Deep Learning Basics

Lecture7 : Recurrent Neural Networks

최성준 (고려대학교 인공지능학과)

boostcamp ai tech

Sequential Model

Sequential Model

: video, audio, motion, ...

- Naive sequence model

(기본 sequence model)

The number of inputs varies.

$$\text{Input} \quad \boxed{x_{t-2}} \longrightarrow \boxed{x_{t-1}} \longrightarrow \boxed{x_t}$$

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

→ 이전 데이터를 가지고 다음 데이터를 예측할 때, 시간이 지날수록 고려해야 할 과거의 변수들이 계속 늘어남.

Sequential Model

- Autoregressive model

Fix the the past timespan

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

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

=> 고정된 시간 만큼만 과거의 데이터를 봄.

Sequential Model

- Markov model (first-order autoregressive model)

⇒ 현재는 바로 전 과거에만 의존한다.
(대과거 데이터는 다 제외하므로 많은 정보를 버리게 됨. 그러나 간단함)

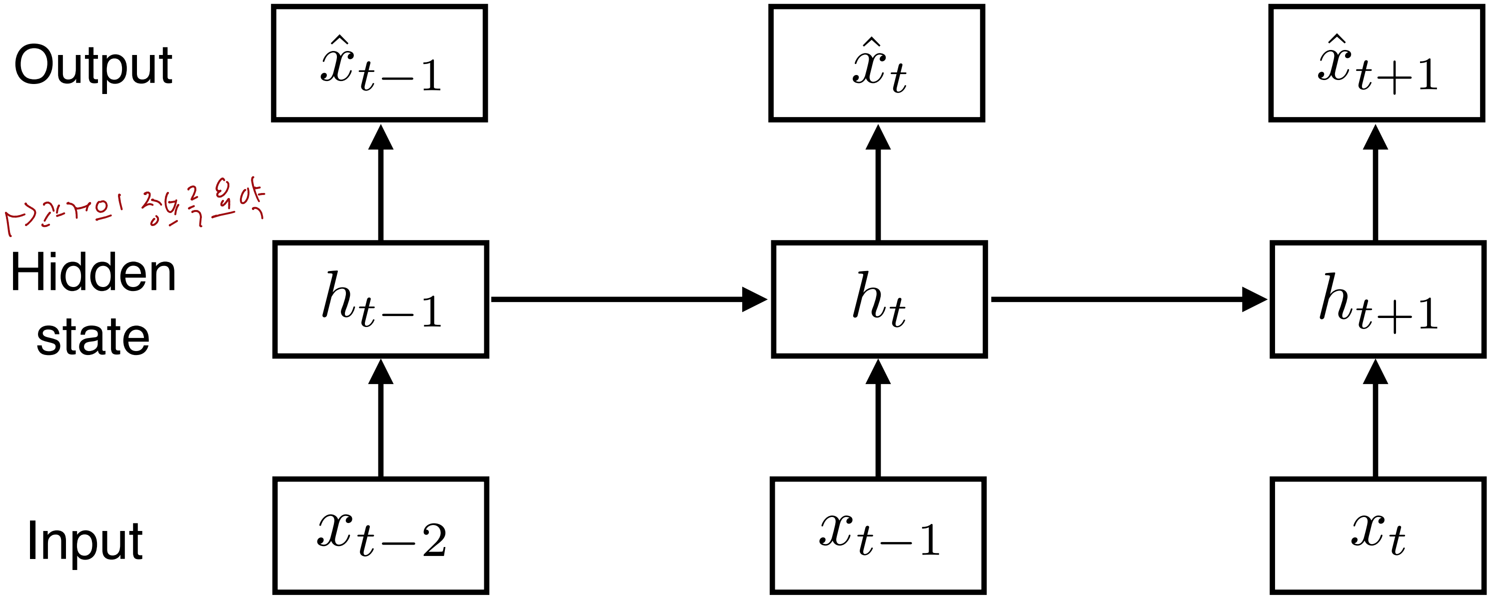
Easy to express the joint distribution !

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

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

Sequential Model

- Latent autoregressive model



$$\hat{x} = p(x_t | h_t)$$

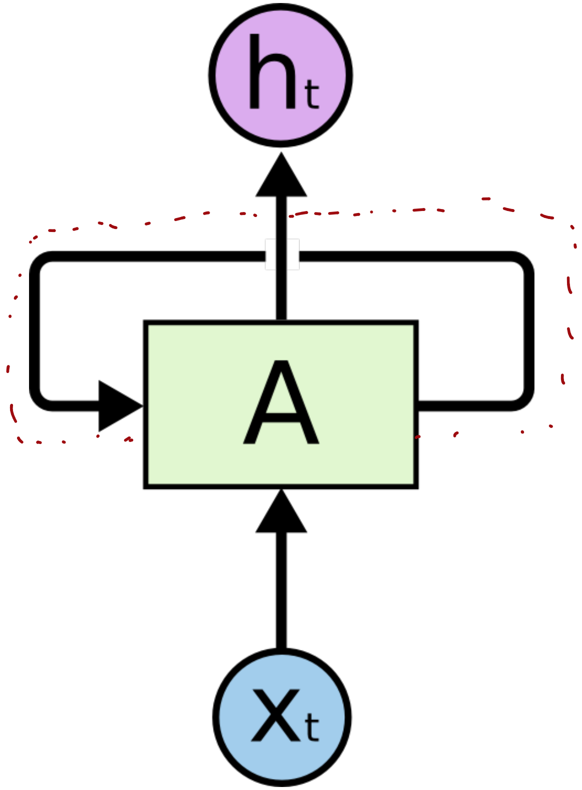
$$h_t = g(h_{t-1}, x_{t-1})$$

summary of the past

요약된 과거의 정보를 고려.

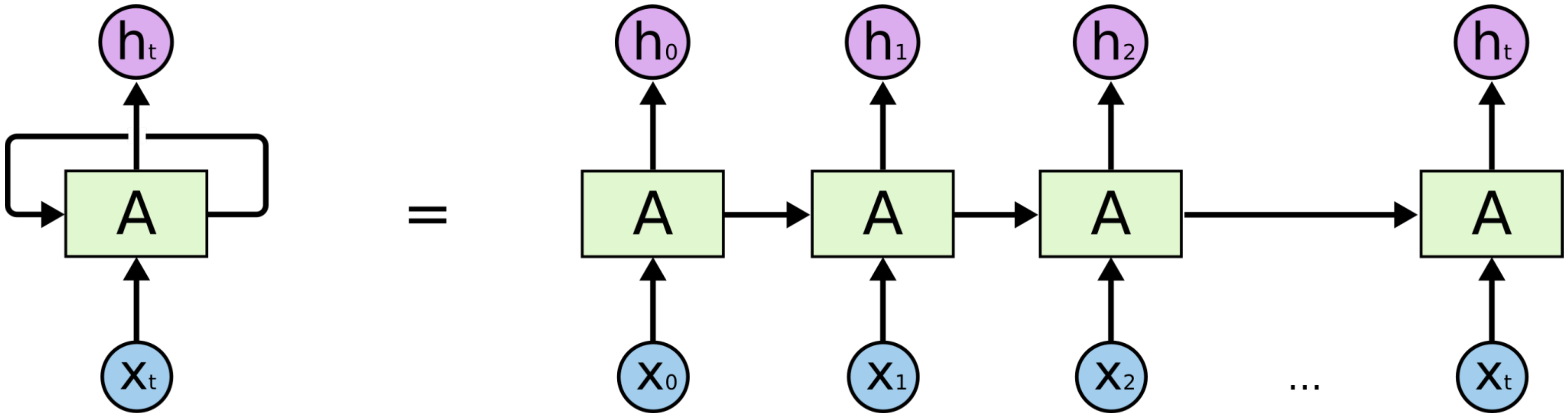
Recurrent Neural Network

Recurrent Neural Network



C. Olah

Recurrent Neural Network

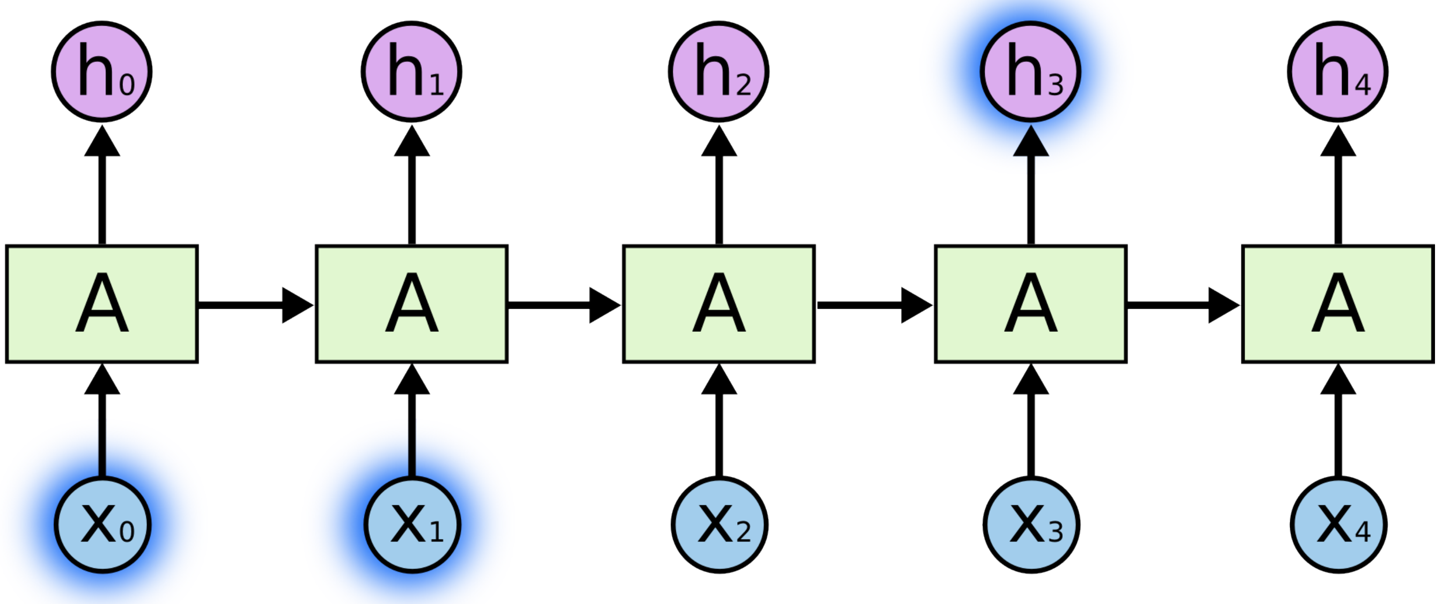


⇒ RNN을 풀어서 보면.
입력과 hidden layer가 많은 MLP모형이 됨.

Recurrent Neural Network

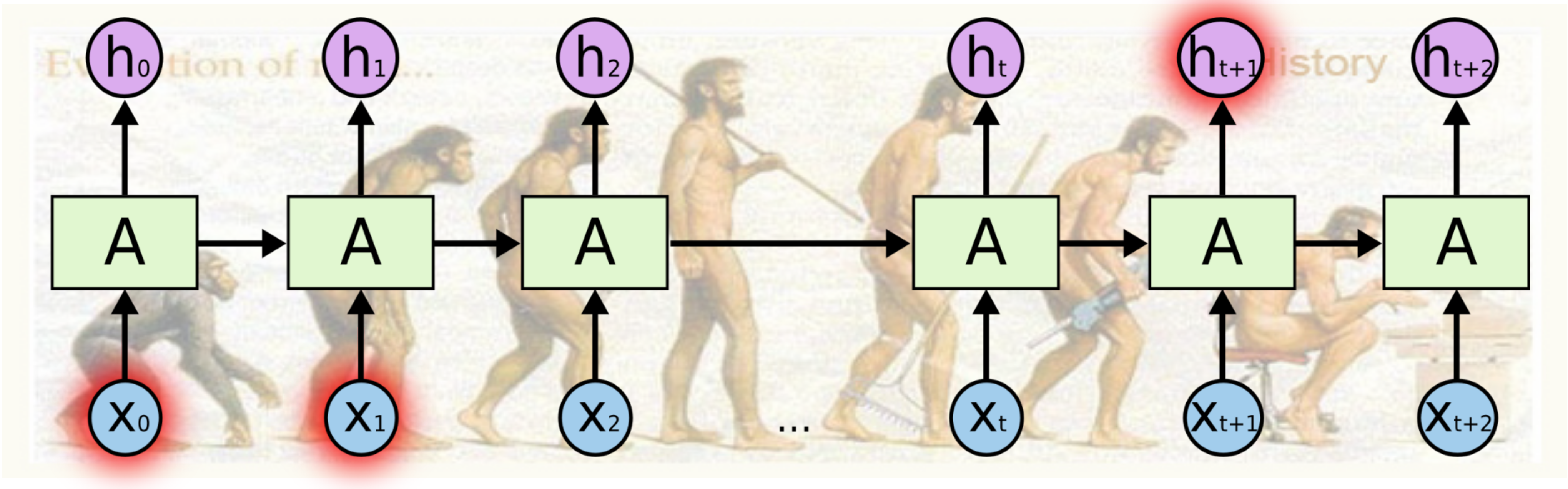
- Short-term dependencies

과거의 정보가 취합되어 현재로 전달되는데, 이걸 다 보니 먼 과거의 정보는 살아남기 힘들게 됨.
(짧은 기간 정보 의존)

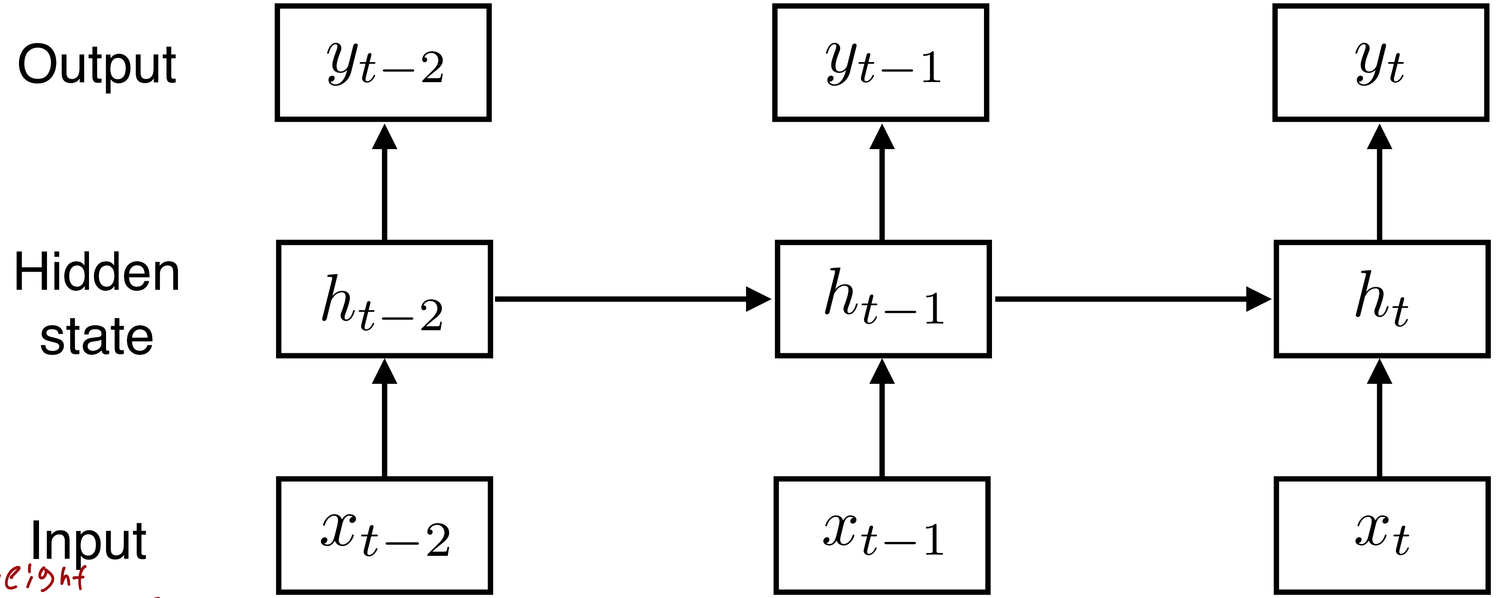


Recurrent Neural Network

- **Long**-term dependencies



Recurrent Neural Network



$$h_1 = \phi(\overbrace{W^T h_0 + U^T}^{\text{weight}} x_1)$$

$$h_2 = \phi(W^T \phi(W^T h_0 + U^T x_1) + U^T x_2)$$

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

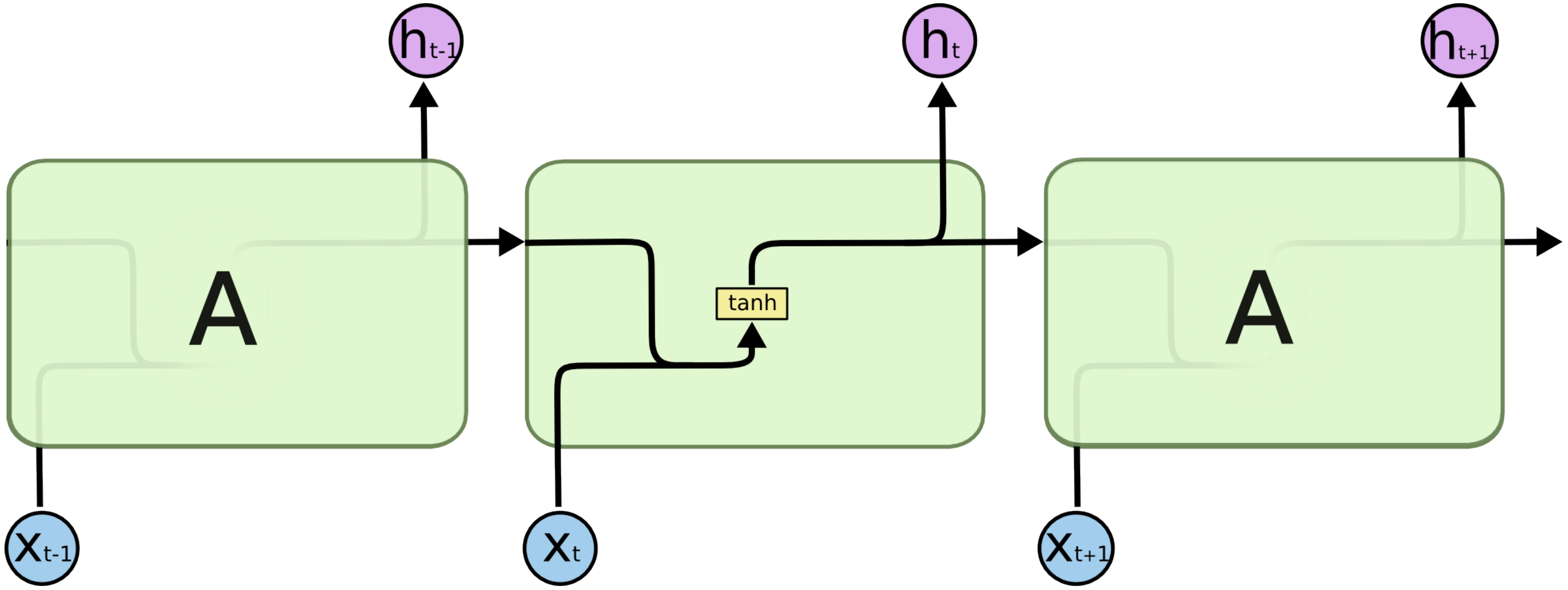
$$h_4 = \phi(W^T \phi(W^T \phi(W^T \phi(W^T h_0 + U^T x_1) + U^T x_2) + U^T x_3) + U^T x_4)$$

Vanishing / exploding gradient : 값이 의미없어지거나 너무 커지게 됨.

Long Short Term Memory

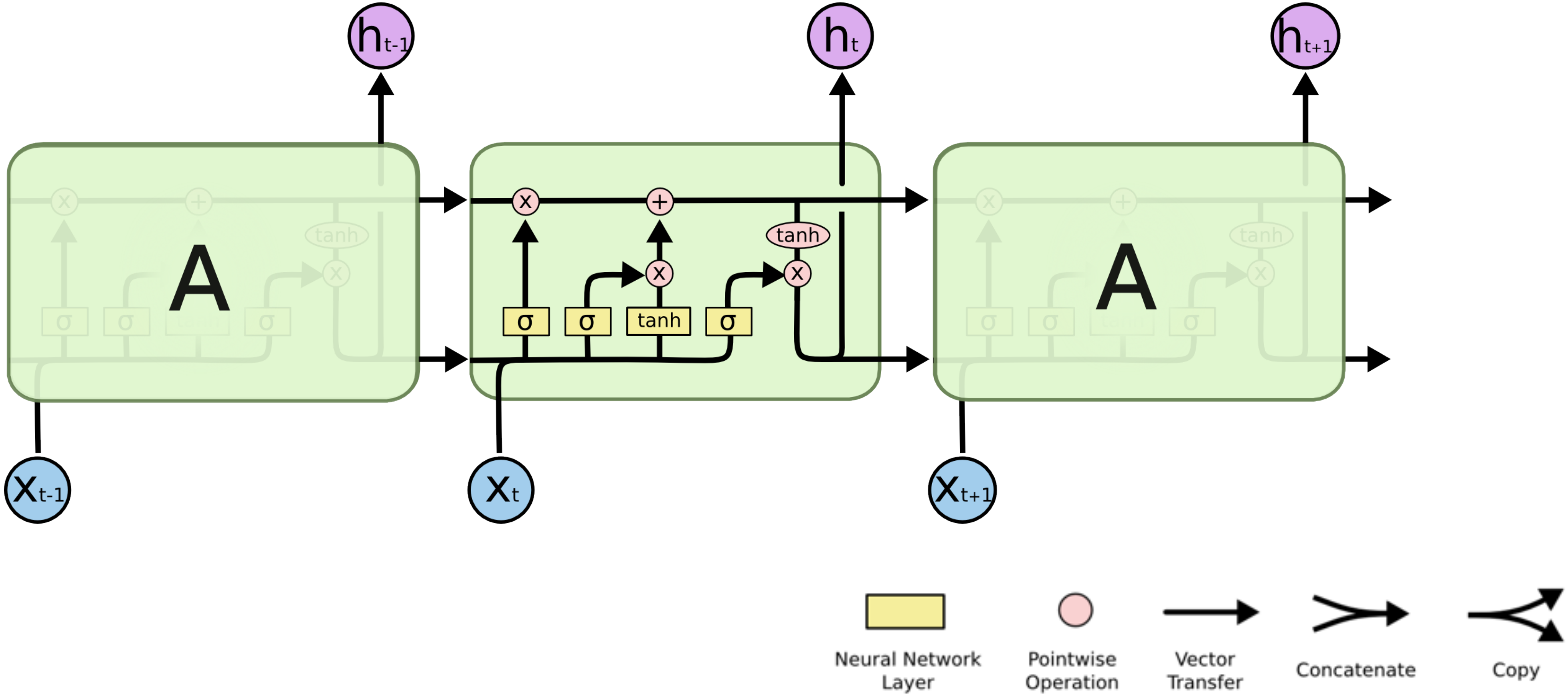
Recurrent Neural Network

가장 기본적인 RNN



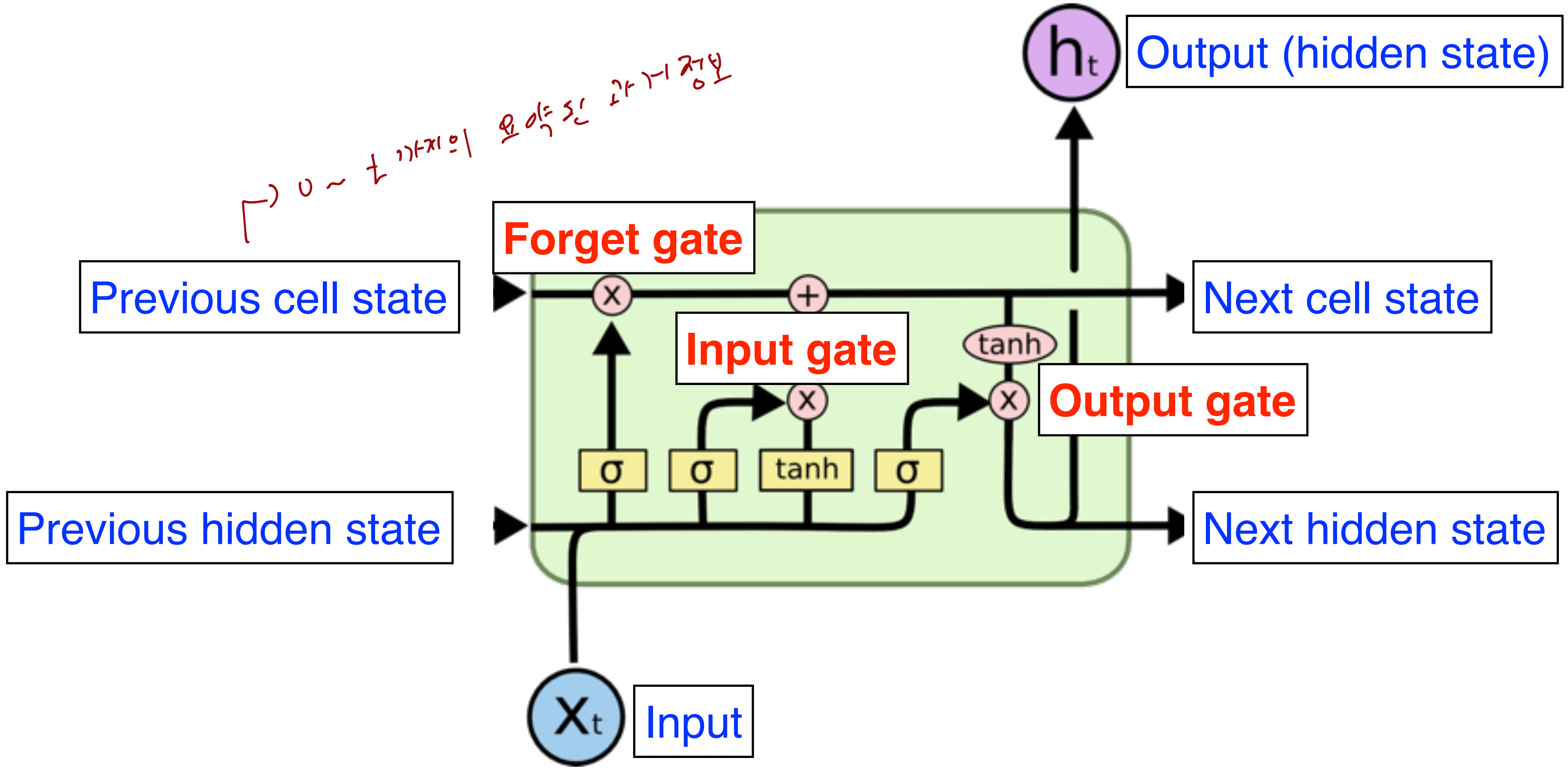
Long Short Term Memory

LSTM



Long Short Term Memory

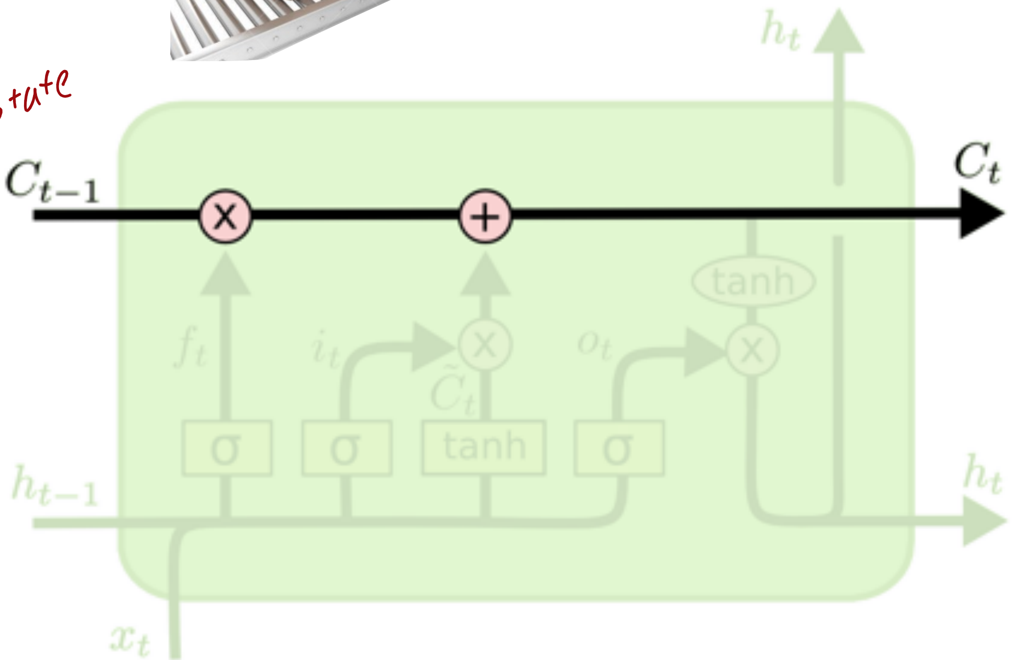
0 ~ t 까지의 요약된 과거 정보

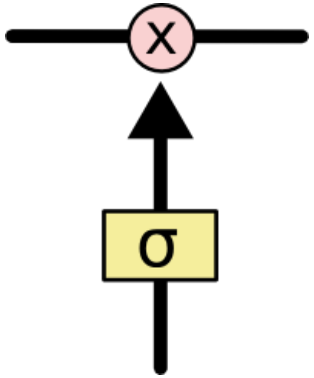


Long Short Term Memory

- Core idea

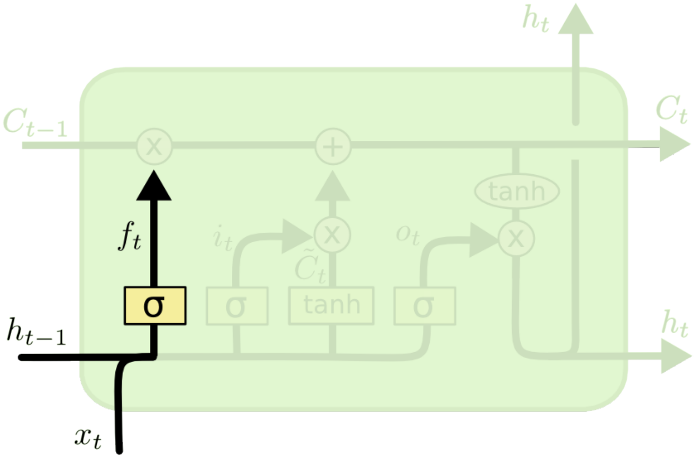
요약된 cell state





Long Short Term Memory

Forget Gate



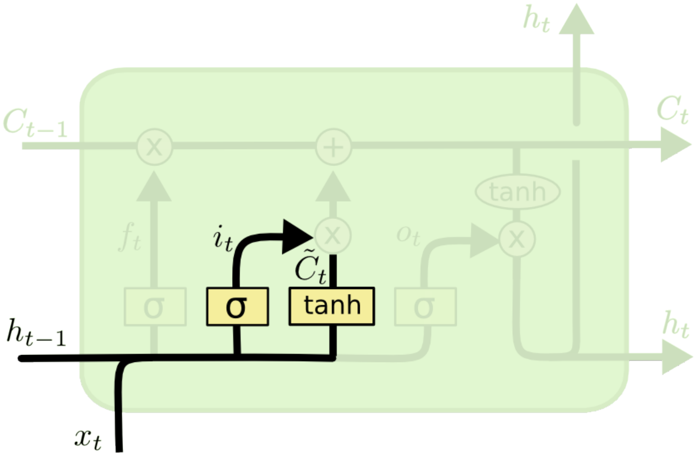
0~1

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

: 입력값 W_f와 가중치 W_f를 통해 cell state에서 나올 정보 중 무엇을 버릴지 선택.

Decide which information to **throw** away

Input Gate



어떤 정보를 추가할지.

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

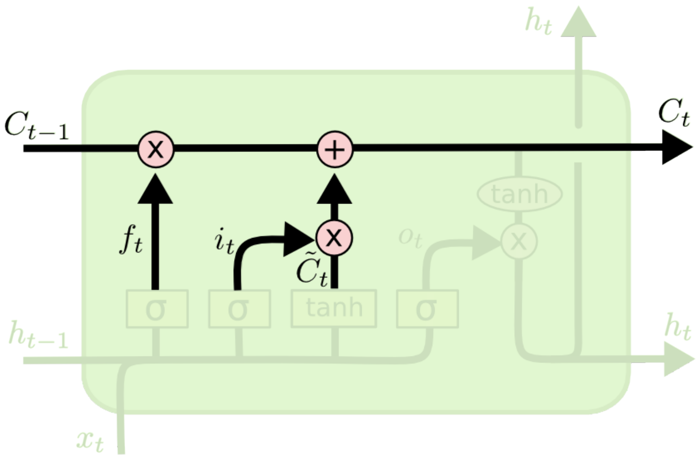
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \ + \ b_C)$$

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

$\tilde{C}_t$: 현재정보와 이전 출력값을 가지고 만들어지는 cell state 예비군.
이전에 취합했던 cell state와 현재정보, 그리고 이전 output으로 만들어지는 $\tilde{C}_t$(예비군)을 잘 섞어서 cell state에 올리게 됨.

Long Short Term Memory

Update cell

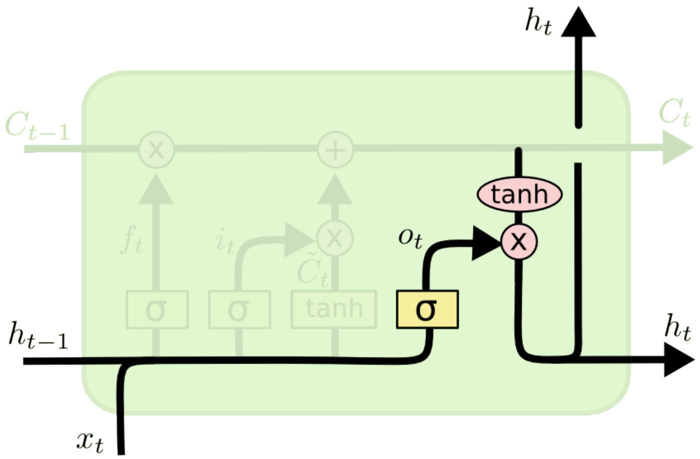


$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \ + \ b_i\right)$$

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

Update the cell state.: 버릴 건 버리고, 쓸 건 쓰고.

Output Gate



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

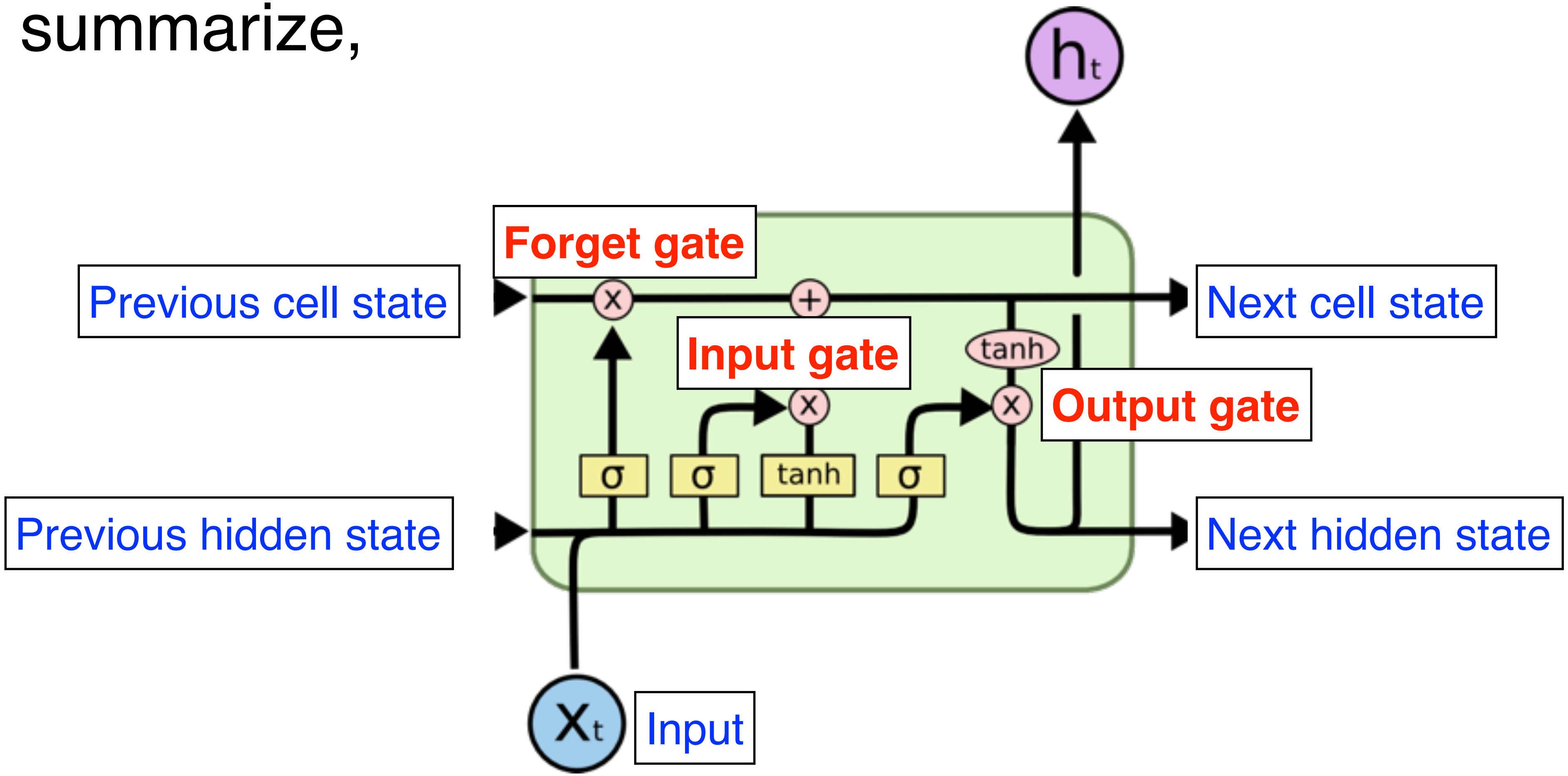
$$h_t = o_t * \tanh\left(C_t\right)$$

Make output using the updated cell state.

: 업데이트된 cell state 에서 어떤 값을 내보낼지 정하는 output Gate를 만들고, output Gate 만큼 곱해서 ht로 현재 output이 나오게 됨

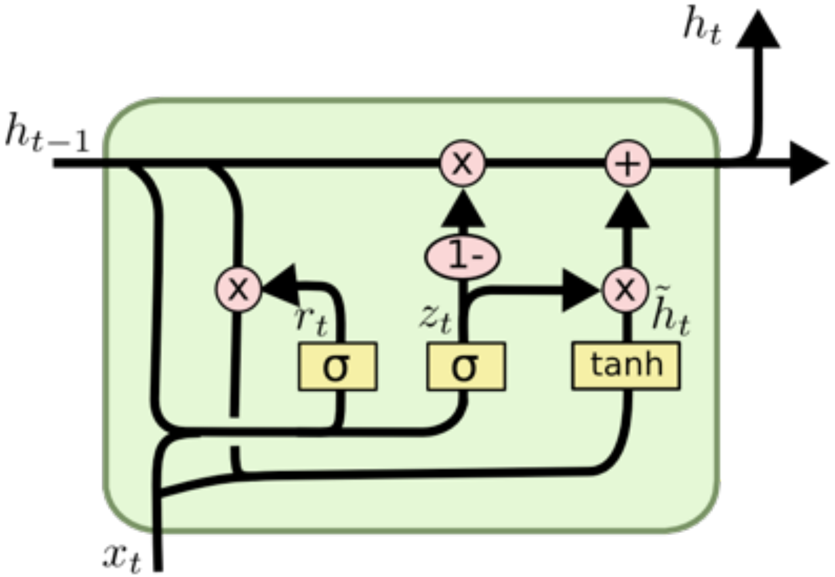
Long Short Term Memory

- To summarize,



Gated Recurrent Unit GRU

= 간단하므로 (parameter 가 적음)
Generalization 이 올라갈 수 있음



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

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

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

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

≈ forget gate

≈ input ⊕ output

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

Thank you for listening