

삼성전기 AI전문가 양성과정 - 프로젝트 실습 (비영상)

자연어처리를 위한 RNN (Recurrent Neural Network)

현청천

2022.02.28

What is RNN

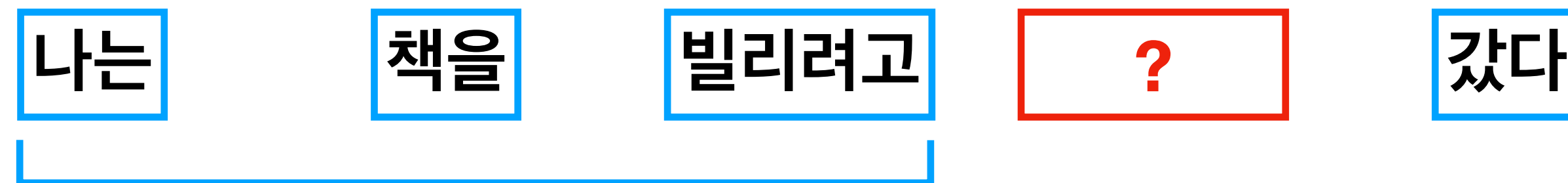
- 시계열(Sequential Data) 데이터를 처리하기 위한 Neural Network
 - 음성, 언어, 주가, 센서 데이터 등



What is RNN

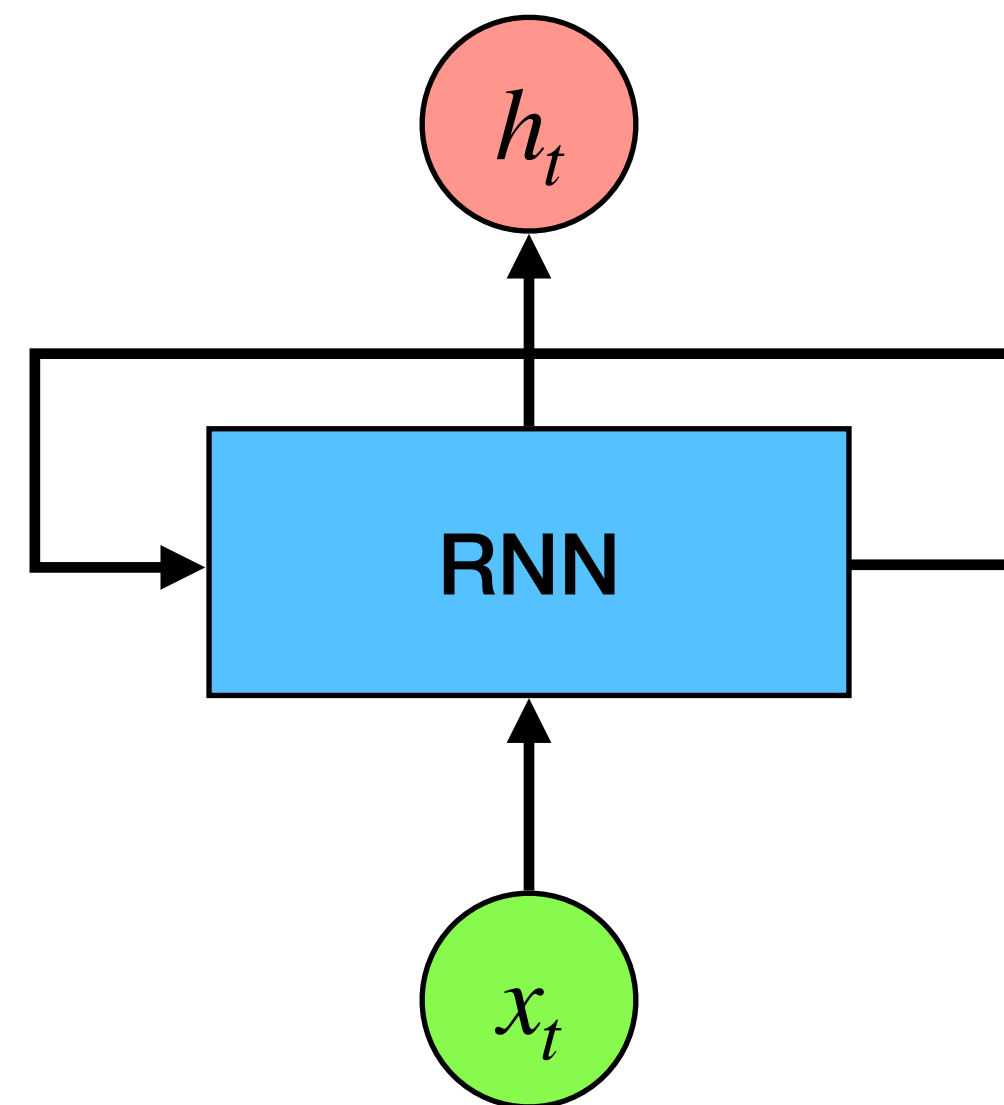
- 시계열(Sequential Data) 데이터를 처리하기 위한 Neural Network
 - 음성, 언어, 주가, 센서 데이터 등
- 문장에서 이전에 나온 단어를 보가 다음단어를 예측

나는 책을 빌리려고 ? 갔다



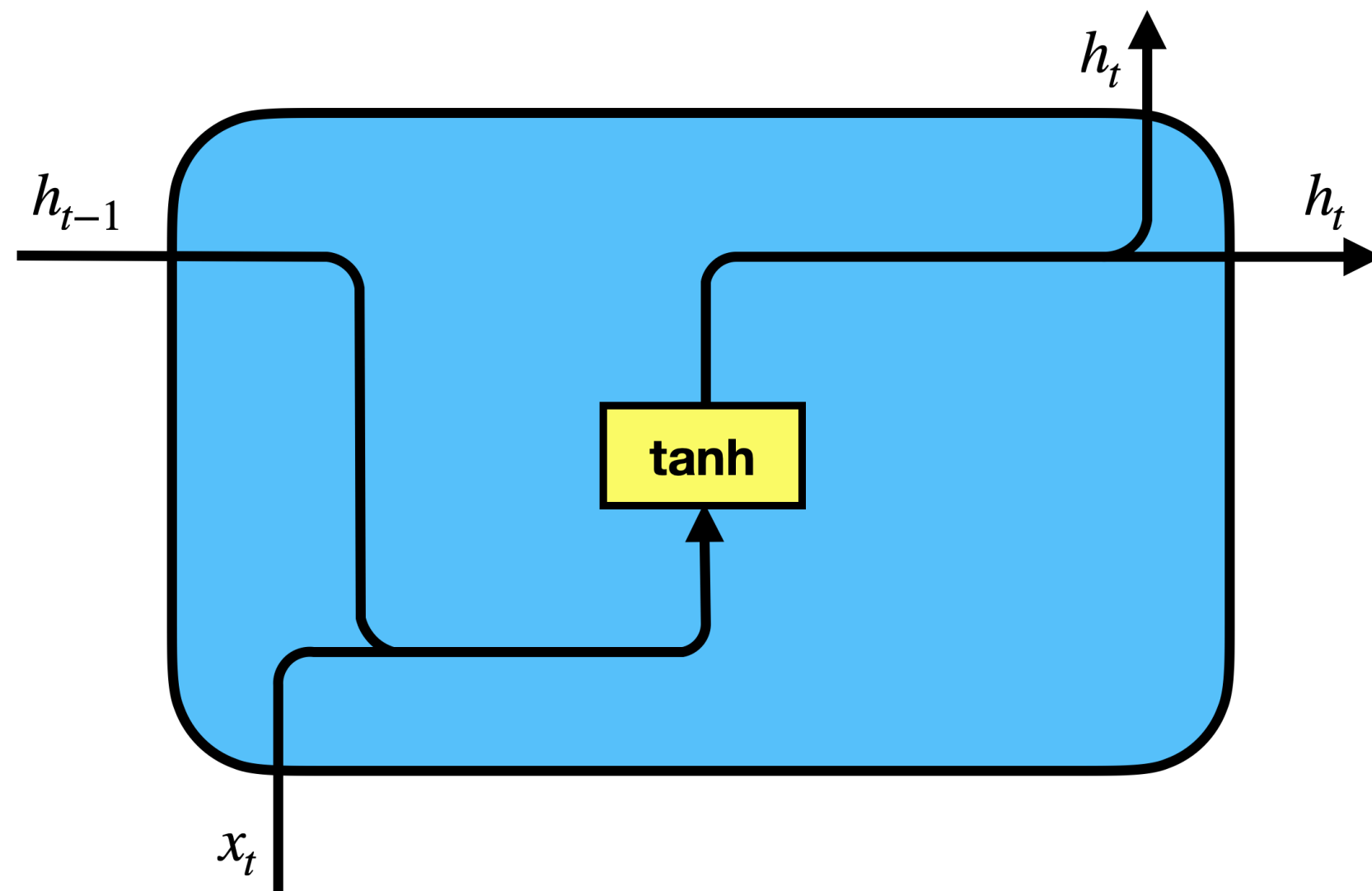
What is RNN

- 시계열(Sequential Data) 데이터를 처리하기 위한 모델
 - 음성, 언어, 주가, 센서 데이터 등
- 문장에서 이전에 나온 단어를 보아 다음 단어를 예측
- 동일한 Weight와 Bias가 모든 입력 값에 대해서 동일하게 사용 됨



RNN

$$h_t = \tanh(W_h h_{t-1} + W_x x_t + b)$$



Sequential
Data

I

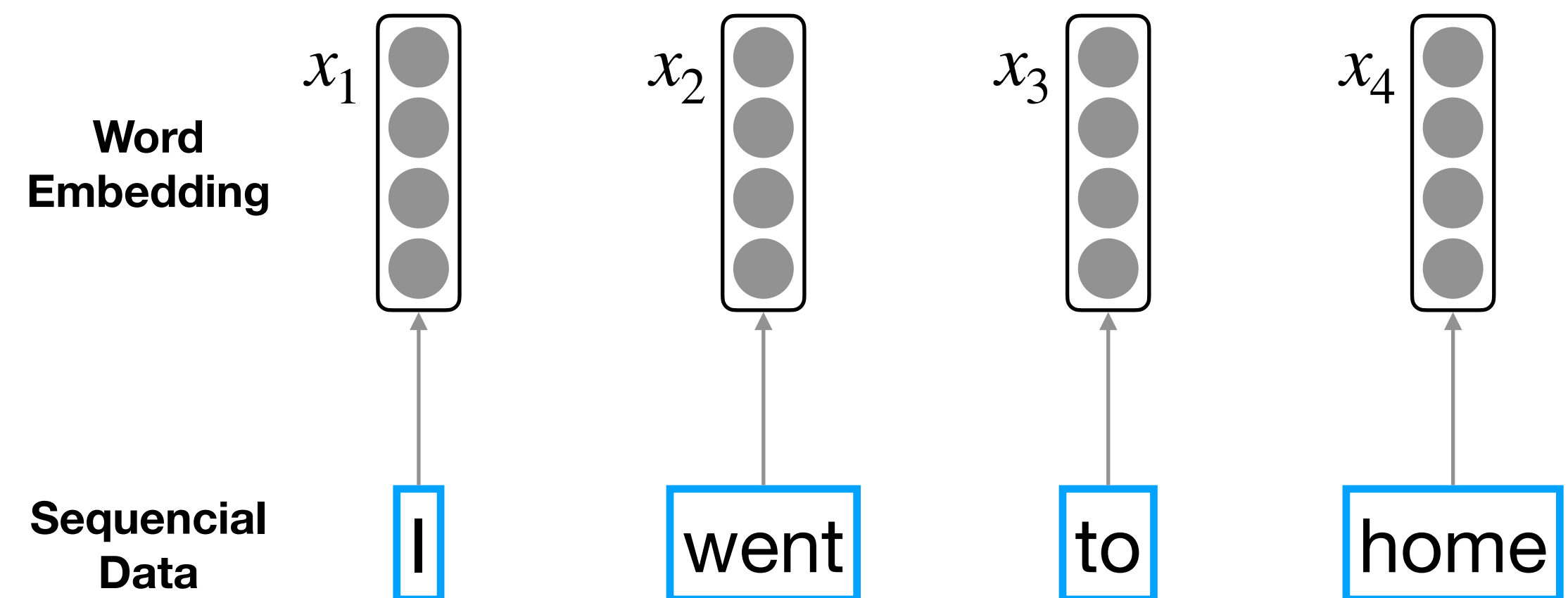
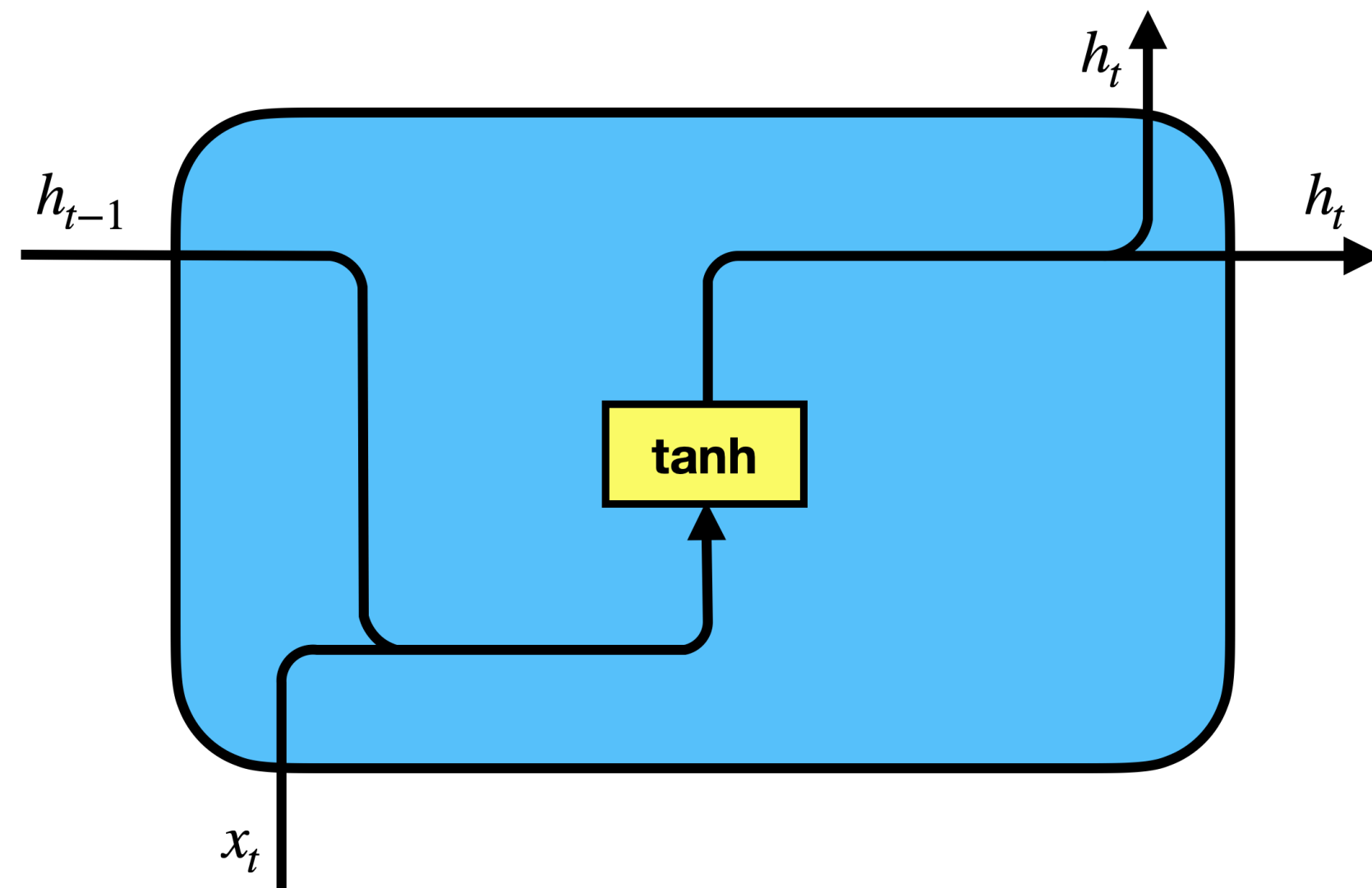
went

to

home

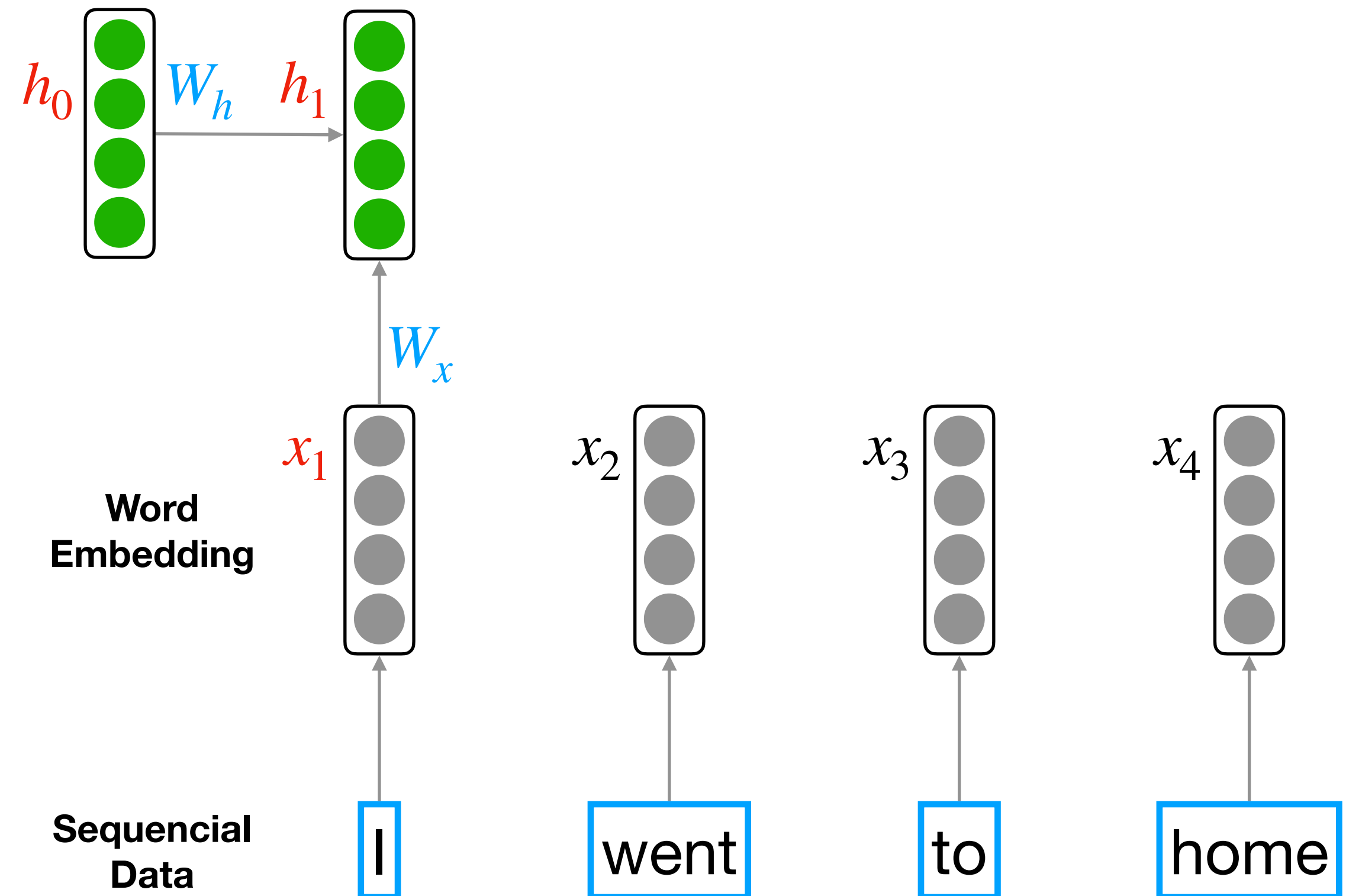
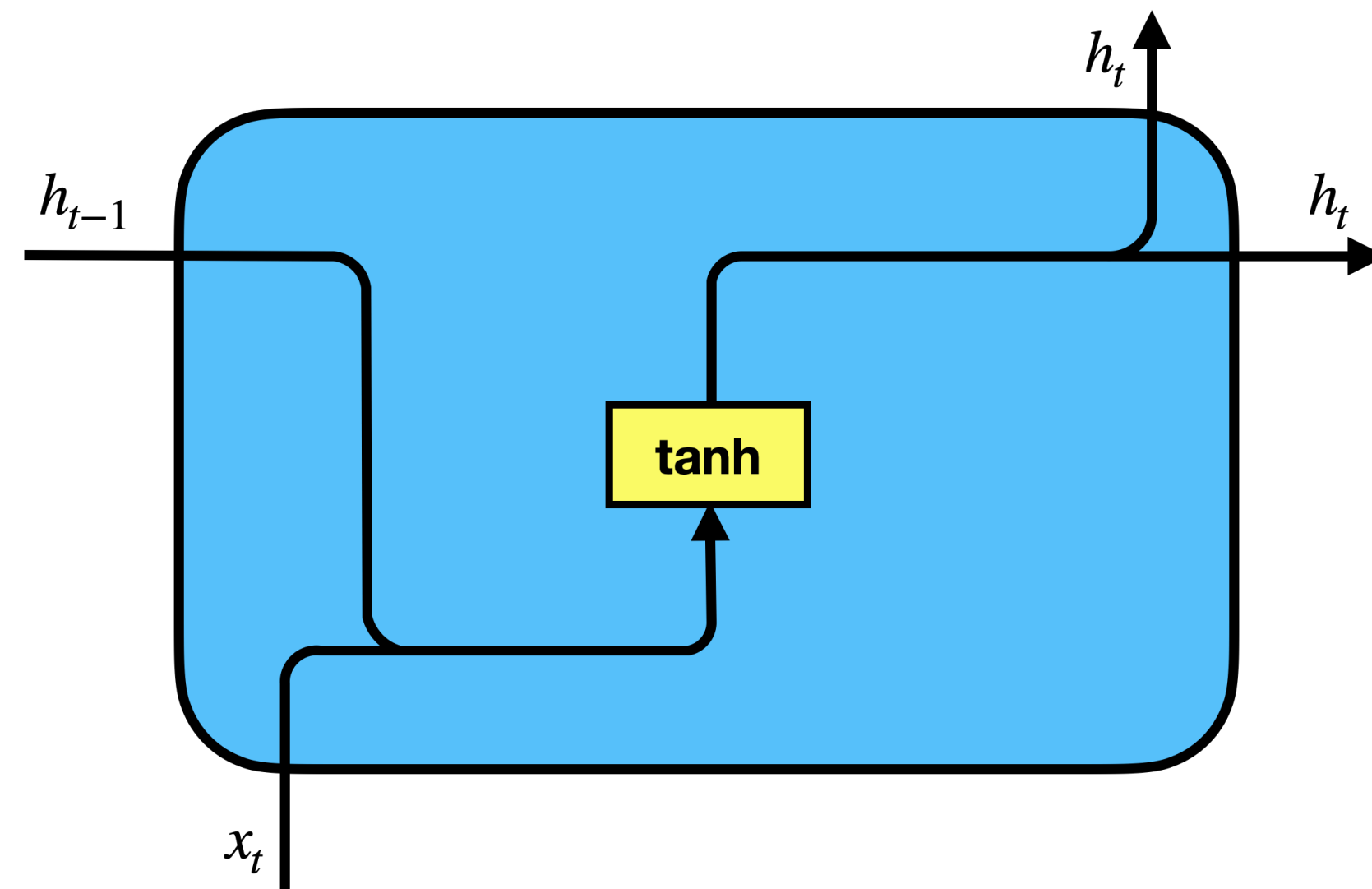
RNN

$$h_t = \tanh(W_h h_{t-1} + W_x x_t + b)$$



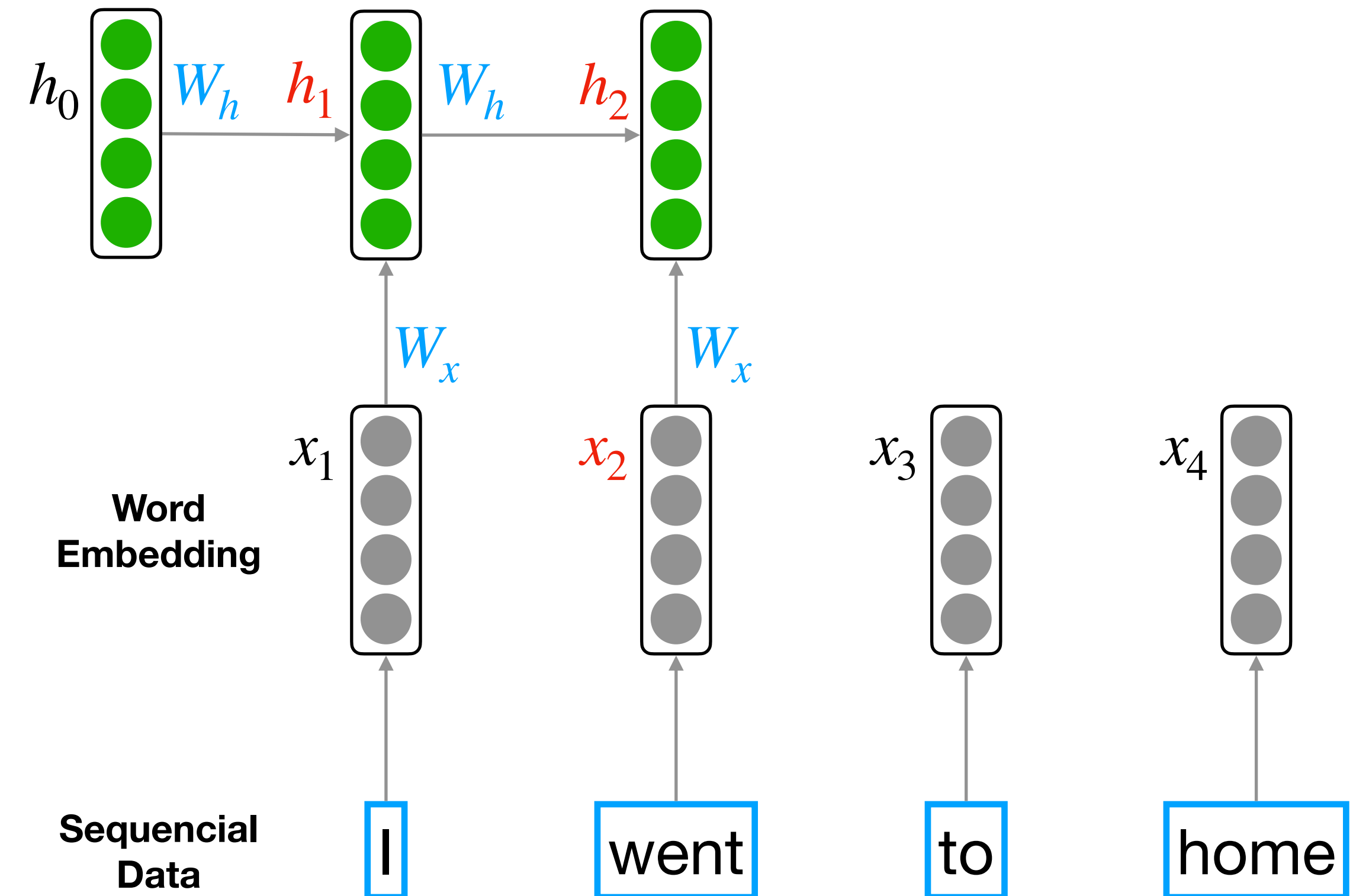
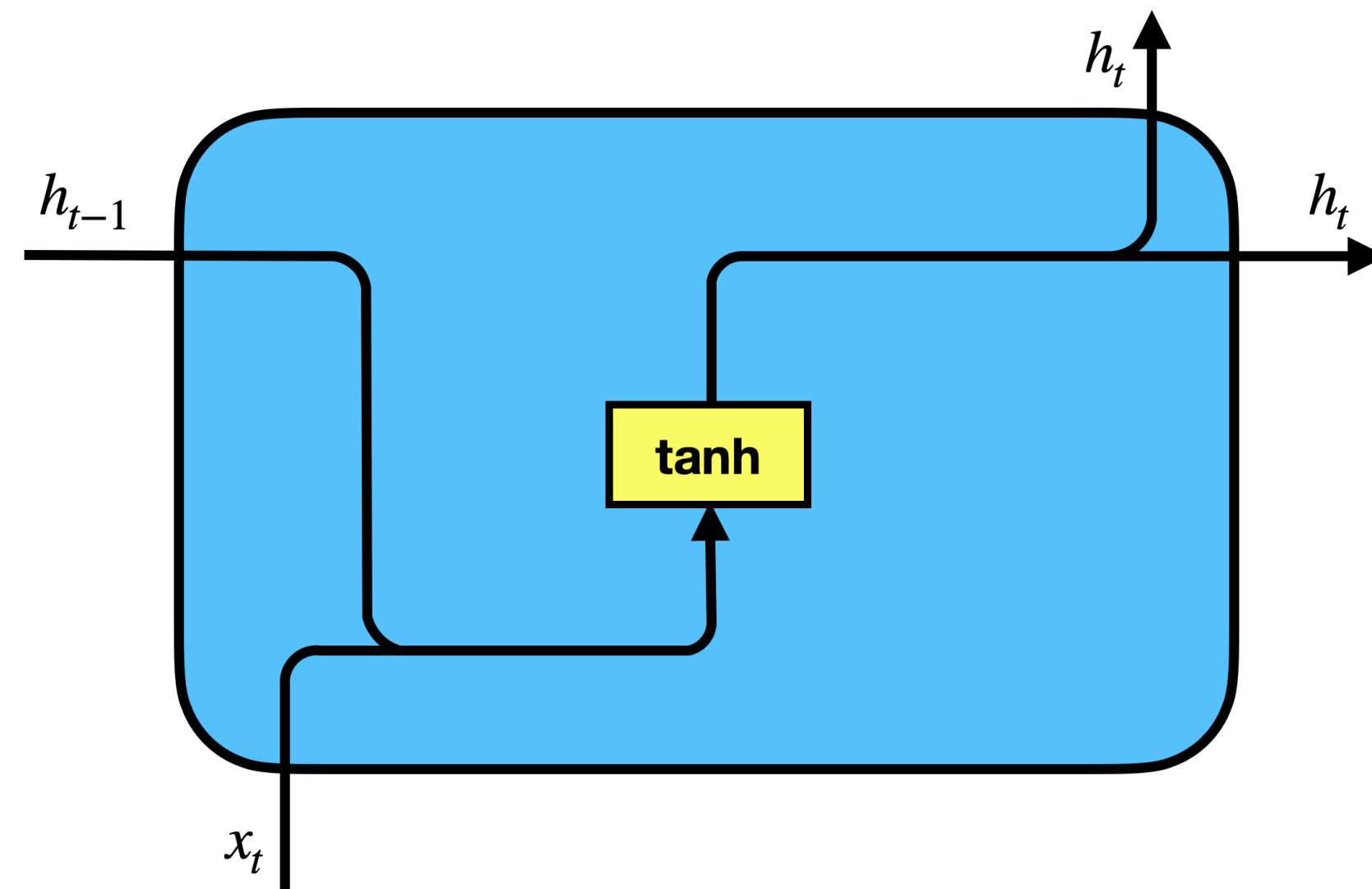
RNN

$$h_t = \tanh(W_h h_{t-1} + W_x x_t + b)$$



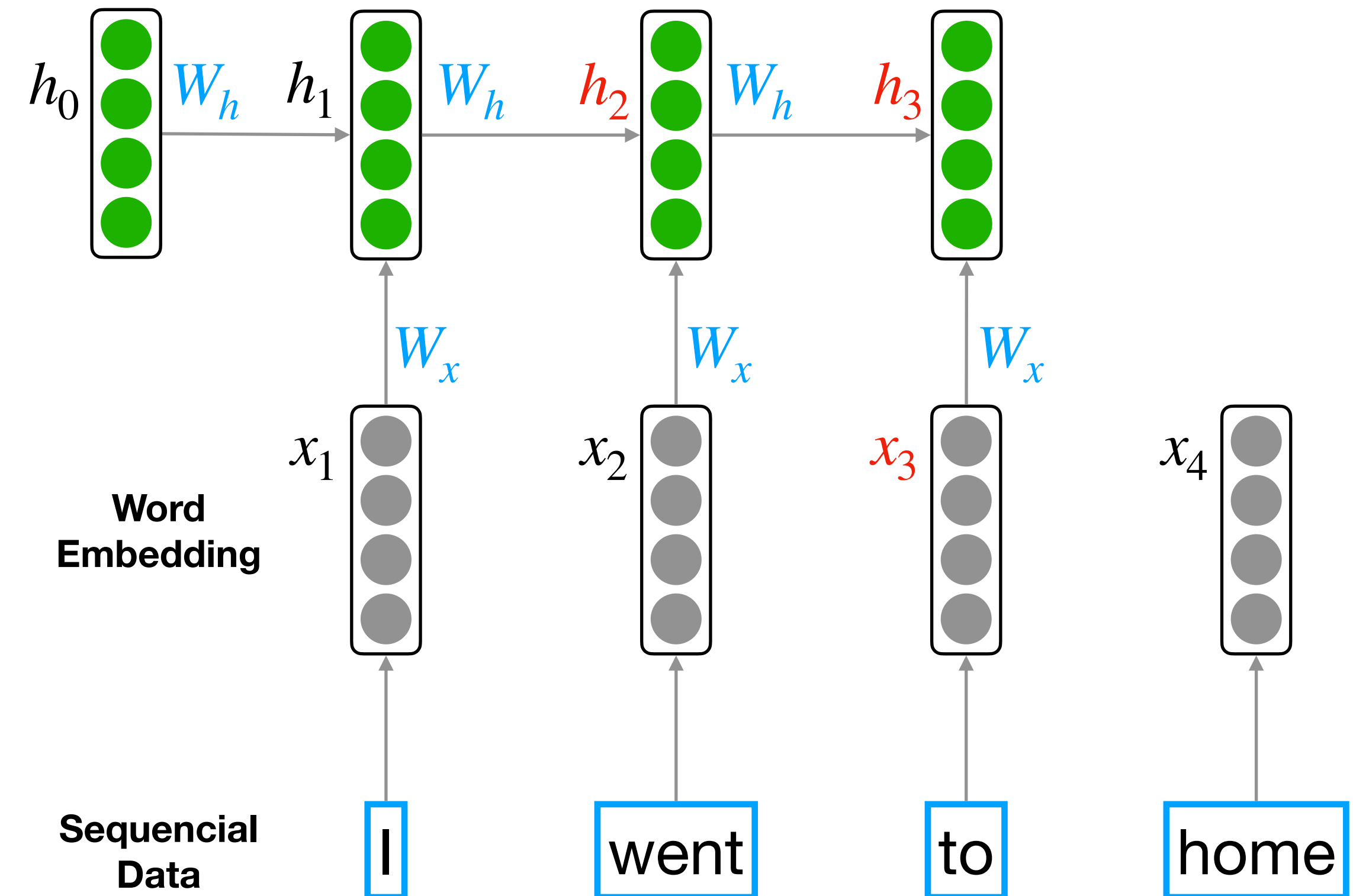
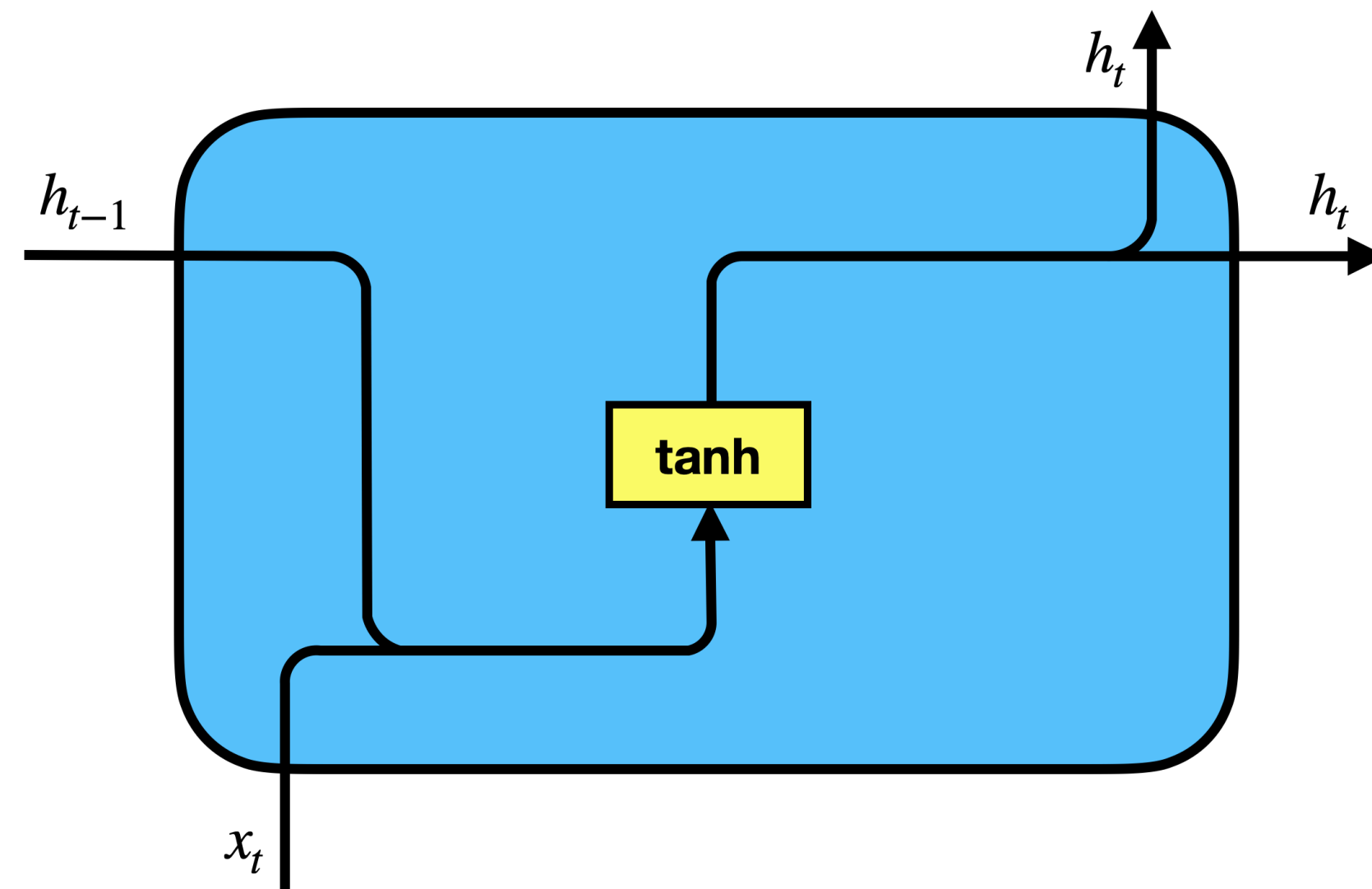
RNN

$$h_t = \tanh(W_h h_{t-1} + W_x x_t + b)$$



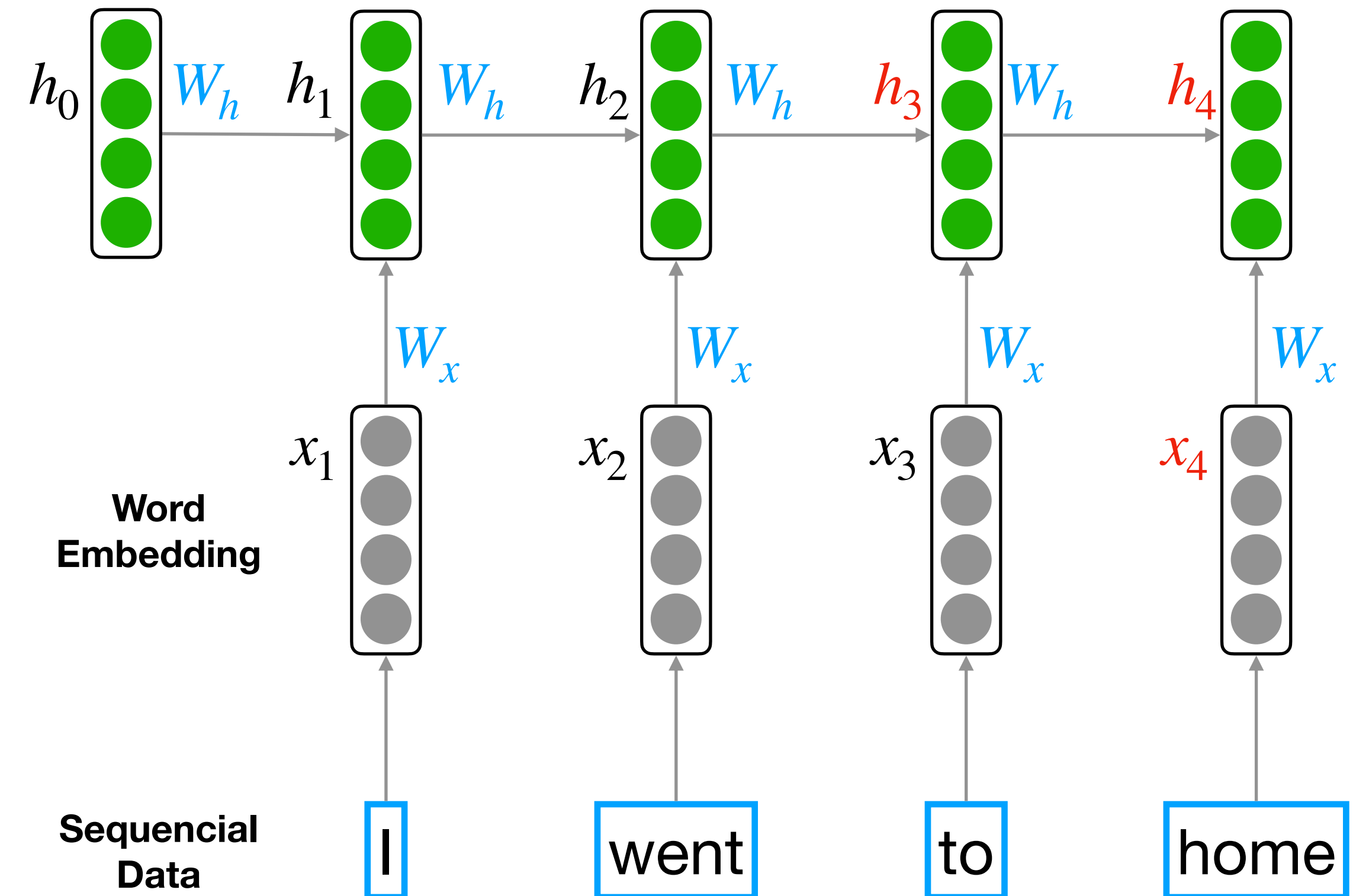
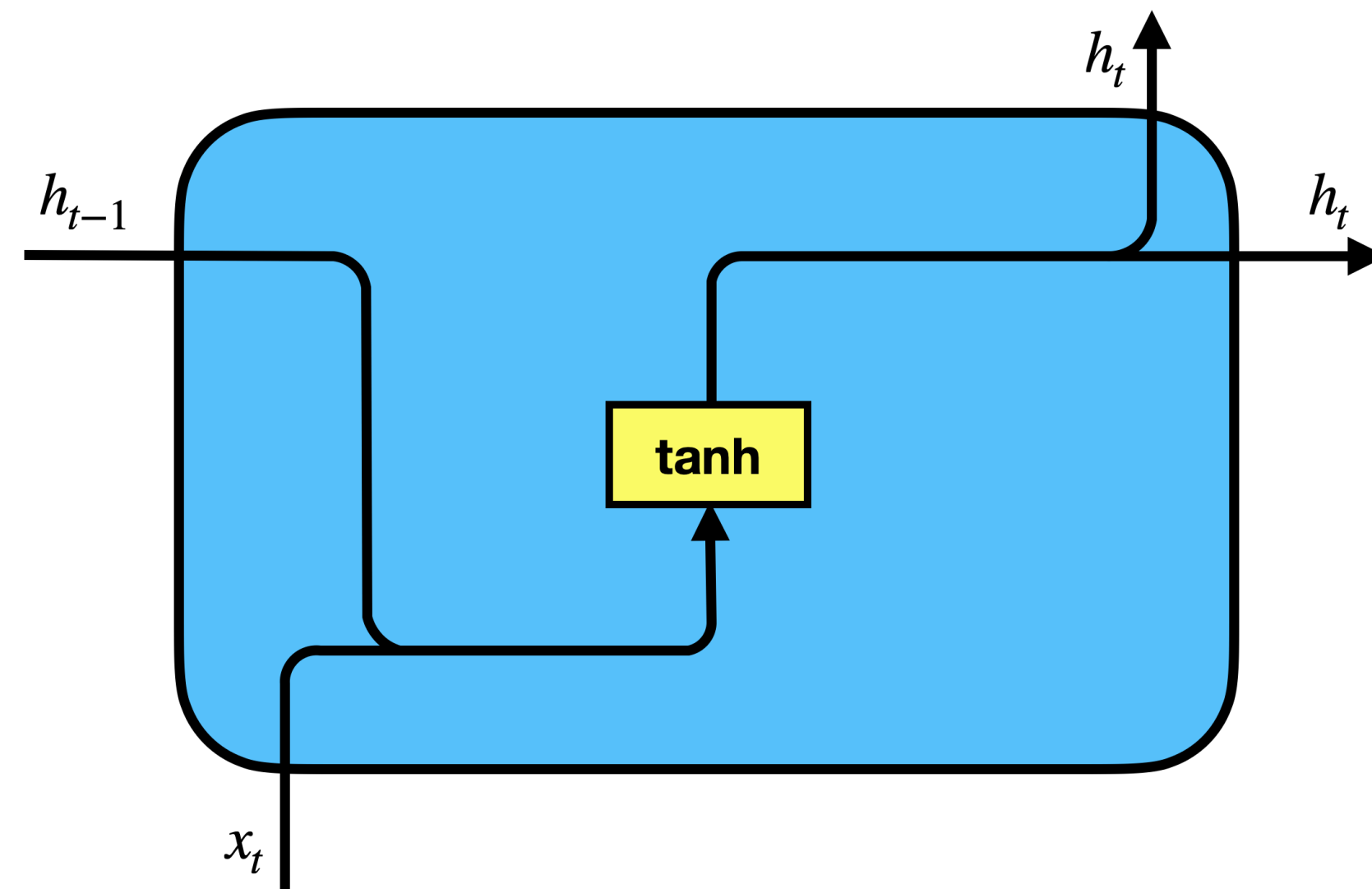
RNN

$$h_t = \tanh(W_h h_{t-1} + W_x x_t + b)$$



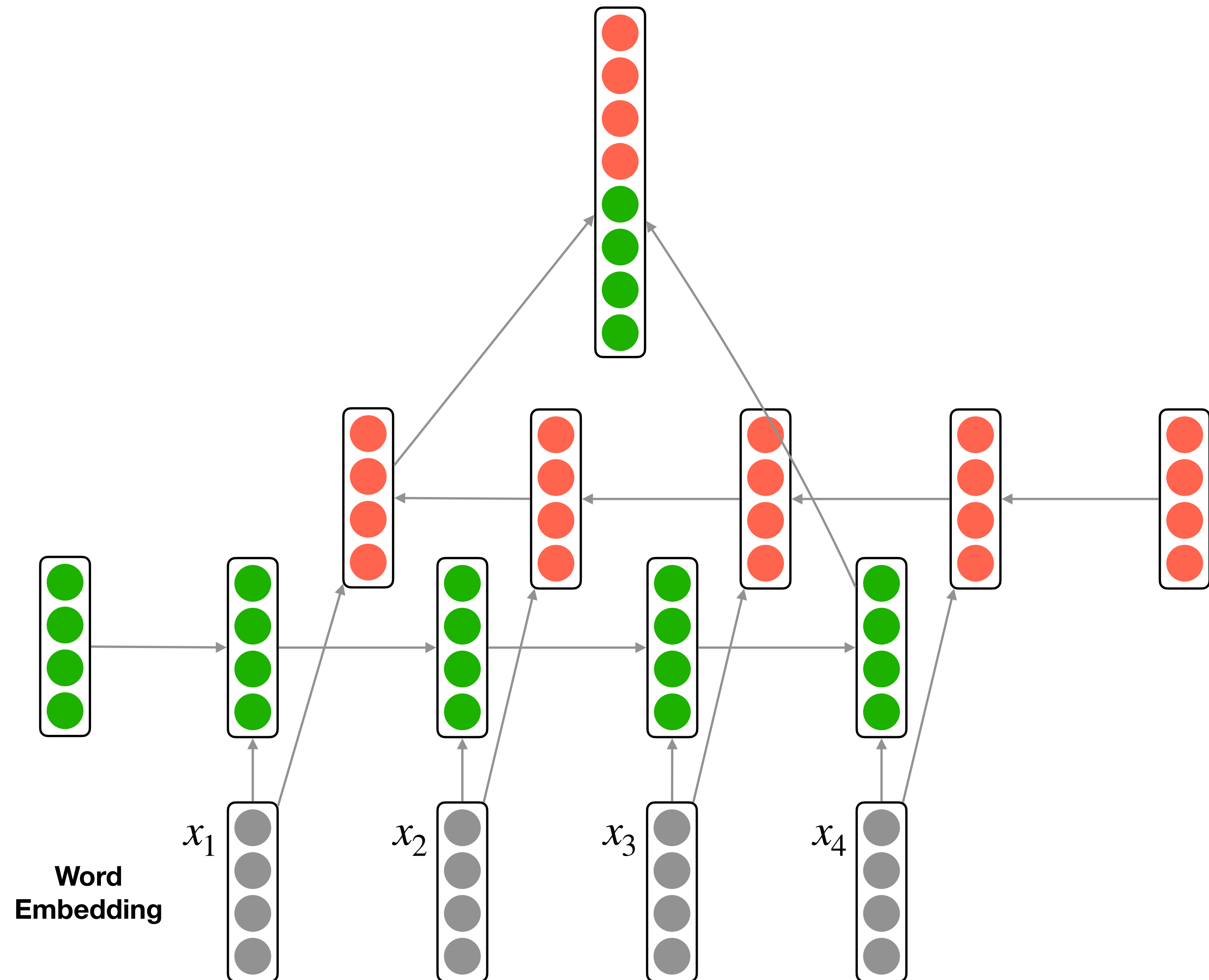
RNN

$$h_t = \tanh(W_h h_{t-1} + W_x x_t + b)$$



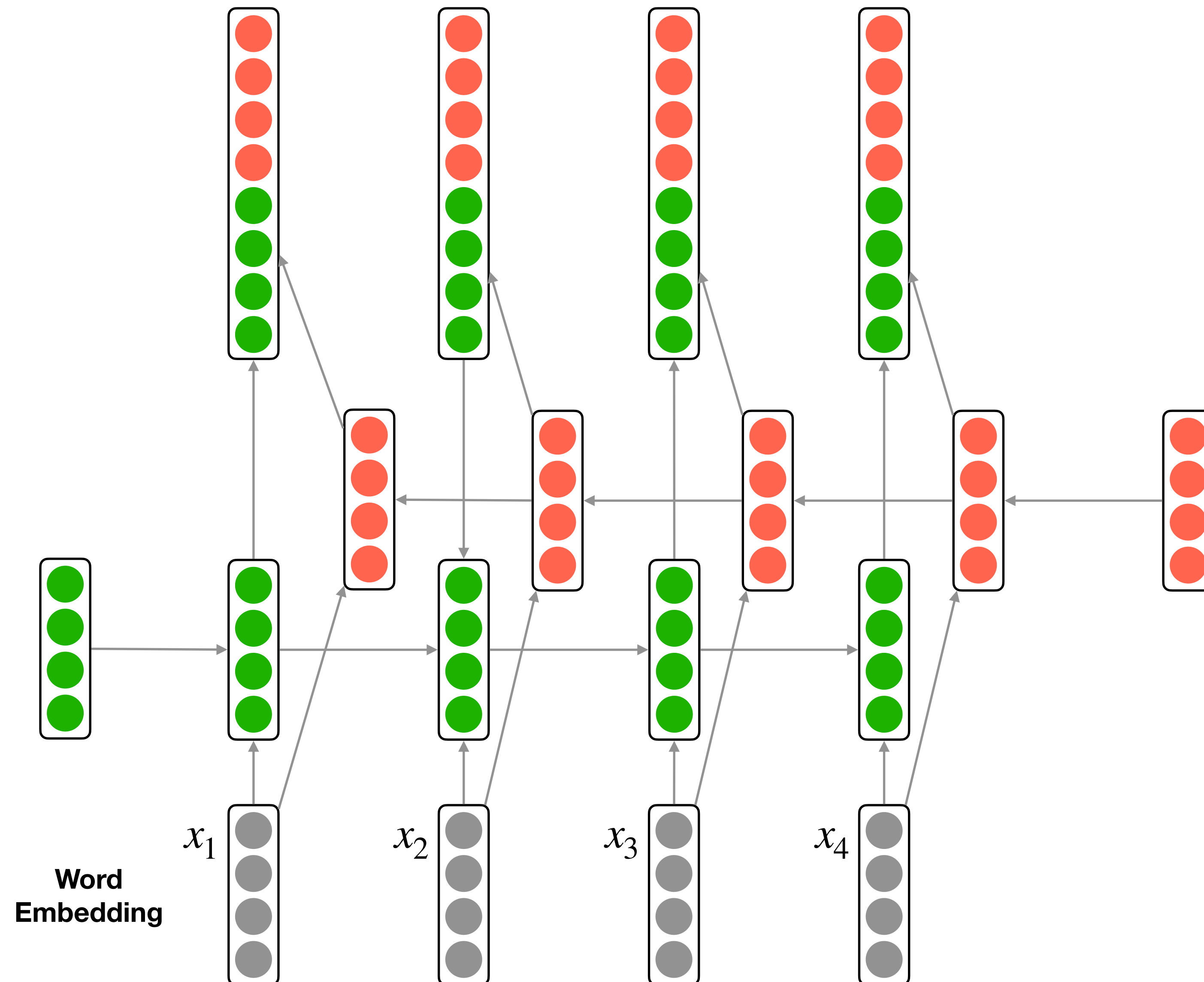
RNN (Bi-directional)

문장의 특징 추출



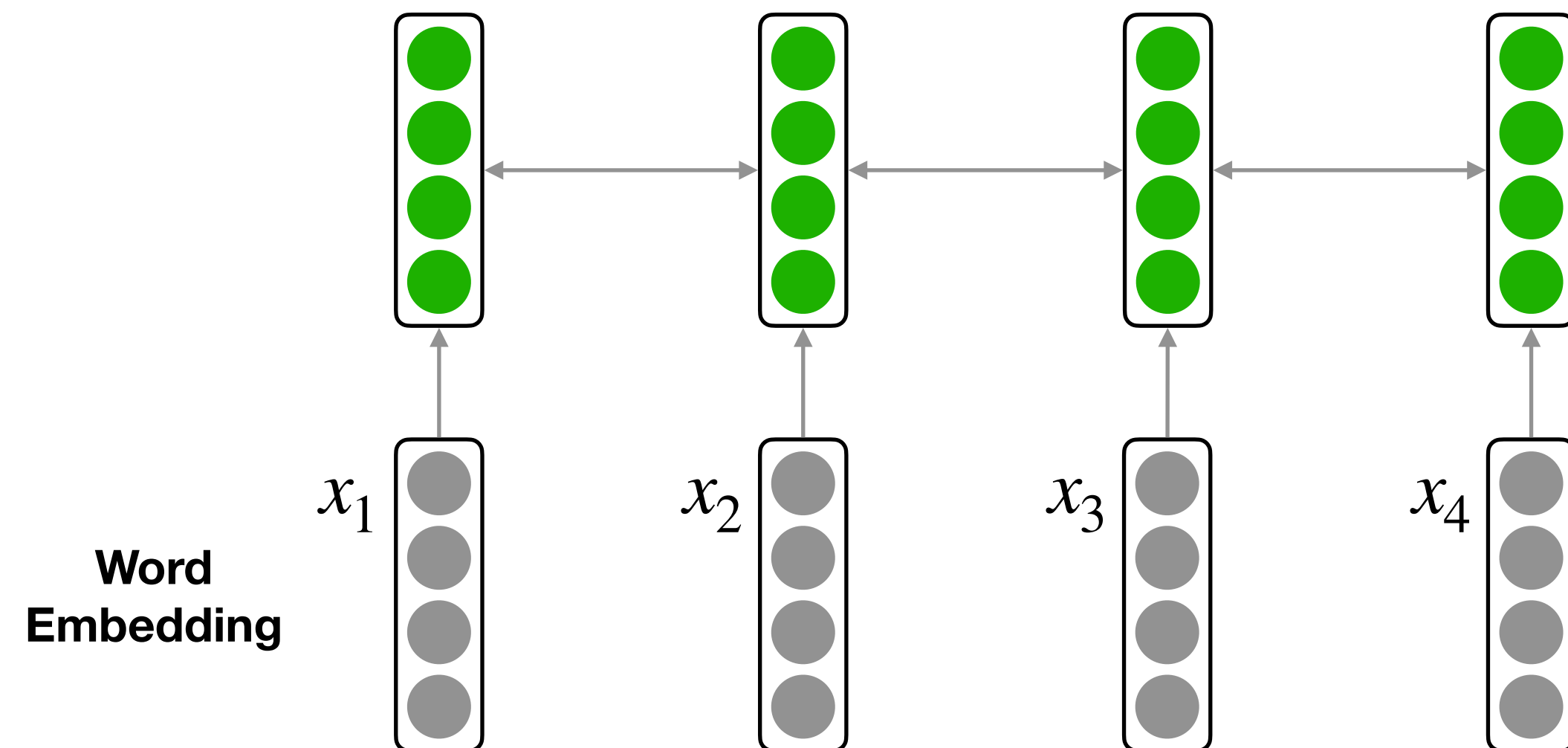
RNN (Bi-directional)

단어별 특징 추출



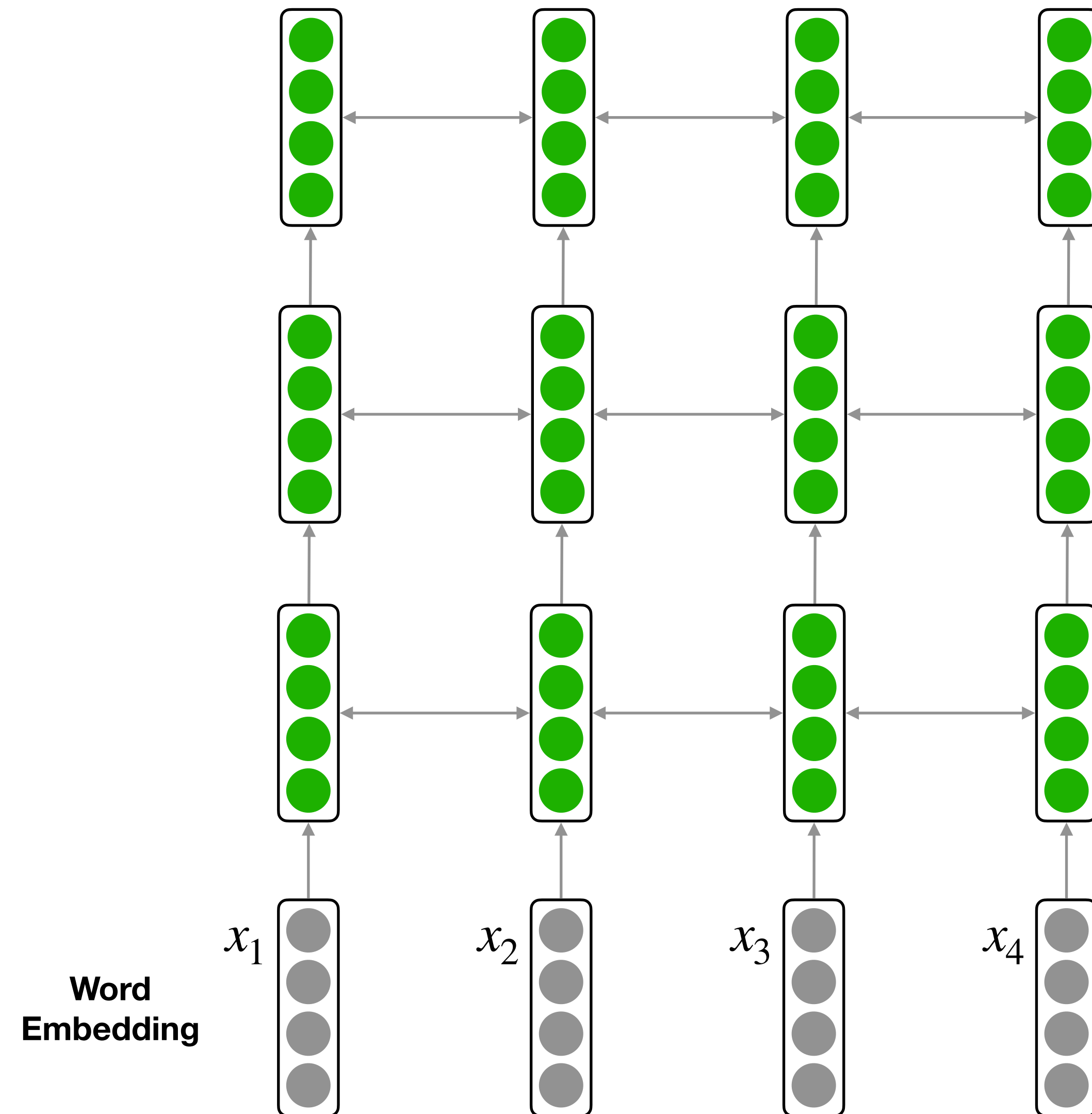
RNN (Bi-directional)

간단한 표현



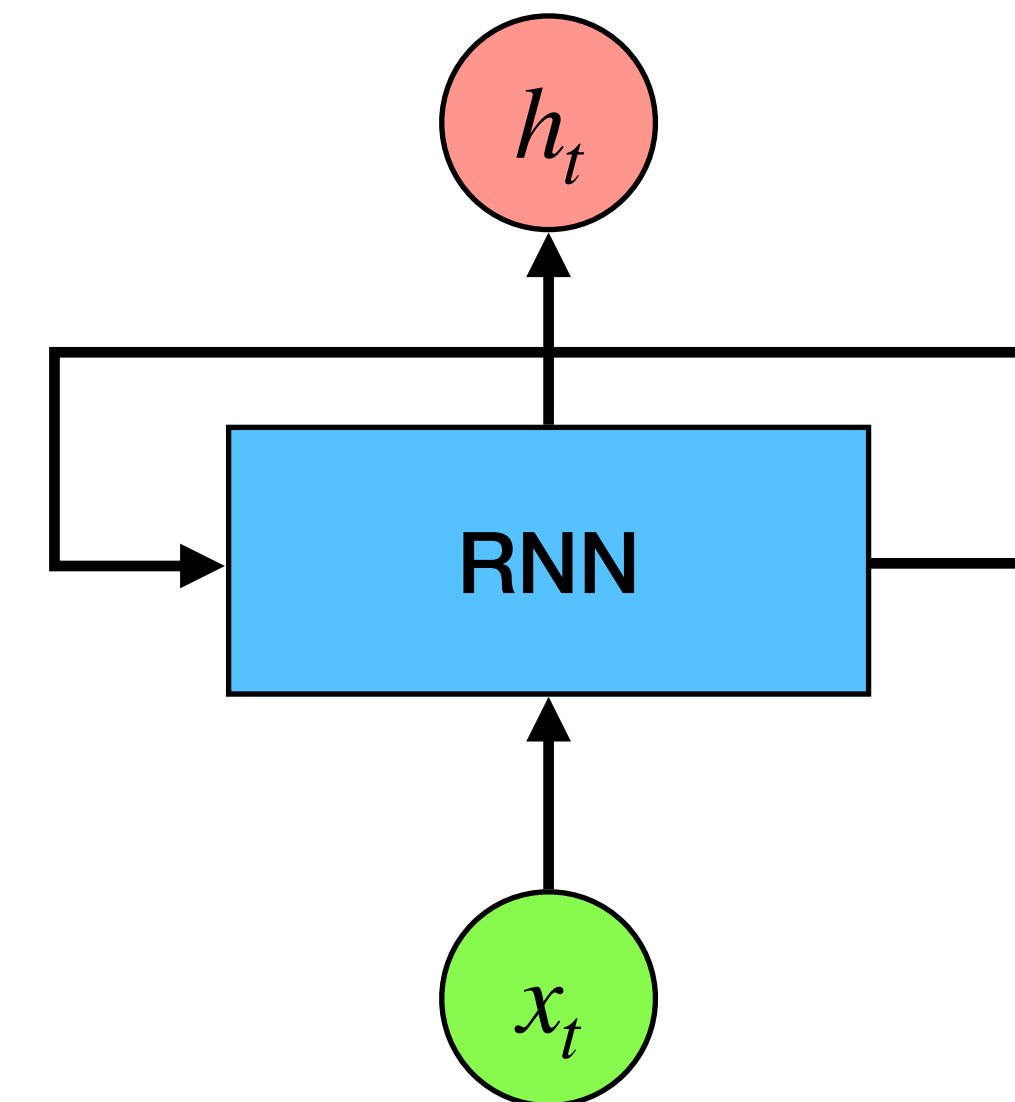
RNN (Bi-directional)

Multi layer stack



RNN

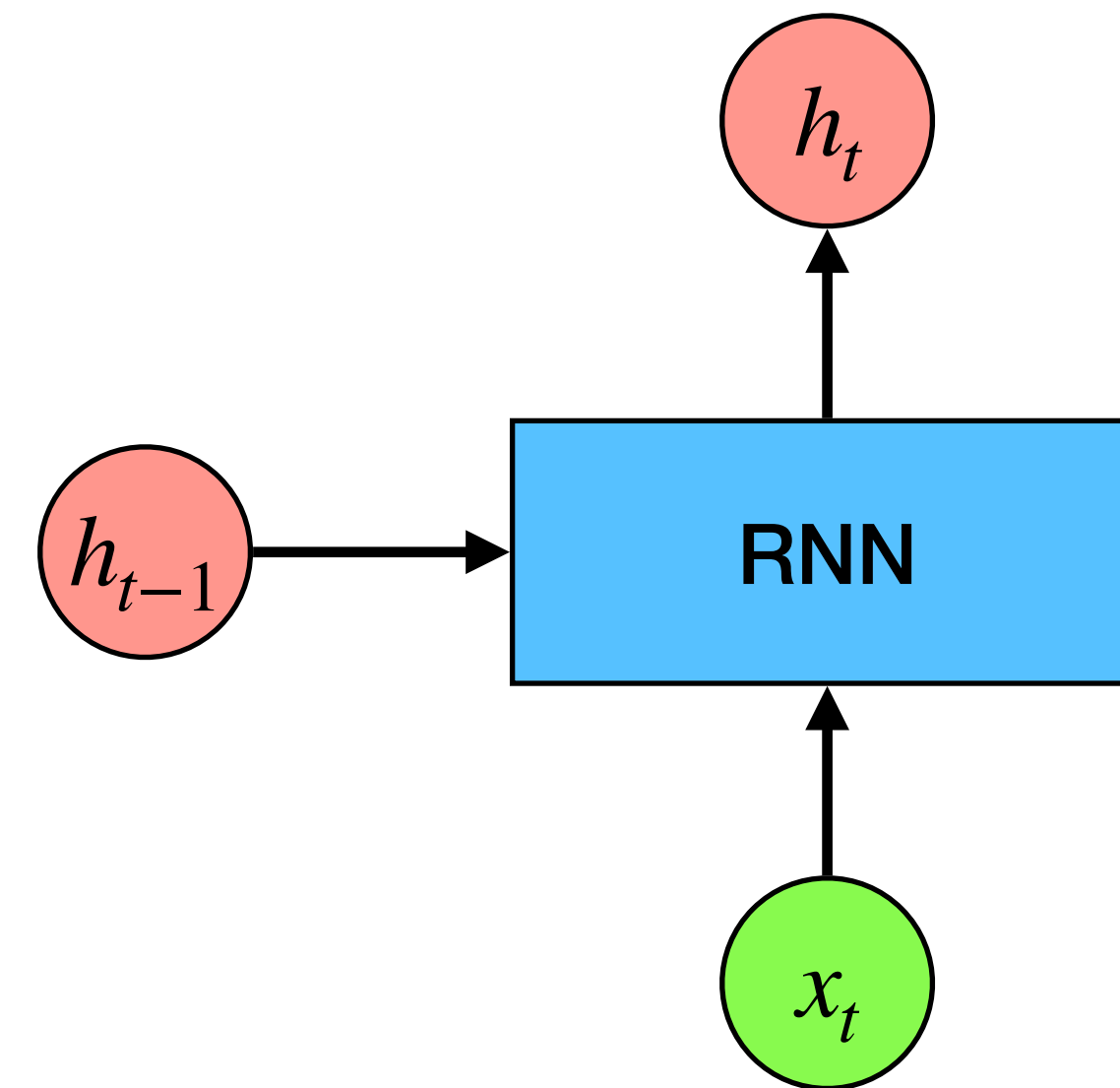
- 길이에 상관 없음
- 순서대로 처리해야 하므로 느림
- 오랜 과거의 정보를 접근하기 어려움
- Vanishing gradient problem
- Exploding gradient problem



RNN 네트워크를 재귀적으로 사용하므로 길이에 상관 없음

RNN

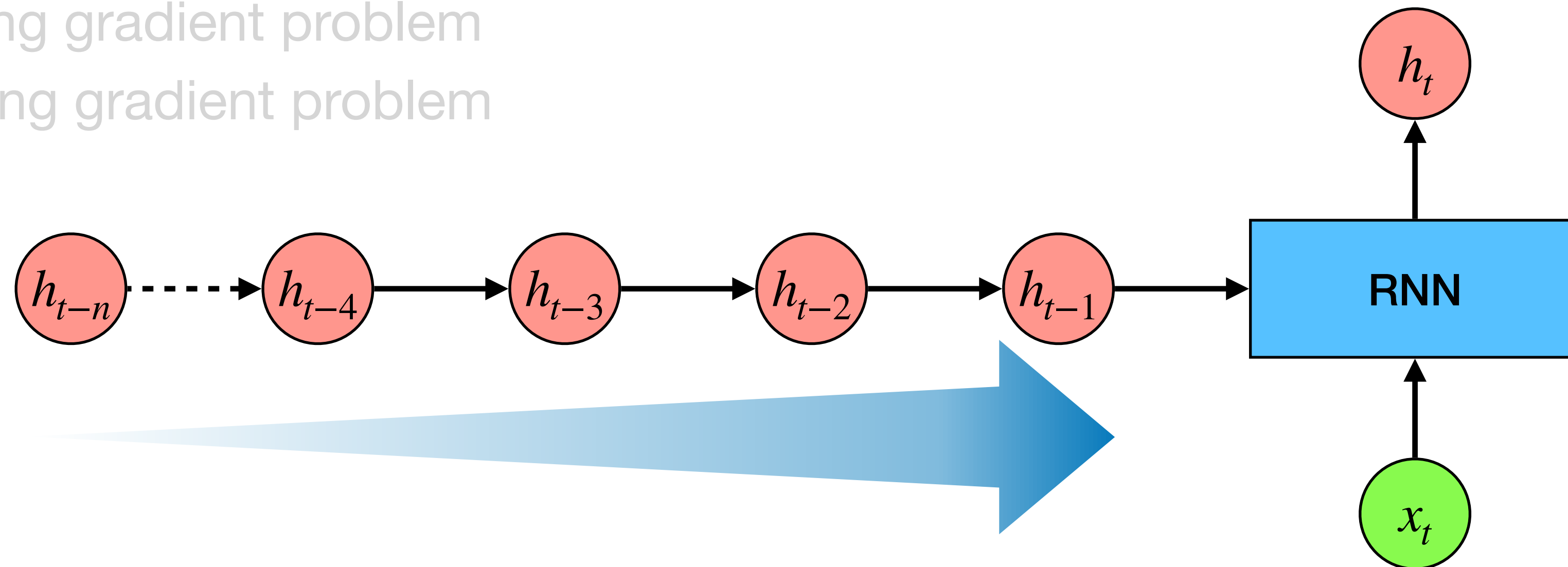
- 길이에 상관 없음
- **순서대로 처리해야 하므로 느림**
- 오랜 과거의 정보를 접근하기 어려움
- Vanishing gradient problem
- Exploding gradient problem



h_t 를 계산하기 위해서는 h_{t-1} 이 필요해서 병렬처리 불가

RNN

- 길이에 상관 없음
- 순서대로 처리해야 하므로 느림
- **오랜 과거의 정보를 접근하기 어려움**
- Vanishing gradient problem
- Exploding gradient problem



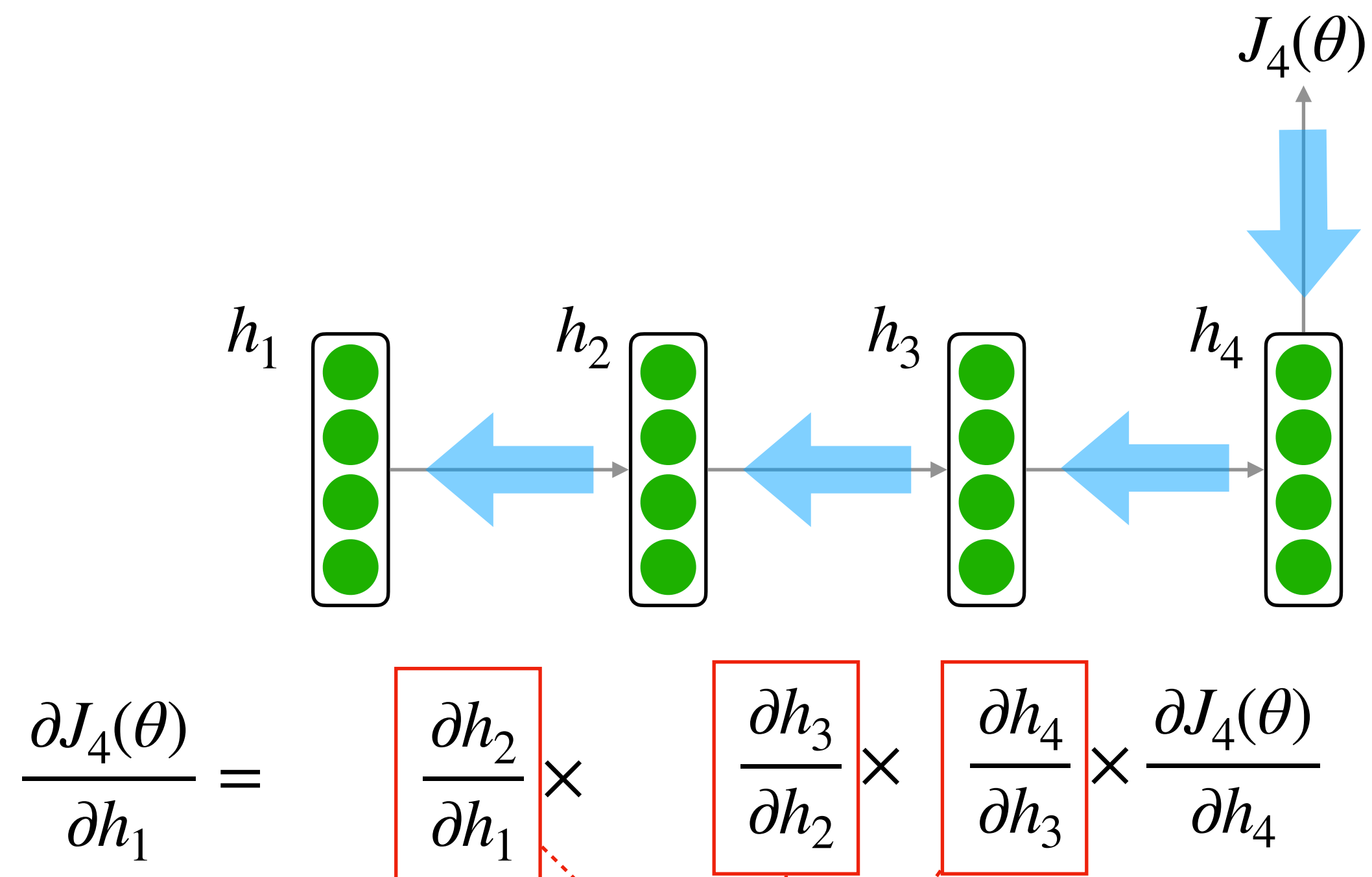
h_t 는 최근 정보가 강하고 과거의 정보는 점차 약해짐

RNN

- 길이에 상관 없음
- 순서대로 처리해야 하므로 느림
- 오랜 과거의 정보를 접근하기 어려움
- **Vanishing gradient problem**
- Exploding gradient problem

$$h_t = \tanh(W_h h_{t-1} + W_x x_t + b)$$

Chain rule!

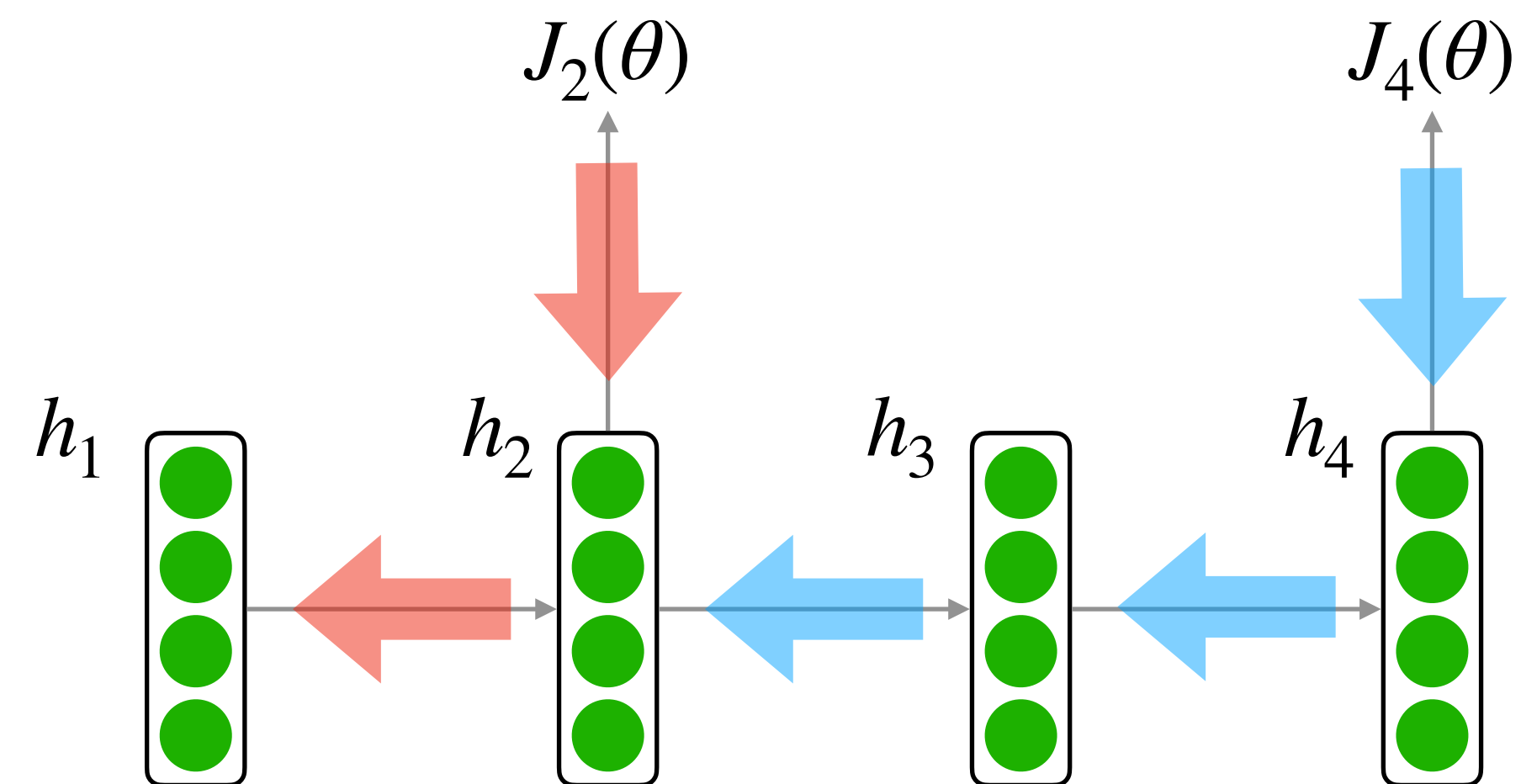


이 값들이 작으면 거리가 멀 수록 gradient가 감소함

RNN

- 길이에 상관 없음
- 순서대로 처리해야 하므로 느림
- 오랜 과거의 정보를 접근하기 어려움
- **Vanishing gradient problem**
- Exploding gradient problem

$$h_t = \tanh(W_h h_{t-1} + W_x x_t + b)$$

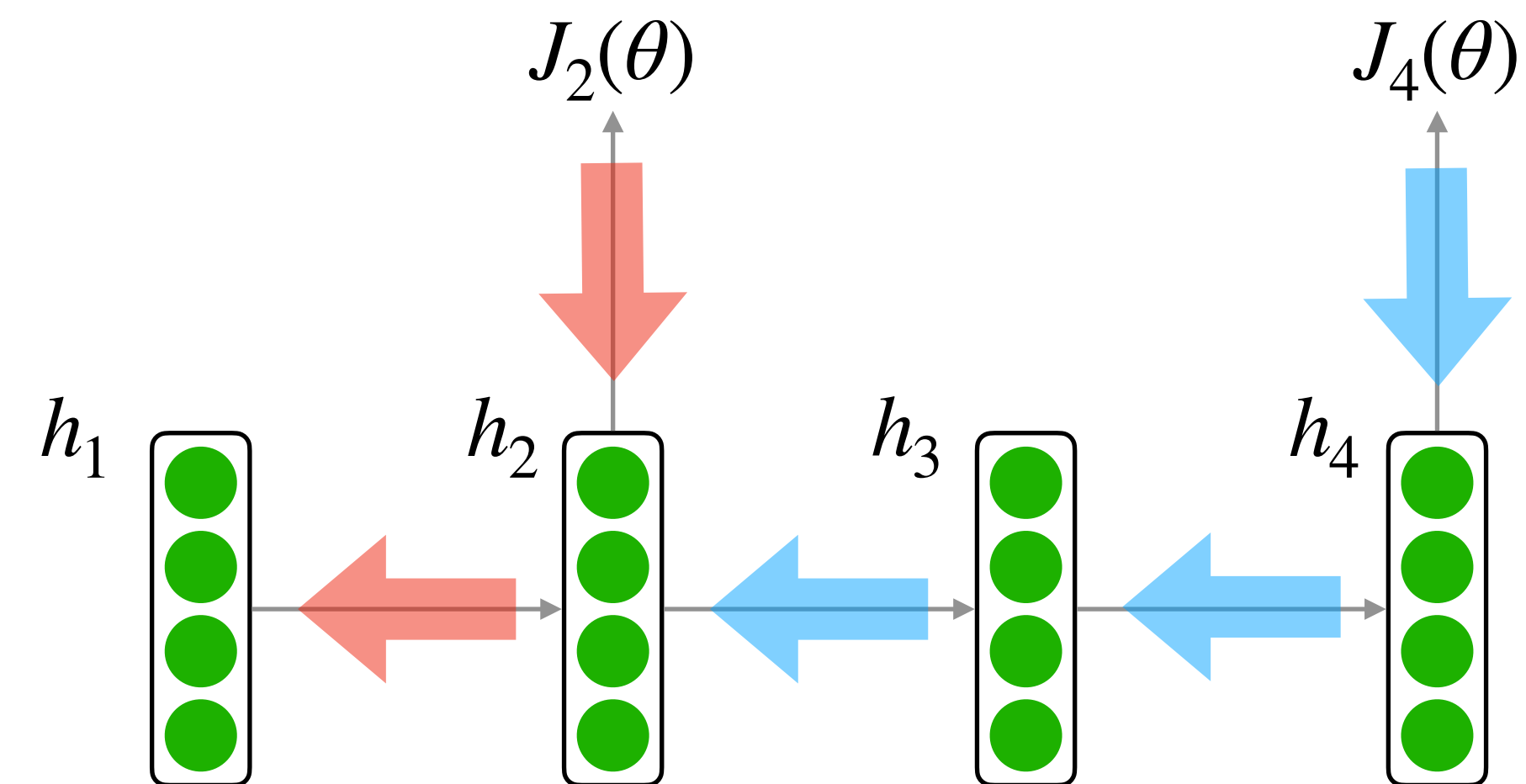


먼 거리의 gradient는 사라지고
가까운 거리의 gradient만 영향을 받음

RNN

- 길이에 상관 없음
- 순서대로 처리해야 하므로 느림
- 오랜 과거의 정보를 접근하기 어려움
- **Vanishing gradient problem**
- Exploding gradient problem

$$h_t = \tanh(W_h h_{t-1} + W_x x_t + b)$$



The **writer** of the **books** _____

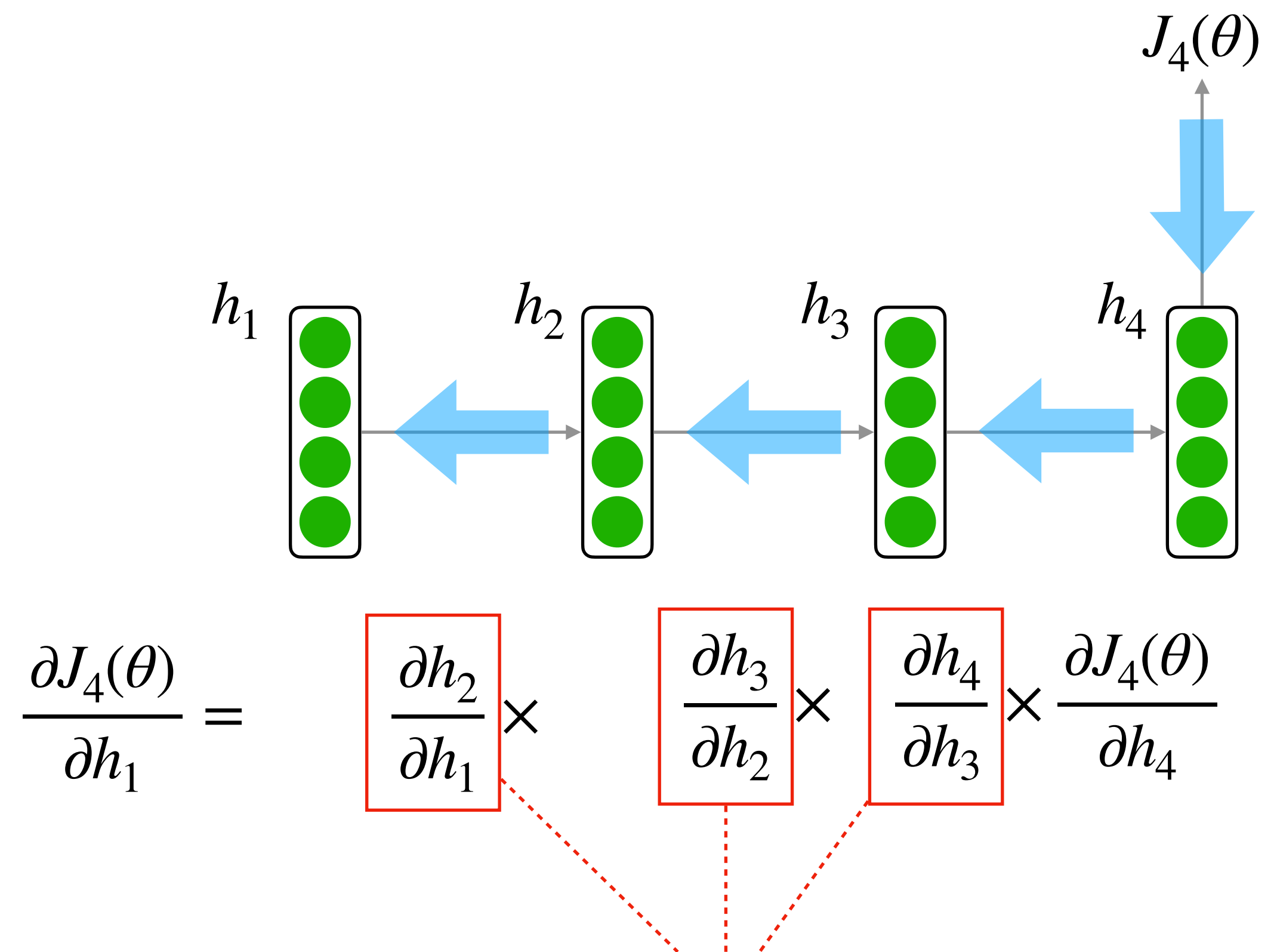
are

is

RNN

- 길이에 상관 없음
- 순서대로 처리해야 하므로 느림
- 오랜 과거의 정보를 접근하기 어려움
- Vanishing gradient problem
- **Exploding gradient problem**

$$h_t = \tanh(W_h h_{t-1} + W_x x_t + b)$$

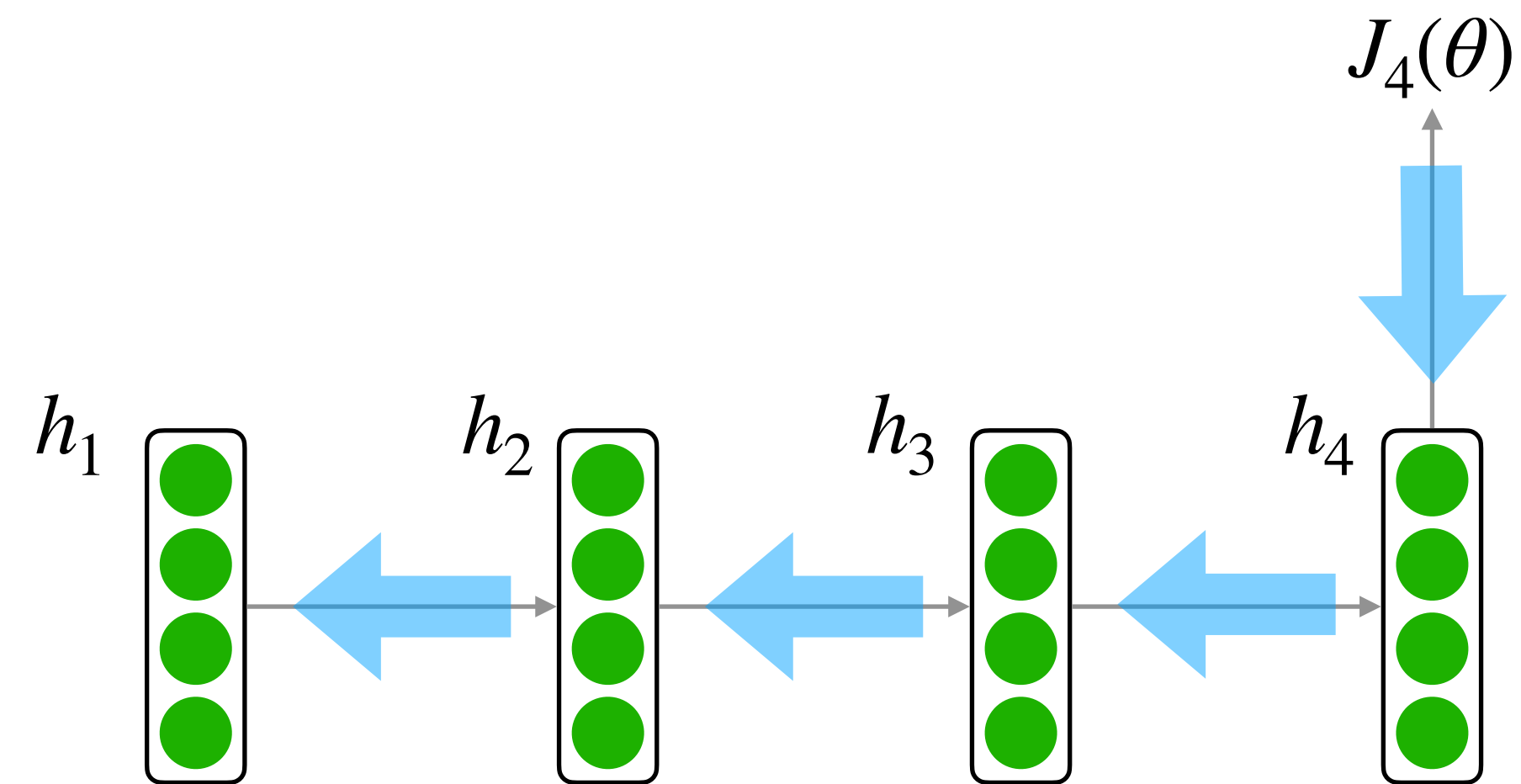


이 값들이 크면 거리가 멀 수록 gradient가 증가함

RNN

- 길이에 상관 없음
- 순서대로 처리해야 하므로 느림
- 오랜 과거의 정보를 접근하기 어려움
- Vanishing gradient problem
- **Exploding gradient problem**

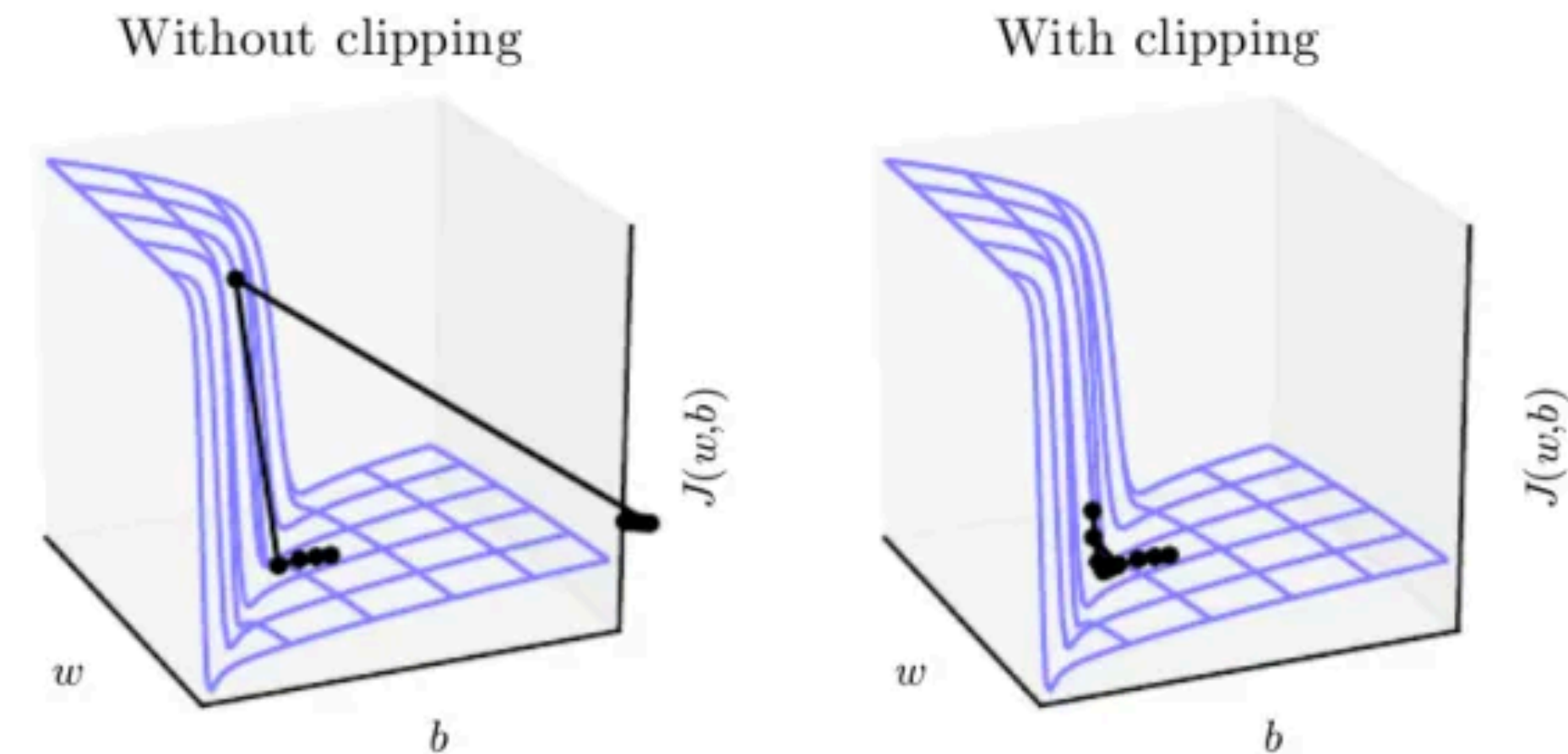
$$h_t = \tanh(W_h h_{t-1} + W_x x_t + b)$$



학습이 잘 안되거나
Inf, NaN 등의 loss가 발생할 수 있음

RNN

- 길이에 상관 없음
- 순서대로 처리해야 하므로 느림
- 오랜 과거의 정보를 접근하기 어려움
- Vanishing gradient problem
- **Exploding gradient problem**

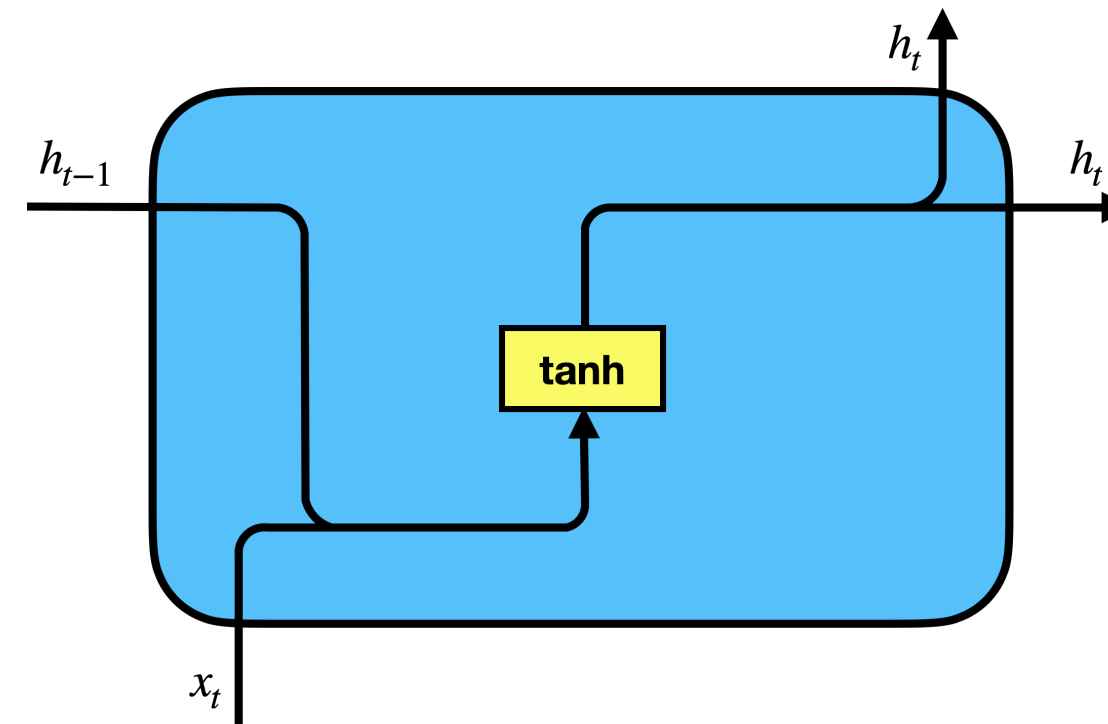


Algorithm 1 Pseudo-code for norm clipping the gradients whenever they explode

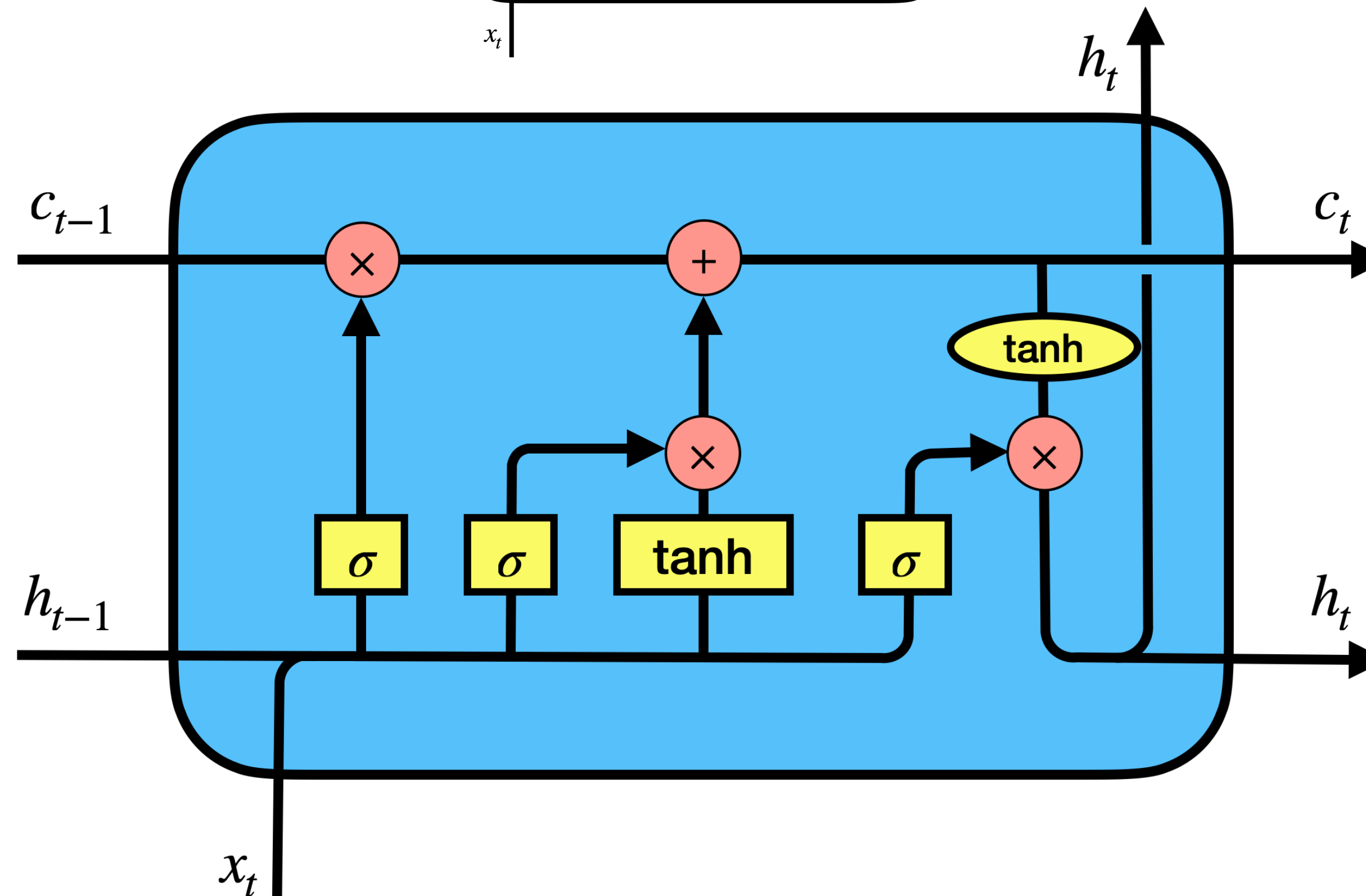
$$\hat{\mathbf{g}} \leftarrow \frac{\partial \mathcal{E}}{\partial \theta}$$
$$\text{if } \|\hat{\mathbf{g}}\| \geq \text{threshold} \text{ then}$$
$$\quad \hat{\mathbf{g}} \leftarrow \frac{\text{threshold}}{\|\hat{\mathbf{g}}\|} \hat{\mathbf{g}}$$
$$\text{end if}$$

LSTM (Long-Short Term Memory)

RNN

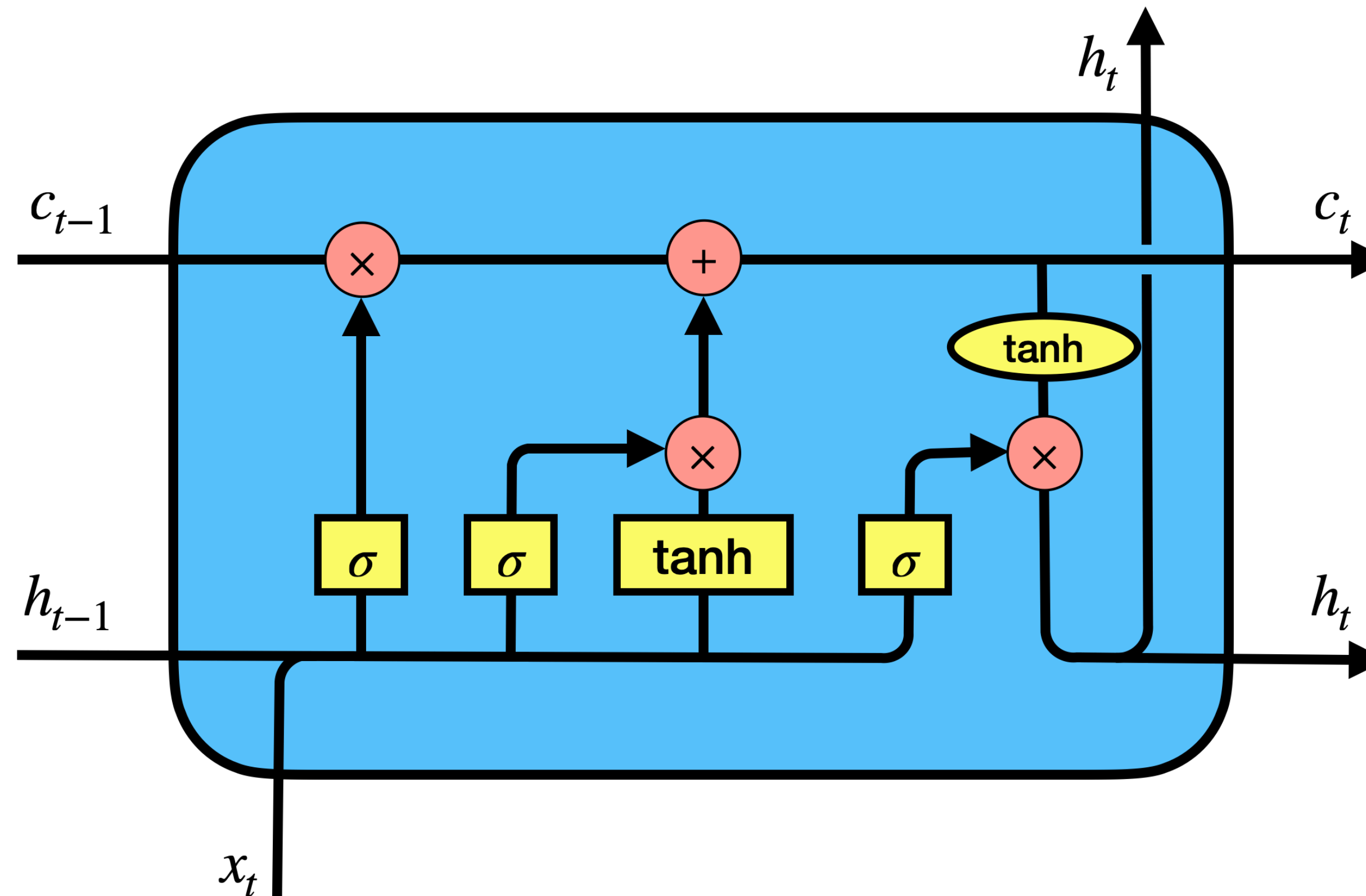


LSTM



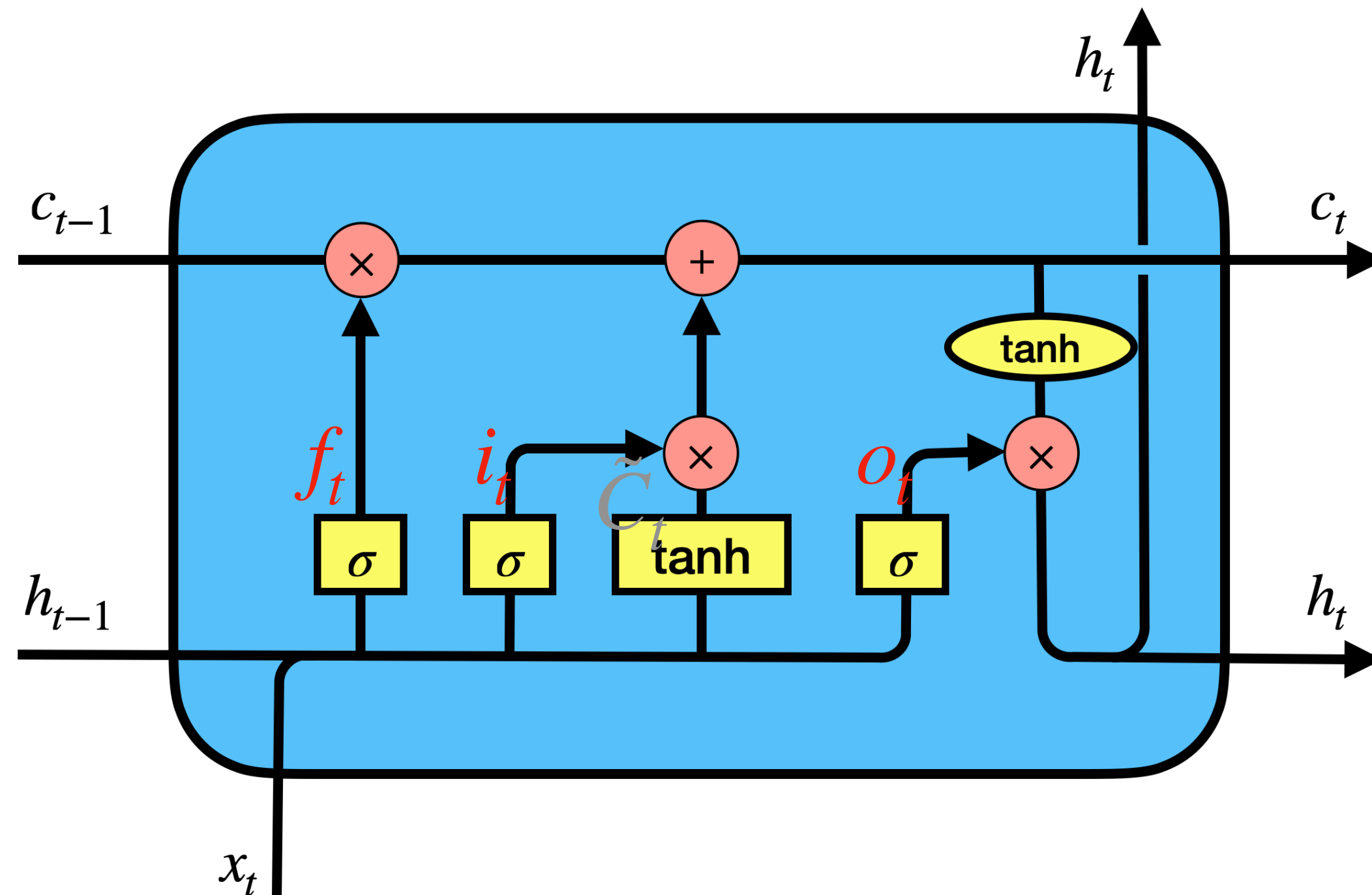
What is LSTM

- RNN의 Vanishing gradient 문제를 해결하기 위한 모델
 - Memory cell을 추가 함 (long term state)
 - $c_t = c_{t-1} + function(h_{t-1}, x_t)$



LSTM (gates)

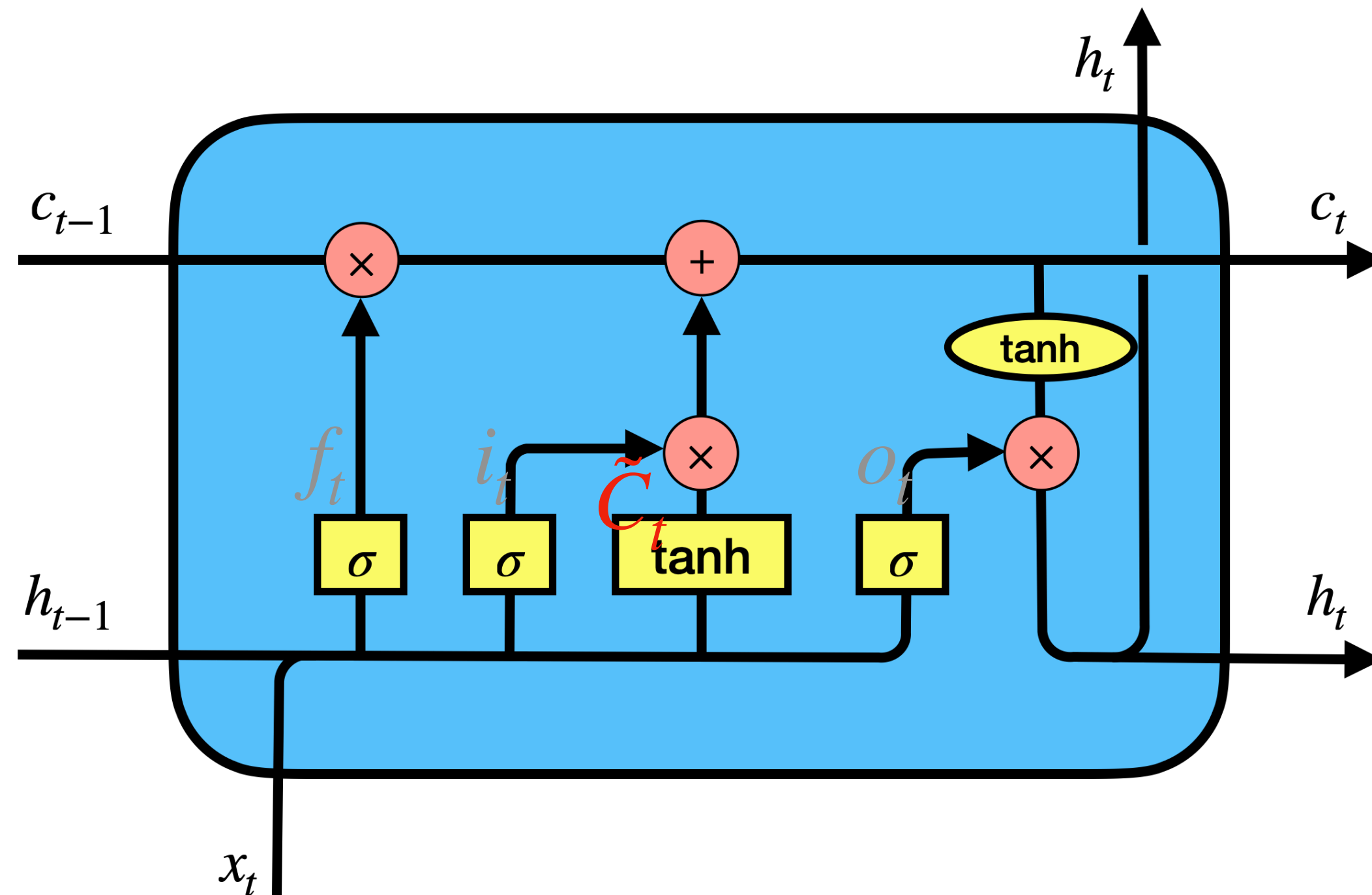
- Forget Gate: 이전 step Memory cell의 정보를 얼마나 사용할지 결정
- Input Gate: 새로 계산된 Memory cell을 얼마나 사용할지 결정
- Output Gate: Memory cell값을 얼마만큼 사용할 것인지 결정



- $f_t = \sigma(W_{hf}h_{t-1} + W_{xf}x_t + b_f)$
- $i_t = \sigma(W_{hi}h_{t-1} + W_{xi}x_t + b_i)$
- $o_t = \sigma(W_{ho}h_{t-1} + W_{xo}x_t + b_o)$
- $\tilde{C}_t = \tanh(W_{hc}h_{t-1} + W_{xc}x_t + b_c)$
- $C_t = f_t \odot C_{t-1} + i_t \odot \tilde{C}_t$
- $h_t = o_t \odot \tanh(C_t)$

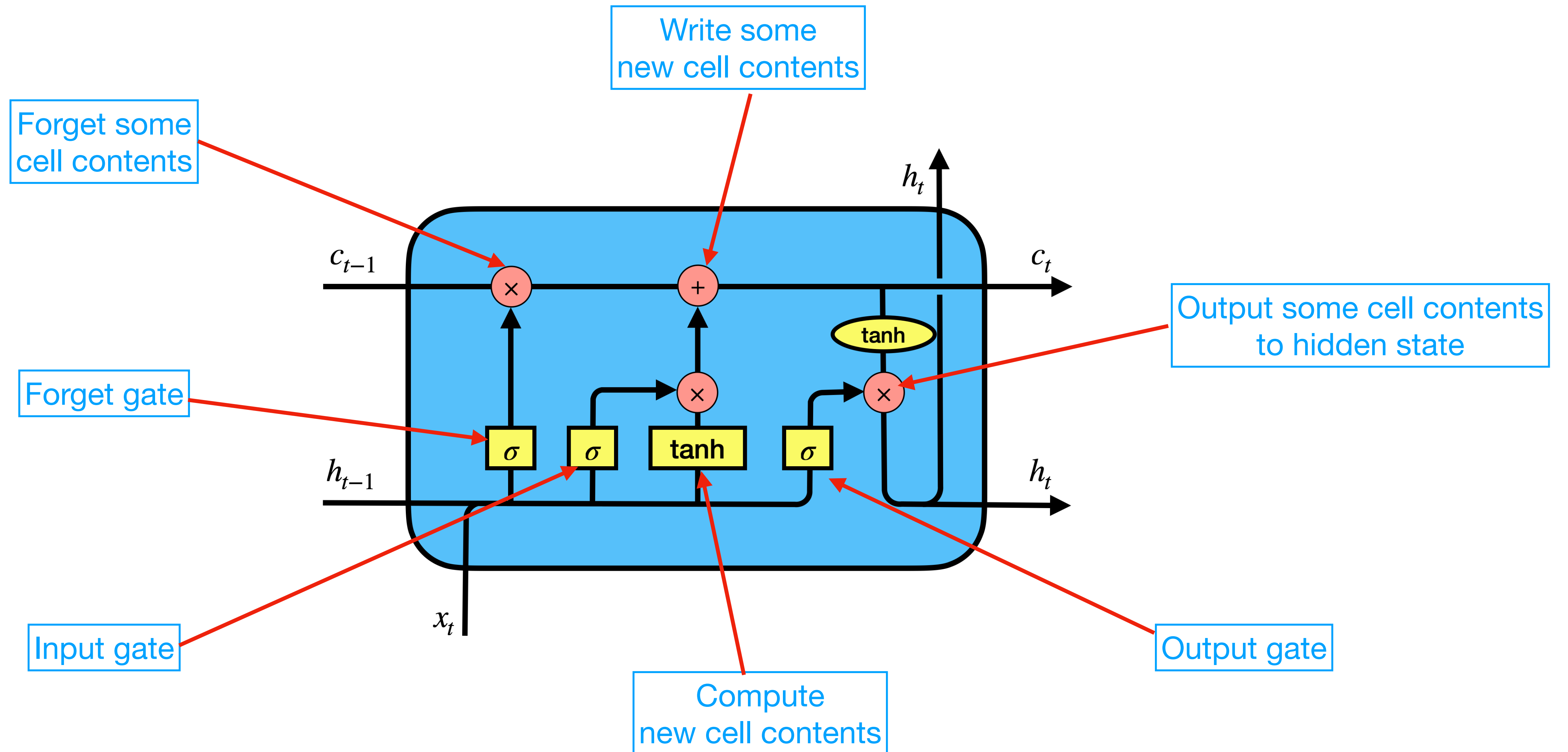
LSTM (cell & hidden)

- New Cell Contents: 현재 시점의 Memory cell 값
- Cell State: 과거 Cell State와 현재 시점의 Cell Content의 합
- Hidden State: 현재 시점의 출력 값



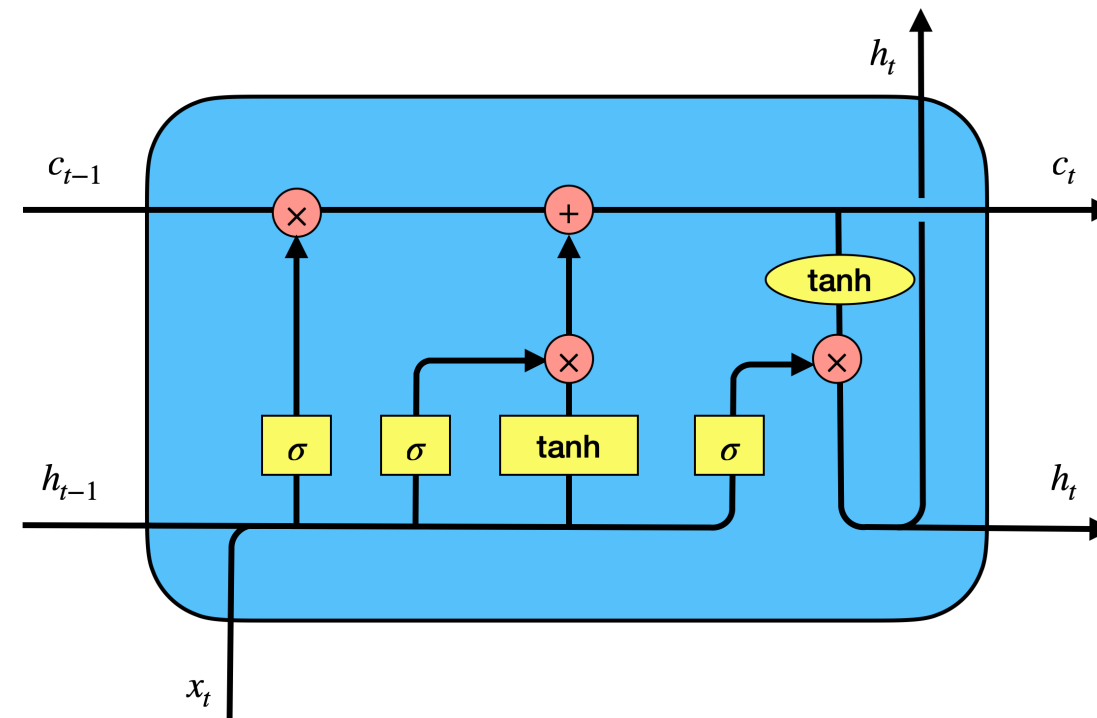
- $f_t = \sigma(W_{hf}h_{t-1} + W_{xf}x_t + b_f)$
- $i_t = \sigma(W_{hi}h_{t-1} + W_{xi}x_t + b_i)$
- $o_t = \sigma(W_{ho}h_{t-1} + W_{xo}x_t + b_o)$
- $\tilde{C}_t = \tanh(W_{hc}h_{t-1} + W_{xc}x_t + b_c)$
- $C_t = f_t \odot C_{t-1} + i_t \odot \tilde{C}_t$
- $h_t = o_t \odot \tanh(C_t)$

LSTM

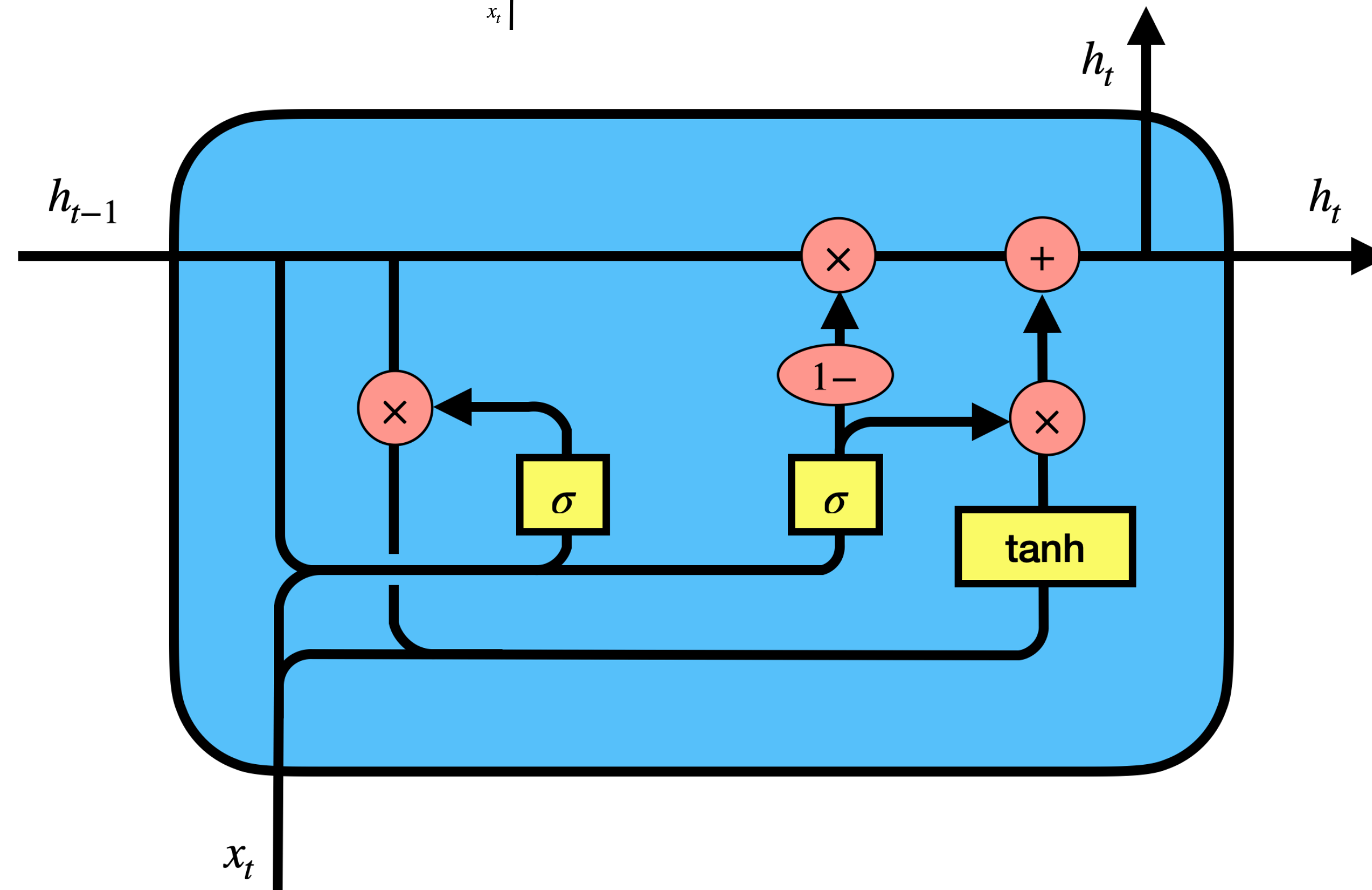


GRU (Gated Recurrent Unit)

LSTM

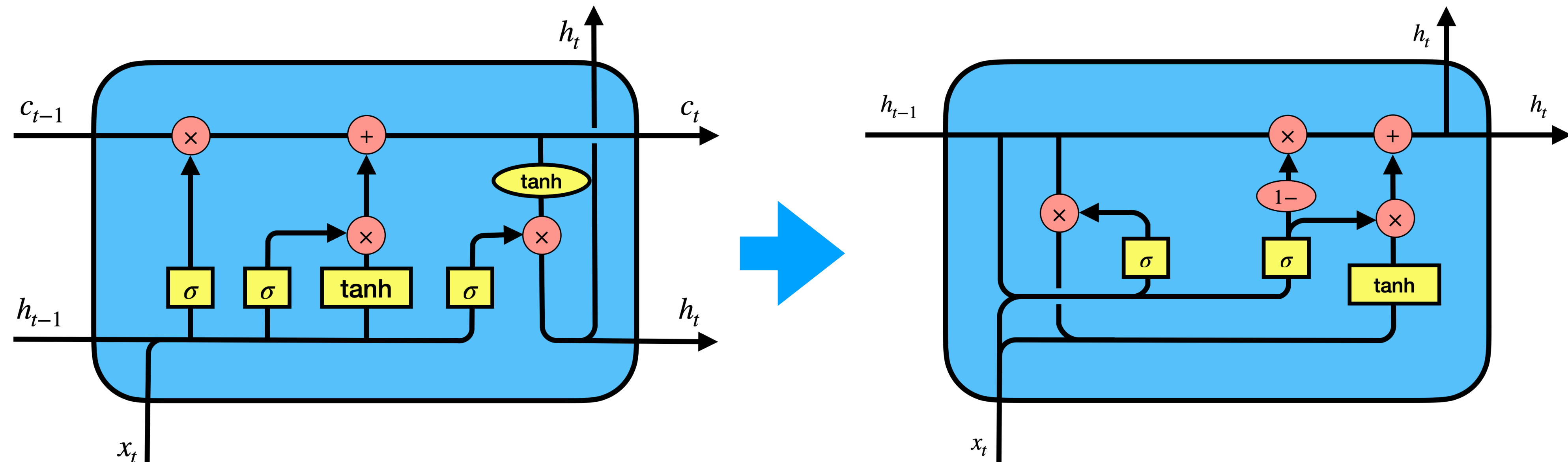


GRU



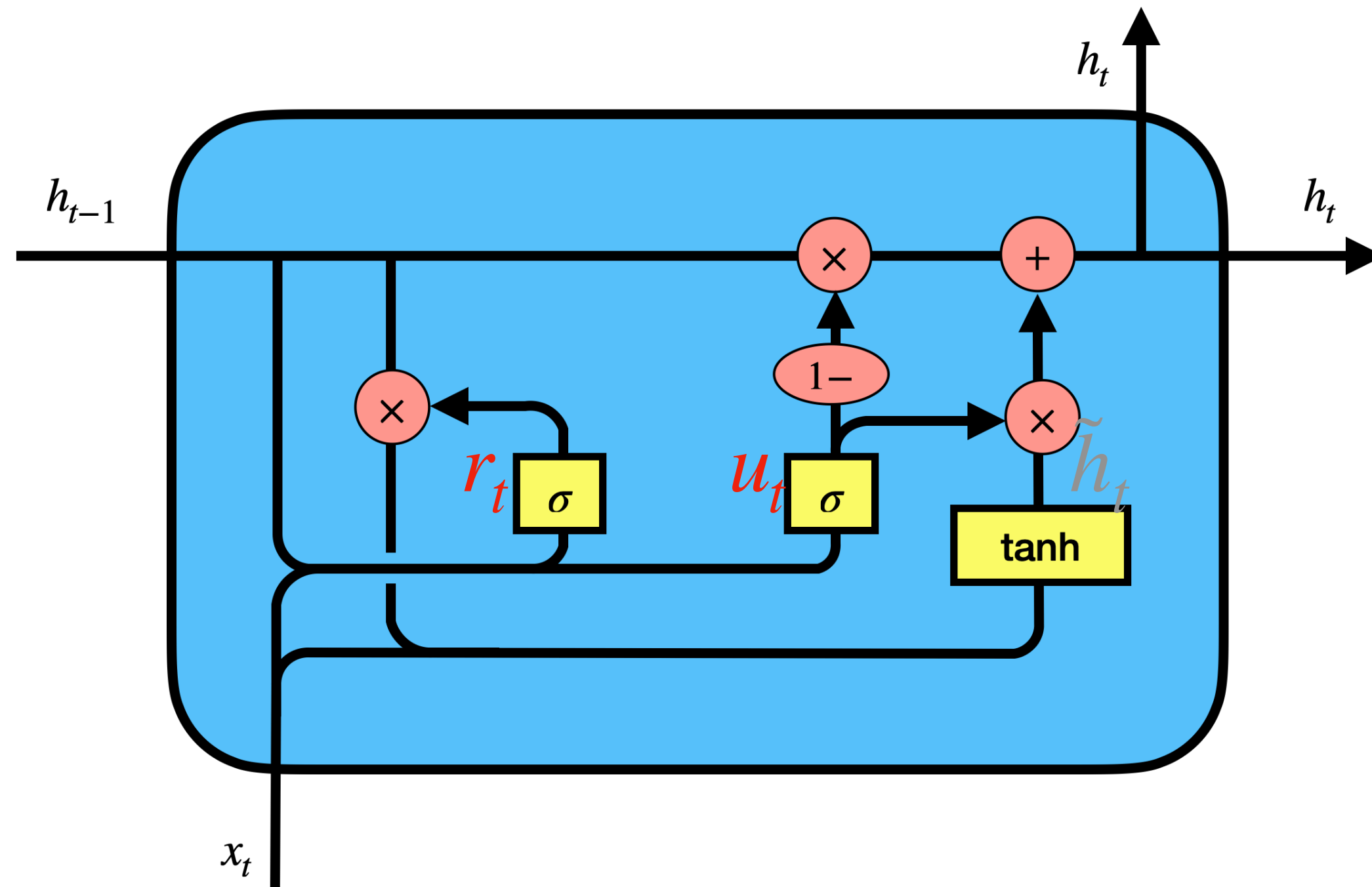
What is GRU

- LSTM보다 단순한 구조 이면서도 긴 데이터를 잘 처리함
 - Memory cell을 사용 안함
 - Gate 숫자를 2개로 줄임



GRU (gates)

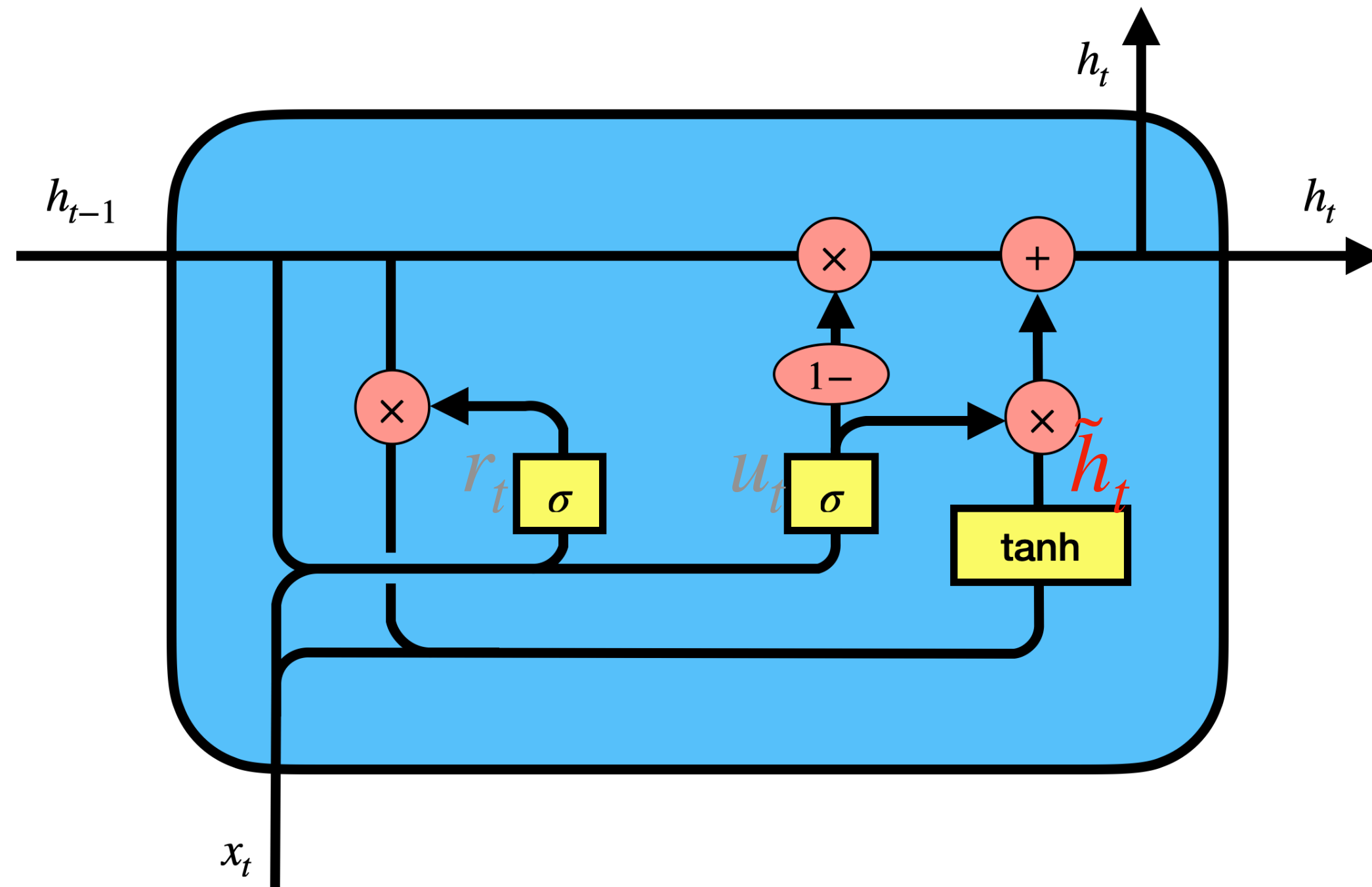
- Reset Gate: 이전 시점 h_{t-1} 의 정보를 얼마나 사용할지 결정
- Update Gate: 현재 시점의 \tilde{h}_t 의 정보를 얼마나 사용할지 결정



- $r_t = \sigma(W_{hr}h_{t-1} + b_{hr} + W_{xr}x_t + b_{xr})$
- $u_t = \sigma(W_{hz}h_{t-1} + b_{hz} + W_{xz}x_t + b_{xz})$
- $\tilde{h}_t = \tanh(r \odot (W_{hg}h_{t-1} + b_{hg}) + W_{xg}x_t + b_{xg})$
- $h_t = u_t \odot h_{t-1} + (1 - u_t) \odot \tilde{h}_t$

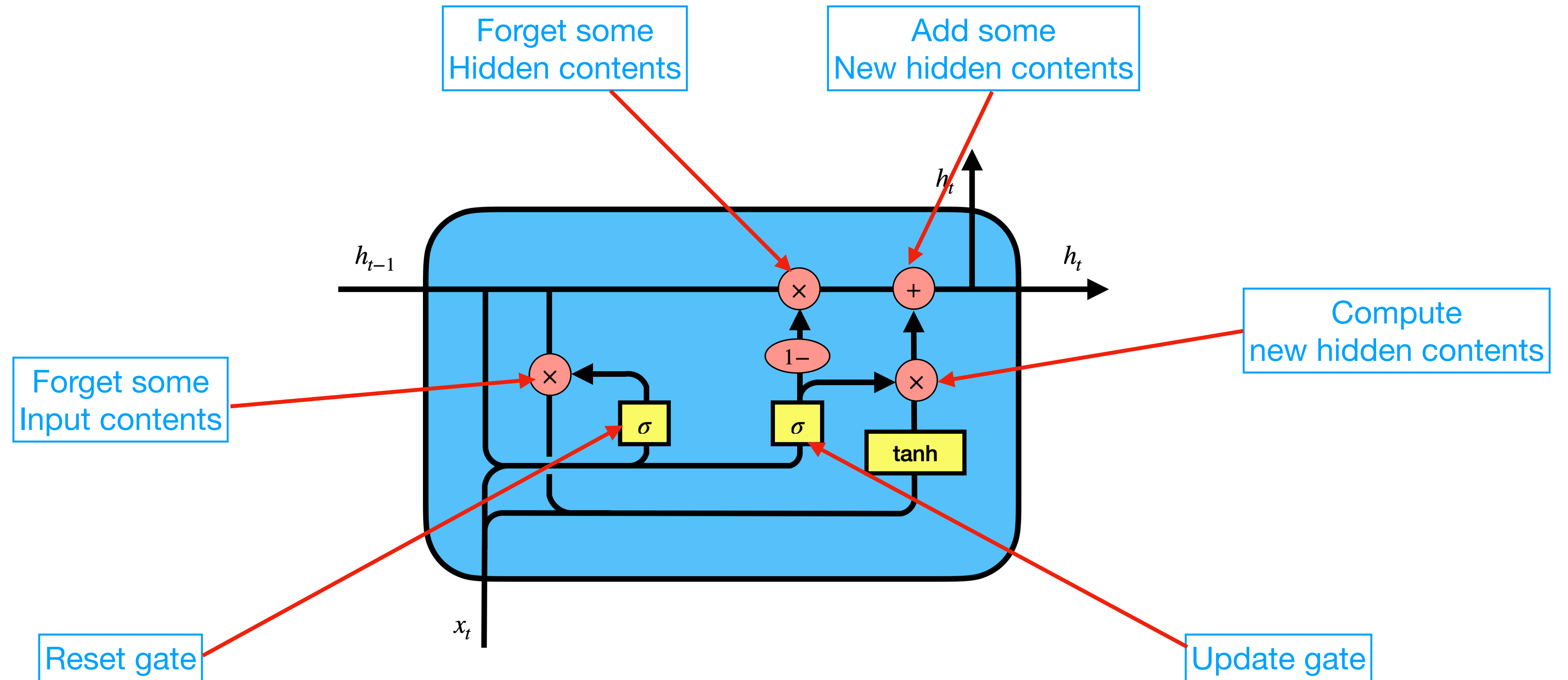
GRU (hidden)

- New Hidden content: 현재 시점의 Hidden state 값
- Hidden state: 현재 시점의 출력 값



- $r_t = \sigma(W_{hr}h_{t-1} + b_{hr} + W_{xr}x_t + b_{xr})$
- $u_t = \sigma(W_{hz}h_{t-1} + b_{hz} + W_{xz}x_t + b_{xz})$
- $\tilde{h}_t = \tanh(r \odot (W_{hg}h_{t-1} + b_{hg}) + W_{xg}x_t + b_{xg})$
- $h_t = u_t \odot h_{t-1} + (1 - u_t) \odot \tilde{h}_t$

GRU



감사합니다.