RNN

2022.09.01 / 7기 김채은

0. 목차

1. RNN

- Recurrent Neural Networks
- RNN for Sentiment Classification
- RNN: Backprop

2. LSTM

- Long Short-Term Memory
- Cell State
- Forget Gate
- Input Gate
- Update
- Output Gate
- 그래서 어떻게 LSTM이 기울기 소실을 막을 수 있는데?

0. 목차

3. GRU

- LSTM의 간소화 버전, GRU
- GRU 구조
- Reset Gate
- Update Gate
- GRU Backpropagation

4. seq2seq

- Sequence to Sequence
- Teacher Forcing

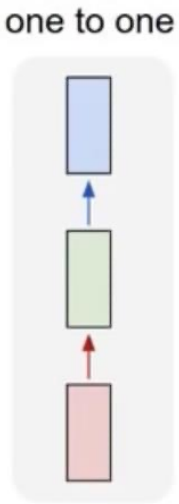
5. Summary

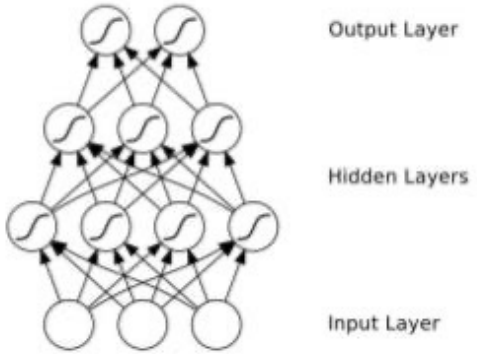
1. RNN

1. RNN

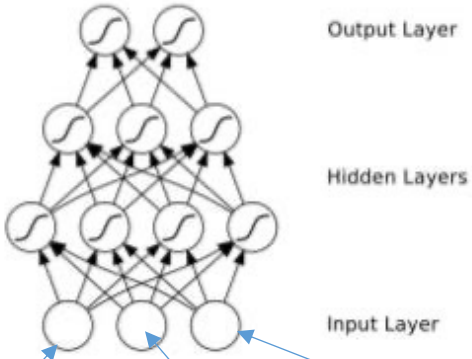
Recurrent Neural Networks 순환 신경망

vs Feed Forward Neural Network





Sequence Data: 시점에 따라 달라지는 데이터
ex) 자연어처리, 시계열 데이터, 영상 처리



1. RNN

Recurrent Neural Networks 순환 신경망

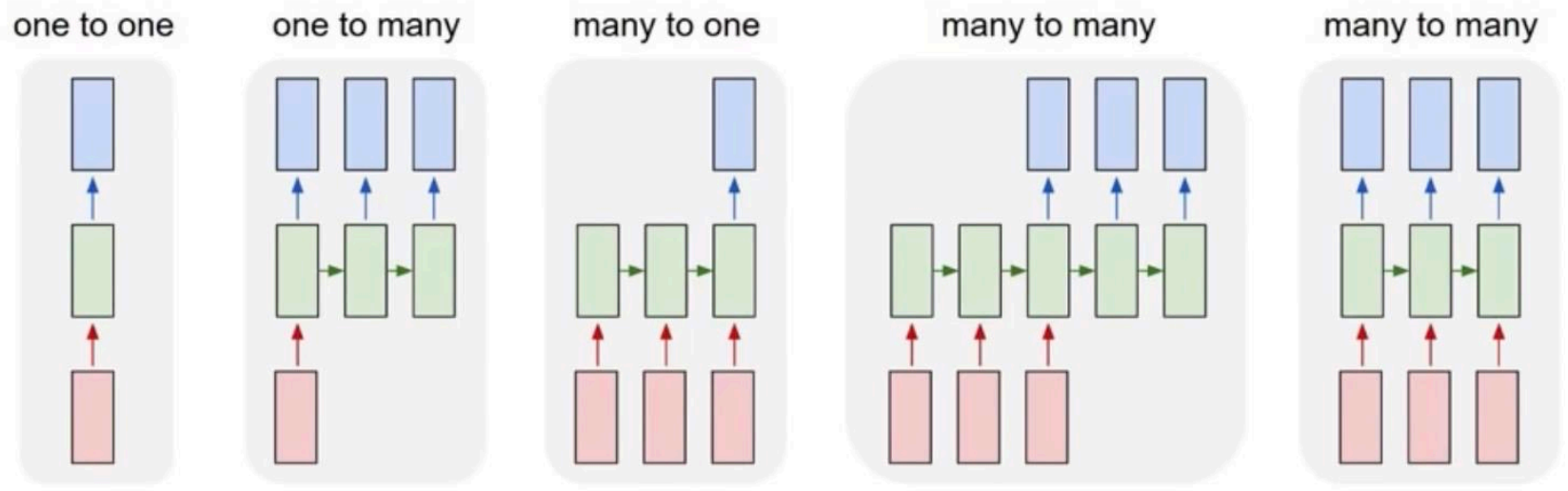


Image to category (Vanilla NN)

Image captioning

Sentiment analysis

Machine translation

POS(품사) tagging

1. RNN

Recurrent Neural Networks 순환 신경망

In each neuron of RNN, the output of previous time step is fed as input of the next time step.

We can process a sequence of vectors **x** by applying a **recurrence formula** at every time step:

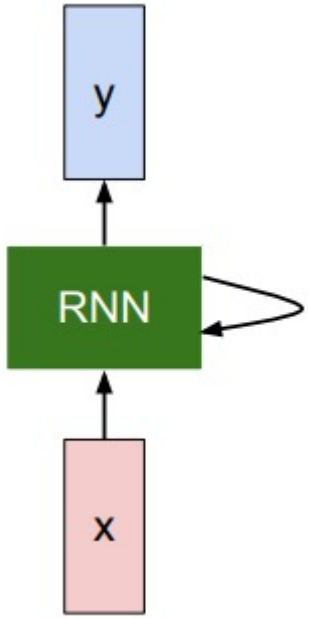
$$h_t = f_W(h_{t-1}, x_t)$$

new state

old state

input vector at some time step

some function with parameters W

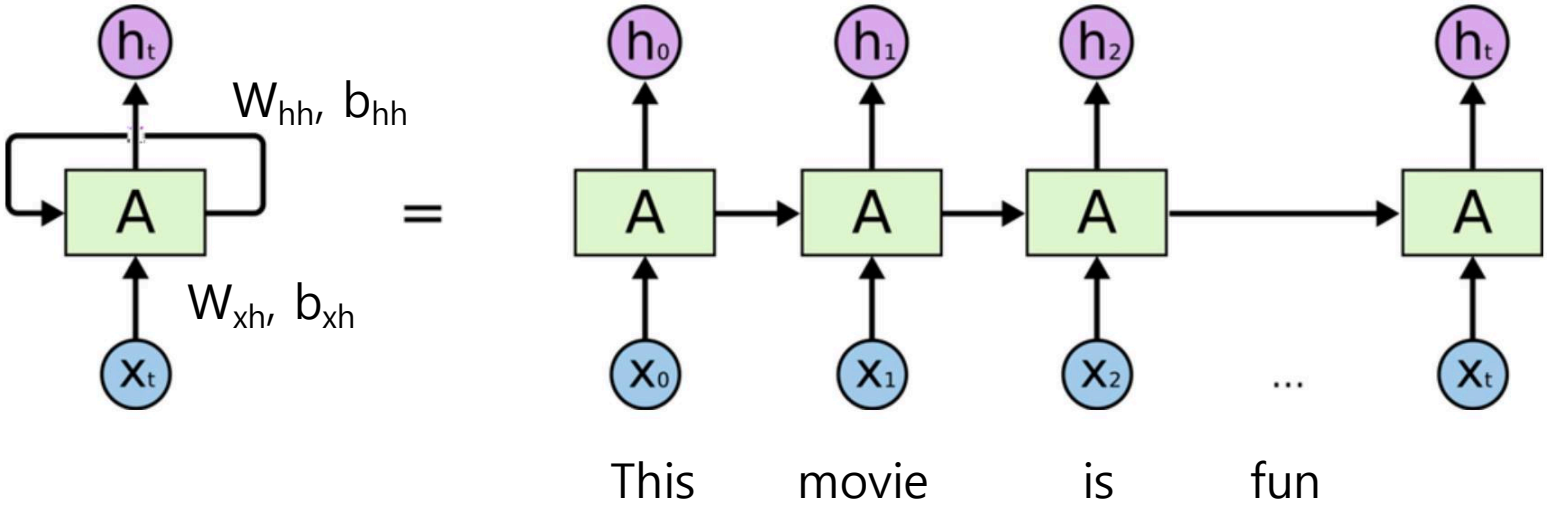


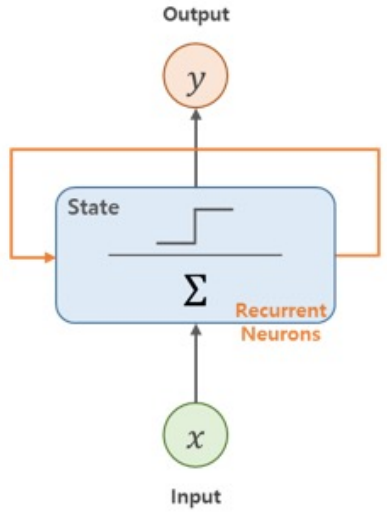
Notice!!!
The same function and the same set of parameters are used at every time step.

1. RNN

Recurrent Neural Networks 순환 신경망

$x_t \in \mathrm{R}^{|V|}$, $h_t \in \mathrm{R}^d$ where $|V| \gg d$



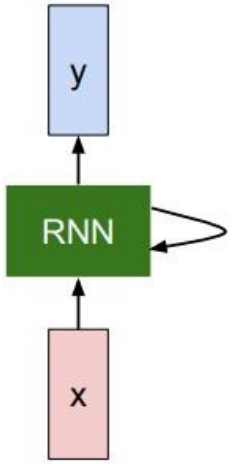


$$h_t = f(x_t, h_{t-1}; \theta) = \tanh(W_{xh}x_t + b_{xh} + W_{hh}h_{t-1} + b_{hh})$$
$$\text{where } \theta = \{W_{xh}, b_{xh}, W_{hh}, b_{hh}\}$$

1. RNN

Recurrent Neural Networks 순환 신경망

The state consists of a single *"hidden"* vector **h**:

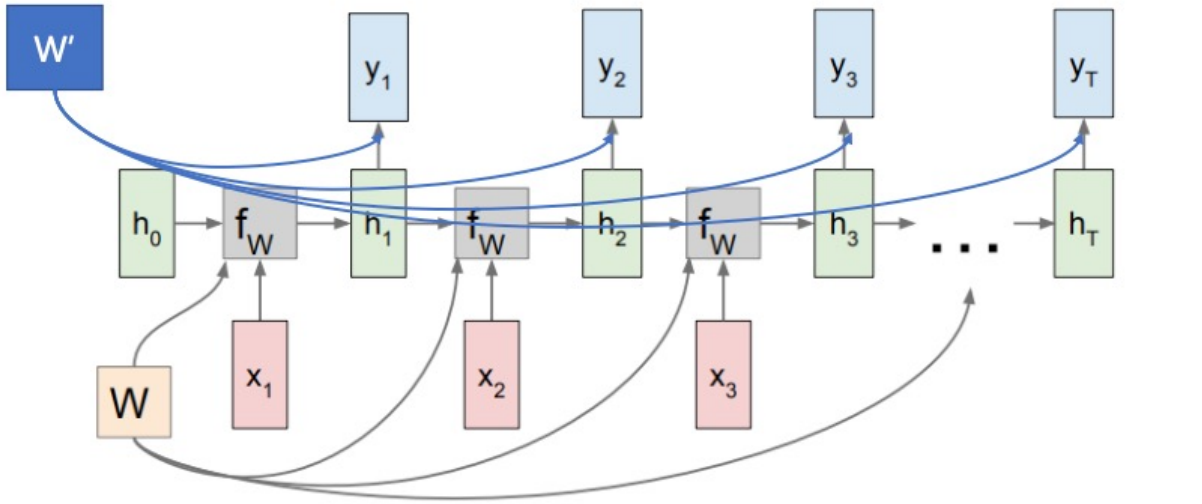


$$h_t = f_W(h_{t-1}, x_t)$$

$$h_t = \tanh(W_{hh}h_{t-1} + W_{xh}x_t)$$

$$y_t = W_{hy}h_t$$

$$y_t = W_{hy}h_t$$



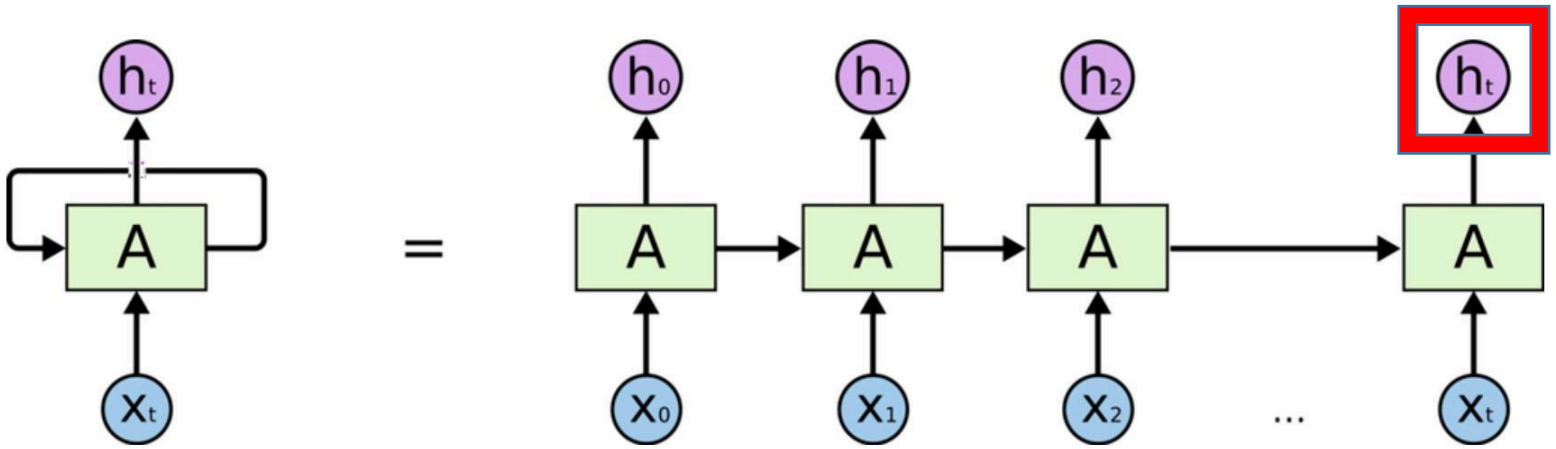
1. RNN

RNN for Sentiment Classification

This movie is fun…
[one-hot representation]

[hidden representation]

[class probabilities]



$x_t \in R^{|v|}$

$h_t \in R^{|v|}$

$\hat{y} = \text{sigmoid}(W_{hy}h_t)$

where $W_{hy} \in R^{1 \times d}$

Class $\in$ {positive, negative}

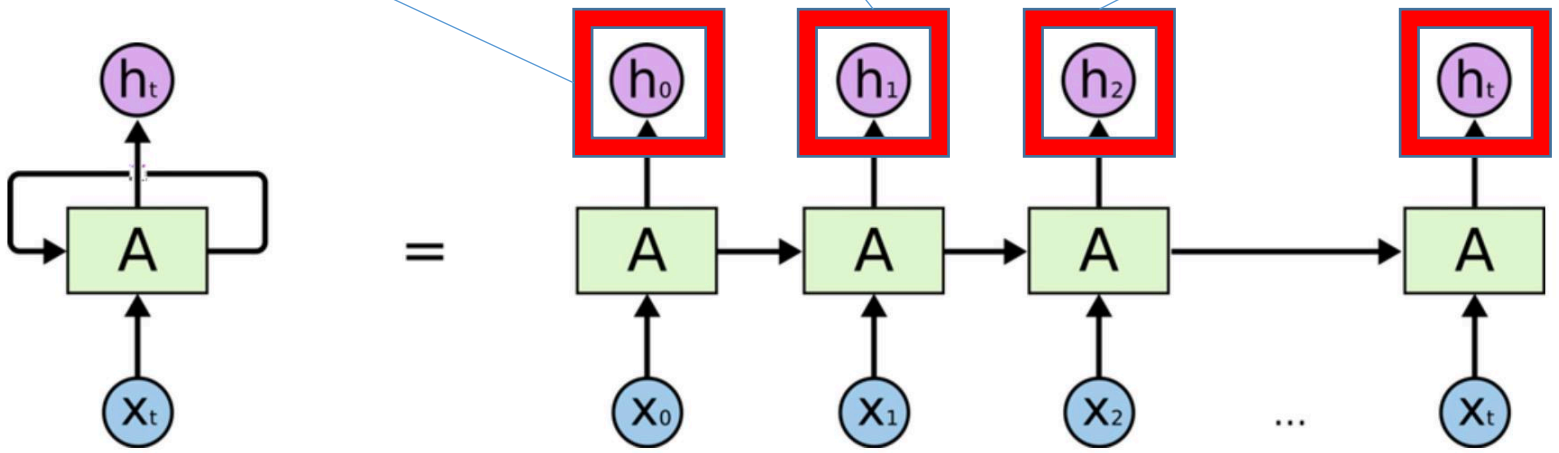
1. RNN

RNN for POS Tagging

$\hat{y}_0 = \text{sigmoid}(W_{hy}h_0)$ where $W_{hy} \in R^{36\times d}$

$\hat{y}_1 = \text{sigmoid}(W_{hy}h_1)$ where $W_{hy} \in R^{36\times d}$

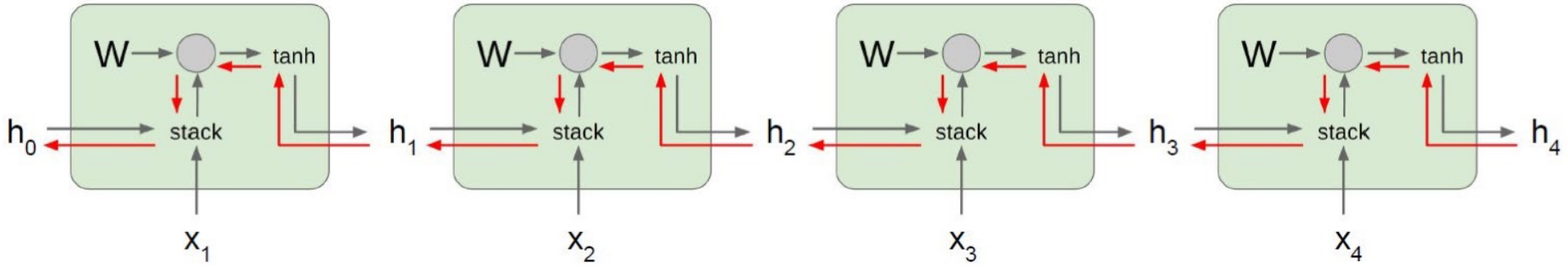
$\hat{y}_2 = \text{sigmoid}(W_{hy}h_2)$ where $W_{hy} \in R^{36\times d}$



Class ∈ {1.NN, 2.VB, … , 36.JJ}

1. RNN

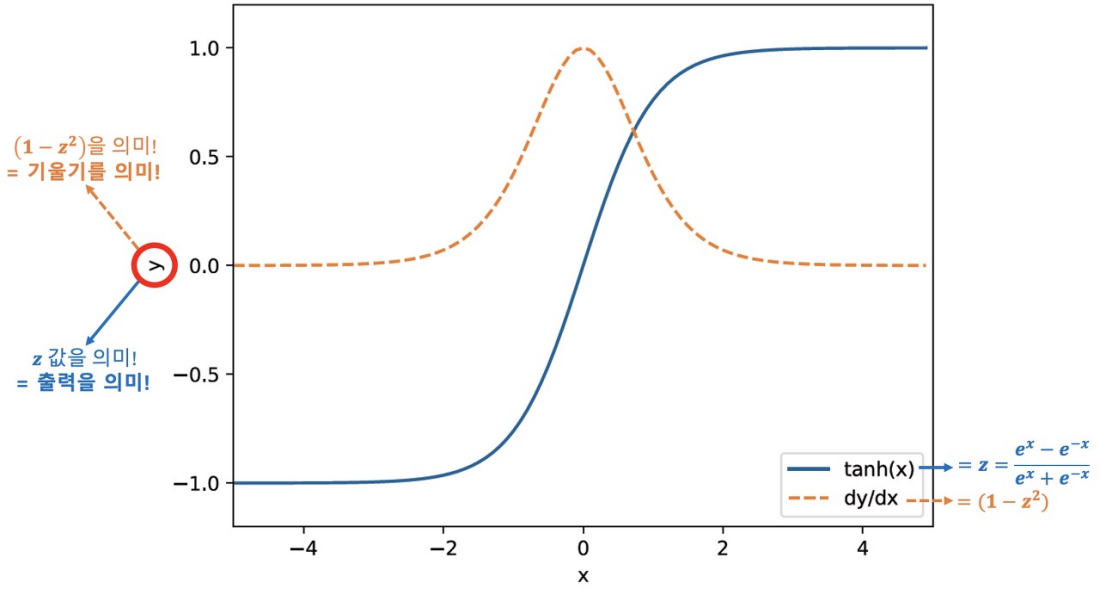
RNN: Backprop



h_0의 gradient를 구하려면 수많은 W들이 곱해져야함!

W의 Largest singular value > 1: Exploding gradients
W의 Largest singular value < 1: Vanishing gradients

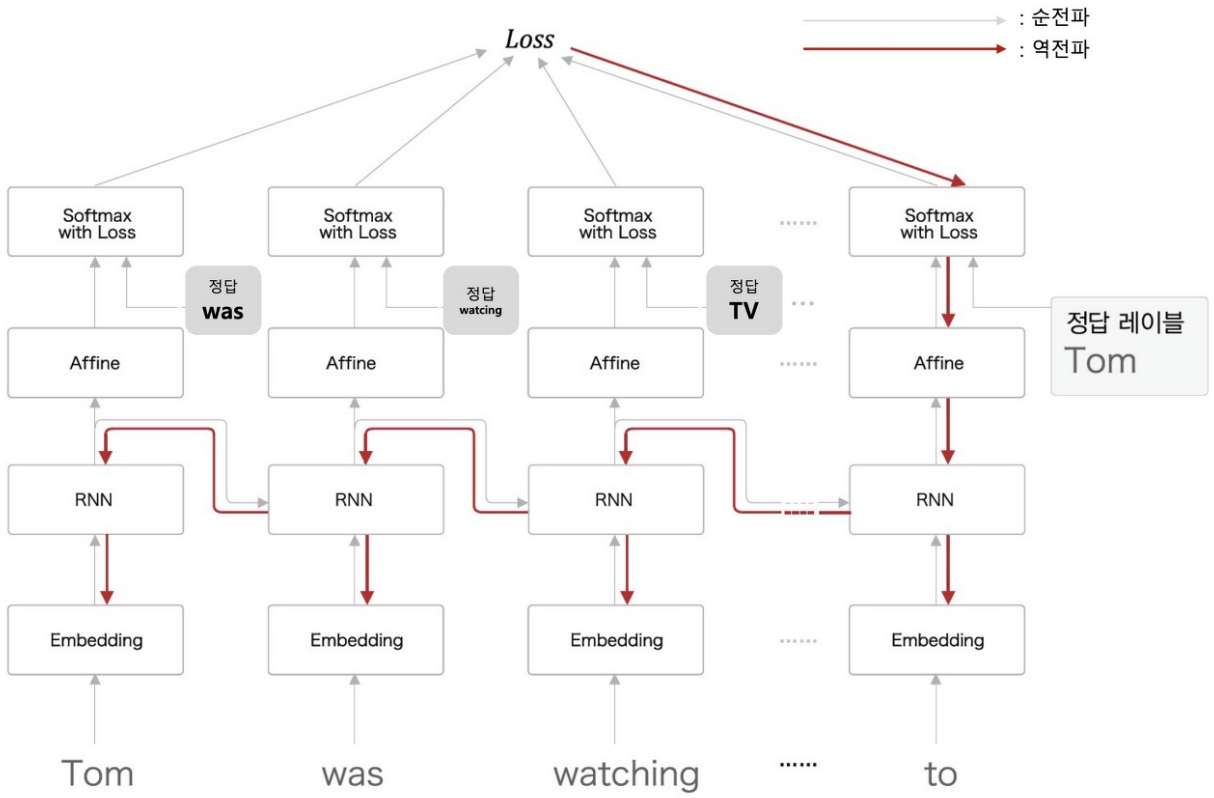
+) tanh의 기울기 소실 문제



RNN: Backprop

그림 6-3 "?"에 들어갈 단어는?: (어느 정도의) 장기 기억이 필요한 문제의 예

Tom was watching TV in his room. Mary came into the room. Mary said hi to ?



정답인 Tom을 맞추기 위해 학습 과정에서
예측값과 정답 간의 차이를 작게 만드는 방향으로
기울기 값을 역방향으로 전달하는
역전파과정을 수행할 것이다.

2. LSTM

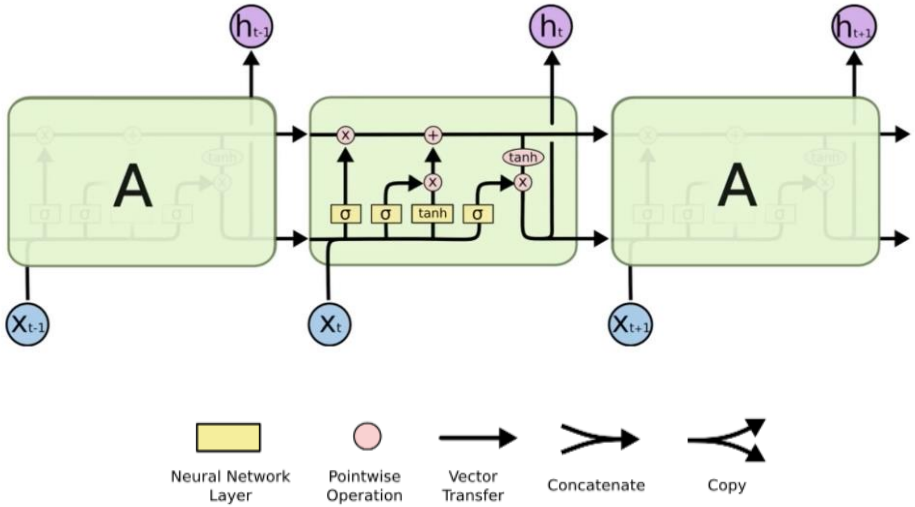
응 기울기 소실해봐~
넘기면 그만이야

2. LSTM

Long Short-Term Memory

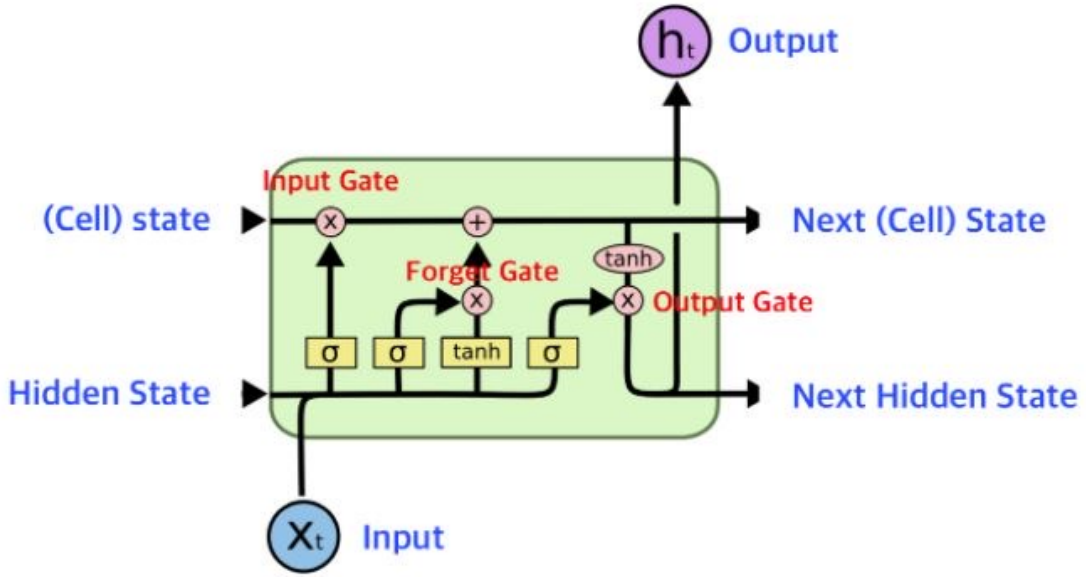
순환 신경망의 문제: 은닉층을 거친 결과값을 재사용한다.
→ RNN의 기울기 소실 & 폭주 문제를 해결하기 위해 등장!

핵심: LSTM은 결과값이 다음 시점으로 넘어갈 때 결과값을 넘길지 말지 결정하는 단계가 추가된다!



2. LSTM

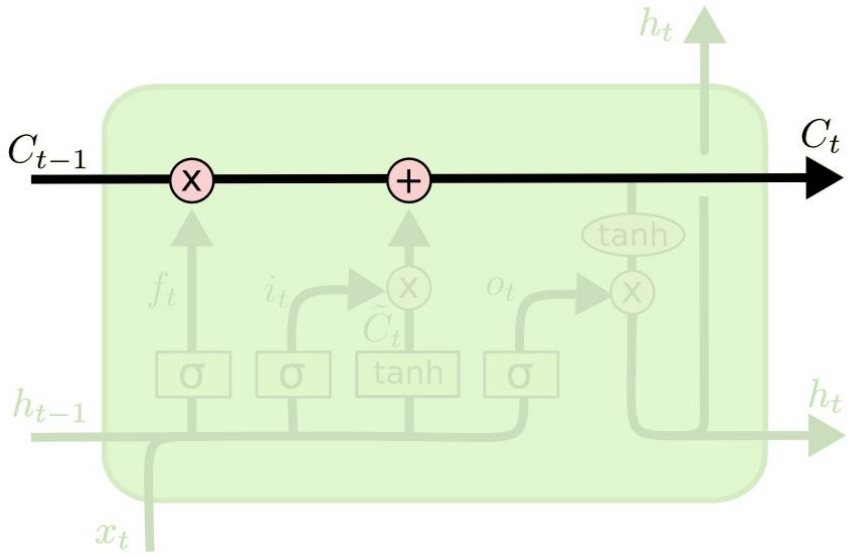
Long Short-Term Memory



- 망각 게이트 Forget Gate: 과거의 기억을 남길 비율을 조정함
- 입력 게이트 Input Gate: 새로운 기억을 추가하는 비율을 조정함
- 출력 게이트 Output Gate: 기억 셀 내부를 출력에 반영하는 비율을 조정함

2. LSTM

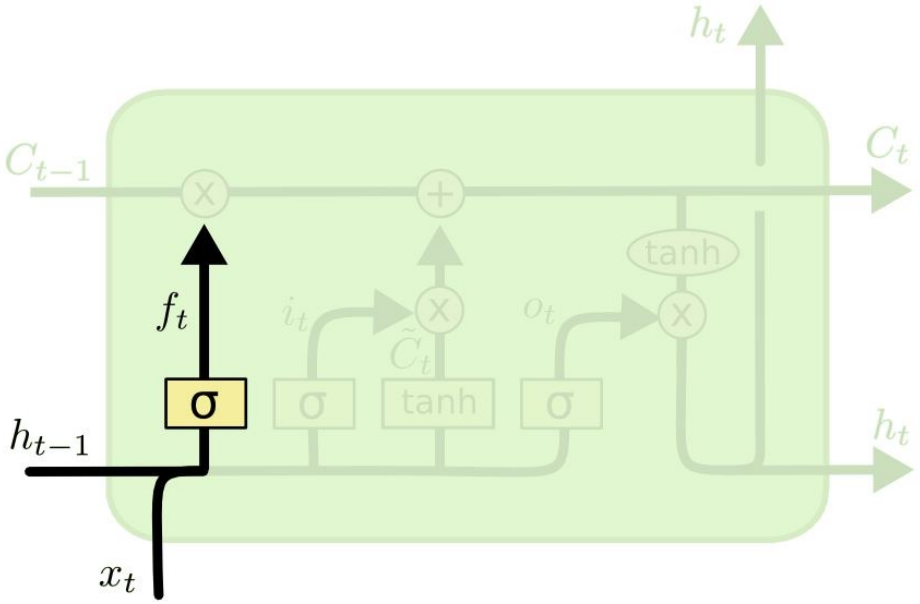
Cell State



정보가 전혀 바뀌지 않고 그대로 흐르게 하는 LSTM의 핵심 부분!
- State가 오래 경과하더라도 gradient가 잘 전파된다
- gate에 의해 정보가 추가되거나 제거되며, gate는 어떤 정보를 유지하고 버릴지 training

2. LSTM

Forget Gate

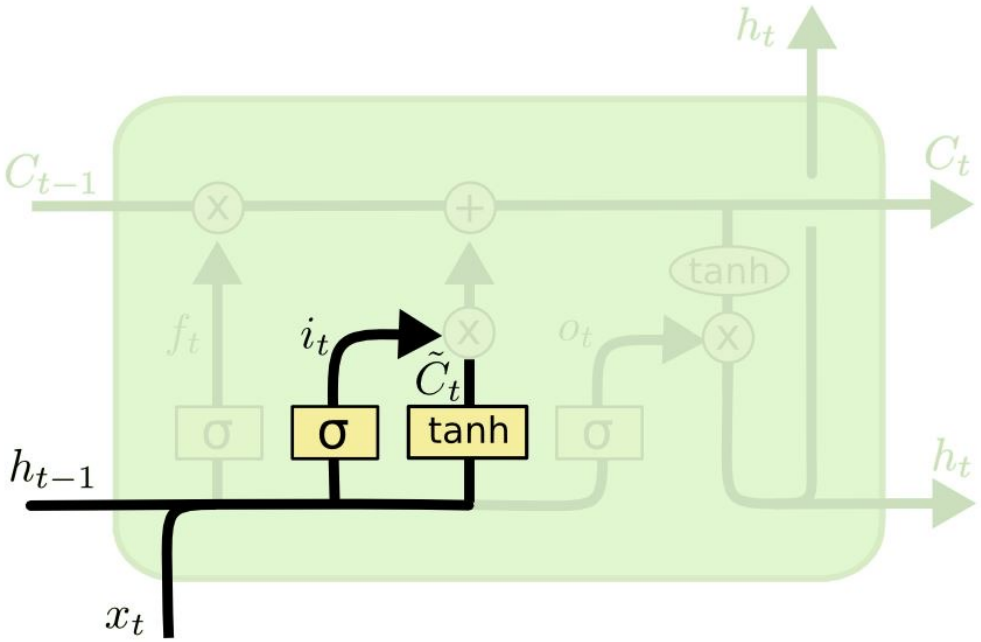


$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$$

과거의 기억을 남길 비율을 조정함
- h_{t-1}과 x_t를 받아 [0,1] 사이의 값을 C_{t-1}에 전달
- C_{t-1} == 1: 모든 정보를 보존해라!
- C_{t-1} == 0: 죄다 갖다 버려라!

2. LSTM

Input Gate



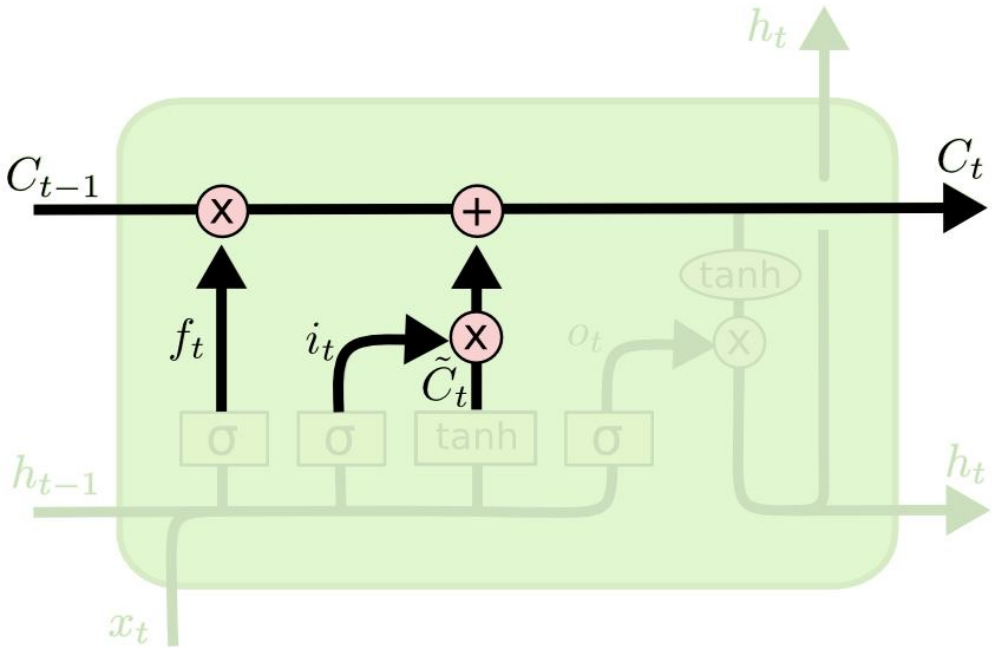
$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \; + \; b_i\right)$$

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

새로운 기억을 추가하는 비율을 조정함
- 현재 cell state값에 얼마나 더할지?
- 정말 필요한 정보만을 가져오자!

2. LSTM

Update

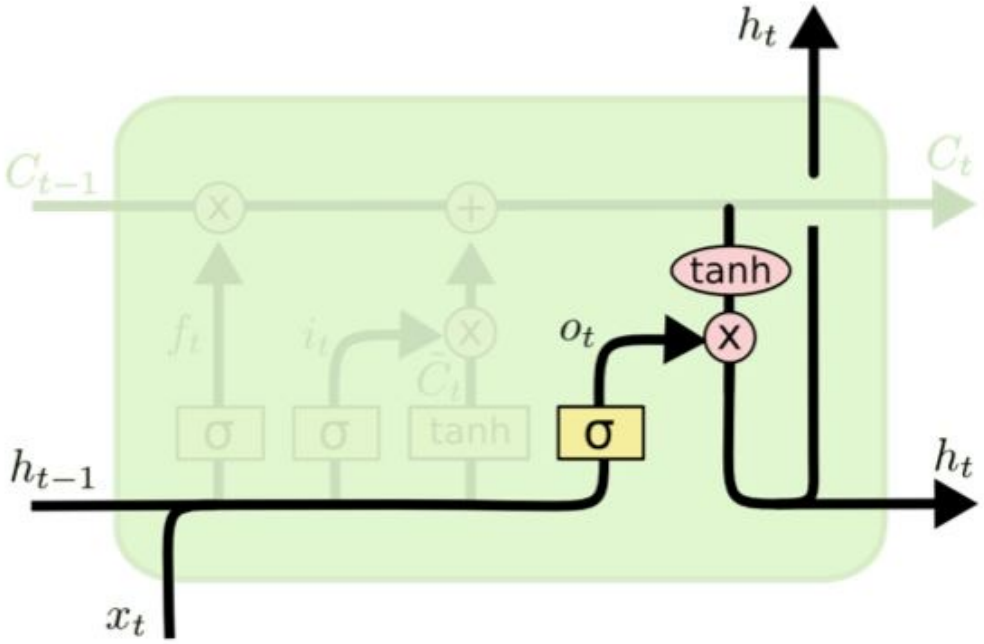


$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

$C_{t-1} \rightarrow C_t$
Forget Gate: 얼마나 버릴지?
Input Gate: 얼마나 더할지?

2. LSTM

Output Gate



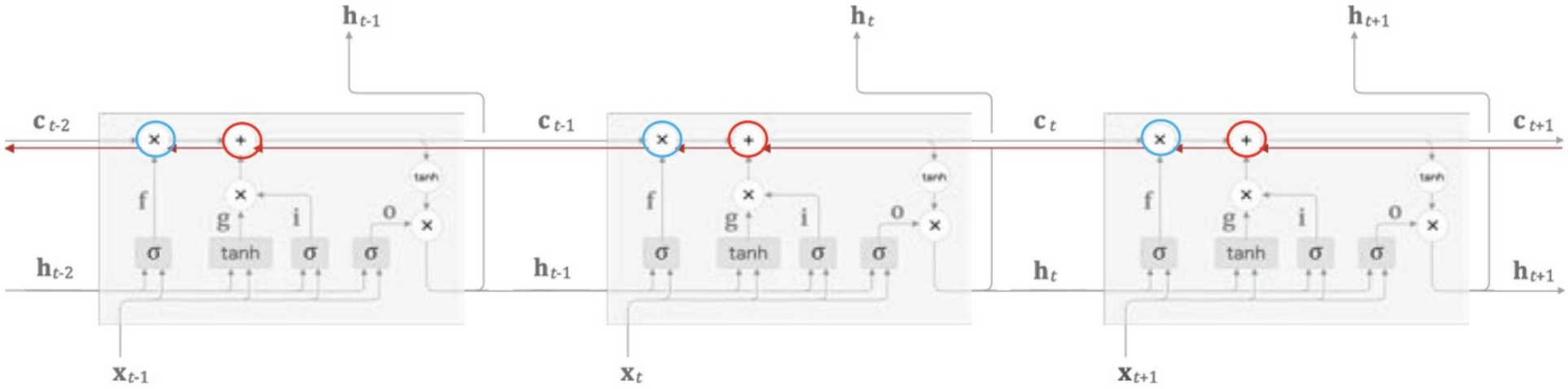
$$o_t = \sigma\left(W_o\left[h_{t-1}, x_t\right] + b_o\right)$$

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

기억 셀 내부를 출력에 반영하는 비율을 조정함

2. LSTM

그래서 어떻게 LSTM이 기울기 소실을 막을 수 있는데?



빨간색 동그라미는 뎃셈 연산
- 덧셈 연산은 이전으로부터 흘러들어오는 국소적인 미분값을 건드리지 않고 그냥 그대로 흘려 보낸다

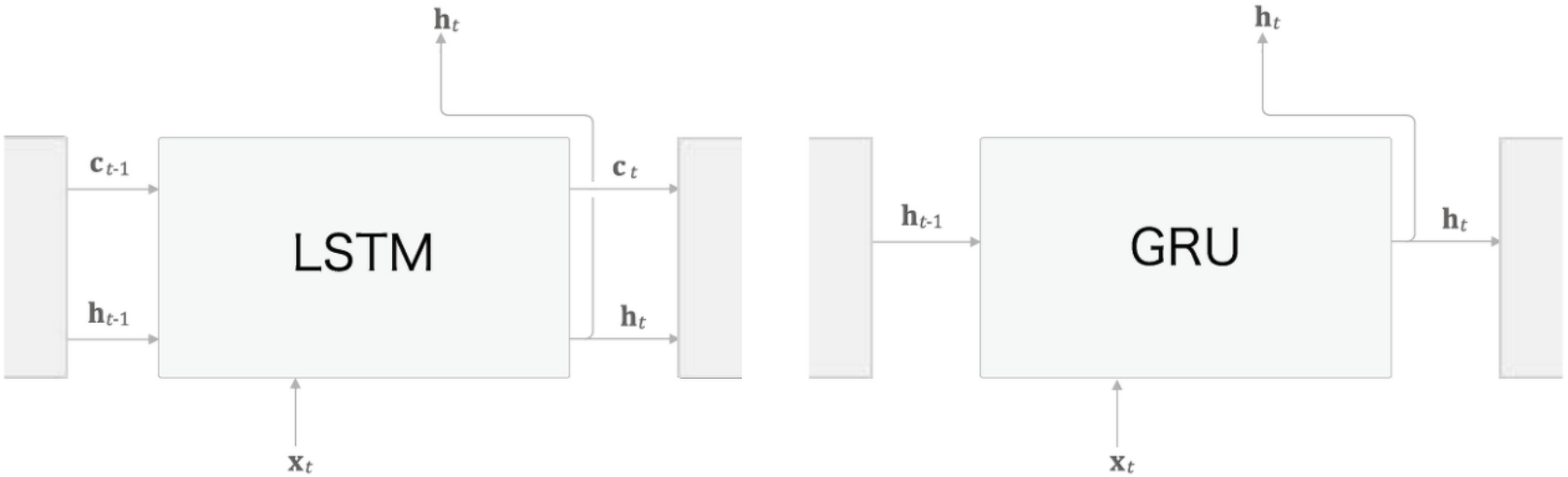
→ 기울기의 변화가 일어나지 않기 때문에 기울기 소실이 발생할 수 없다!

3. GRU

3. GRU

LSTM의 간소화 버전, GRU

그림 C-1 LSTM과 GRU 비교



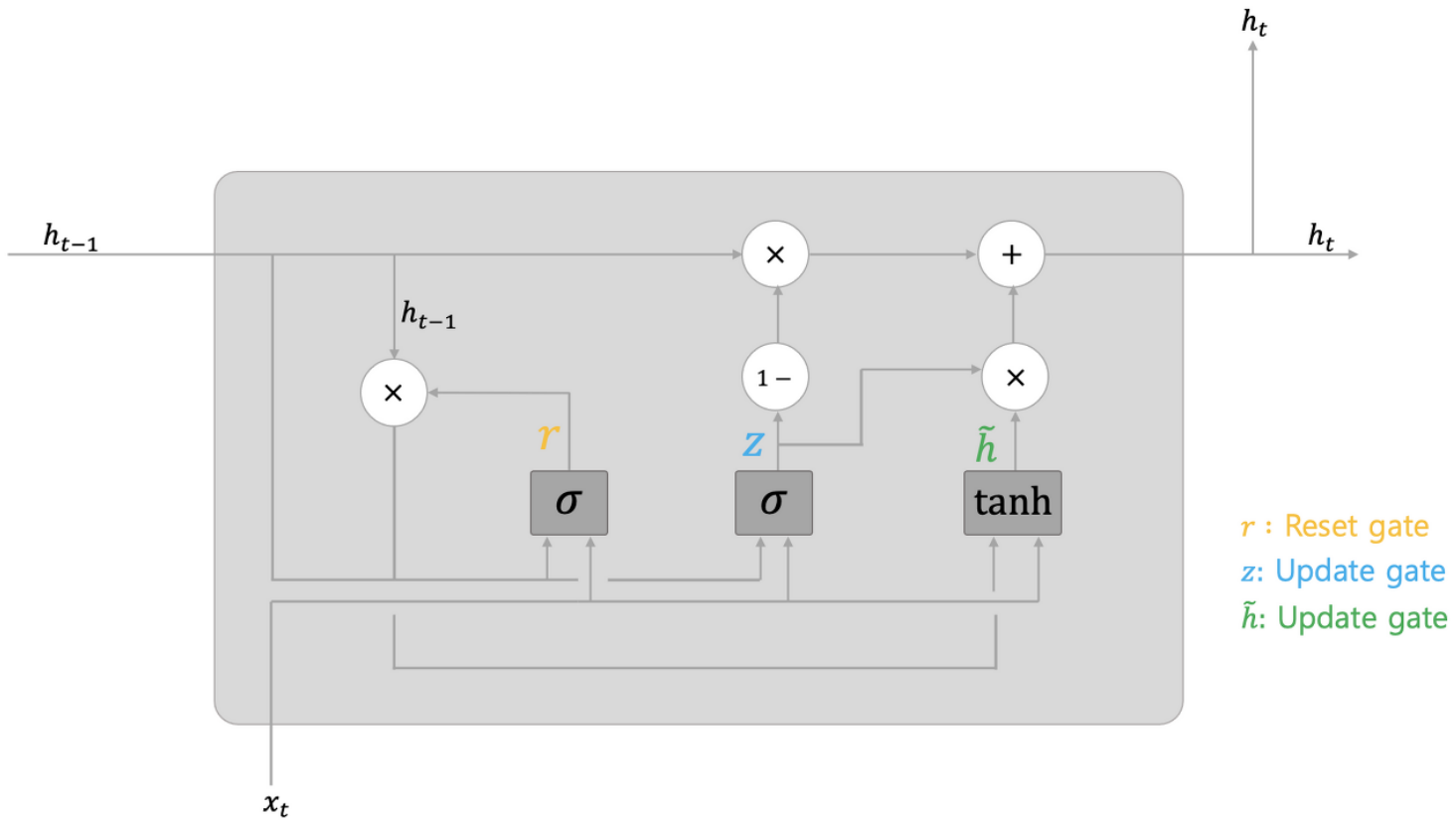
LSTM의 단점
학습할 파라미터가 많아지면,
계산이 오래걸린다

GRU
LSTM처럼 게이트의 기능은 유지하되,
LSTM보다 파라미터 개수는 줄이자!

기억 셀(C)이 사라지고 은닉 상태 벡터(h)만 사용

3. GRU

GRU 구조

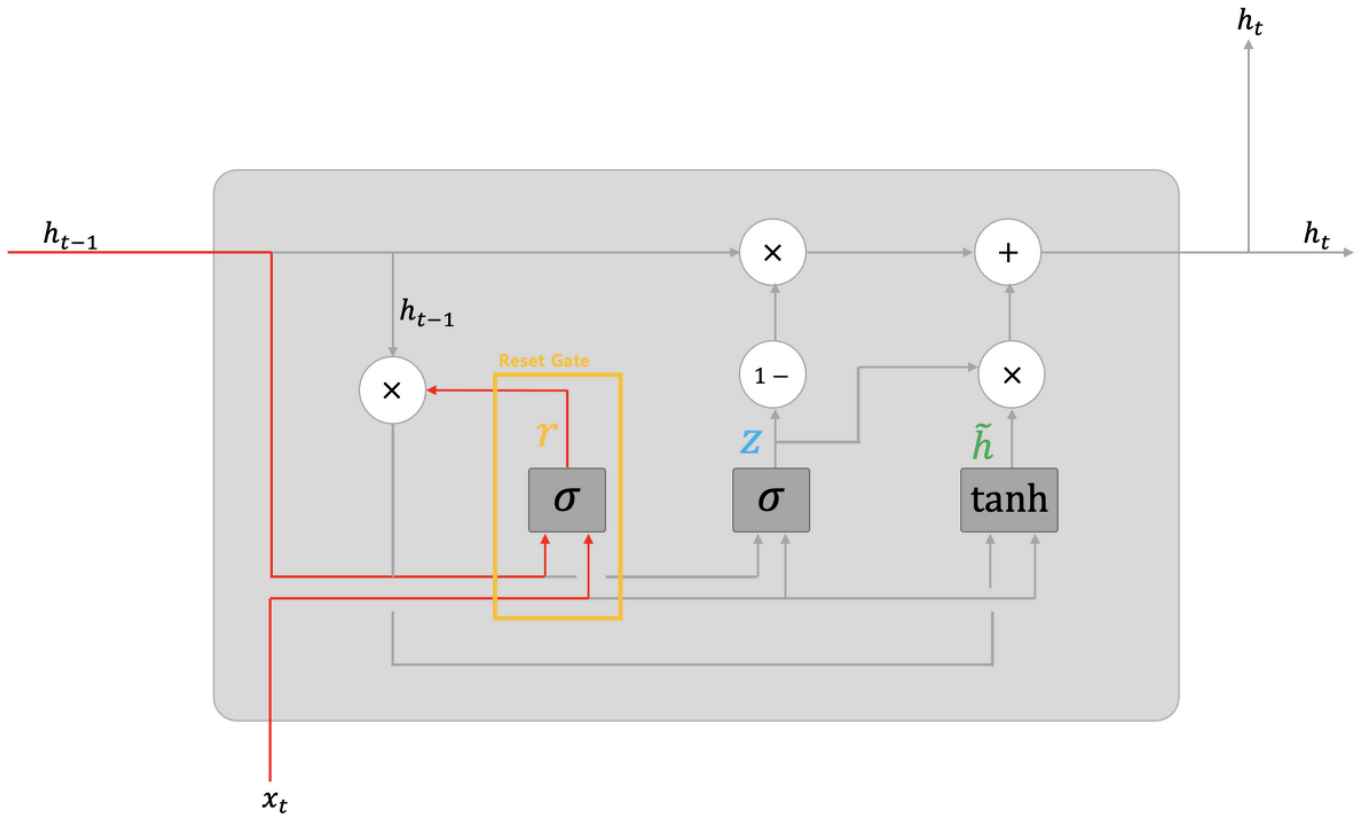


Reset Gate (← Output Gate)

Update Gate = Input Gate + Forget Gate

3. GRU

Reset Gate

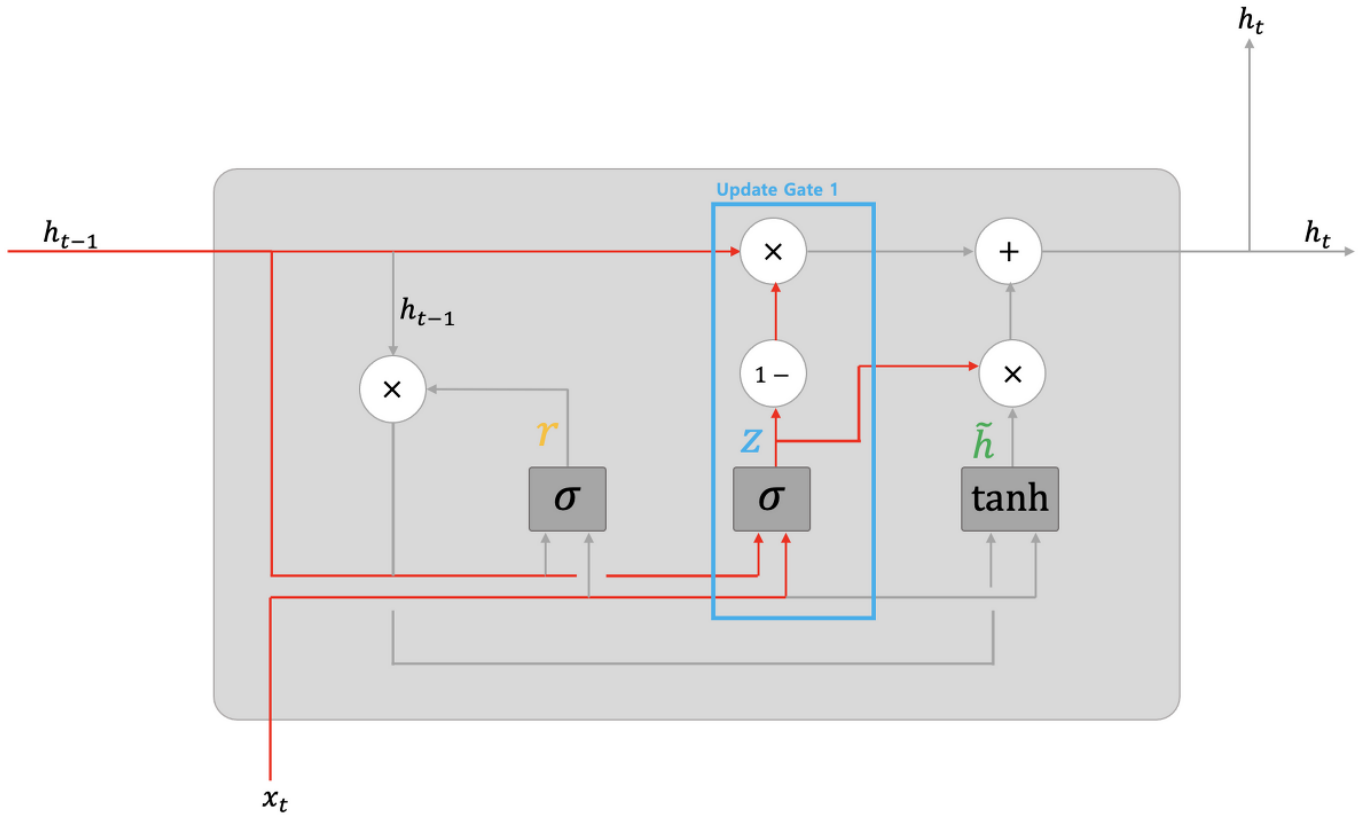


h_{t-1}을 얼마나 반영할지 결정
r == 0 : h_{t-1}을 모두 무시해!

$$r = \sigma(x_t W_x^{(r)} + h_{t-1} W_h^{(r)} + b^{(r)})$$

3. GRU

Update Gate 1 (Forget Gate 역할)



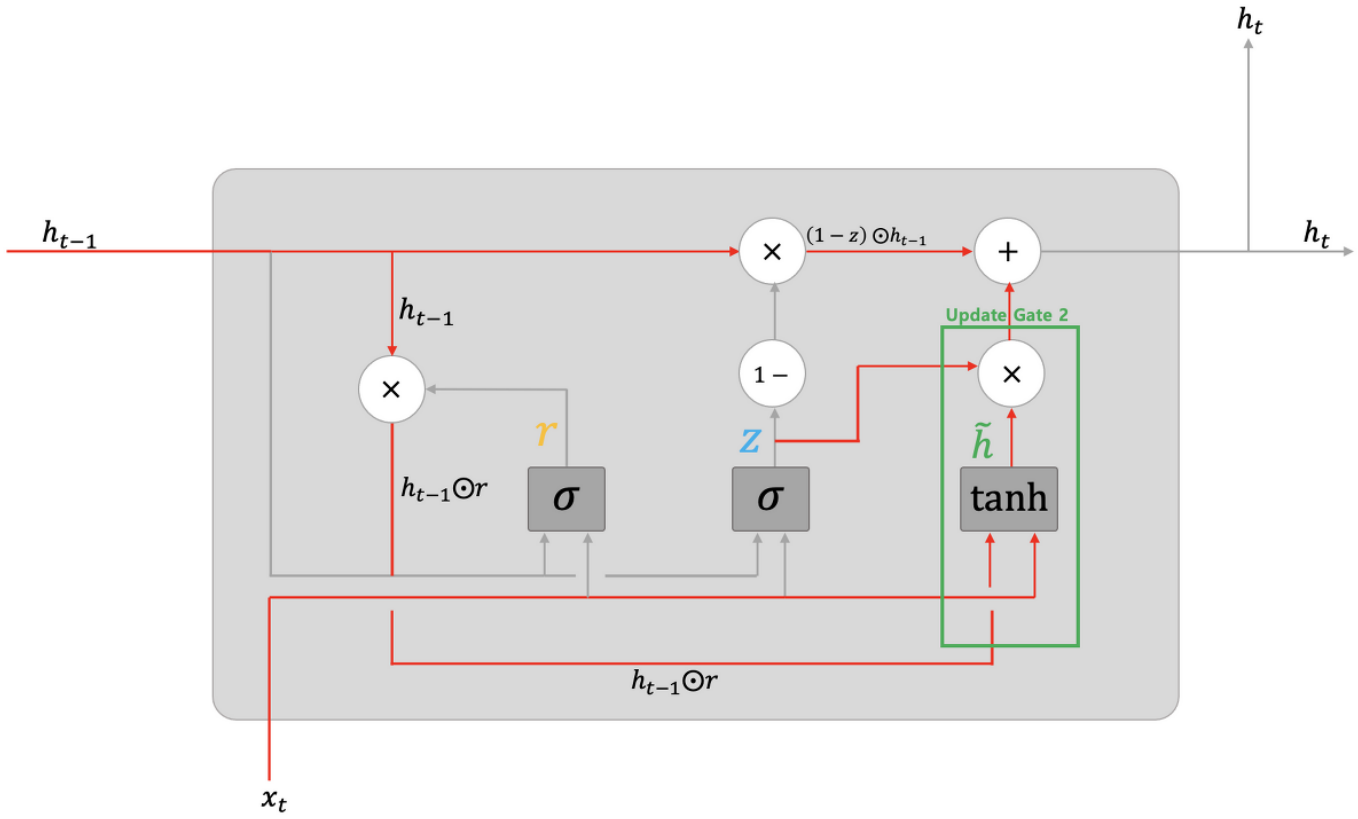
h_{t-1} 갱신: 불필요한 정보는 버리고 필요한 정보만 취사선택

$$z = \sigma(x_t W_x^{(r)} + h_{t-1} W_h^{(r)} + b^{(r)})$$

$$(1 - z) \odot h_{t-1}$$

3. GRU

Update Gate 2 (Input Gate 역할)

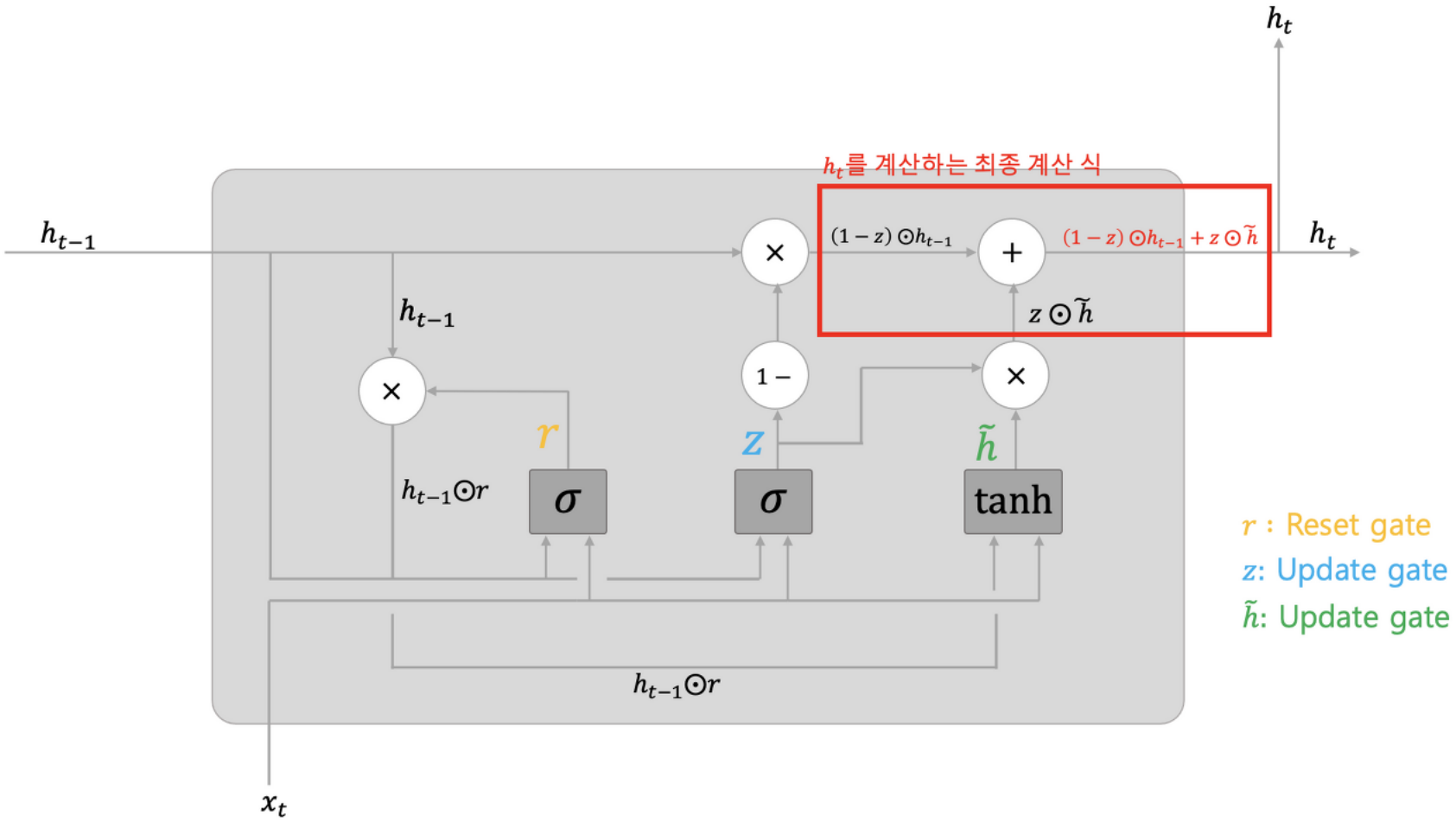


새로운 기억

$$\tilde{h} = tanh(x_t\, W_x \;\; + (h_{t-1}\odot r)W_h \;\; + \; b \;\;)$$

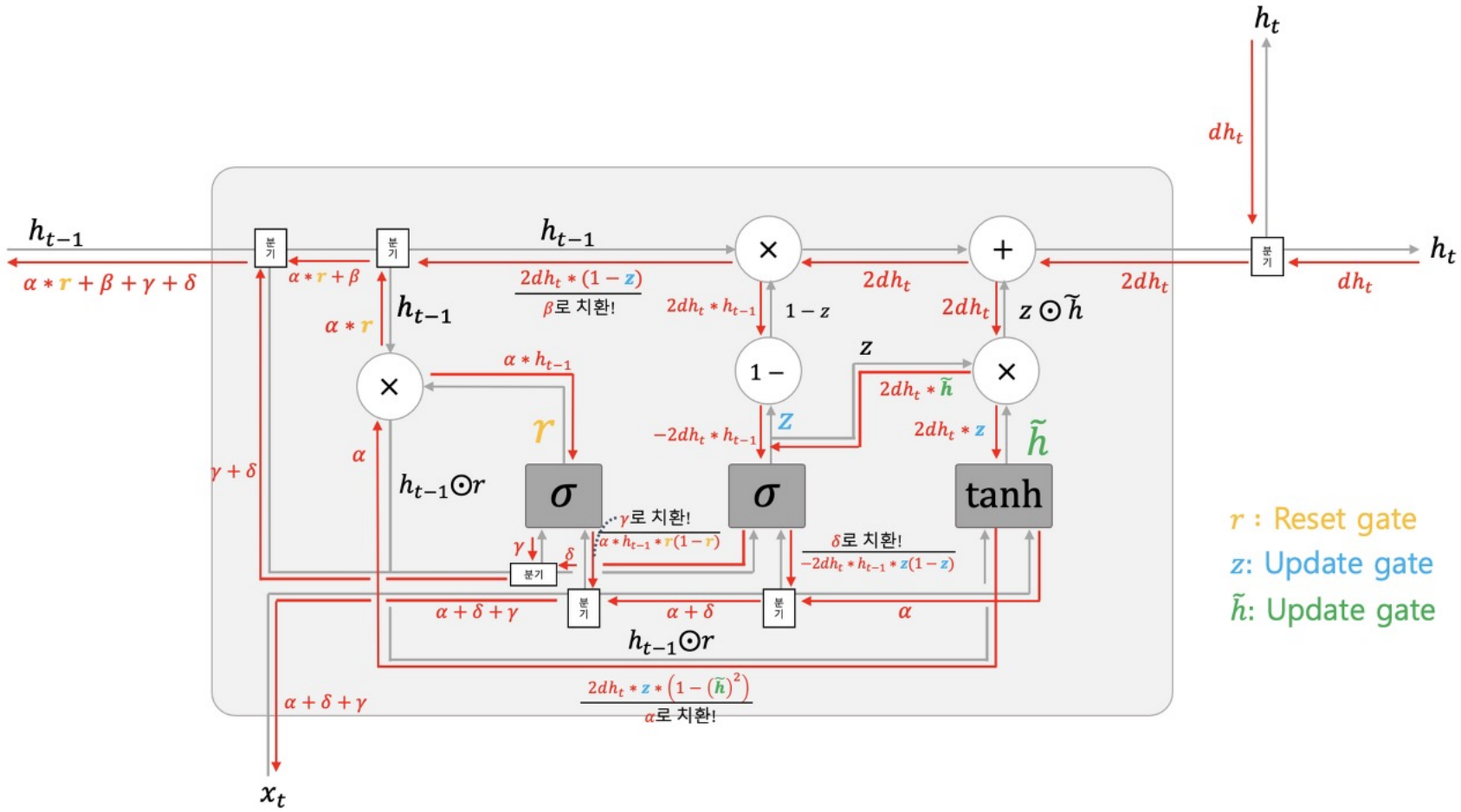
3. GRU

Update Gate h_t 최종 계산식



3. GRU

GRU Backpropagation



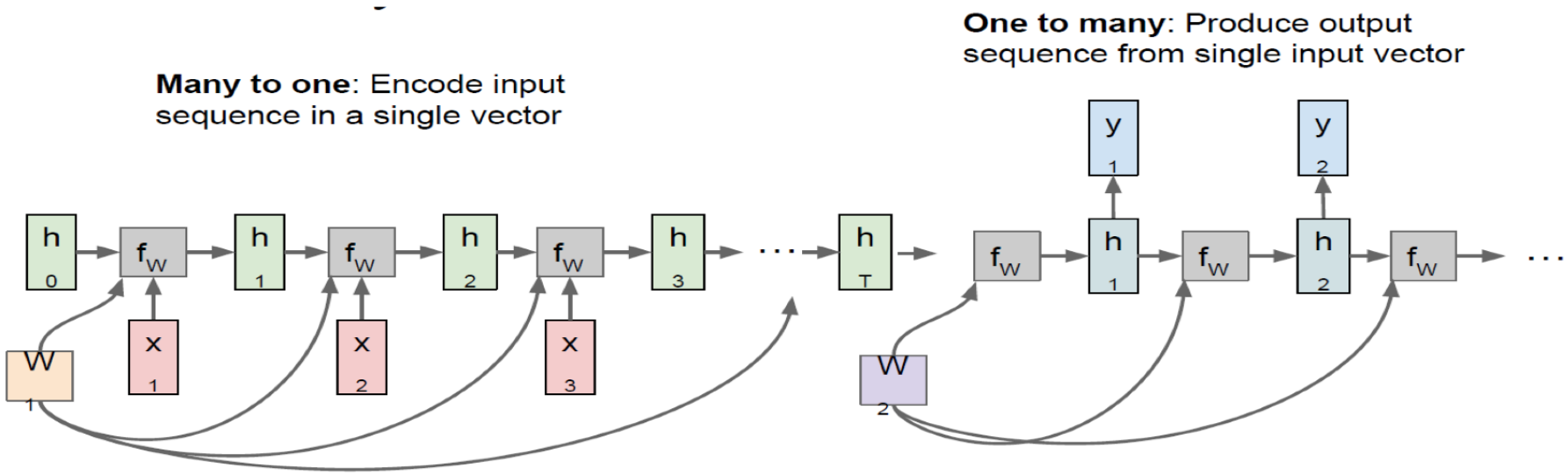
4. seq2seq

4. seq2seq

Sequence to Sequence

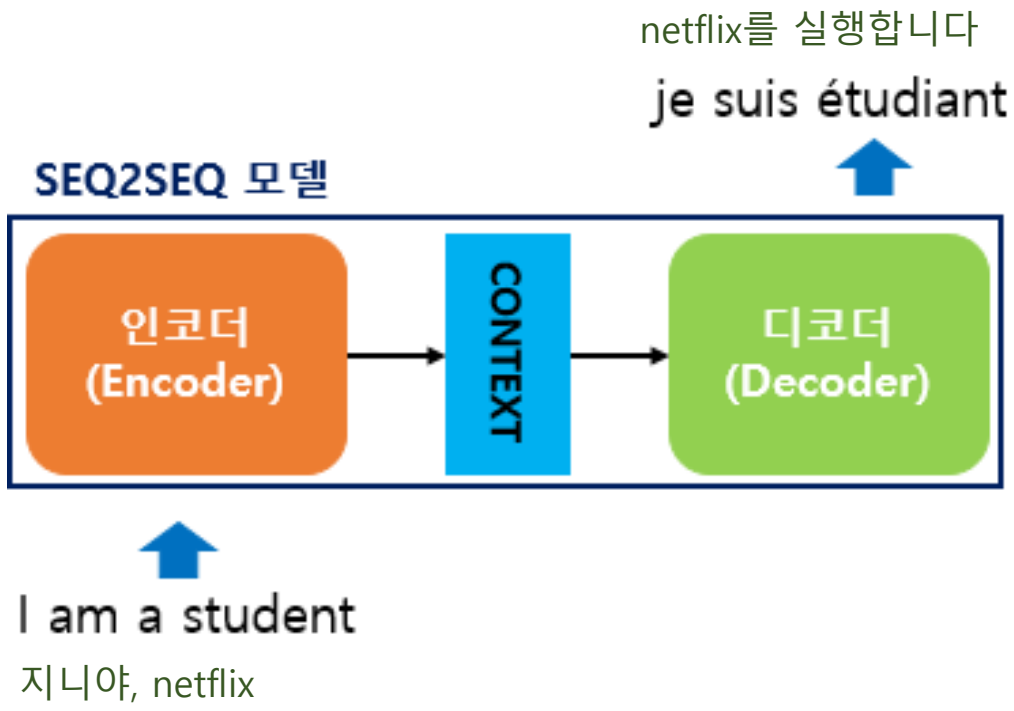
시계열 데이터들을 다른 시계열 데이터들로 변환할 때 사용!
ex) 변역, 음성 인식, …

구조: Many to One(Encoder) + One to Many(Decoder)



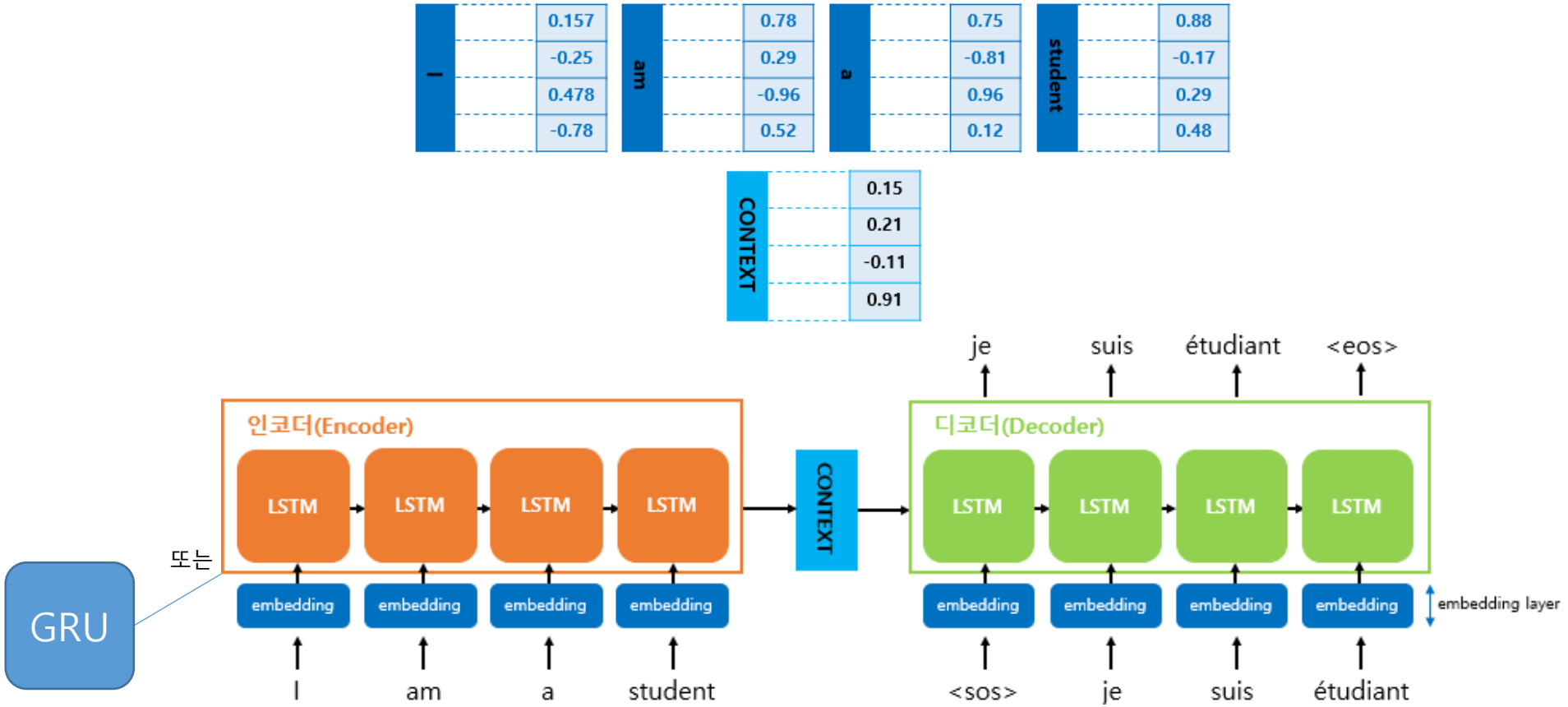
4. seq2seq

Sequence to Sequence



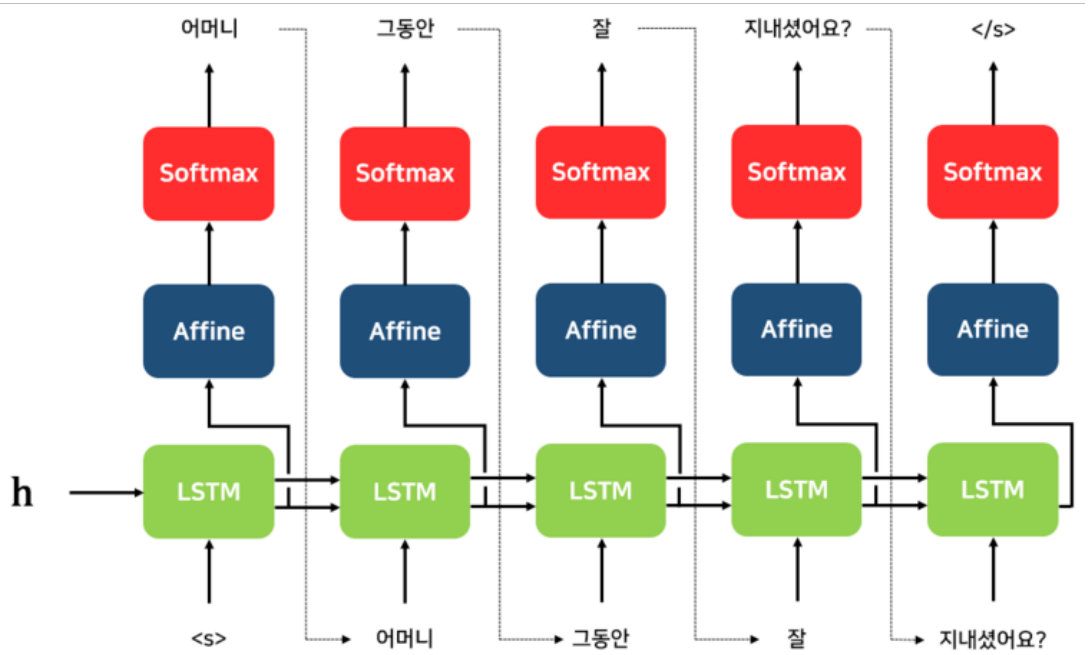
4. seq2seq

Sequence to Sequence



4. seq2seq

Teacher Forcing

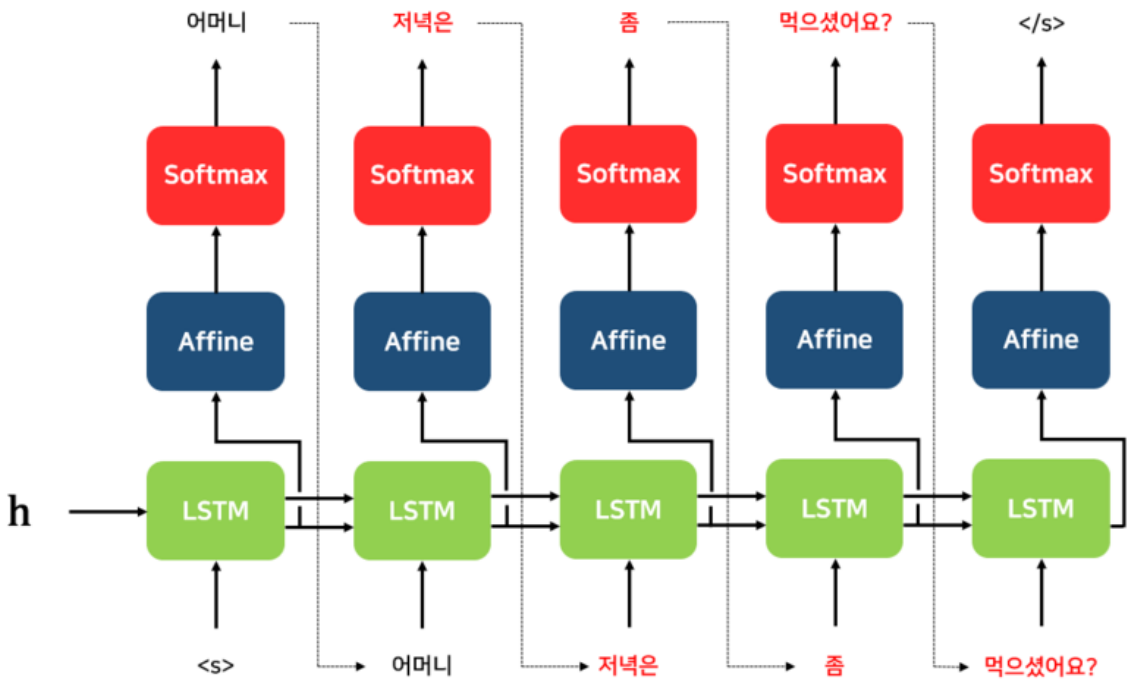


지금까지 말한 General Seq2Seq

t-1번째의 디코더 셀이 예측한 값을
t번째 디코더의 입력으로 넣어준다.

4. seq2seq

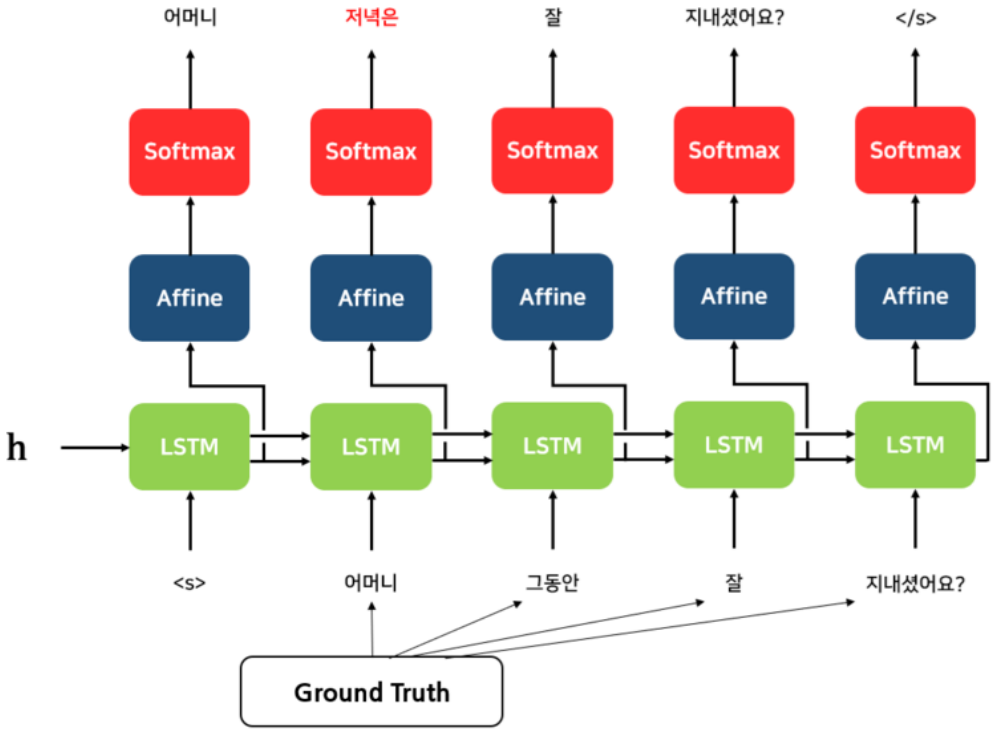
Teacher Forcing



만약 t-1번째 디코더 셀에서
잘못된 단어가 예측되어
t번째 디코더에 전달된다면?

4. seq2seq

Teacher Forcing



학습 vs 테스트

학습 시 입력으로
Ground Truth를 넣어준다.

5. Summary

5. Summary

Summary

✓ RNN

1) sequence data란?
2) input/output 수에 따라 다양하게 적용 가능
3) Backpropagation에서 기울기 소실 문제 발생

✓ LSTM

1) 결과값이 다음 시점으로 넘어갈 때 결과값을 넘길지 말지 결정하는 단계 추가
2) Cell state
3) 3개의 Gate: Forget, Input, Output

5. Summary

Summary

✓ GRU

1) LSTM의 간소화 버전 – 기억 셀(C)이 사라지고 은닉 상태 벡터(h)만 사용
2) 2개의 Gate: Reset, Update
3) GRU backpropagation

✓ seq2seq

1) 구조: Many to One(Encoder) + One to Many(Decoder)
2) Teacher Forcing

5. Summary

Reference

여진영 교수님 빅데이터 강의안

핵심 딥러닝 입문: RNN, LSTM, GRU, VAE, GAN 구현 - 아즈마 유키나가

6기 안민용 선배님 - RNN 세션 강의 자료

https://wooono.tistory.com/223

https://wikidocs.net/152773

https://techblog-history-younghunjo1.tistory.com/481