2024 D&A

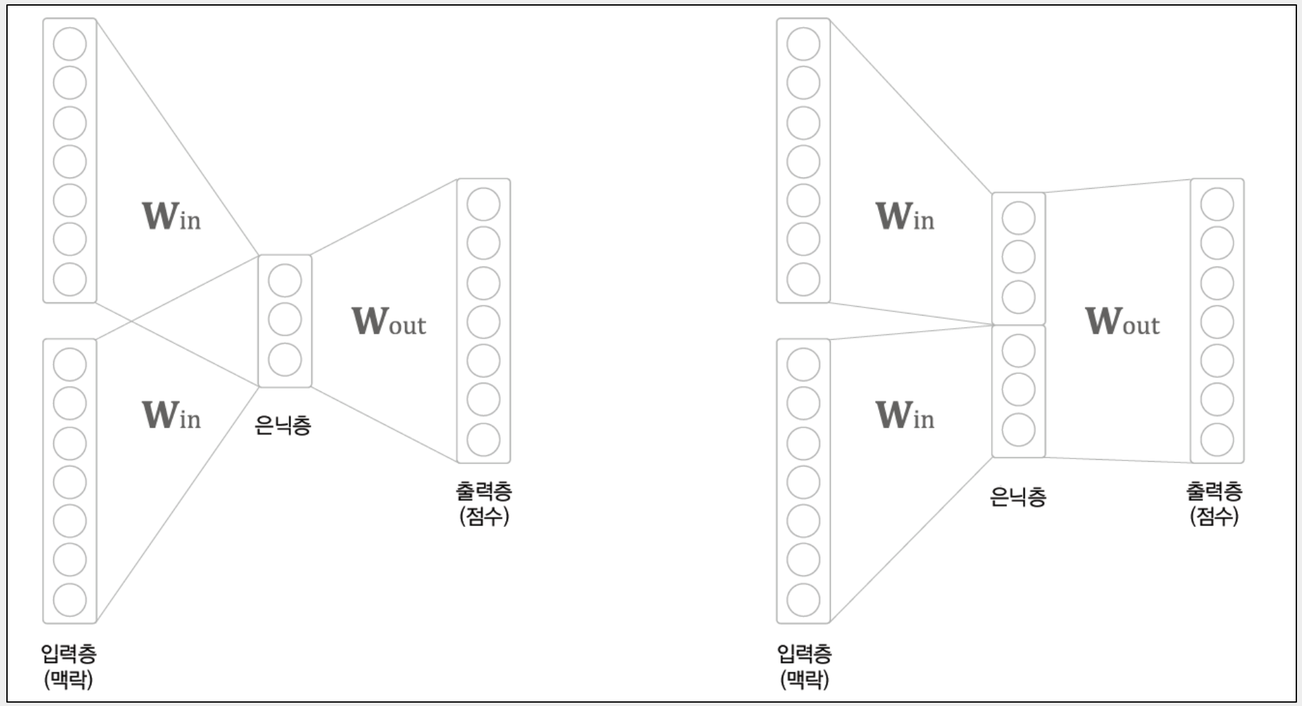
NLP II

Deep Session 8차시

CONTENTS.

지난 7차시

Word2Vec - CBOW (continuous bag of words)



문제점

(왼쪽) CBOW 모델에서는 맥락 안의 단어 순서 무시

(오른쪽) 맥락의 단어 벡터를 은닉층에서 연결
그러나 맥락의 크기에 비례해 가중치 매개변수도 늘어남

해결

RNN 사용하여 해결(순환 신경망)

맥락이 아무리 길더라도 맥락의 정보를 기억하는 매커니즘

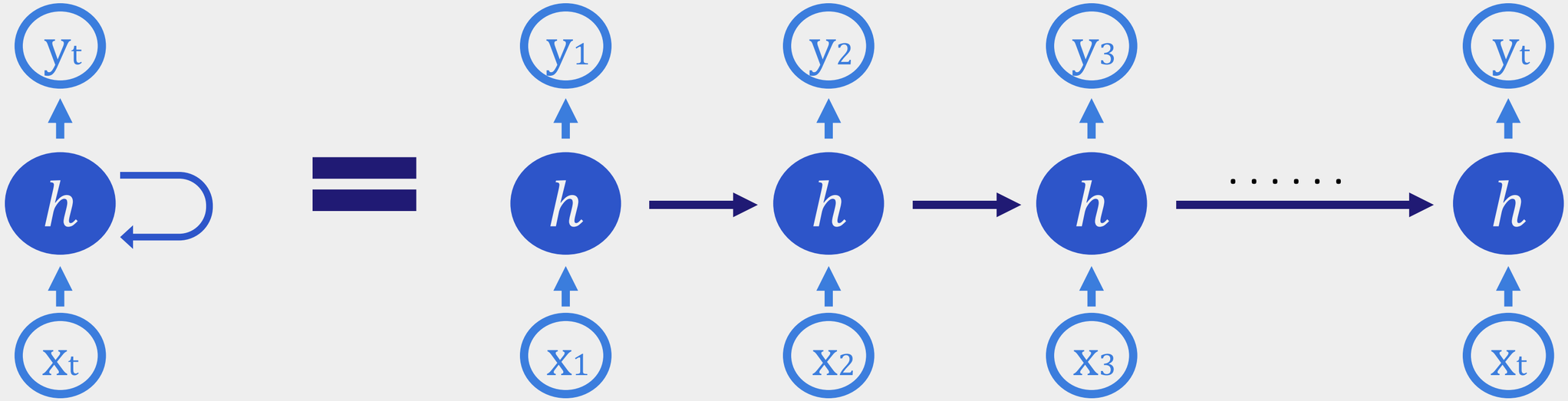
Recurrent Neural Network (순환 신경망)

RNN이란?

시계열 데이터(Sequential data)를 처리하도록 고안된 모델 중 하나

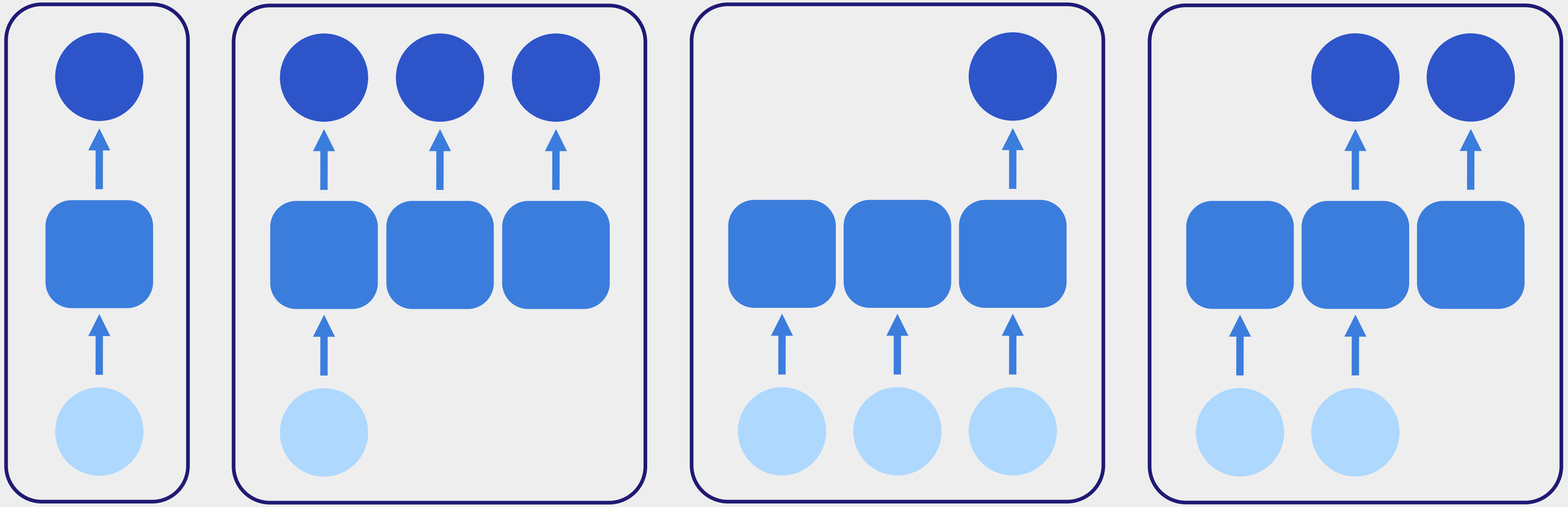
데이터가 반복적으로 '순환'하는 형태이기 때문에 순환 신경망이라고 부름

-> 과거의 정보를 기억하는 동시에 최신 데이터로 갱신



RNN

Task에 따른 RNN의 구조



One to One

Vanilla Neural Network

One to Many

ex) Image Captioning
(image → seq of words)

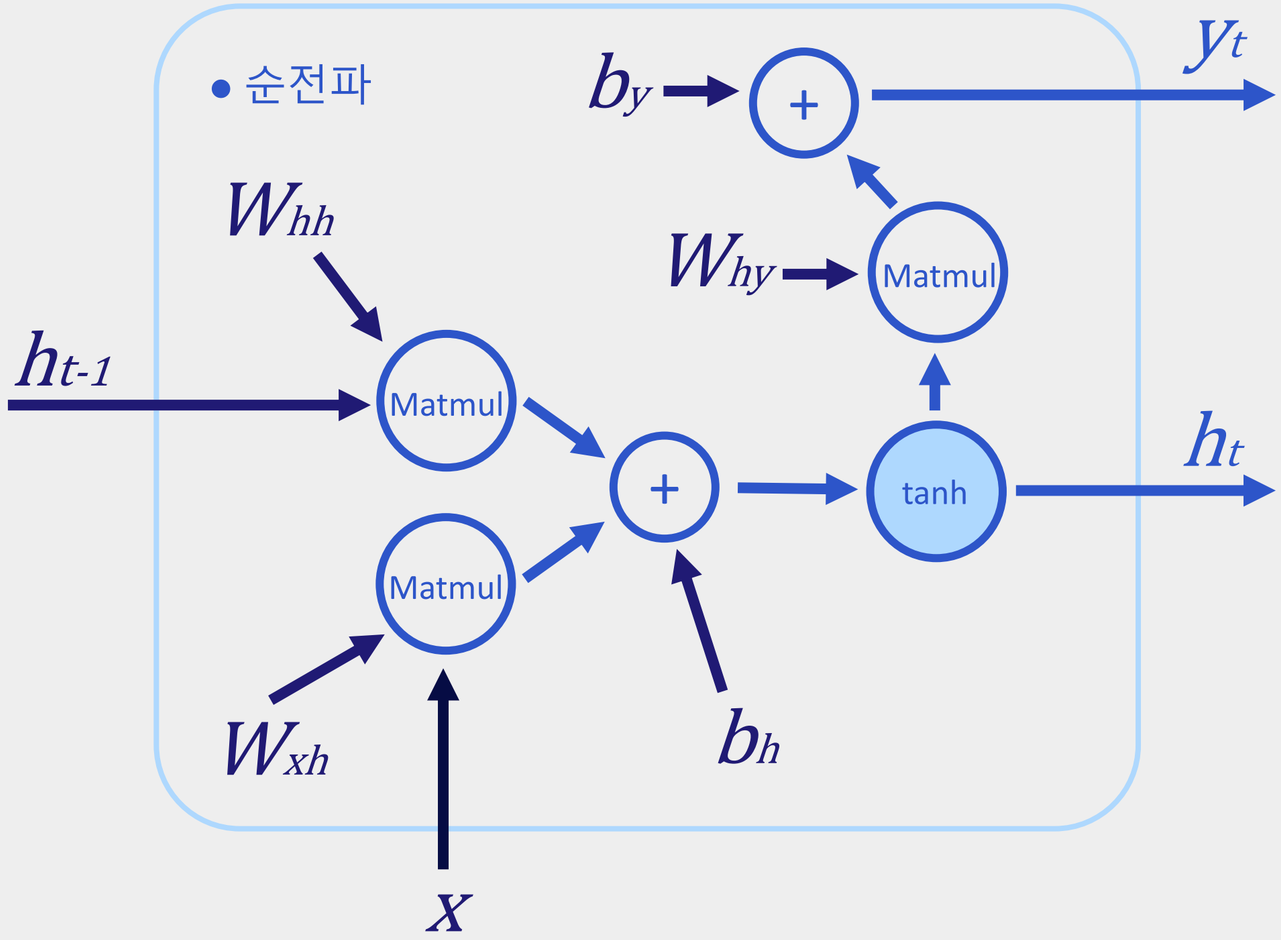
Many to One

ex) Sentiment Classification
(seq of words → sentiment)

Many to Many

ex) Machine Translation
(seq of words → seq of words)

Vanilla RNN



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

$$y_t = W_{hy}\, h_t + b_y$$

* 가중치

W_{hh} : 이전 hidden state를 현재 hidden state로 변환

W_{xh} : input x를 hidden state로 변환

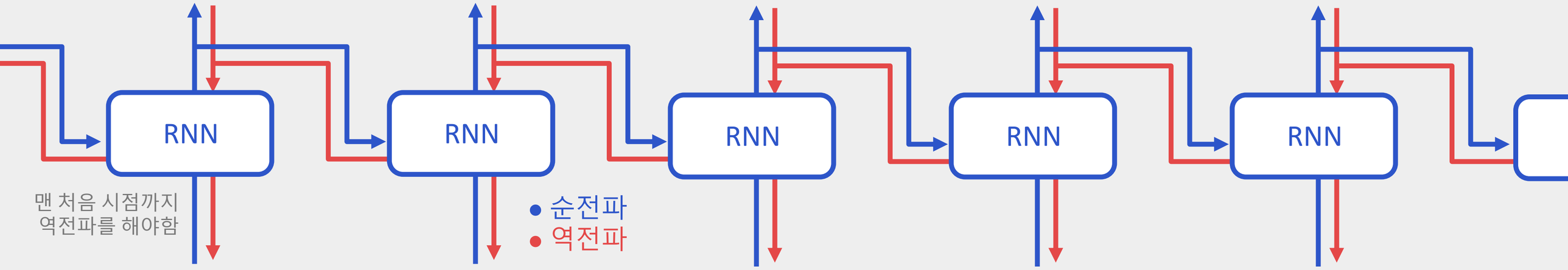
W_{hy} : hidden state를 output y로 변환

BPTT & Truncated BPTT

BPTT (Backpropagation Through Time) - RNN의 역전파 알고리즘

RNN의 **순환 구조를 펼친 상태**에서 시간 방향(오른쪽)을 역행하여 역전파가 이루어짐 (= BPTT)

RNN은 매 시각의 중간 데이터를 메모리에 저장해야 하기 때문에
시계열 데이터의 시간 크기가 커지는 것(길어지는 것)에 비례하여 요구되는 **컴퓨팅 자원 증가, 기울기 불안정**해짐
-> Gradient Vanishing & Gradient Exploding 발생



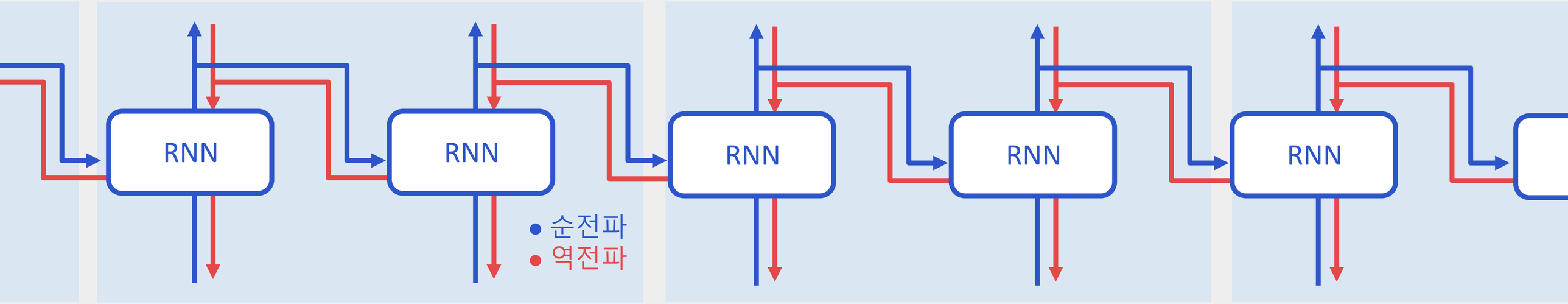
BPTT & Truncated BPTT

Truncated BPTT

이와 같은 문제를 완화하기 위해 시간축 방향으로 너무 길어진 신경망을 적당한 지점에서 잘라내어 오차역전파 수행

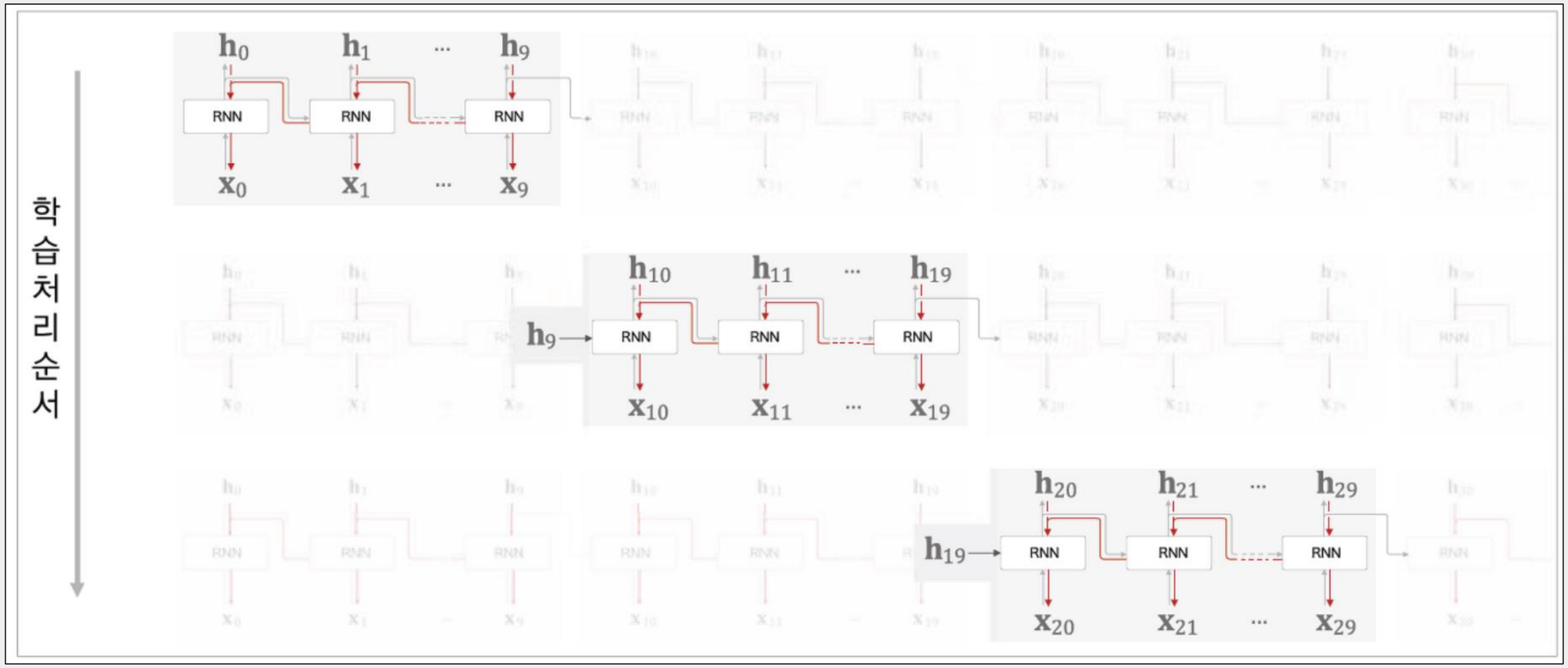
단, 역전파의 연결만 끊을 것(순전파 연결은 유지)

-> 데이터를 '순서대로' 입력해야 함



BPTT & Truncated BPTT

Truncated BPTT



RNNLM (RNN Language Model)

RNNLM이란?

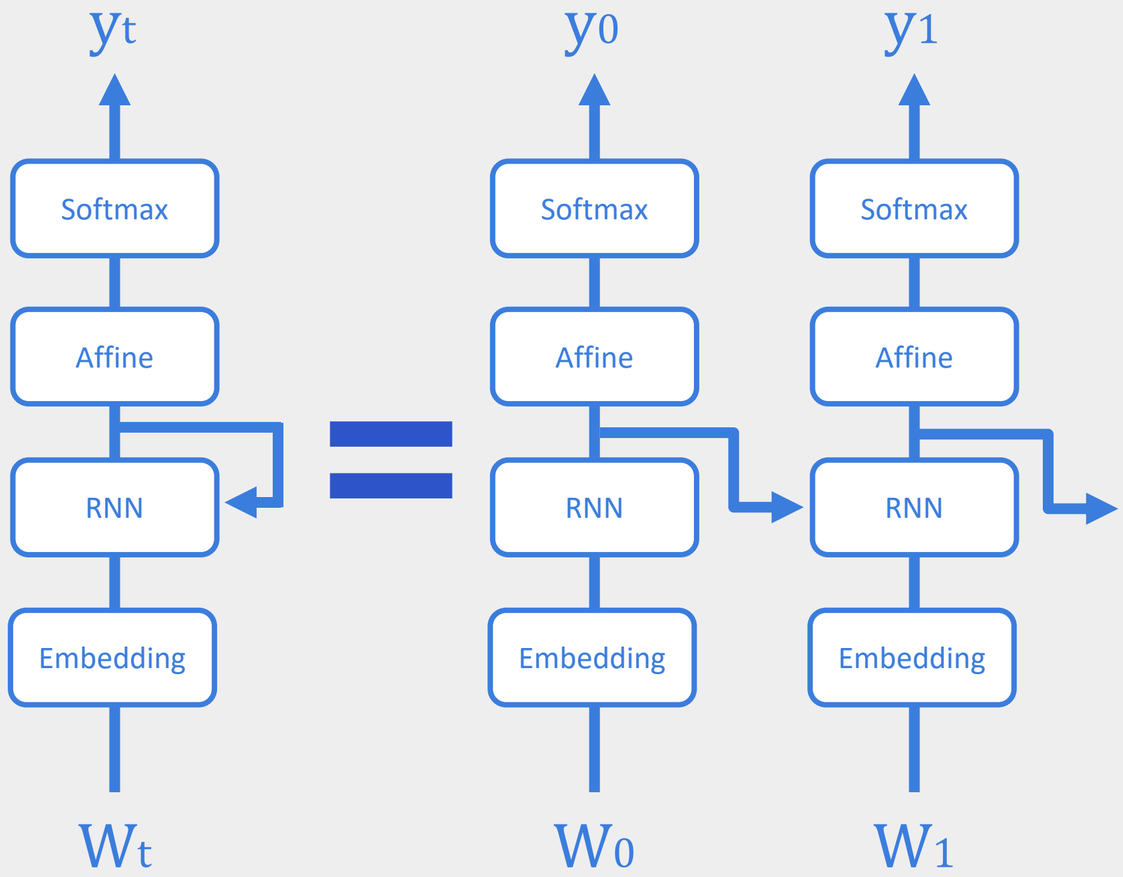
RNN 계층을 이용한 언어 모델

이전 단어들과의 의존 관계 고려하여 예측

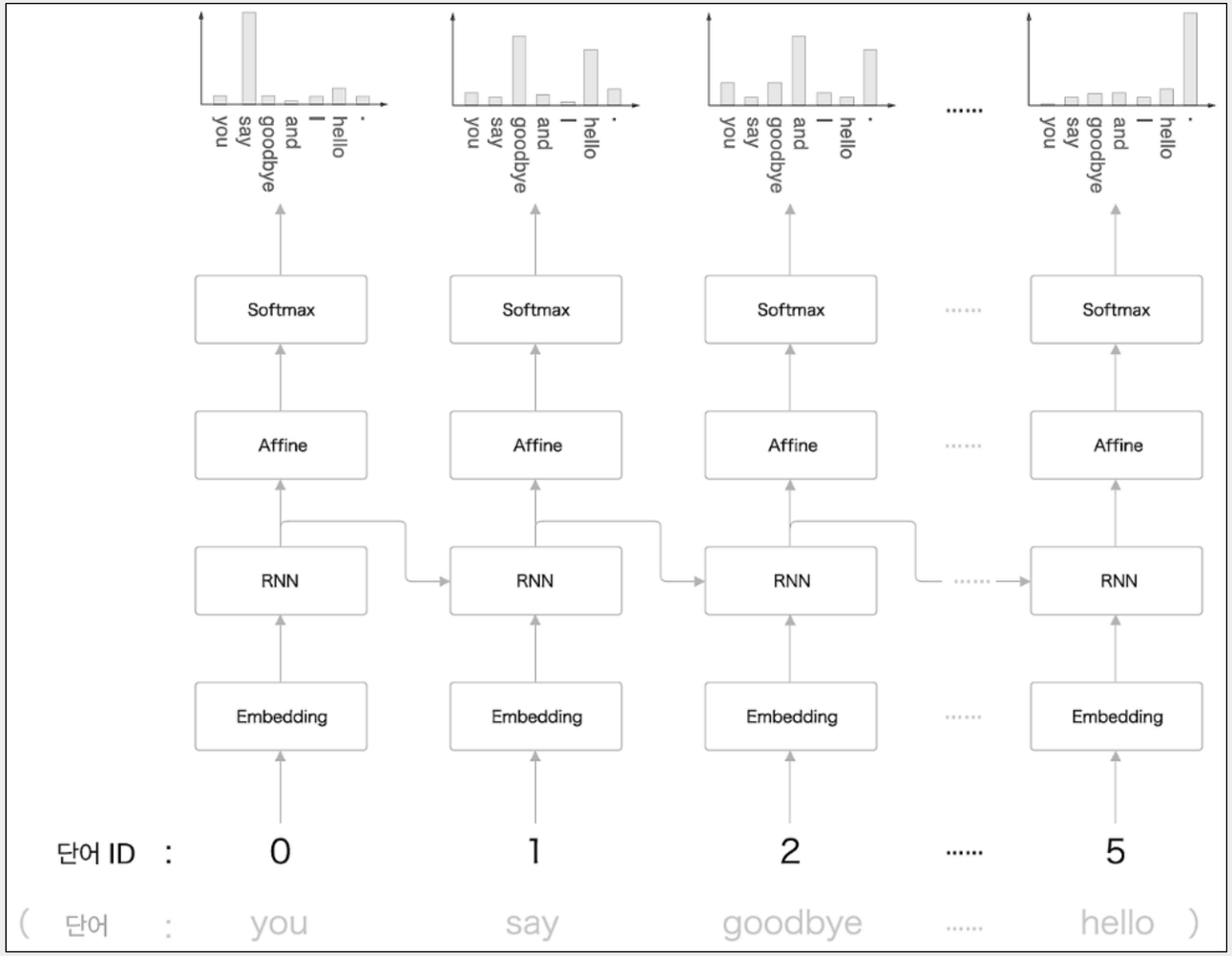
Softmax 함수 사용하여 확률 분포 계산 -> Cross entropy

계층 순서

Input > embedding > RNN > Affine > Softmax > output



RNNLM (RNN Language Model)

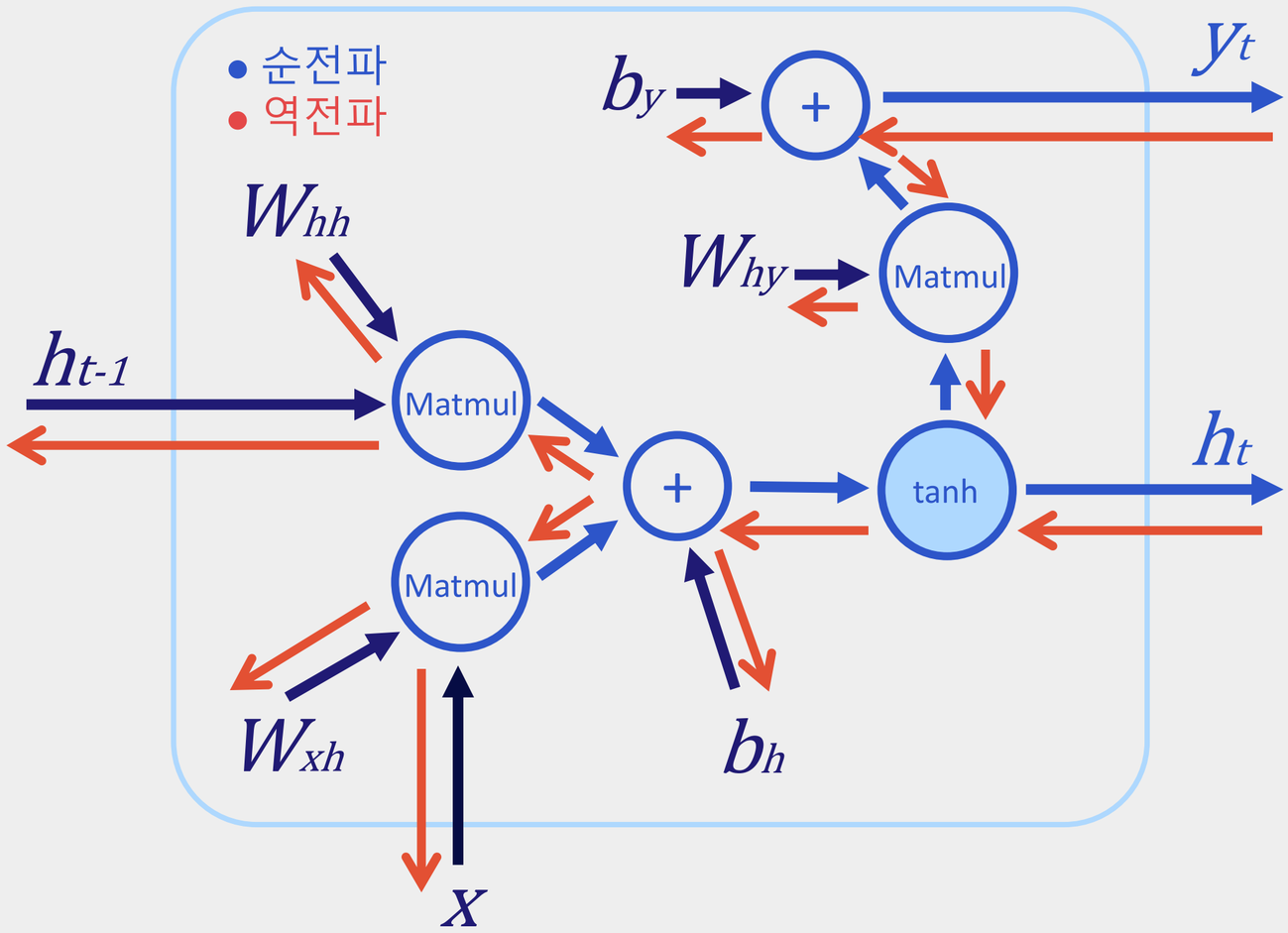


RNN의 문제점

문제점

Gradient Vanishing(기울기 소실) or Gradient Exploding (기울기 폭발) 발생

RNN의 문제점



RNN 계층의 기울기 전파

역전파로 전해지는 기울기는 차례로

'tanh' , '+', 'MatMul' 연산 통과

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

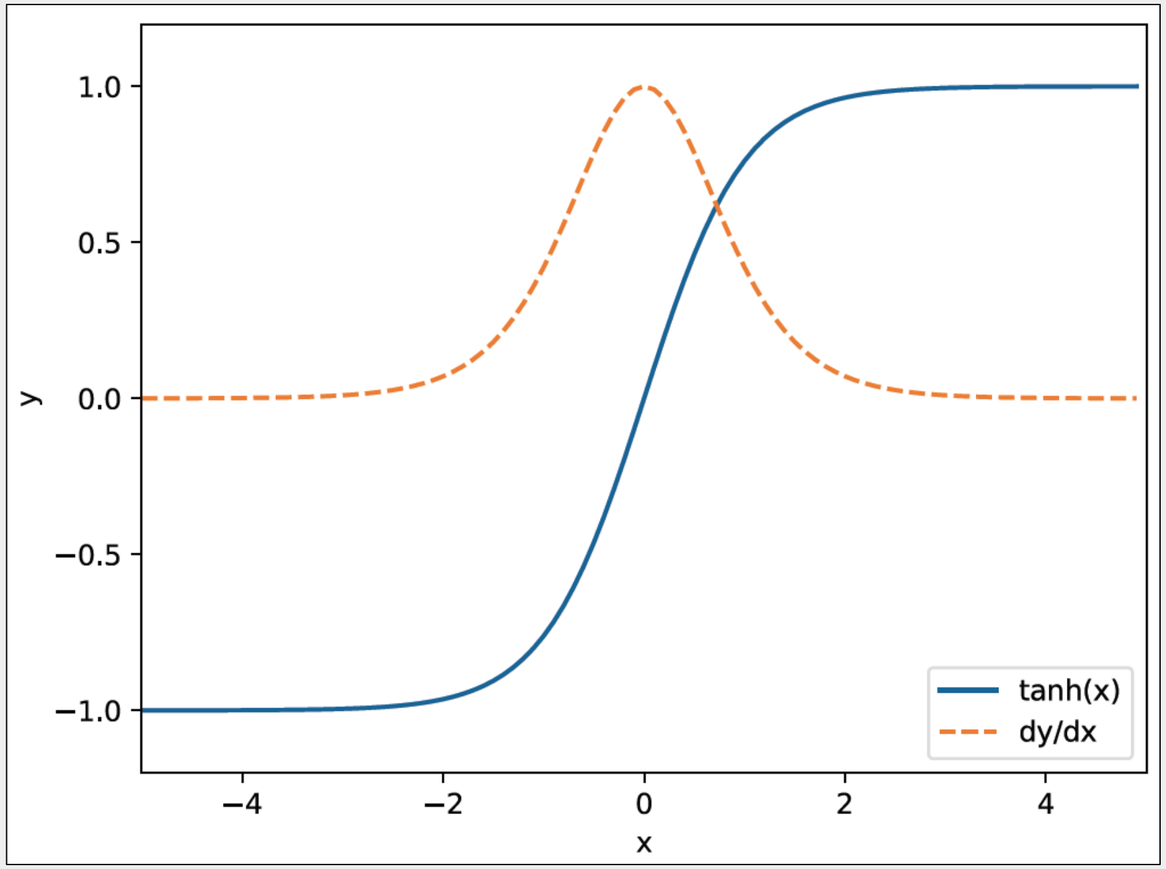
RNN의 문제점

원인(1) - tanh 함수

점선이 y= tanh(x)의 미분

미분값은 1.0 이하이고, x가 0으로부터 멀어질수록 작아짐

=> 역전파시 tanh 노드를 지날 때마다 기울기값 점점 작아짐



RNN의 문제점

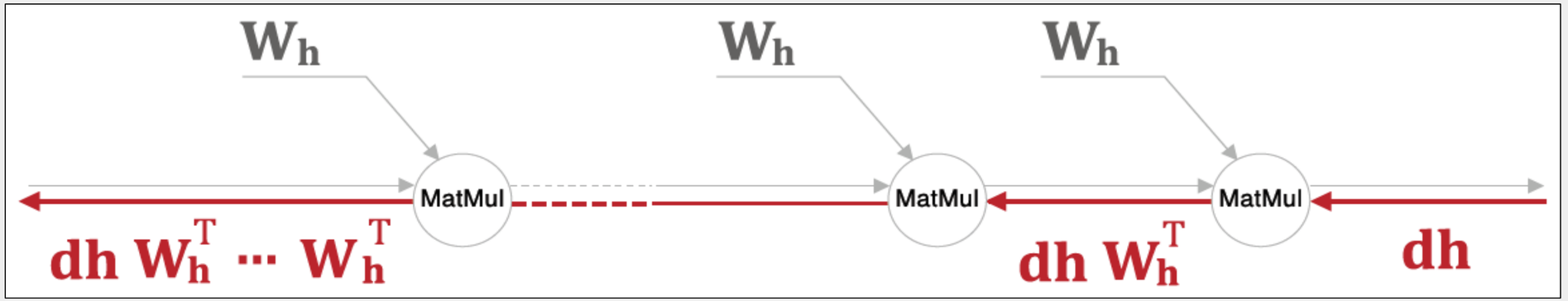
원인(2) - MatMul 연산

MatMul 노드에서의 역전파는 $\mathbf{dh}\boldsymbol{W_h}^{\boldsymbol{T}}$ (W_{hh}, W_{xh}, W_{hy}) 라는 행렬 곱으로 기울기 계산

같은 계산을 시계열 데이터의 시간 크기만큼 반복 (동일한 가중치 $\boldsymbol{W_{hh}}, \boldsymbol{W_{xh}}, \boldsymbol{W_{hy}}$ 사용)

1) 스칼라값일 경우 1이상이면 기울기 폭발, 1이하면 기울기 소실

2) 행렬일 경우 행렬의 특이값(데이터 분포 정도)이 척도가 됨



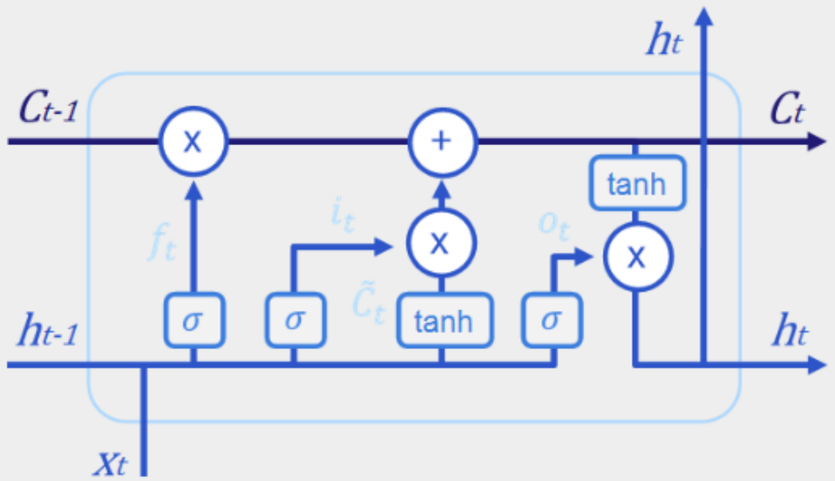
RNN의 문제점

해결방안

1) 기울기 폭주 : 기울기 클리핑(gradient clipping)

$$if \quad \|\hat{g}\| \geq threshold$$
$$:$$
$$\hat{g} = \frac{threshold}{\|\hat{g}\|} \hat{g}$$

2) 기울기 소실 : 장기 의존성 문제 => LSTM

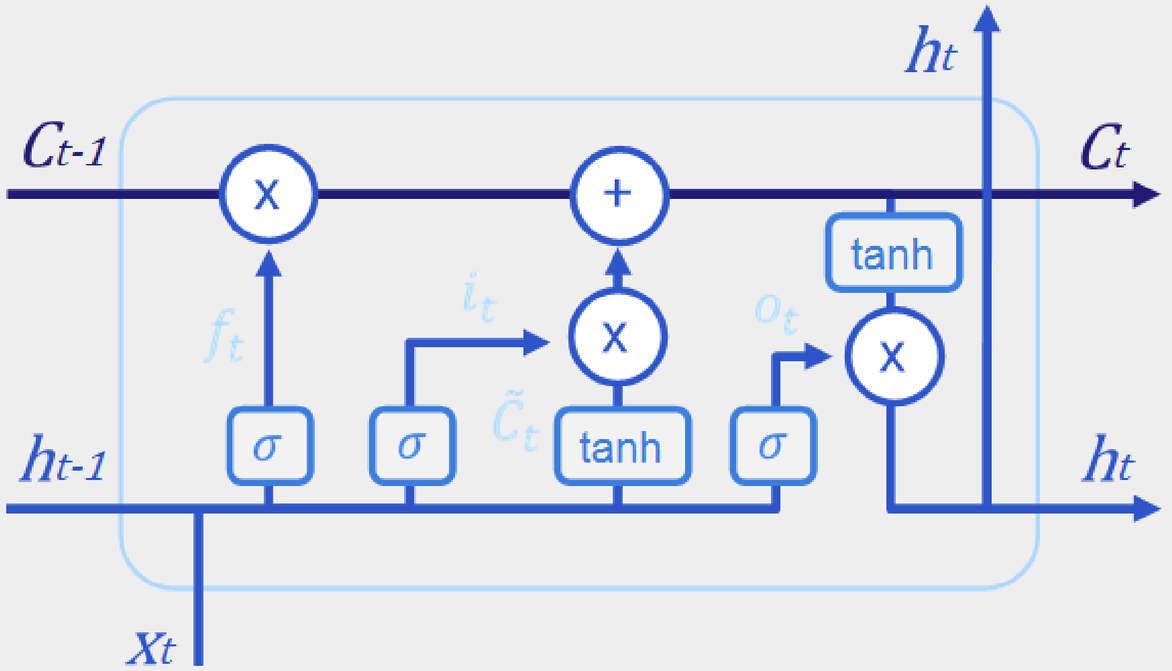


LSTM

Long Short Term Memory Network

LSTM이란?

장기 기억과 단기 기억, '잊어버림' 추가를 통해 기본 RNN의 장기 의존성 문제 해결



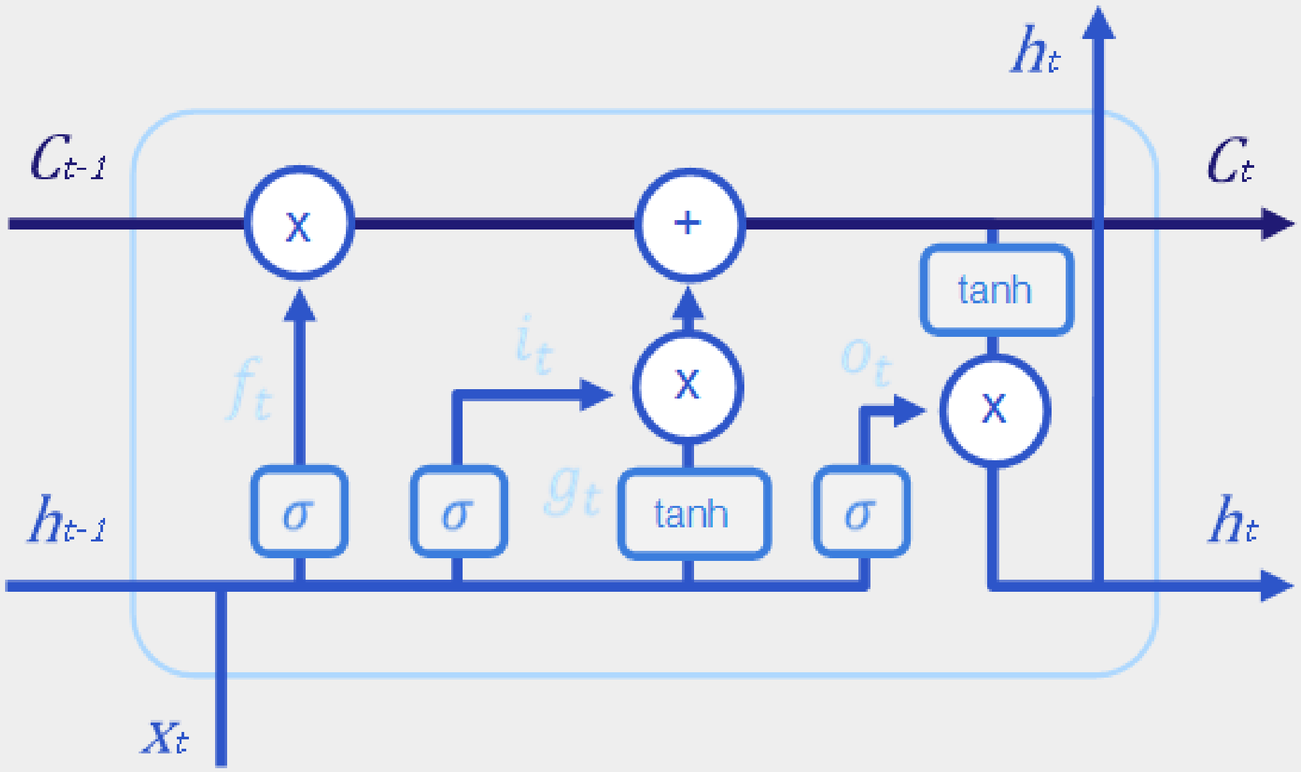
LSTM의 구조 (1) - 핵심 아이디어

Cell state (c)

LSTM의 첫 번째 핵심 아이디어 'Cell state'

-> 장기 기억을 담당하는 메모리와 같은 존재

LSTM 계층 내에서만 완결 (다른 계층으로 출력 X)



Gate

LSTM의 두 번째 핵심 아이디어 'Gate'

-> 3개의 Gate를 통해 정보를 얼마나 기억할지 제어

0.0 ~ 1.0 사이의 실수 (1.0은 완전 개방)

* Gate의 열림 정도는 학습 대상

LSTM의 구조 (2) - forget gate

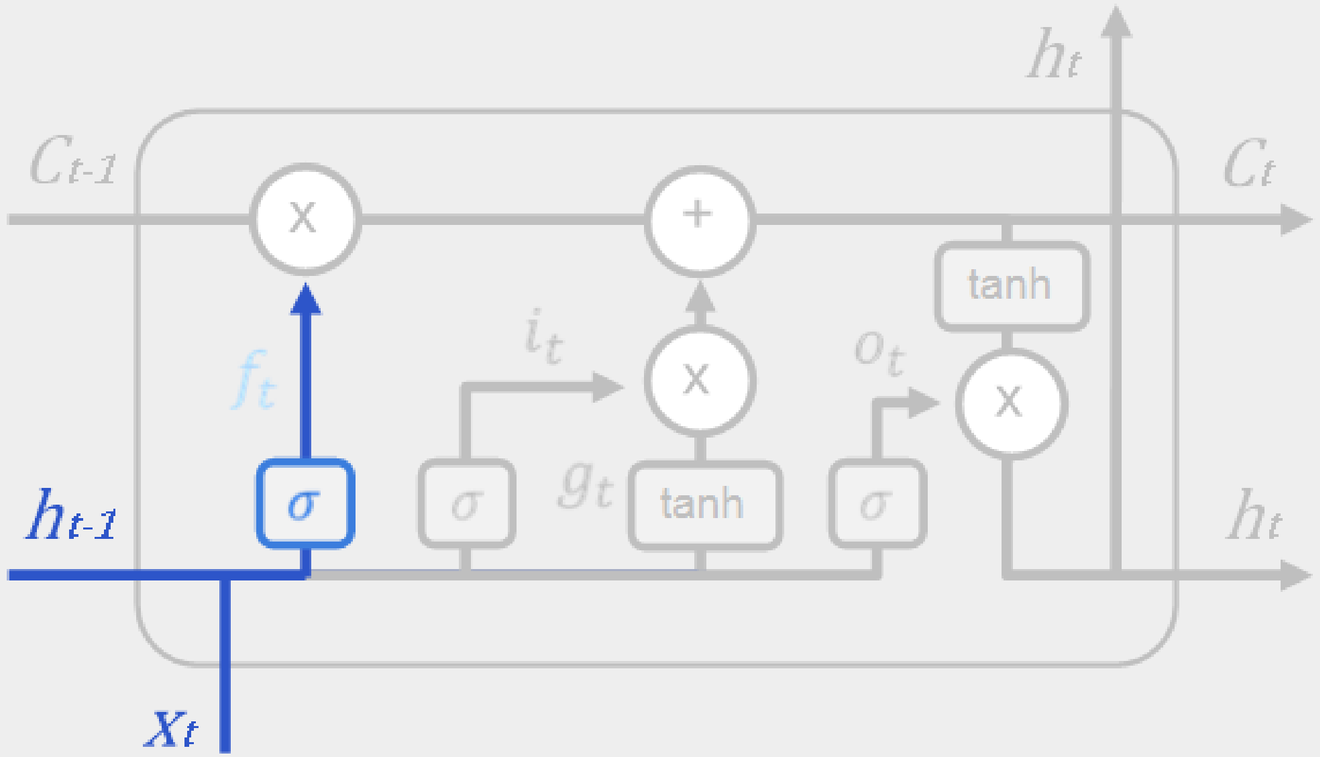
Forget Gate : f_t

과거의 정보를 얼마나 잊을지에 대한 단계

h_{t-1} 과 x_t 를 받아서 Sigmoid 함수 적용

값이 1에 가까울수록 정보 보존 경향이 강해짐

$$f_t = \sigma\left(W_{xh\_f}x_t + W_{hh\_f}h_{t-1} + b_{h\_f}\right)$$



LSTM의 구조 (3) - 새로운 기억 셀

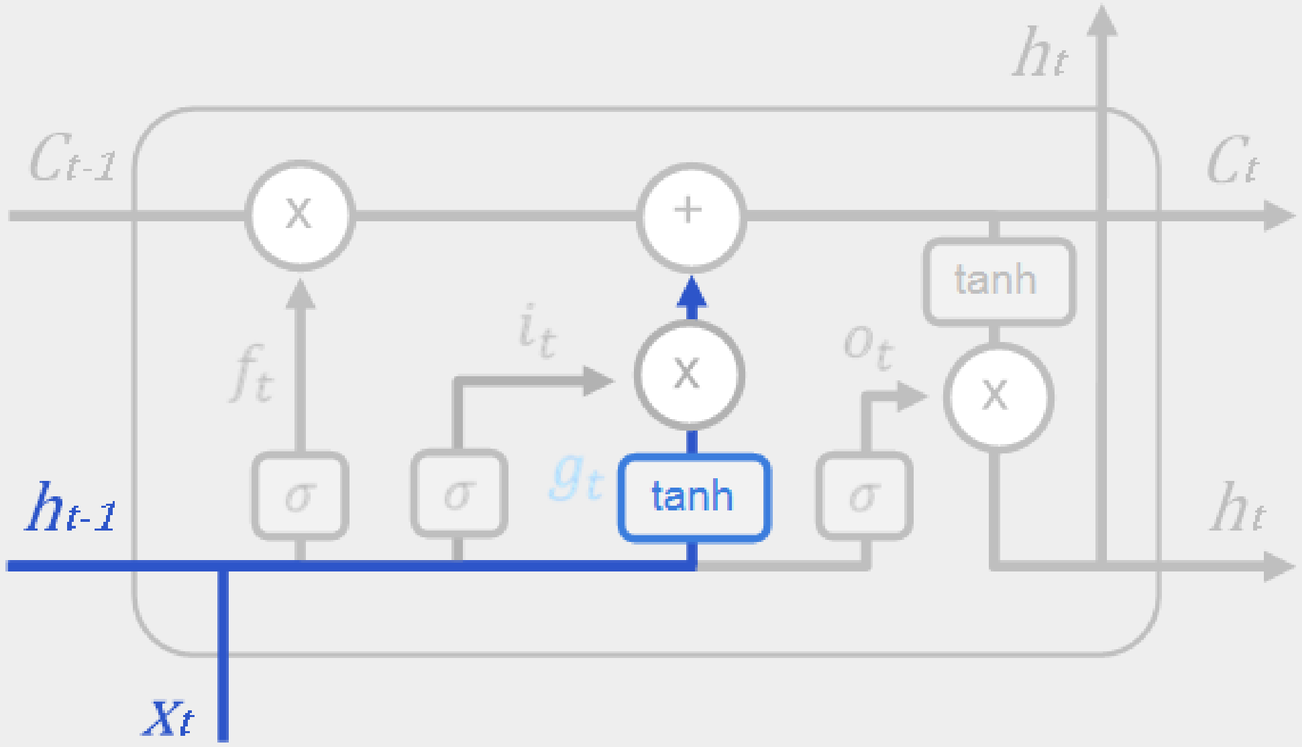
새로운 기억 셀 : g_t

새로 기억해야 할 정보를 기억 셀(c)에 추가

h_{t-1} 과 x_t 를 입력으로 받음

Gate가 아니므로 tanh 함수 사용

$$g_t = tanh(W_{xh\_g}x_t + W_{hh\_g}h_{t-1} + b_{h\_g})$$



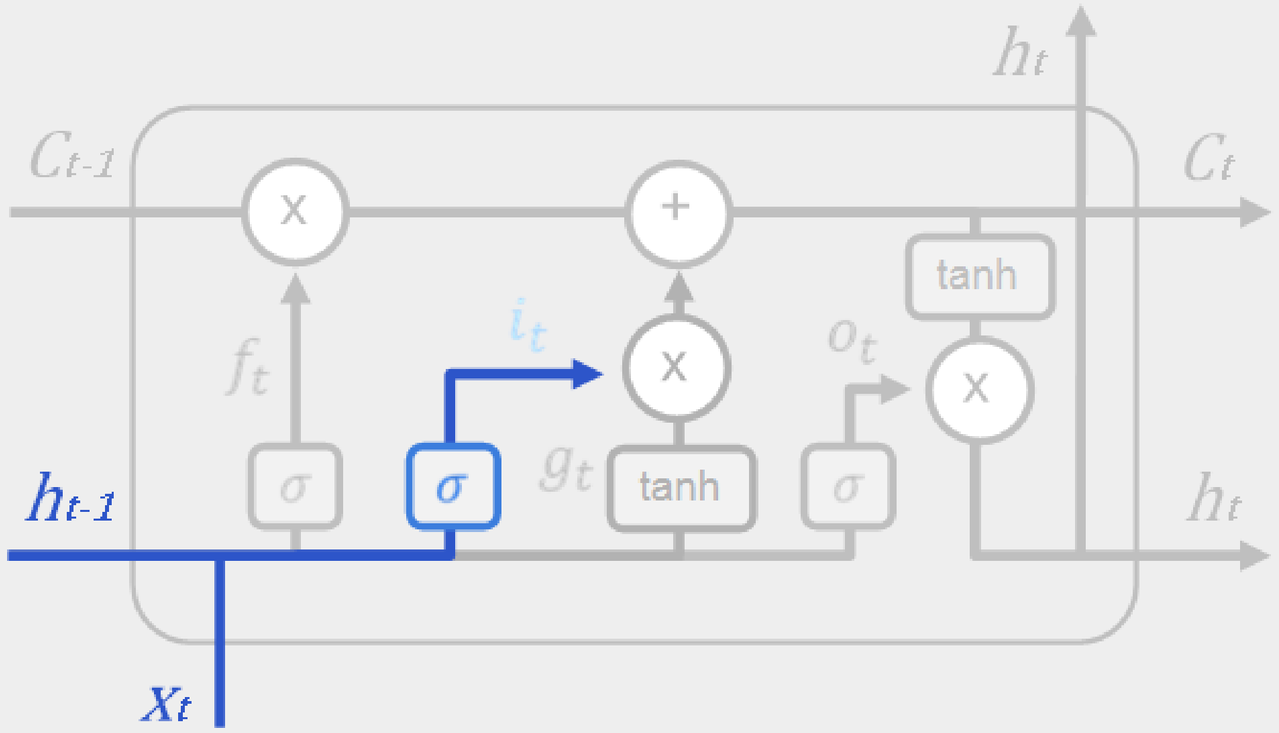
LSTM의 구조 (4) - input gate

Input Gate : i_t

현재 정보를 얼마나 기억할 것인지에 대한 단계

h_{t-1} 과 x_t 를 받아서 Sigmoid 함수 적용

$$i_t = \sigma(W_{xh\_i}x_t + W_{hh\_i}h_{t-1} + b_{h\_i})$$



LSTM의 구조 (5) - state update

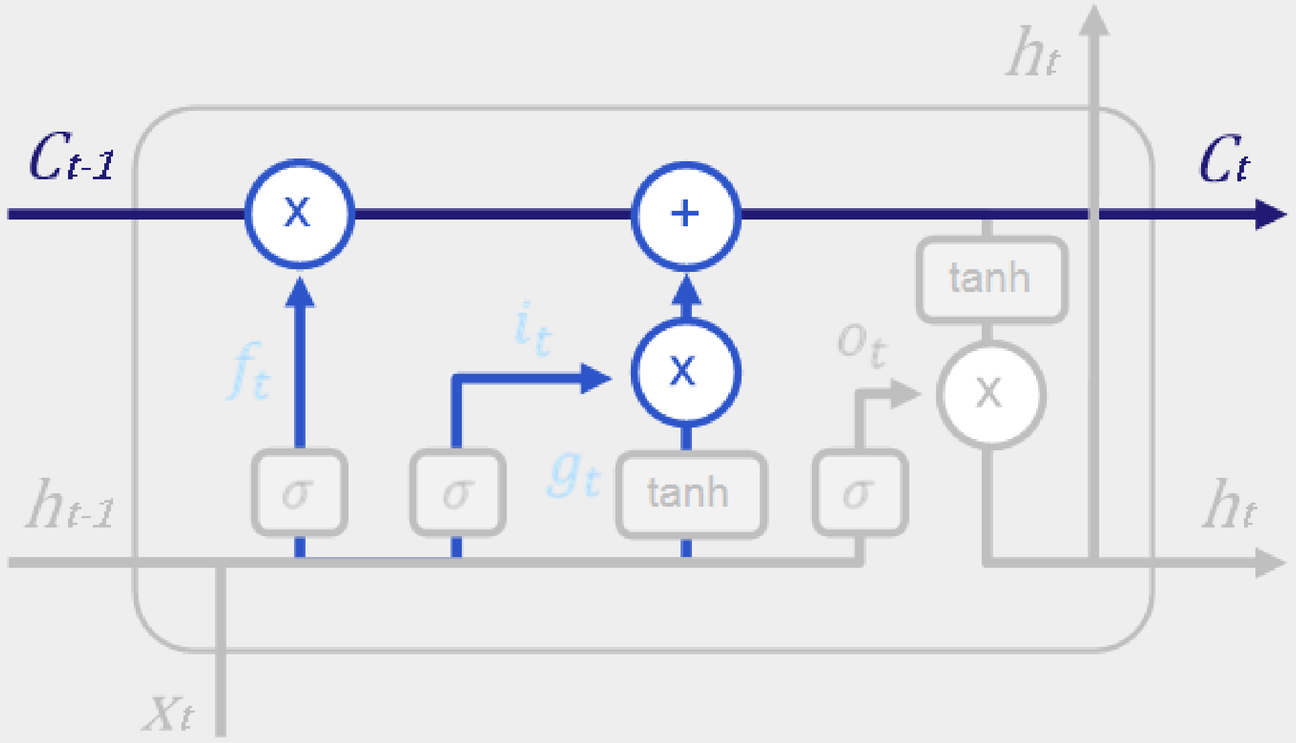
State Update (c_t)

c_{t-1} 에서 c_t 로 Cell state를 갱신하는 단계

f와 c_{t-1} 과의 원소별 곱셈을 통해 c_t 를 구함

i와 g의 원소별 곱셈 결과를 기억 셀에 추가

$$c_t = f_t \odot c_{t-1} + g_t \odot i_t$$



LSTM의 구조 (6) - output gate

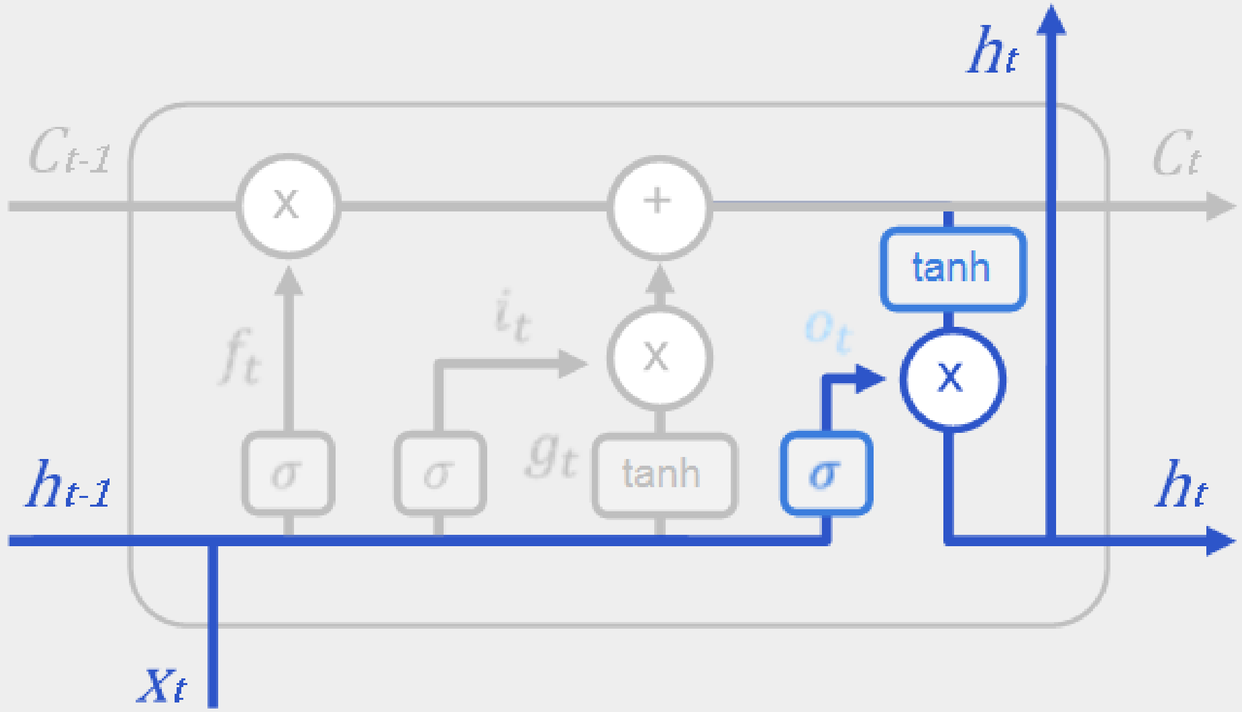
Output Gate : o_t

다음 state로 내보낼 hidden state를 구하는 단계

h_{t-1} 에서 x_t 에 sigmoid 취한 o(gate)를
c_t 에 tanh 적용한 값과 곱하여 hidden state로 내보냄

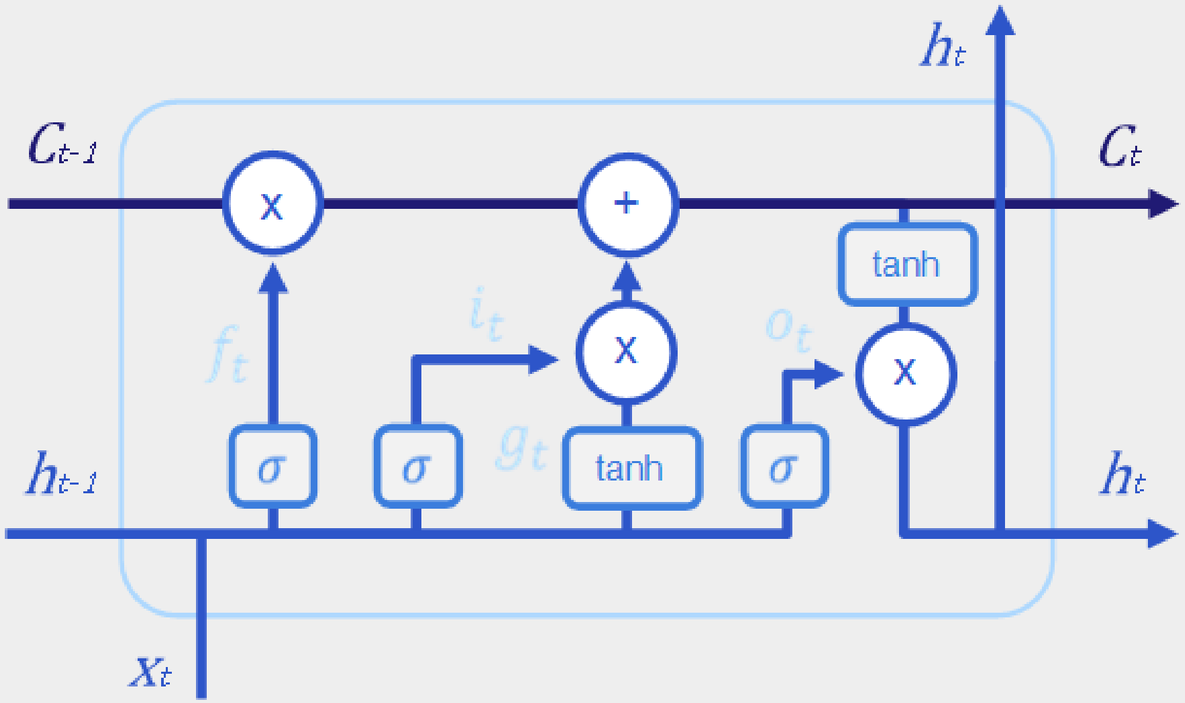
$$o_t = \sigma\left(W_{xh\_o}x_t + W_{hh\_o}h_{t-1} + b_{h\_0}\right)$$

$$h_t = o_t \odot tanh(c_t)$$



LSTM의 구조 정리

정리



$$f_t = \sigma\left(W_{xh\_f}x_t + W_{hh\_f}h_{t-1} + b_{h\_f}\right)$$

$$g_t = tanh(W_{xh\_g}x_t + W_{hh\_g}h_{t-1} + b_{h\_g})$$

$$i_t = \sigma(W_{xh\_i}x_t + W_{hh\_i}h_{t-1} + b_{h\_i})$$

$$o_t = \sigma\left(W_{xh\_o}x_t + W_{hh\_o}h_{t-1} + b_{h\_o}\right)$$

$$c_t = f_t \odot c_{t-1} + g_t \odot i_t$$

$$h_t = o_t \odot tanh(c_t)$$

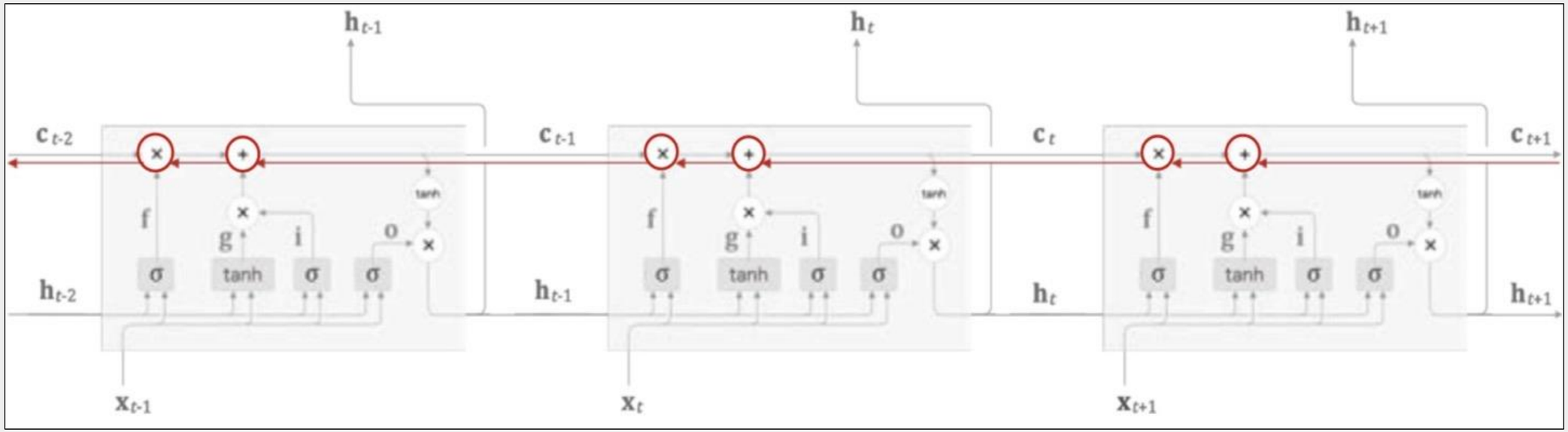
LSTM의 기울기 흐름

‘x’ 노드

‘행렬 곱’ 이 아닌 ‘원소별 곱’으로 계산되어 곱셈 효과 누적 없음

‘x’ 노드는 Forget Gate가 제어하여 ‘잊어서는 안 된다’고 판단한 원소에 대해 그 기울기가 약화되지 않은 채로 과거 반향으로 전달

=> 기울기 감소 X



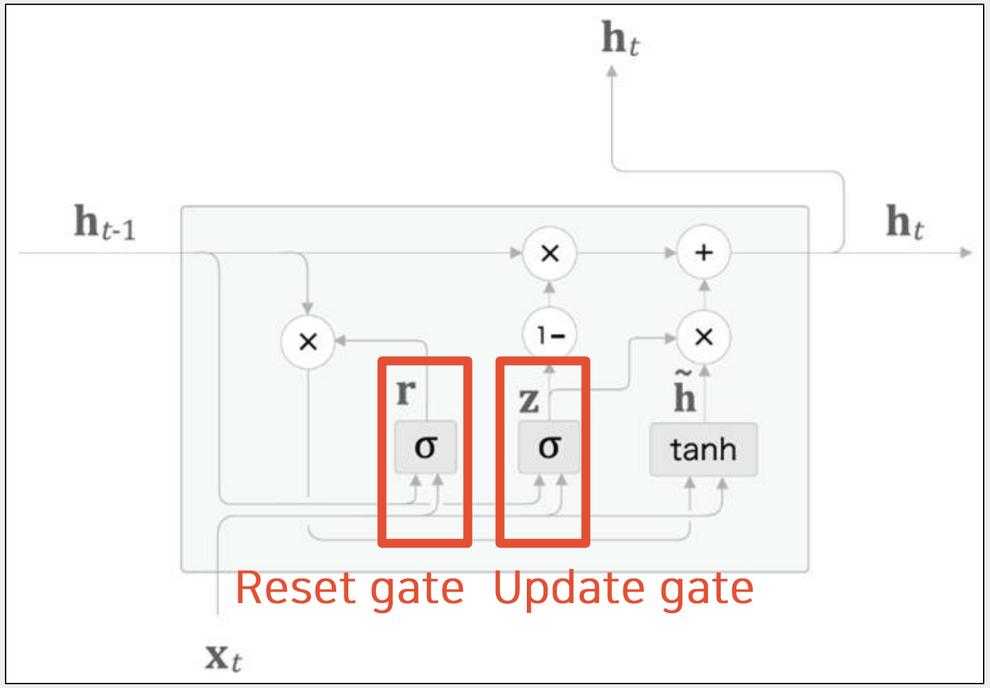
GRU

Gated Recurrent Unit

LSTM과의 차이점

LSTM의 간소화 버전. LSTM과 성능은 비슷하나 학습할 parameter가 적어 학습 속도 빠름

LSTM과 다르게 Gate가 2개이며, Reset gate와 Update gate로 이루어져 있음
Cell state와 hidden state가 hidden state로 통합



GRU의 구조 (1)

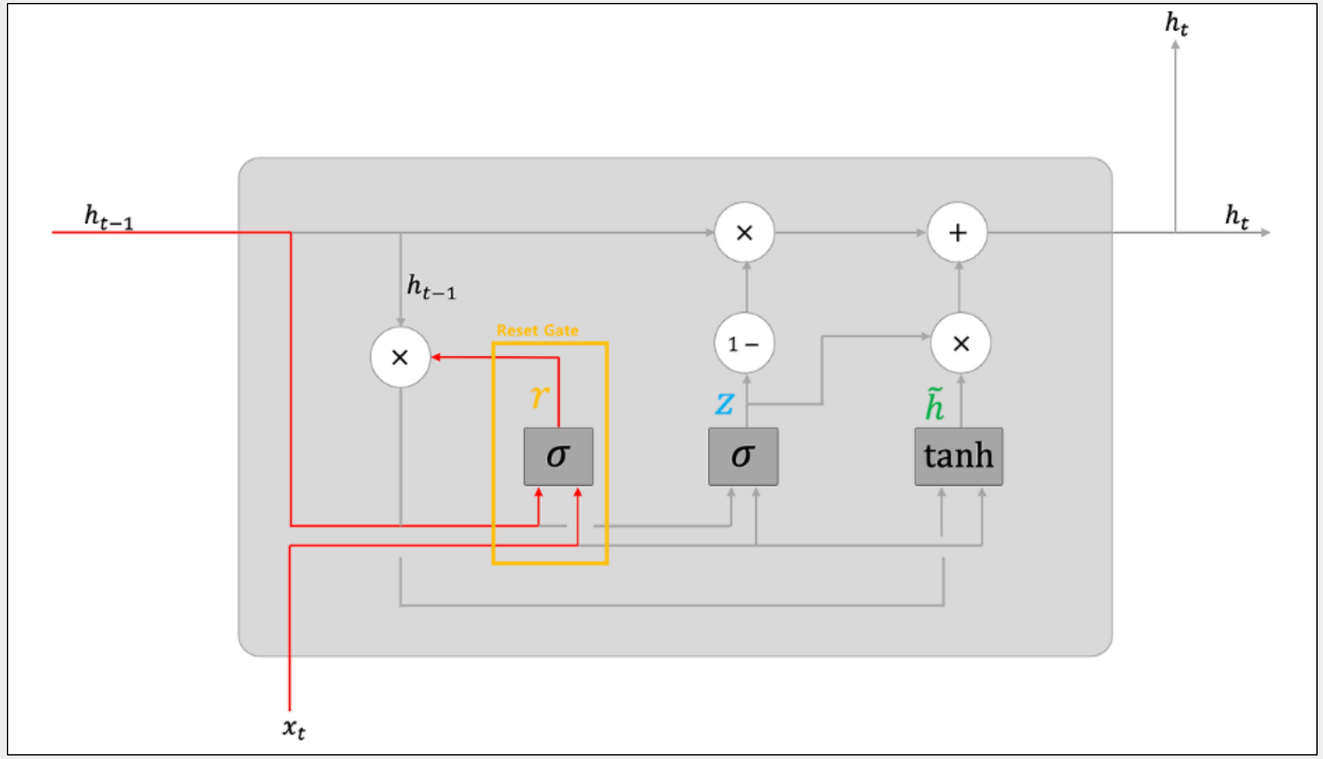
Reset Gate (r)

과거의 은닉 상태를 얼마나 '무시'할지 결정

만약 r이 0이면 새로운 hidden state는x_t 만으로 결정

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

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



GRU의 구조 (2)

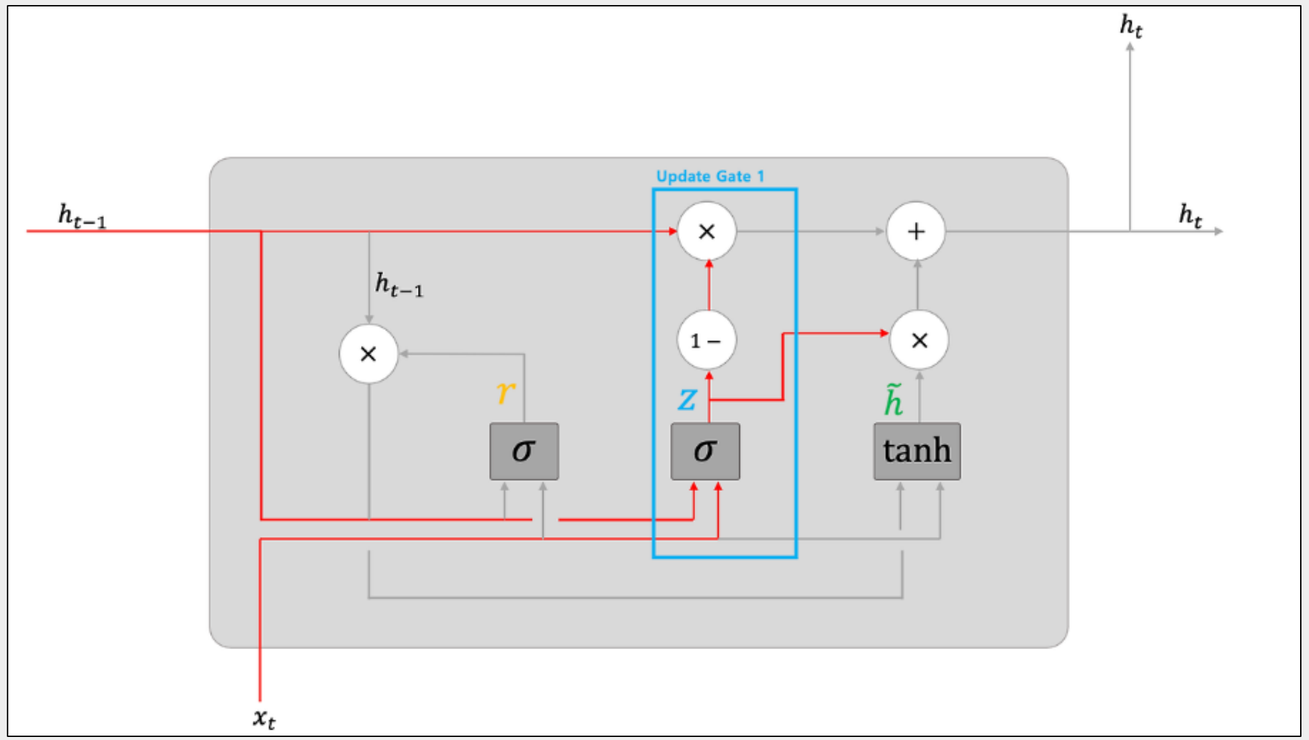
Update Gate (z)

과거와 현재의 정보를 각각 얼마나 반영할지에 대한 비율

Forget Gate와 Input Gate의 역할을 혼자 담당

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

$$h_t = (1 - z) \odot h_{t-1} + z \odot \tilde{h}$$



Seq2Seq

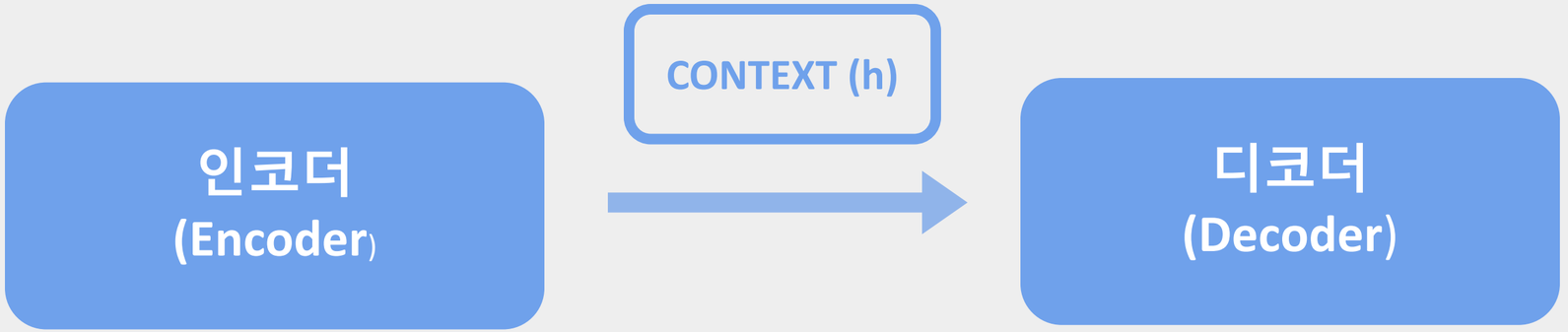
Sequence-to-sequence

Seq2seq이란?

시계열 데이터를 다른 시계열 데이터로 변환하는 모델

RNN 기반의 두 모델을 연결한 모델 -> encoder-decoder 구조

기계 번역과 같은 Many-to-Many task의 수행에 적합

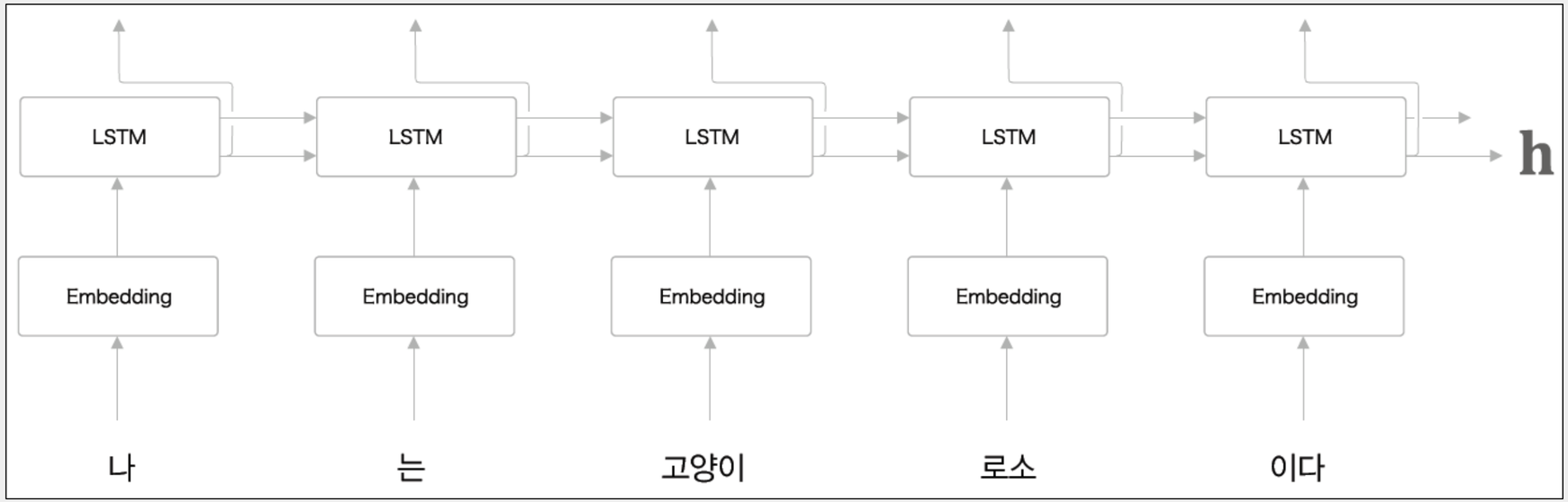


Seq2Seq의 구조

Encoder

RNN 계층을 이어둔 구조
Embedding된 input 값을 받아 RNN 계층 통과
마지막 계층의 hidden state -> context vector (h)

즉, 임의 길이의 시계열 데이터를 고정 길이 벡터 h로 변환함



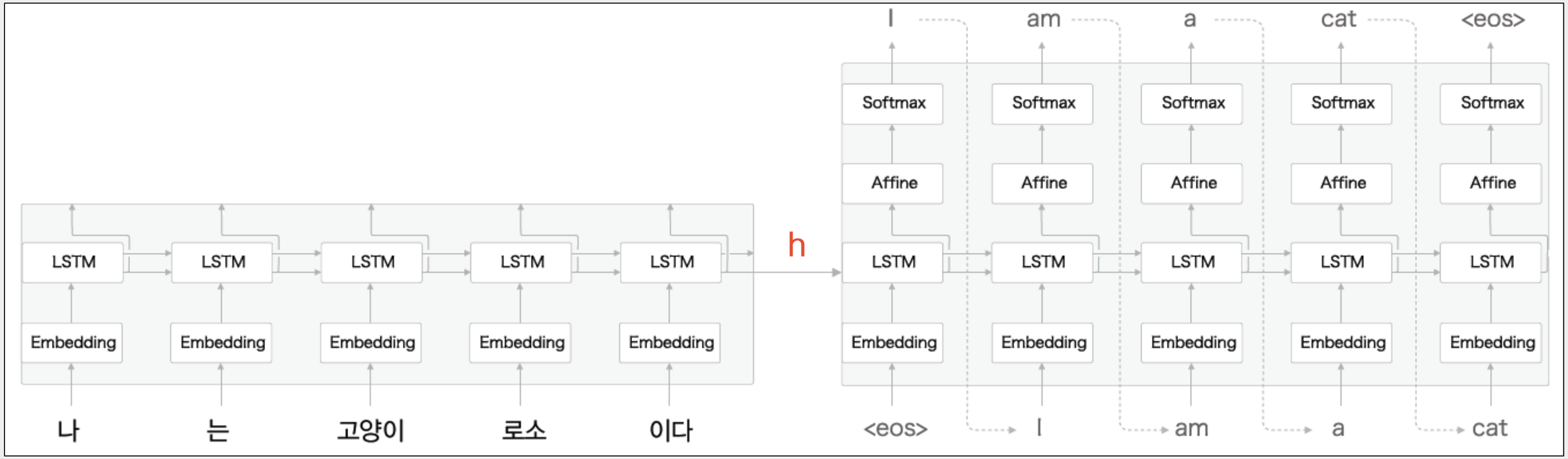
Seq2Seq의 구조

Decoder

RNNLM과 비슷한 구조

BUT context vector (h) 를 추가로 입력 받아 sequence 반환

t 시점의 출력값을 t+1 시점의 입력값으로 사용

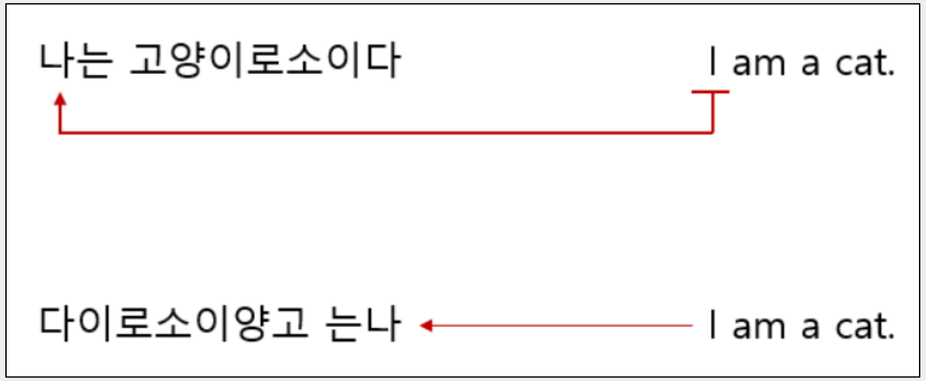


Seq2Seq의 성능 개선 방안

입력 데이터 반전

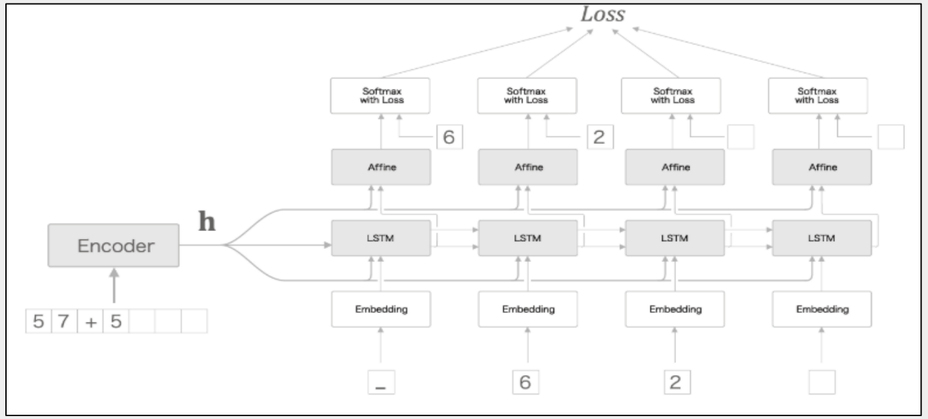
입력 데이터를 반전하는 것만으로도 학습 속도가 개선

입력 데이터를 반전시키면 입력 문장의 첫 부분과
그에 대응하는 변환 후의 단어 사이의 거리가 가까워지기 때문에
기울기 전파가 보다 원활하게 이루어져 학습 효율이 좋아지는 것 (으로 추정)



엿보기 (Peeky)

Decoder가 처리할 수 있는 유일한 정보가 모두 담긴 h를
더 많이 활용하기 위해 h를 decoder의 다른 계층에도 전달



Seq2Seq

Seq2Seq의 문제점

정보 손실 문제

고정 길이 벡터이므로 입력 문장의 길이에 관계 없이 항상 같은 길이의 벡터로 변환
아무리 문장의 길이가 길어져도 항상 똑같은 길이의 벡터에 정보를 밀어넣어야 함
-> 정보 손실 발생

기울기 손실

RNN의 고질적인 문제인 기울기 손실 문제가 여전히 존재

8차시

과제

1. 실습 코드 (RNN, LSTM) 살펴보기

2. Seq2Seq 논문 리뷰

References

밑바닥부터 시작하는 딥러닝 2

https://blog.naver.com/sooftware/221784419691

https://techblog-history-younghunjo1.tistory.com/486

2024 D&A

THANK YOU

2024. 05. 16

Deep Session 8차시