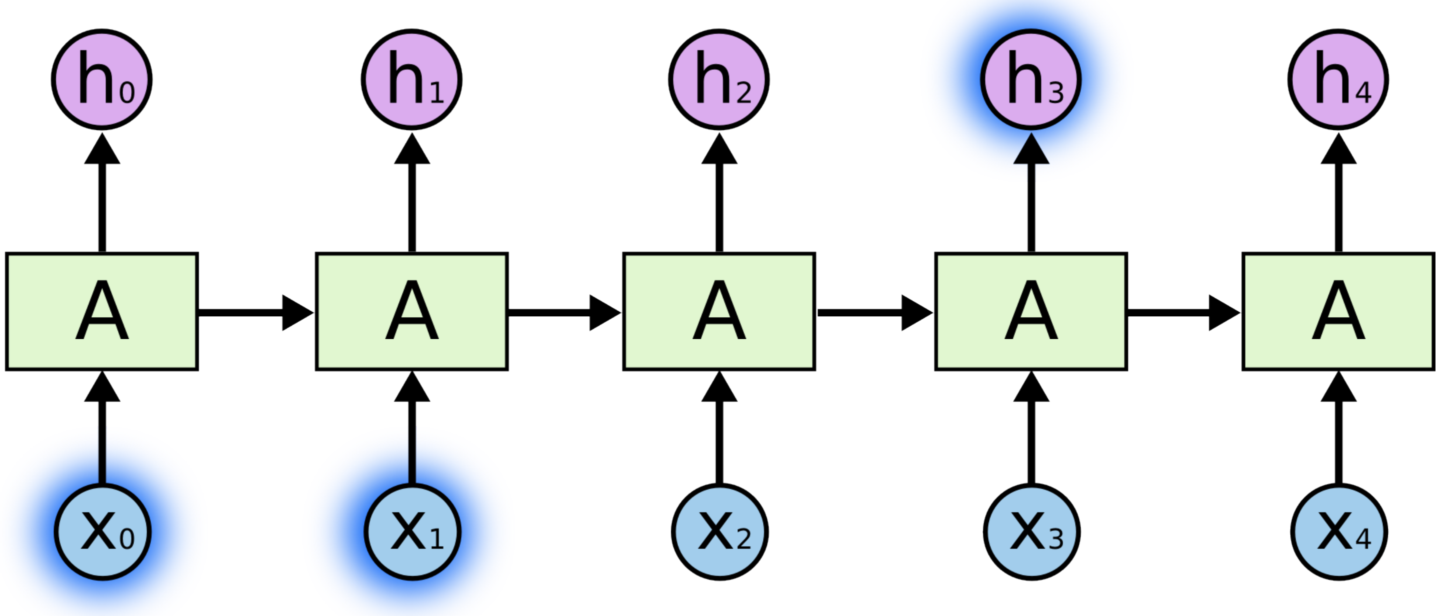
C. Olah

Recurrent Neural Network



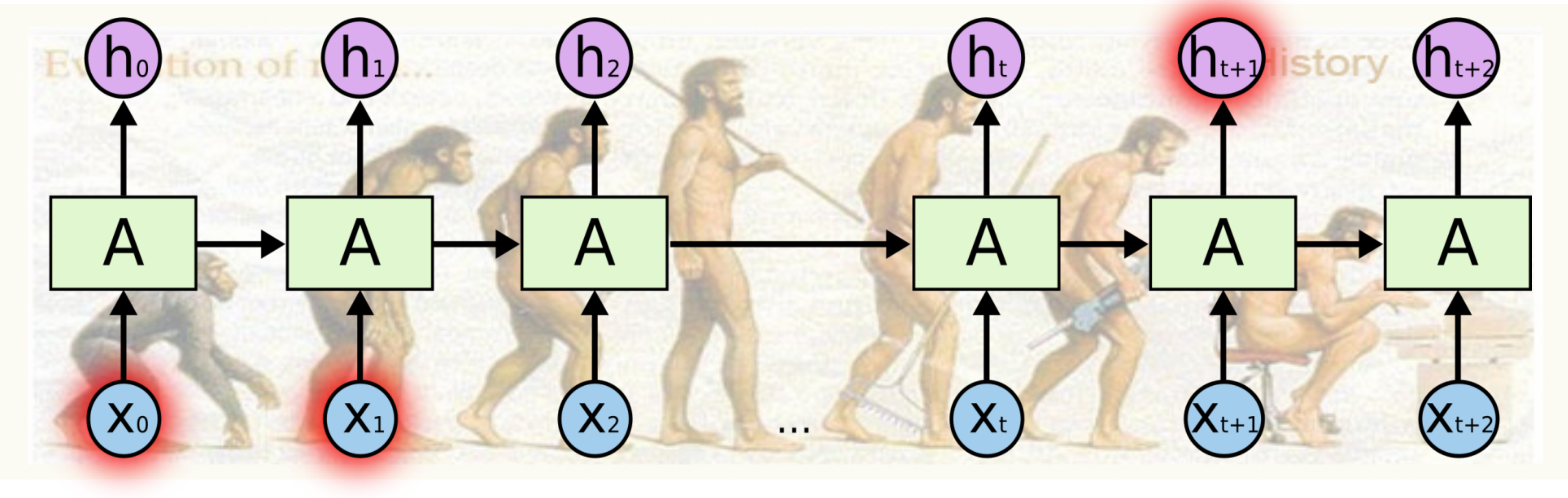
Recurrent Neural Network

- Short-term dependencies

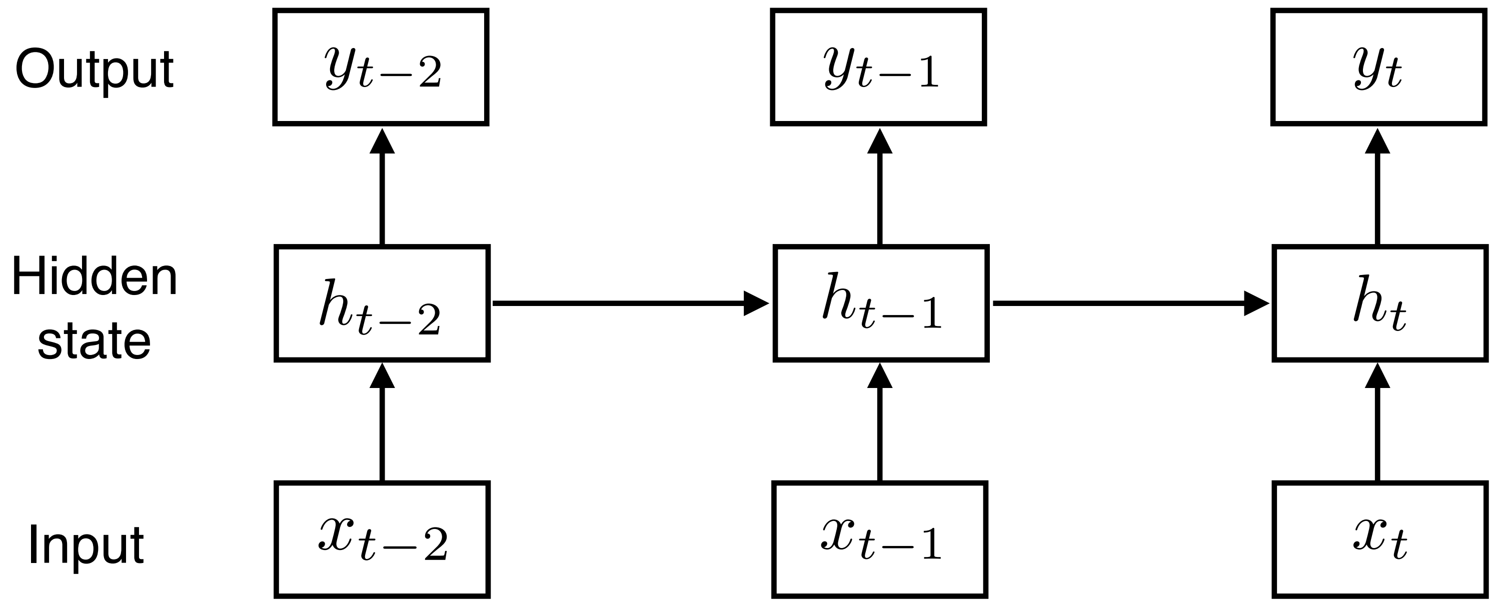


Recurrent Neural Network

- **Long**-term dependencies



Recurrent Neural Network

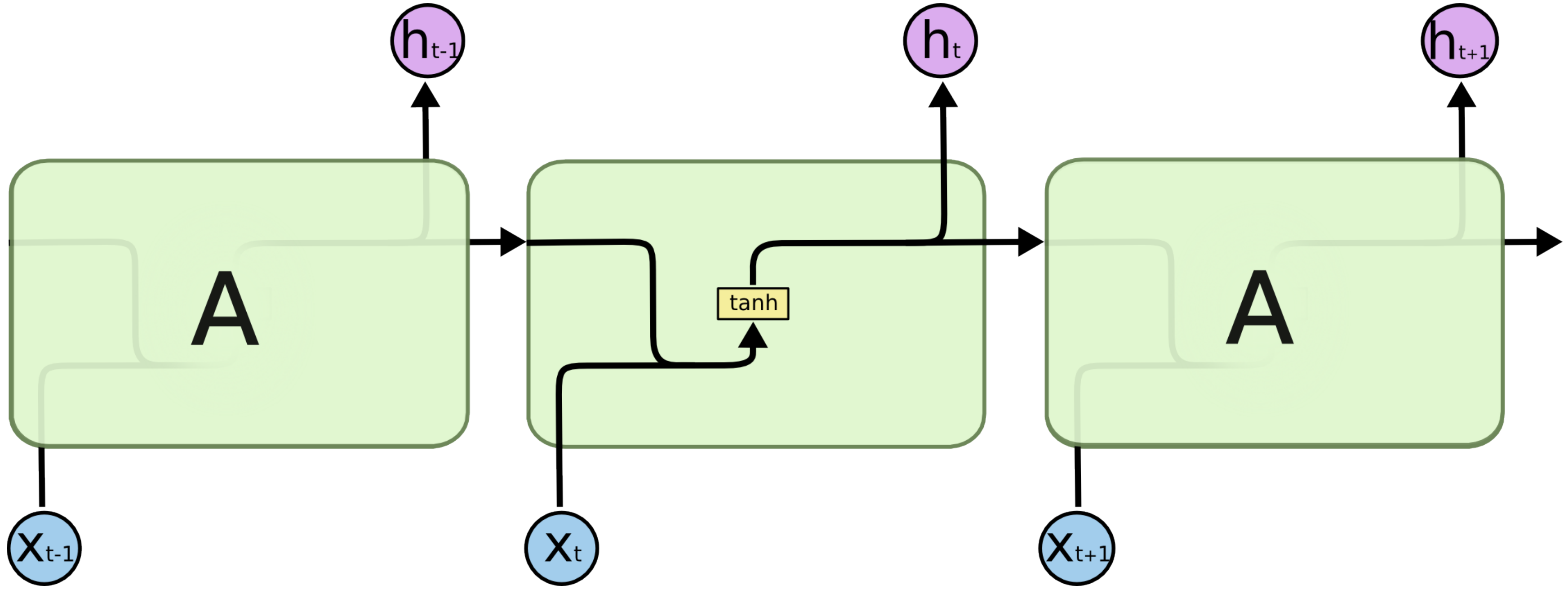


$$h_1 = \phi(W^T h_0 + U^T x_1)$$
$$h_2 = \phi(W^T \phi(W^T h_0 + U^T x_1) + U^T x_2)$$
$$h_3 = \phi(W^T \phi(W^T \phi(W^T \phi(W^T h_0 + U^T x_1) + U^T x_2) + U^T x_1) + U^T x_3)$$
$$h_4 = \phi(W^T \phi(W^T \phi(W^T \phi(W^T \phi(W^T h_0 + U^T x_1) + U^T x_2) + U^T x_1) + U^T x_3) + U^T x_4)$$

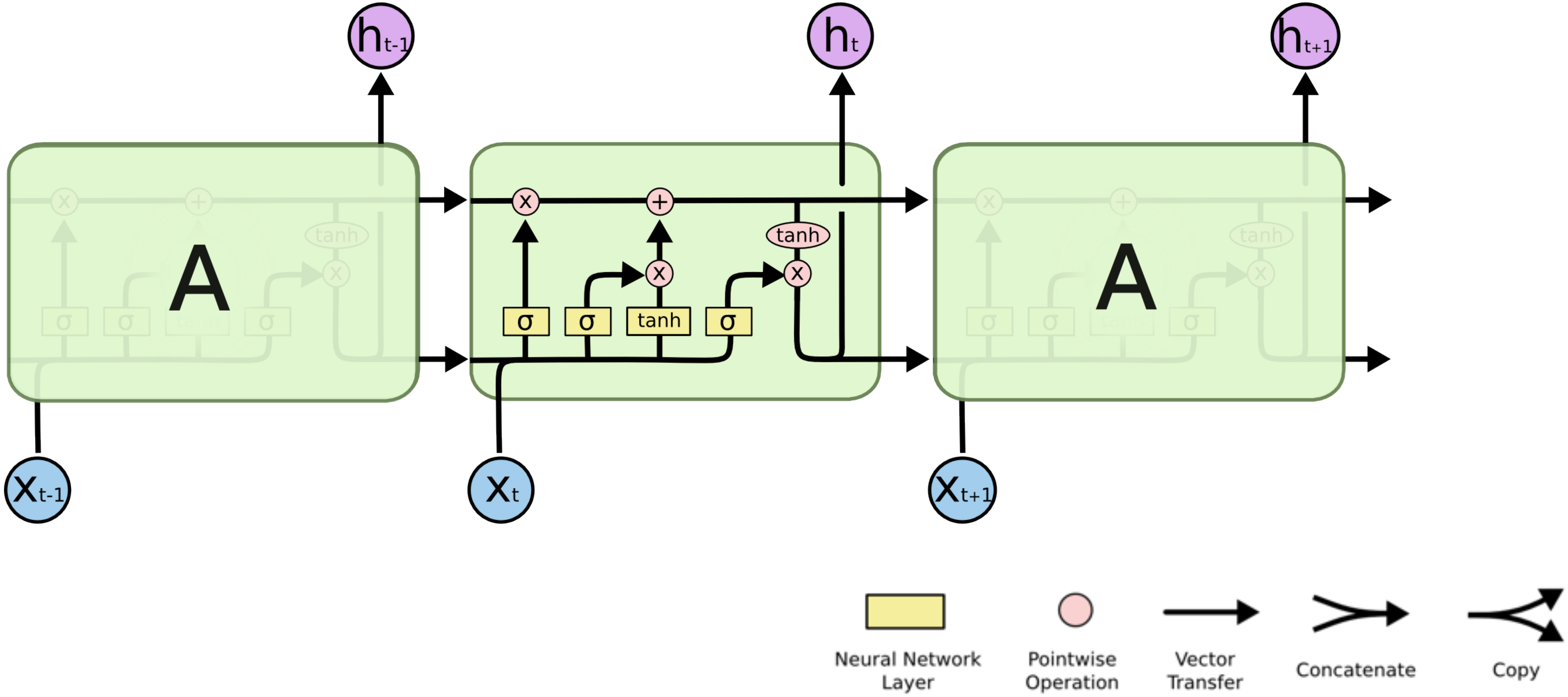
Vanishing / exploding gradient

Long Short Term Memory

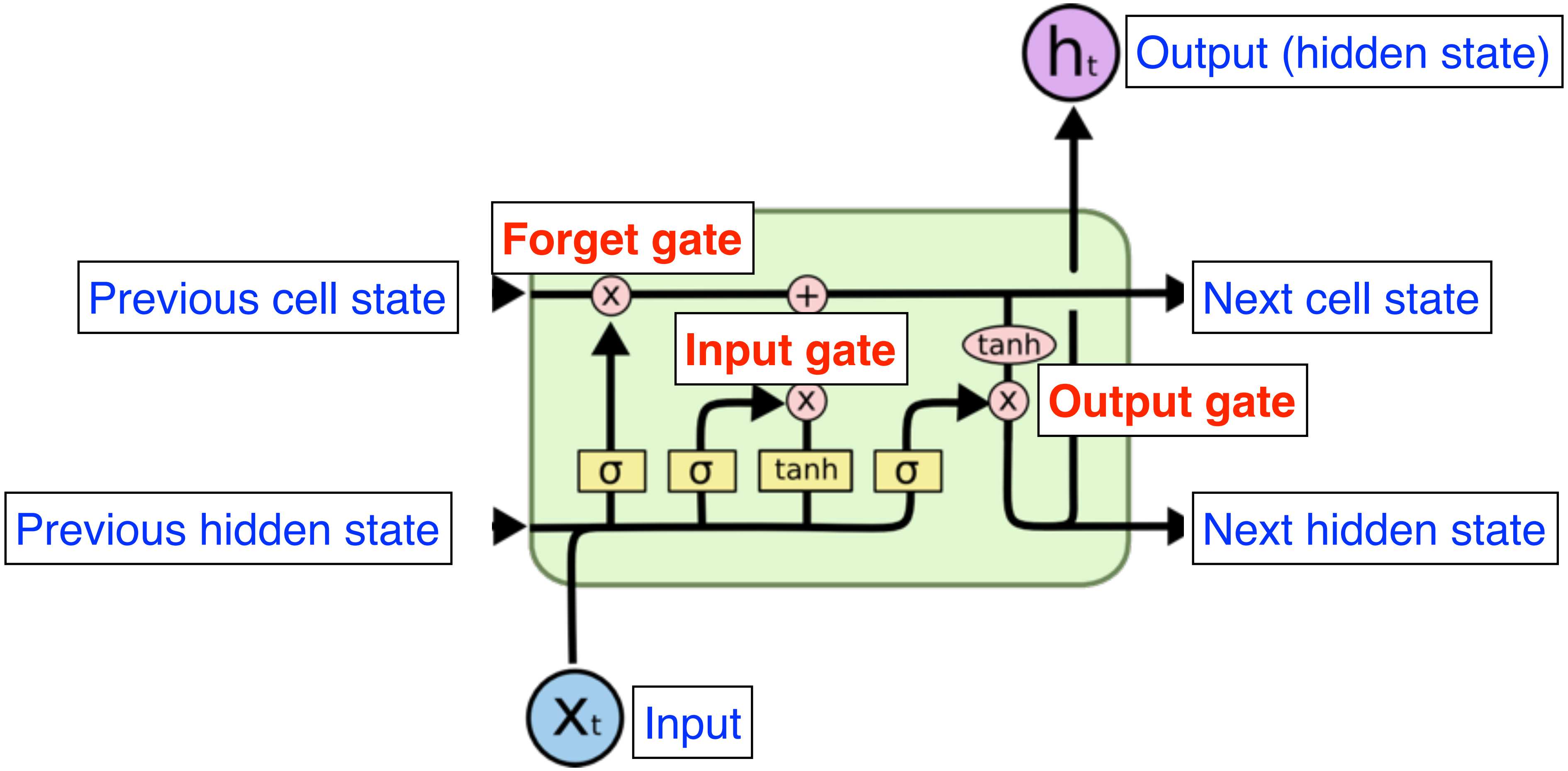
Recurrent Neural Network



Long Short Term Memory

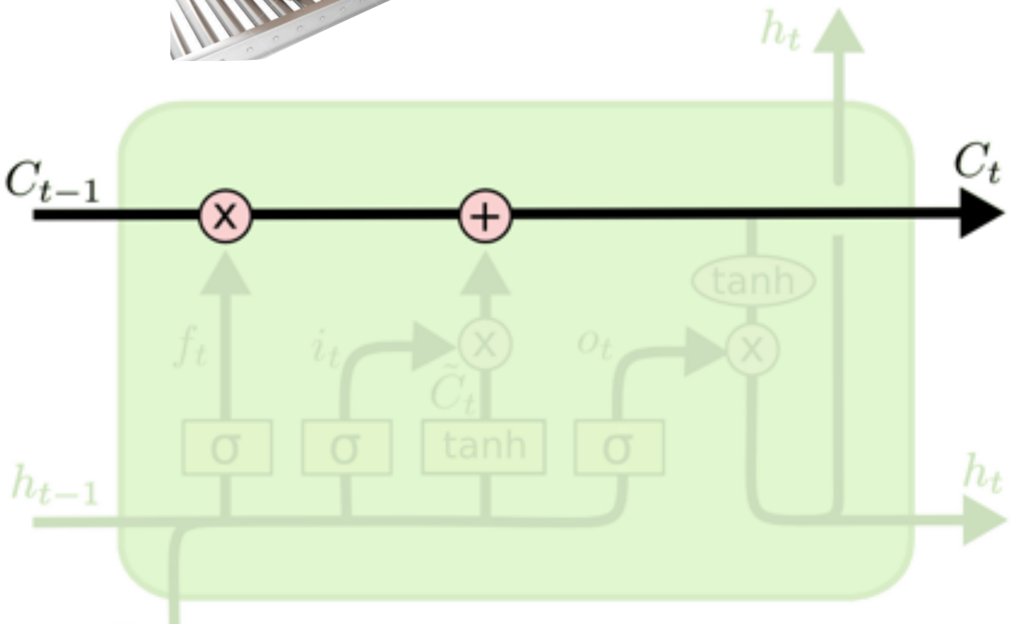


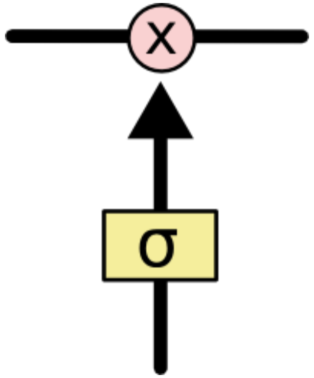
Long Short Term Memory



Long Short Term Memory

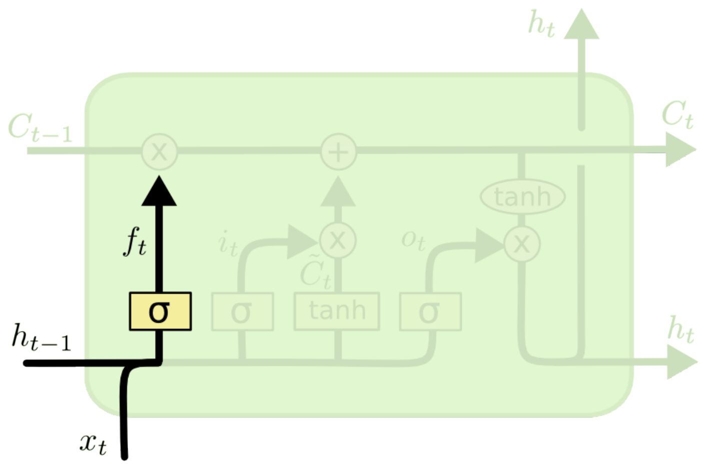
- Core idea





Long Short Term Memory

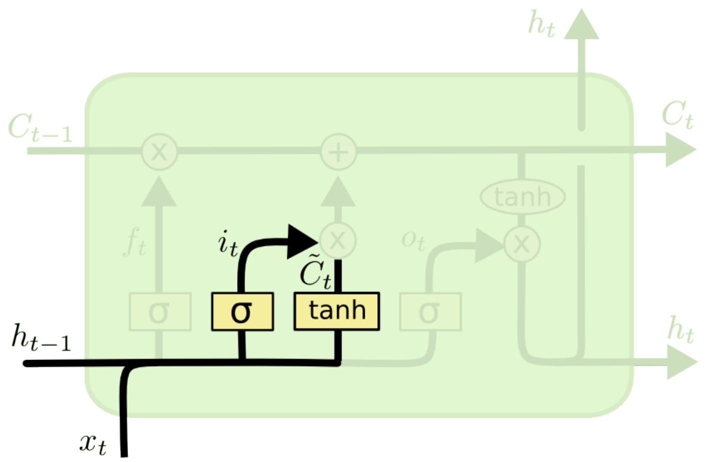
Forget Gate



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

Decide which information to **throw** away

Input Gate



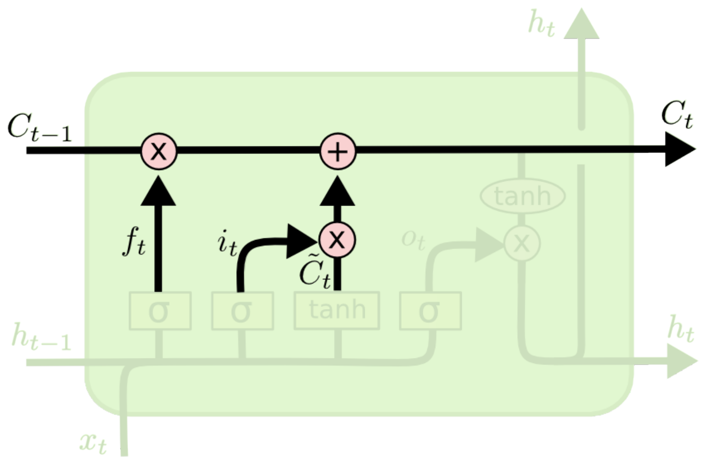
$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \ + \ b_i\right)$$

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

Decide which information to **store** in the cell state

Long Short Term Memory

Update cell

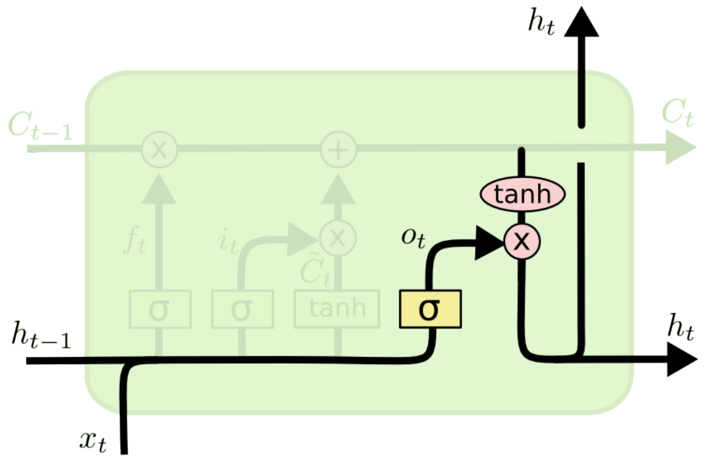


$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \; + \; b_i\right)$$

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

Update the cell state.

Output Gate



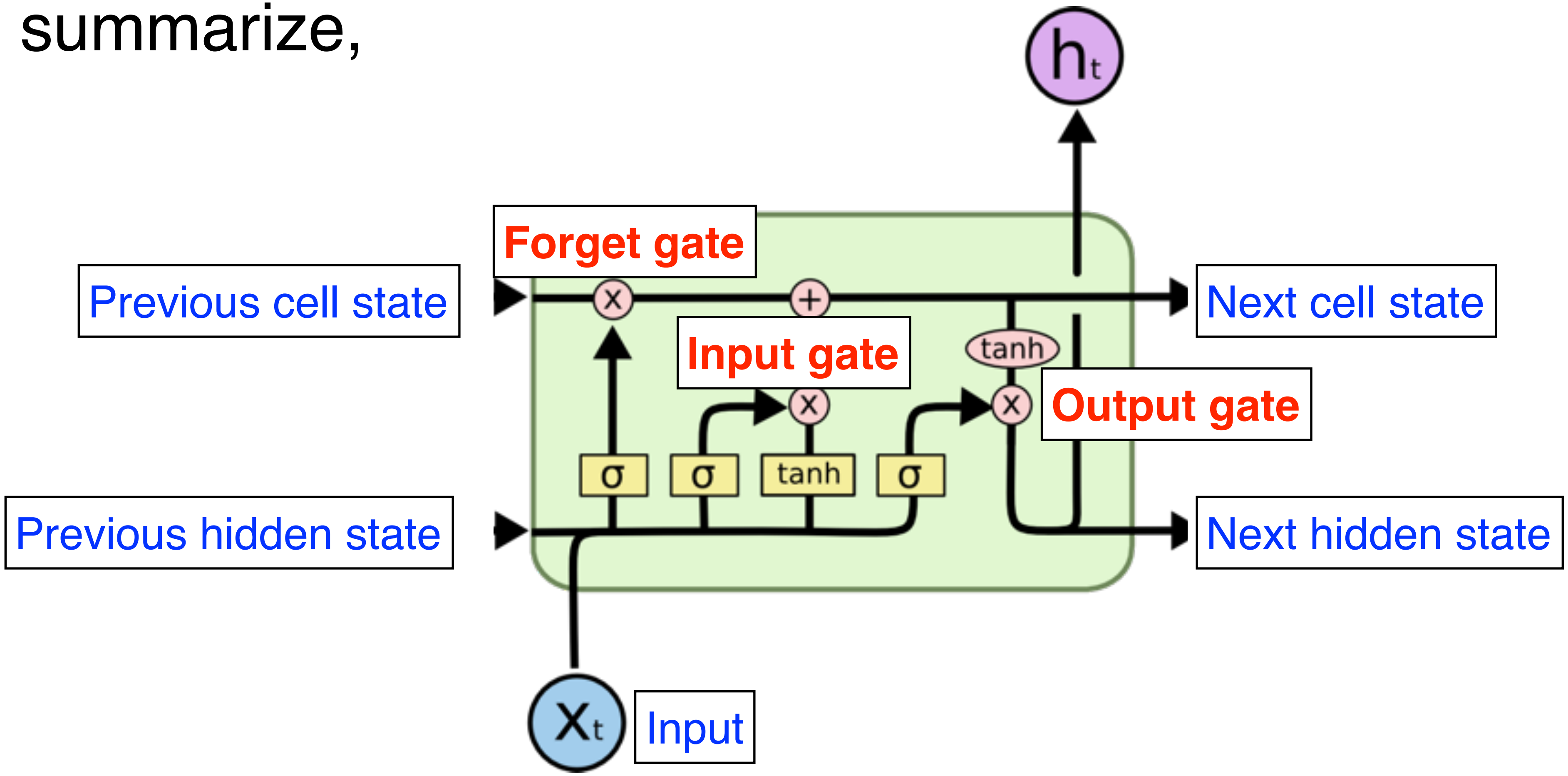
$$o_t = \sigma\left(W_o \; [h_{t-1}, x_t] \; + \; b_o\right)$$

$$h_t = o_t * \tanh\left(C_t\right)$$

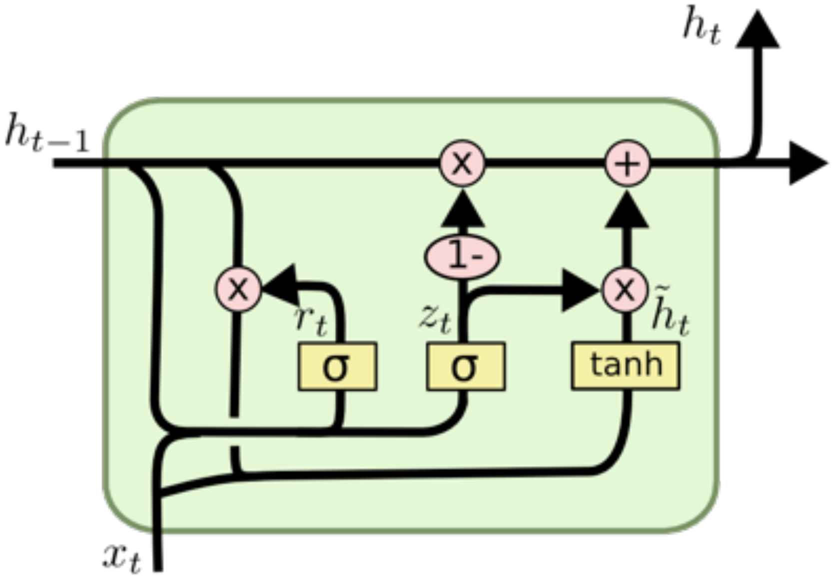
Make output using the updated cell state.

Long Short Term Memory

- To summarize,



Gated Recurrent Unit



$$z_t = \sigma\left(W_z \cdot [h_{t-1}, x_t]\right)$$

$$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t]\right)$$

$$\tilde{h}_t = \tanh\left(W \cdot [r_t * h_{t-1}, x_t]\right)$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

- Simpler architecture with two gates (**reset gate** and **update gate**).
- No **cell state**, just **hidden state**.

Thank you for listening