Deep Learning from Scratch 2

밑바닥부터 시작하는 딥러닝 2

5 ~ 6장 발표

24.07.11

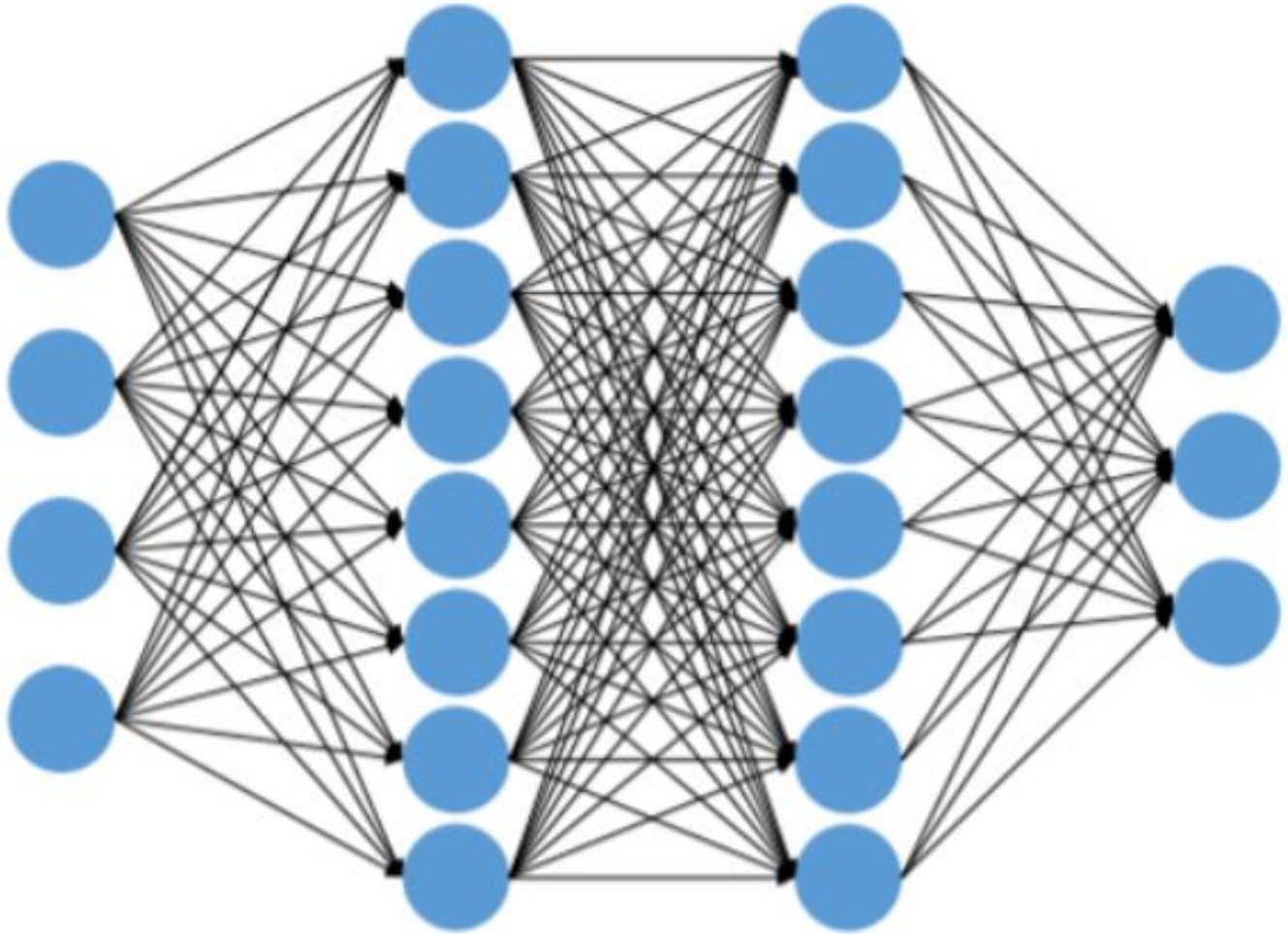
김태균

목차

1. RNN

1.1 도입

배경



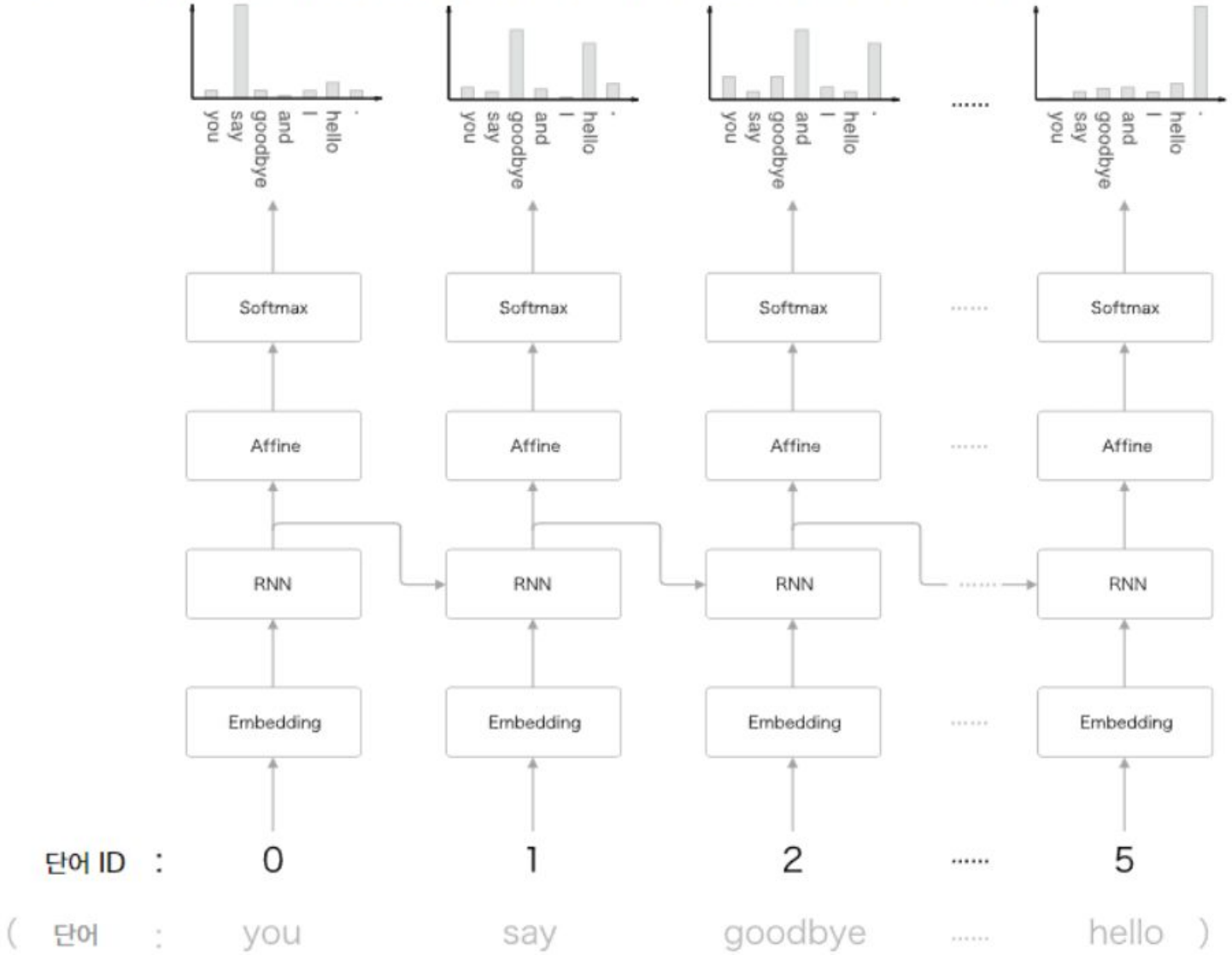
Feed Forward Neural Network

1.1 도입
배경

Q. **언어 모델** 이란?

A. **특정 단어의 시퀀스가 일어날 가능성** 을 확률로 표현한 것

그림 5-26 샘플 말뭉치로 "you say goodbye and I say hello ."를 처리하는 RNNLM의 예



1.1 도입

배경

Q. 다음 □ 에 들어갈 단어는?

“**철수**가 방에서 졸고있다. **어머니**가 그 방에 들어오셨다. **어머니**가 □에게 과일을 주셨다.”

1.1 도입
배경

Q. 다음 □ 에 들어갈 단어는?

“**철수**가 방에서 졸고있다. **어머니**가 그 방에 들어오셨다. **어머니**가 □에게 과일을 주셨다.”

A. □ = 철수

“**철수**가 방에서 졸고있다. **어머니**가 그 방에 들어오셨다. **어머니**가 **철수**에게 과일을 주셨다.”

1.1 도입

배경

“**철수**가 방에서 졸고있다. **어머니**가 그 방에 들어오셨다. **어머니**가 **철수**에게 과일을 주셨다.”

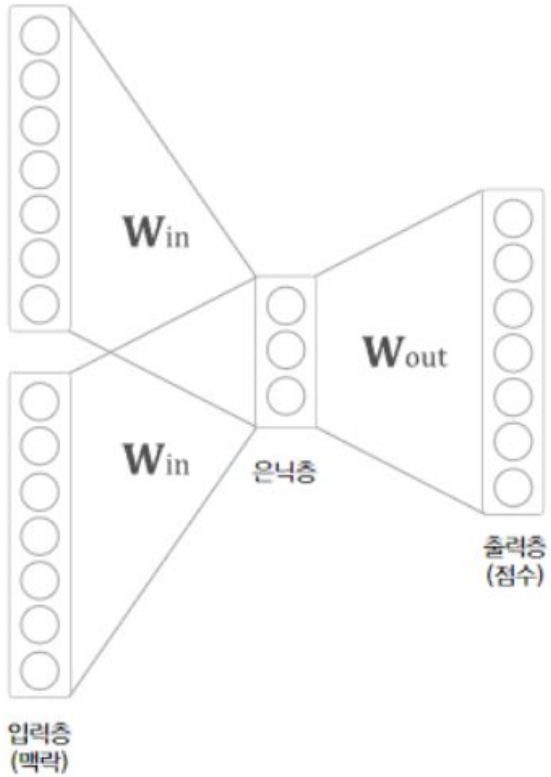
⇒ 단어를 추측하기 위해 긴 맥락이 필요한 상황

1.1 도입

배경

- 언어 모델로서의 CBOW

⇒ 맥락의 단어 **순서를 무시**

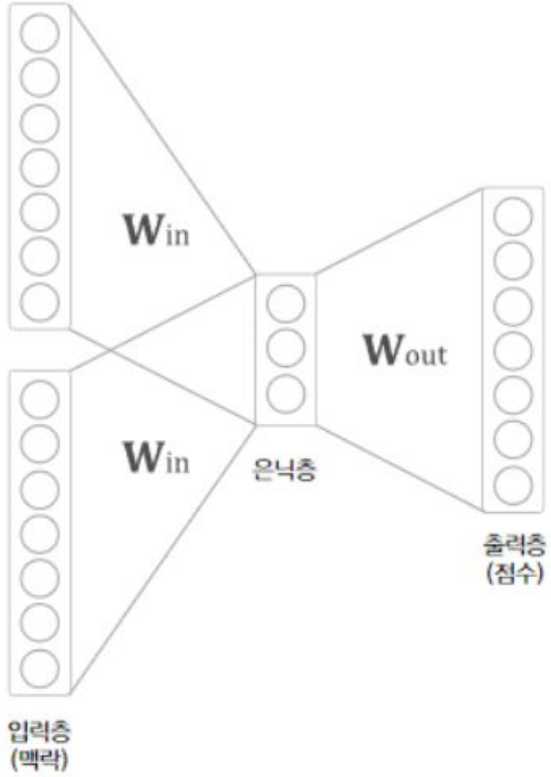


1.1 도입

배경

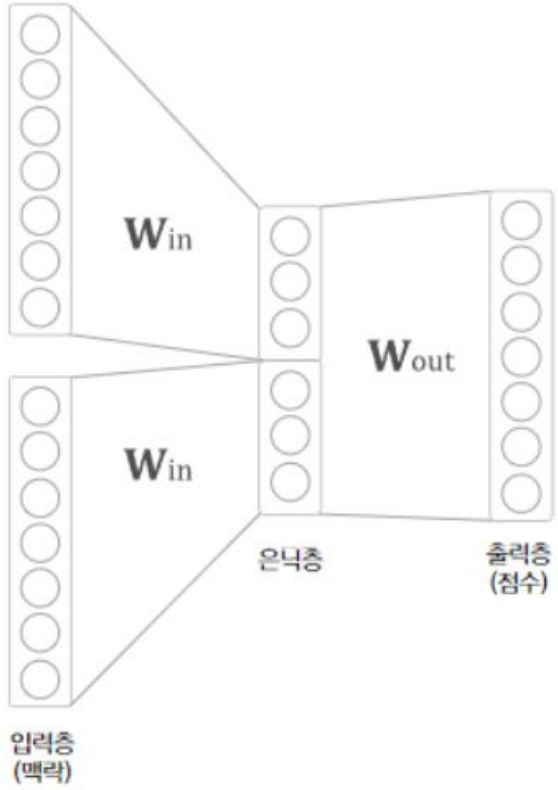
- 언어 모델로서의 CBOW

⇒ 맥락의 단어 **순서를 무시**



- 맥락의 단어 벡터를 연결

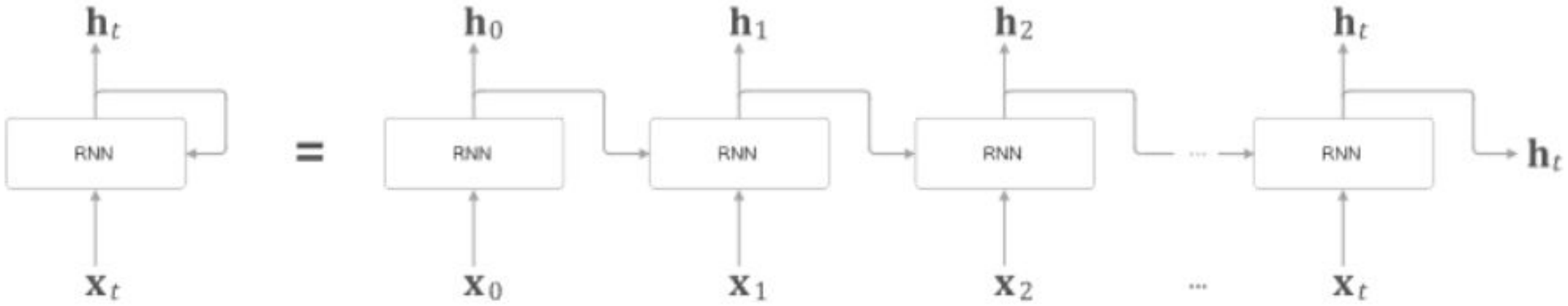
⇒ 맥락의 크기 ∝ **가중치 매개변수**



1.2 RNN이란

- RNN(Recurrent Neural Network)

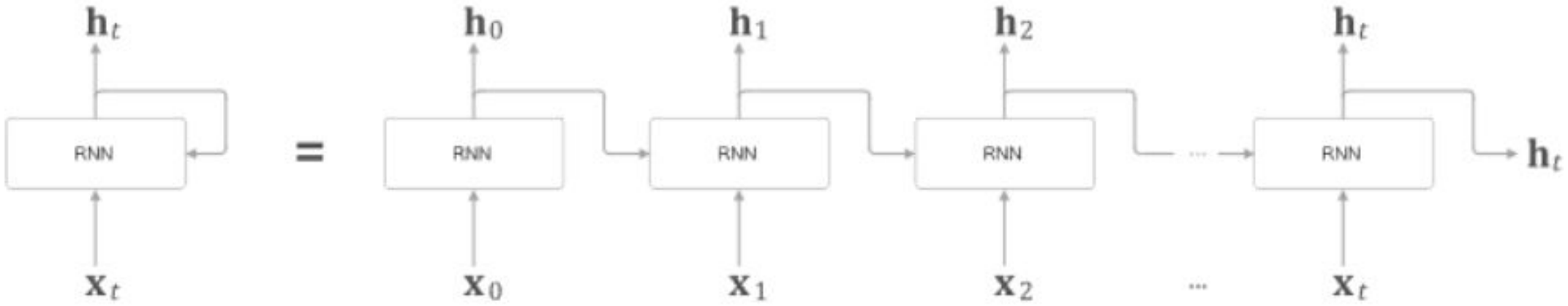
: 순환 신경망



1.2 RNN이란

- RNN(Recurrent Neural Network)

: 순환 신경망

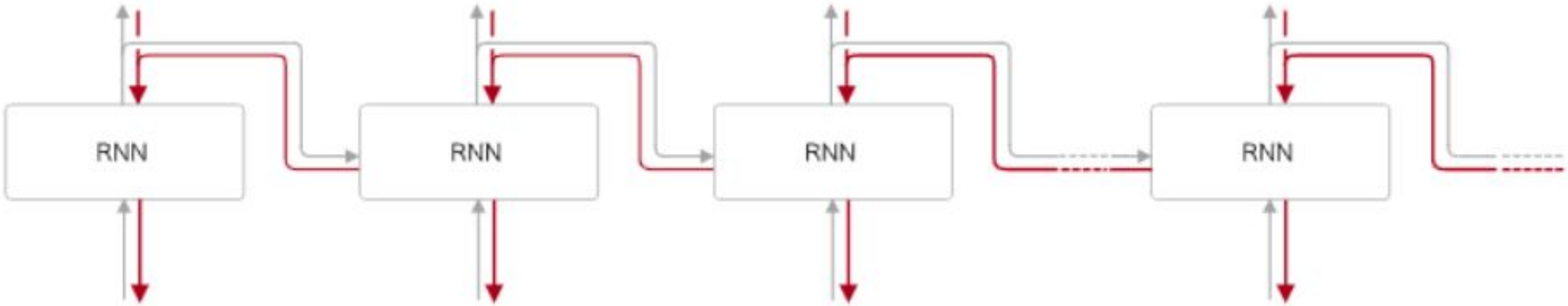


$$\boldsymbol{h}_t = \tanh(\boldsymbol{h}_{t-1}\boldsymbol{W}_h + \boldsymbol{x}_t\boldsymbol{W}_x + \boldsymbol{b})$$

1.2 RNN이란

- BPTT(Backpropagation Through Time)

: RNN에서 시간축을 따라 오차역전파를 확장한 알고리즘

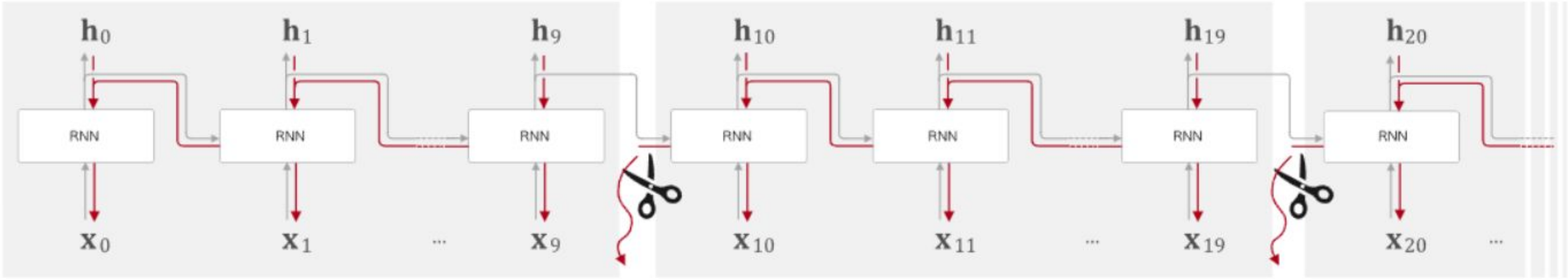


1.2 RNN이란

- Truncated BPTT

: 긴 시계열 데이터를 처리할 때, 일정한 길이로

연결을 잘라서 **부분적으로 역전파를 수행** 하는 알고리즘

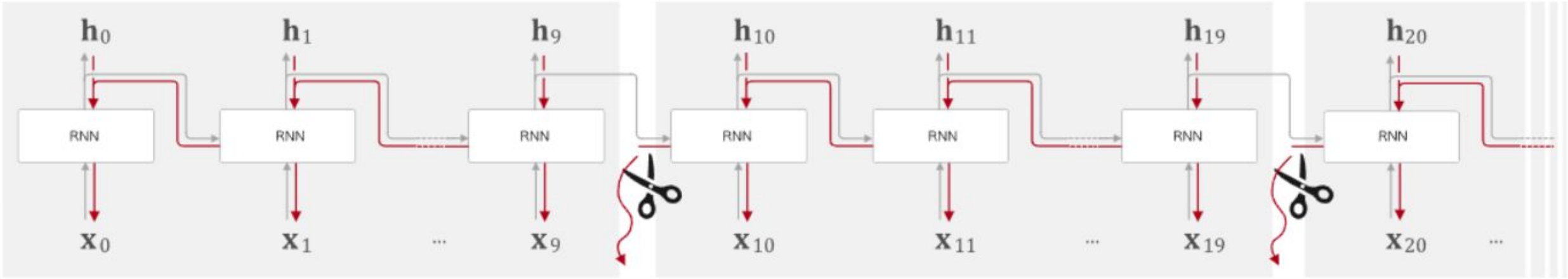


1.2 RNN이란

- Truncated BPTT

: 긴 시계열 데이터를 처리할 때, 일정한 길이로

연결을 잘라서 **부분적으로 역전파를 수행** 하는 알고리즘



∵ **순전파의 연결 유지 ⇒ 순서가 존재하는 데이터의 학습 가능**

1.2 RNN이란

- **Truncated BPTT의 미니배치 학습**

: 시작 위치를 오프셋만큼 이동 후,
데이터를 순서대로 제공

Ex) 길이가 1,000인 시계열 데이터에 대해,
미니배치의 수를 두 개로, 시각의 길이를 10개 단위로 잘라 Truncated BPTT 학습

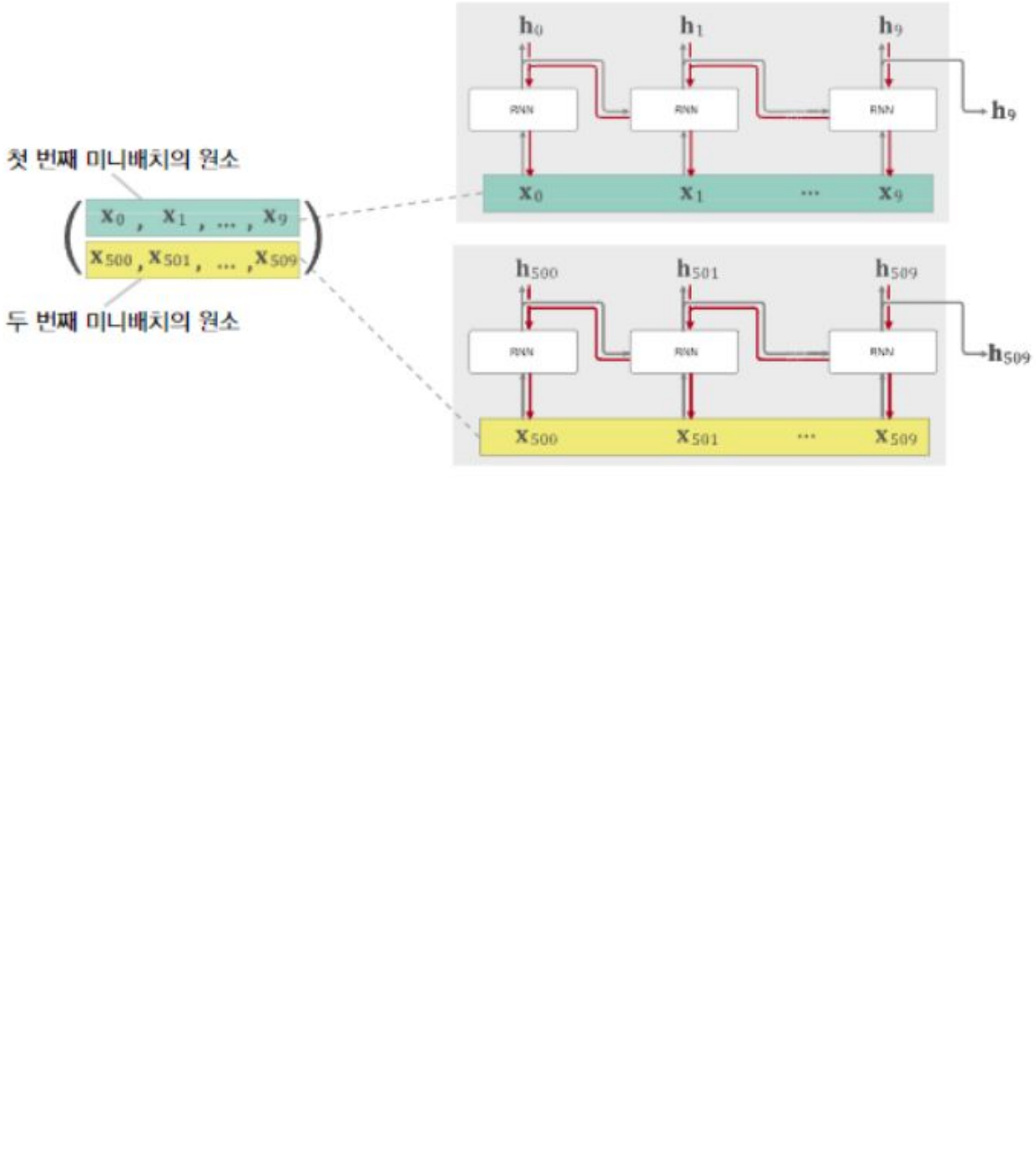
1.2 RNN이란

- **Truncated BPTT의 미니배치 학습**

: 시작 위치를 오프셋만큼 이동 후, 데이터를 순서대로 제공

Ex) 길이가 1,000인 시계열 데이터에 대해,

미니배치의 수를 두 개로, 시각의 길이를 10개 단위로 잘라 Truncated BPTT 학습



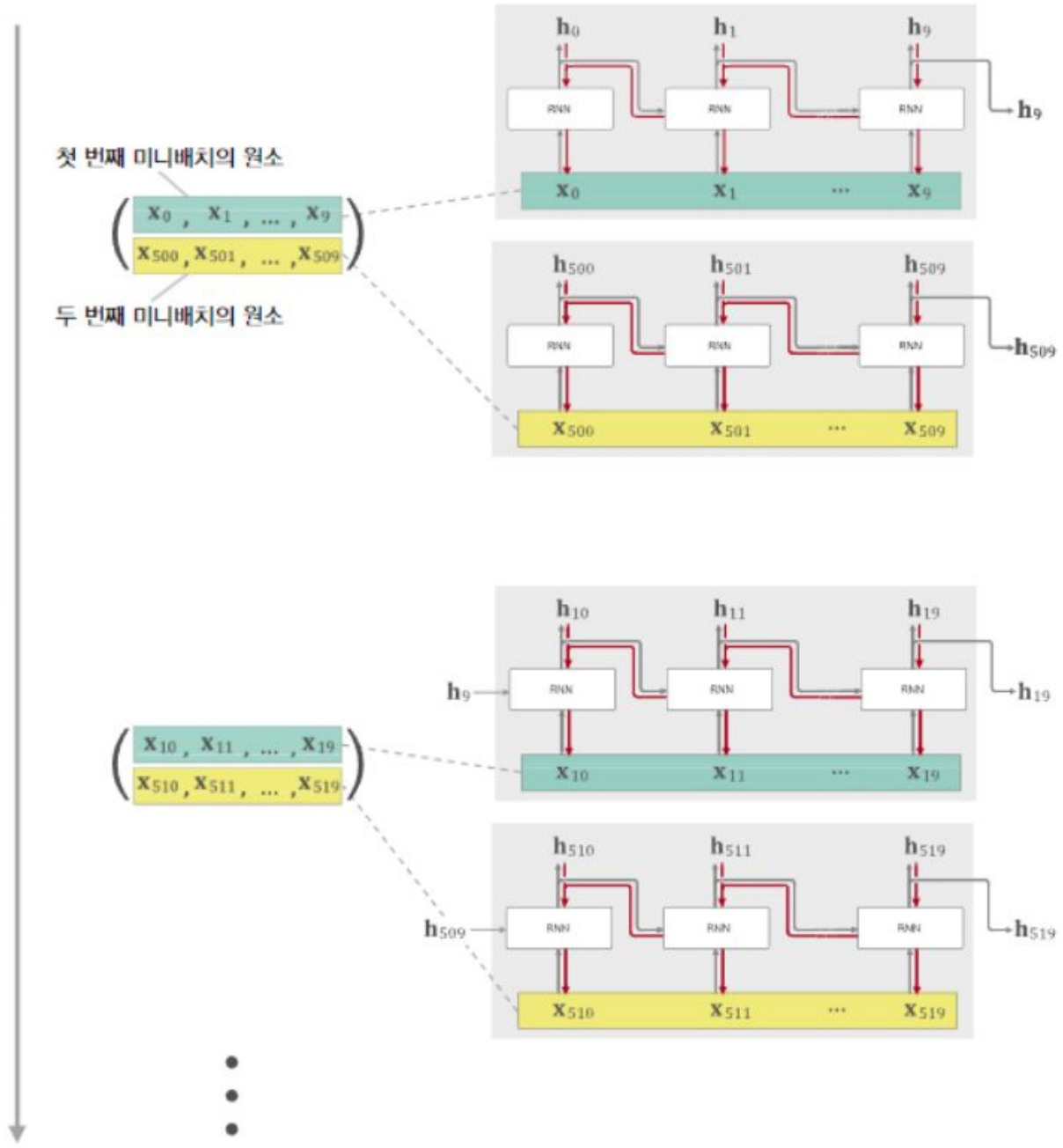
1.2 RNN이란

- **Truncated BPTT의 미니배치 학습**

: 시작 위치를 오프셋만큼 이동 후, 데이터를 순서대로 제공

Ex) 길이가 1,000인 시계열 데이터에 대해,

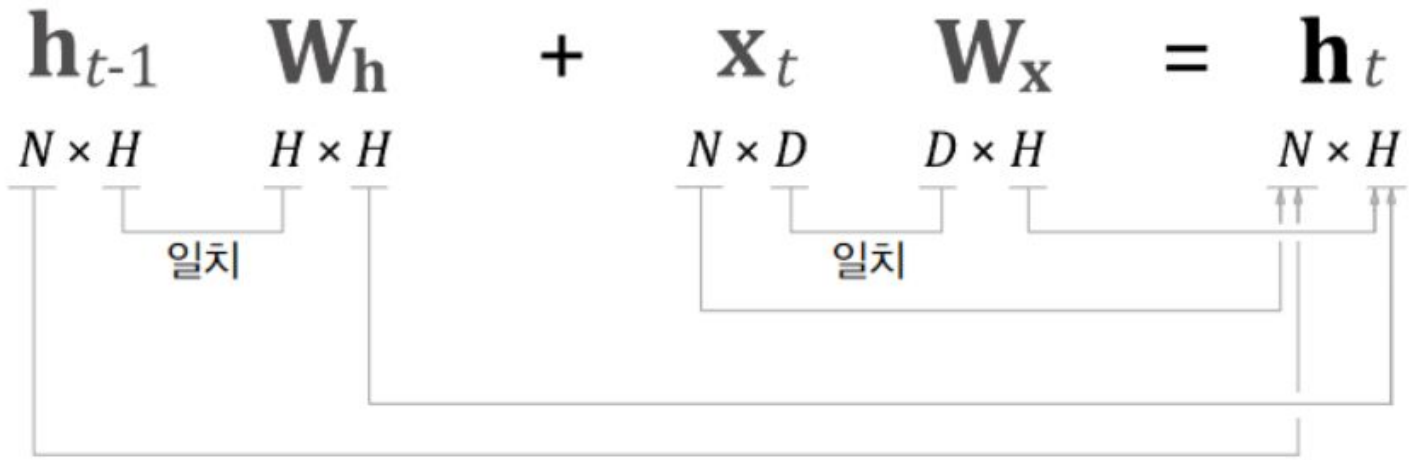
미니배치의 수를 두 개로, 시각의 길이를 10개 단위로 잘라 Truncated BPTT 학습



1.2 RNN이란

- **Time RNN 클래스 구현**
  1. RNN
  2. Time RNN

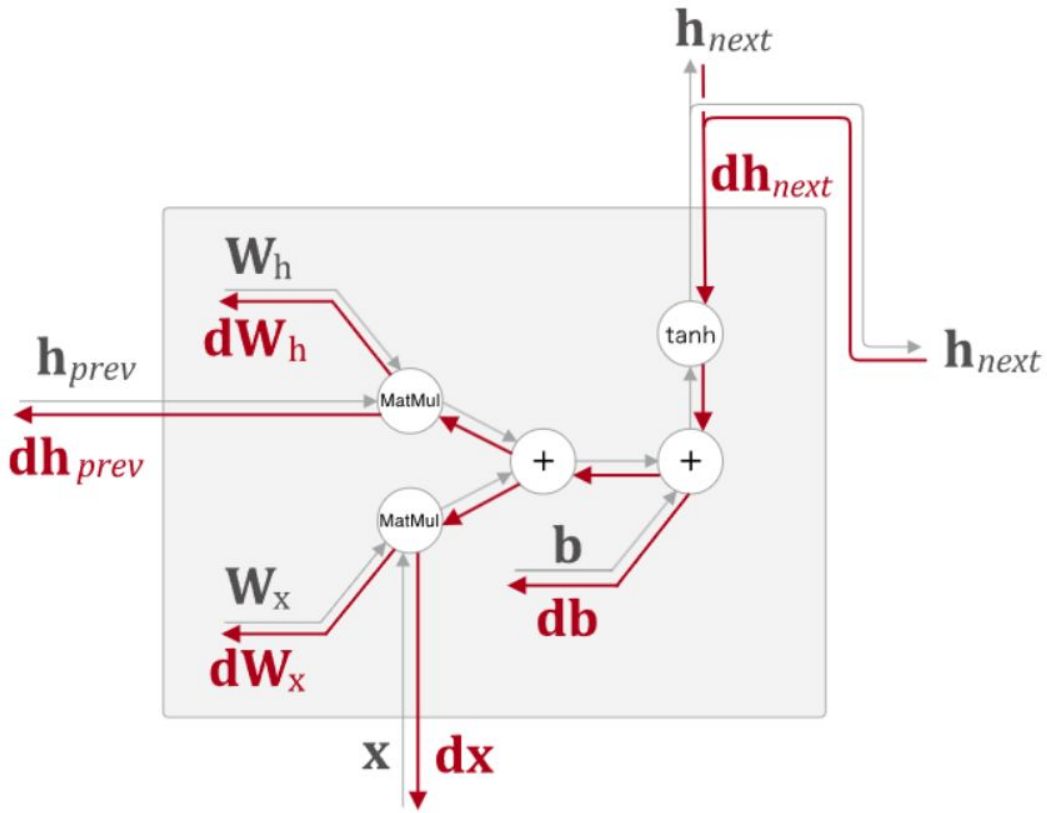
$$\boldsymbol{h}_t = \tanh(\boldsymbol{h}_{t-1}\boldsymbol{W}_h + \boldsymbol{x}_t\boldsymbol{W}_x + \boldsymbol{b})$$



1.2 RNN이란

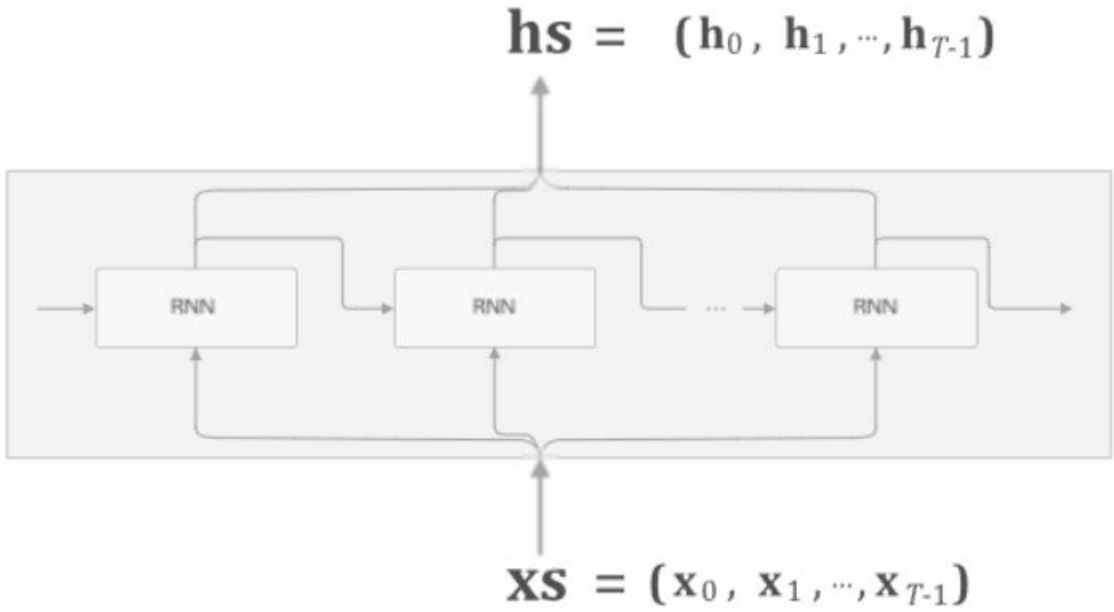
- Time RNN 클래스 구현
  1. RNN
  2. Time RNN

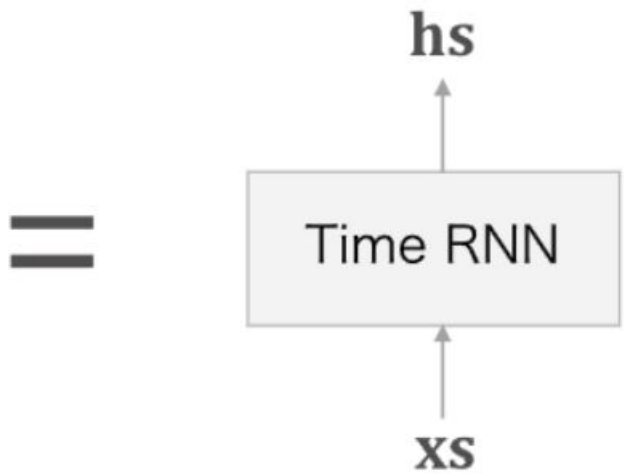
$$\boldsymbol{h}_t = \tanh(\boldsymbol{h}_{t-1}\boldsymbol{W}_h + \boldsymbol{x}_t\boldsymbol{W}_x + \boldsymbol{b})$$



1.2 RNN이란

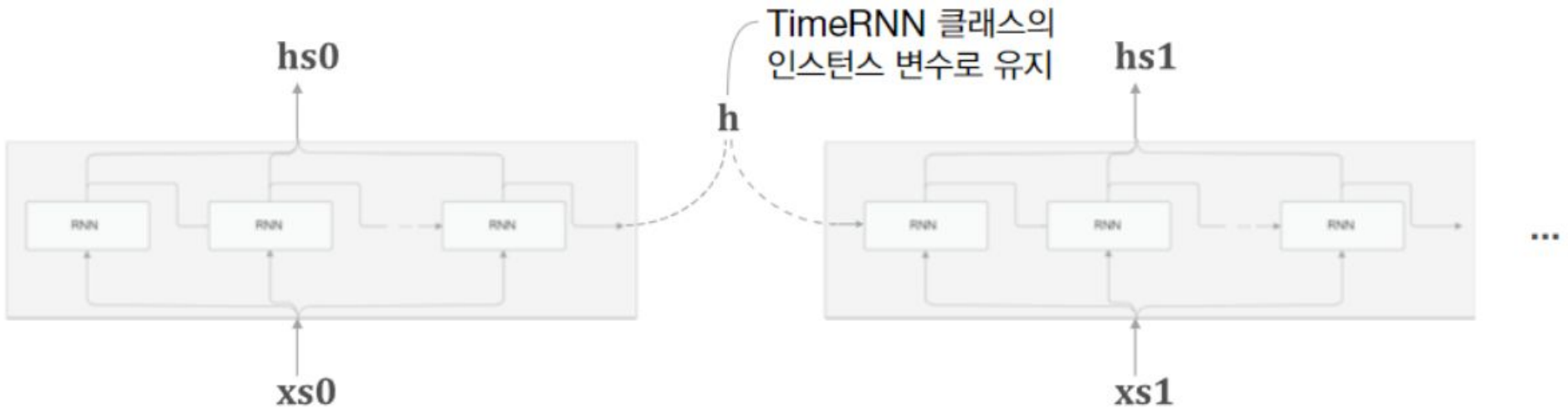
- Time RNN 클래스 구현
  1. RNN
  2. Time RNN





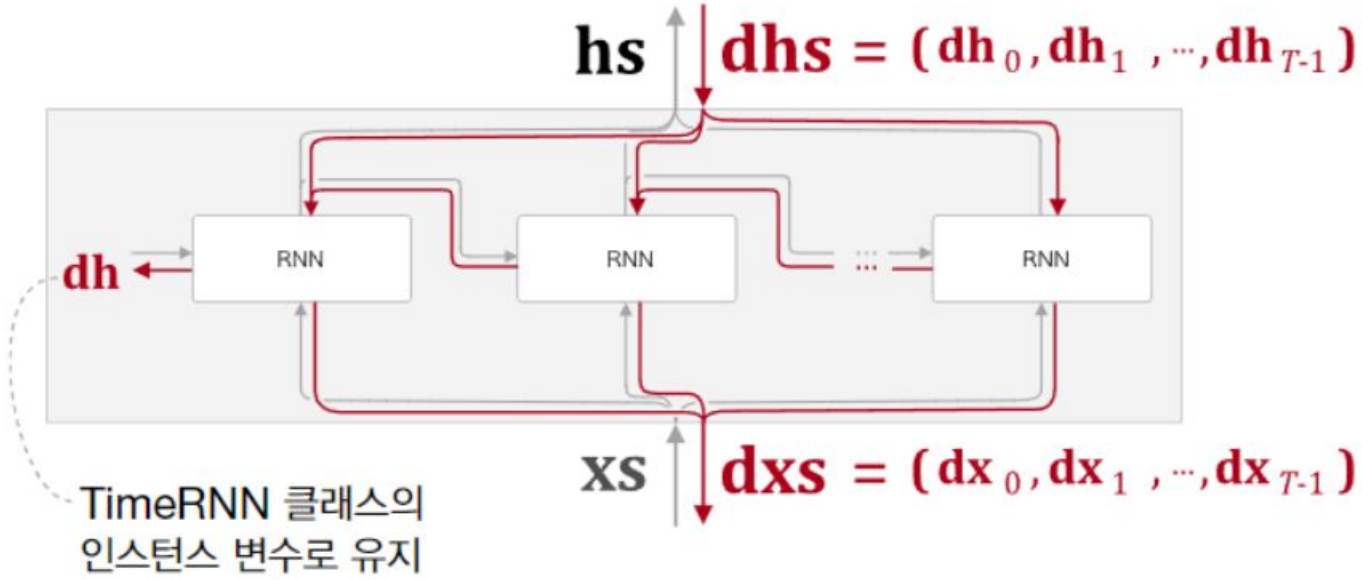
1.2 RNN이란

- Time RNN 클래스 구현
  1. RNN
  2. Time RNN



1.2 RNN이란

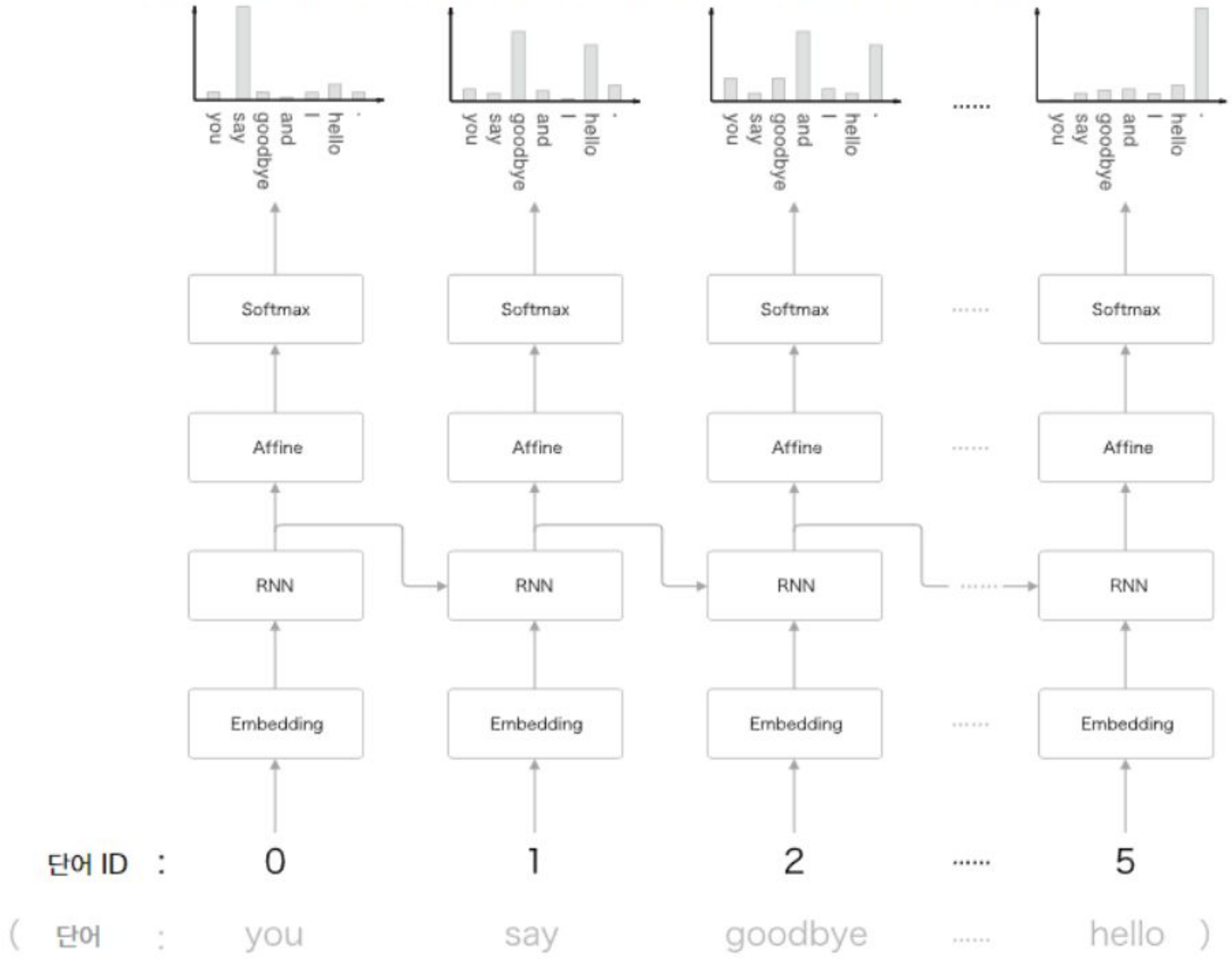
- Time RNN 클래스 구현
  1. RNN
  2. Time RNN



1.3 RNNLM

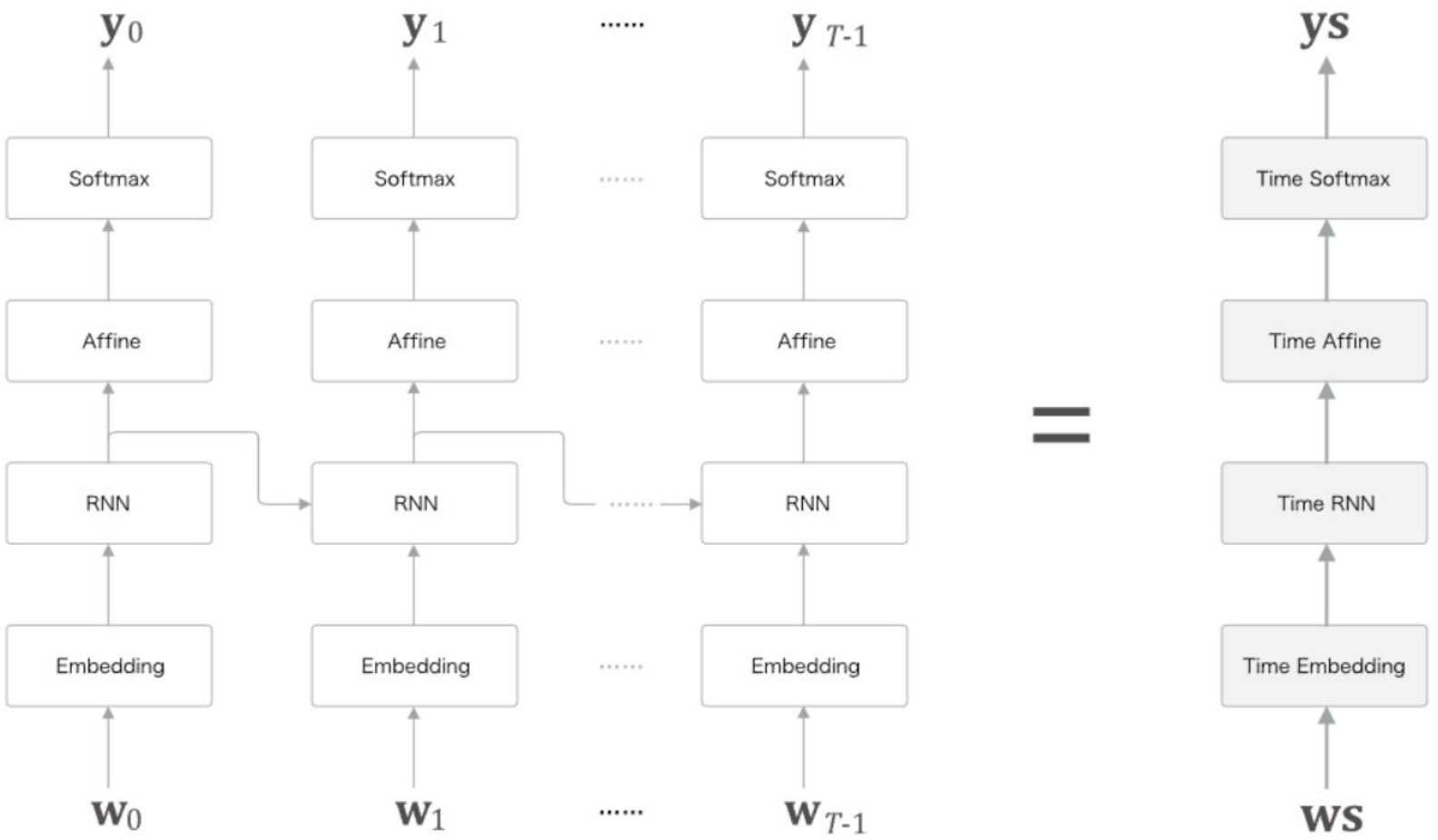
- **RNN을 사용한 언어 모델**
  1. 전체 구조
  2. 추가된 Time 계층

그림 5-26 샘플 말뭉치로 "you say goodbye and I say hello ."를 처리하는 RNNLM의 예



1.3 RNNLM

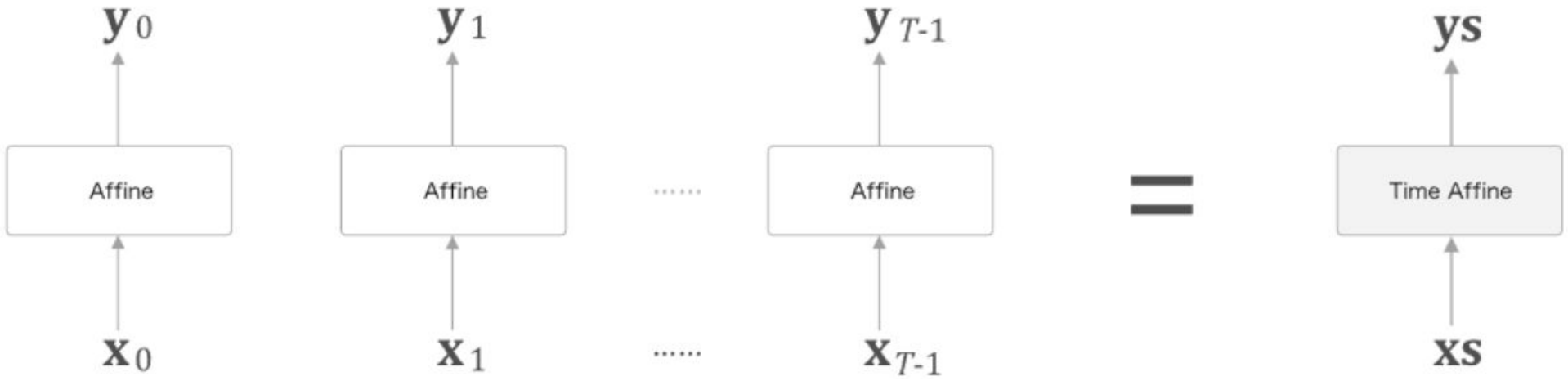
- **RNN을 사용한 언어 모델**
  1. **전체 구조**
  2. 추가된 Time 계층



1.3 RNNLM

- **RNN을 사용한 언어 모델**
  1. 전체 구조
  2. **추가된 Time 계층**
     - **Time Affine과 Time Embedding**

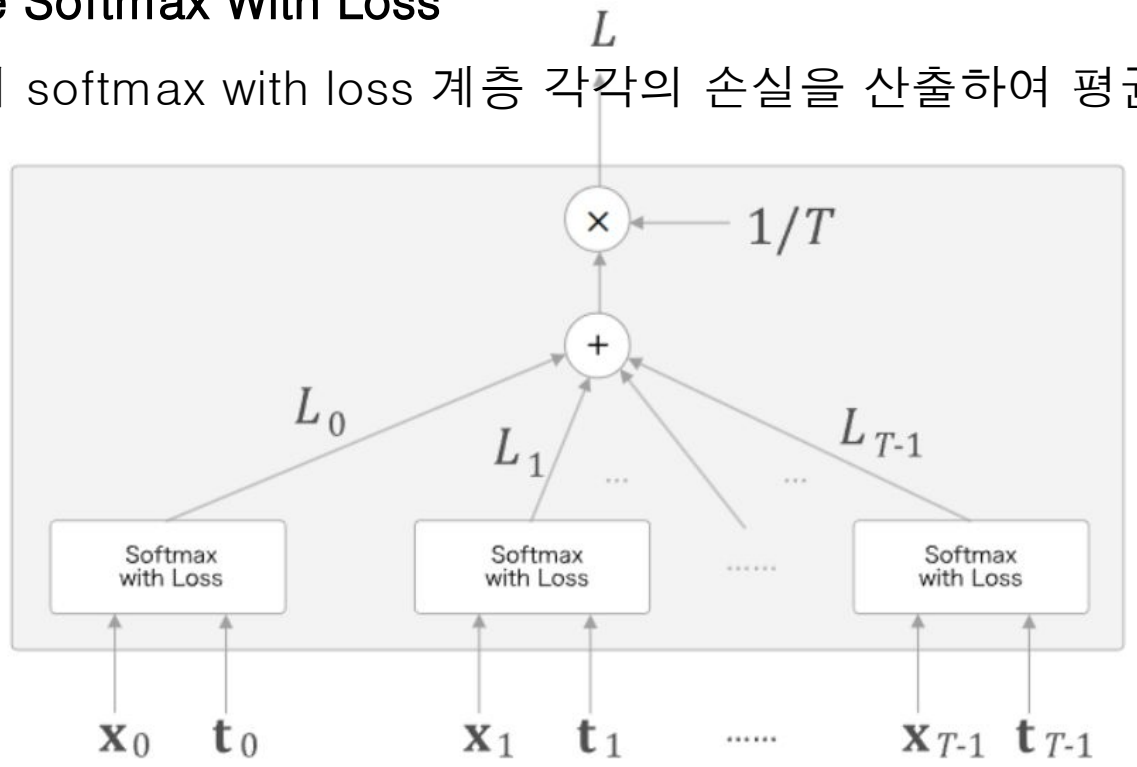
       : T개의 각 계층을 준비하여, 각 시각의 데이터를 개별적으로 처리



1.3 RNNLM

- **RNN을 사용한 언어 모델**
  1. 전체 구조
  2. **추가된 Time 계층**
     - **Time Softmax With Loss**

: T개의 softmax with loss 계층 각각의 손실을 산출하여 평균 값을 계산



=

$$\Rightarrow L = \frac{1}{T}(L_0 + L_1 + \ldots + L_{T-1})$$

1.3 RNNLM

- **언어 모델의 평가**
  - **Perplexity :** 언어 모델의 예측 성능을 측정하는 지표

1.3 RNNLM

- **언어 모델의 평가**
  - **Perplexity :** 언어 모델의 예측 성능을 측정하는 지표

$$L = -\frac{1}{N}\sum_{n}\sum_{k} t_{nk} \log y_{nk}$$

$$\text{Perplexity} = e^{L}$$

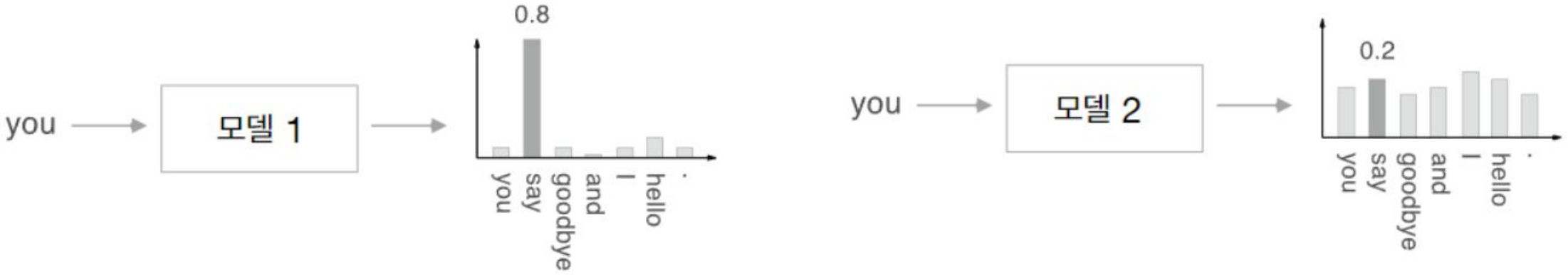
1.3 RNNLM

- **언어 모델의 평가**
  - **Perplexity :** 언어 모델의 예측 성능을 측정하는 지표
  - **분기 수 :** 다음에 출현할 수 있는 단어의 후보 수

1.3 RNNLM

- **언어 모델의 평가**
  - **Perplexity :** 언어 모델의 예측 성능을 측정하는 지표
  - **분기 수 :** 다음에 출현할 수 있는 단어의 후보 수

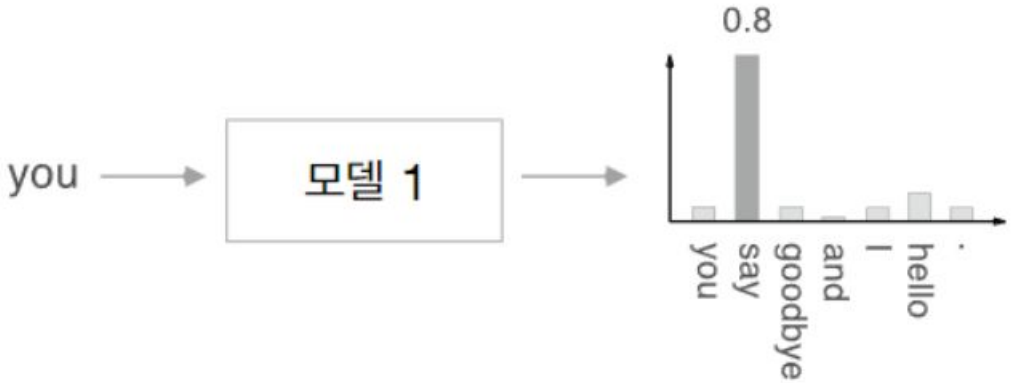
Ex) “**You** say goodbye and I say hello.”



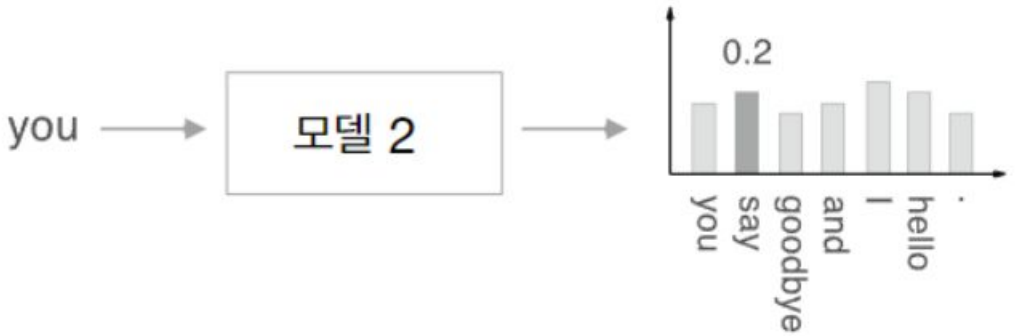
1.3 RNNLM

- **언어 모델의 평가**
  - **Perplexity :** 언어 모델의 예측 성능을 측정하는 지표
  - **분기 수 :** 다음에 출현할 수 있는 단어의 후보 수

Ex) “**You** say goodbye and I say hello.”



‘모델 1’의 perplexity = **1.25**

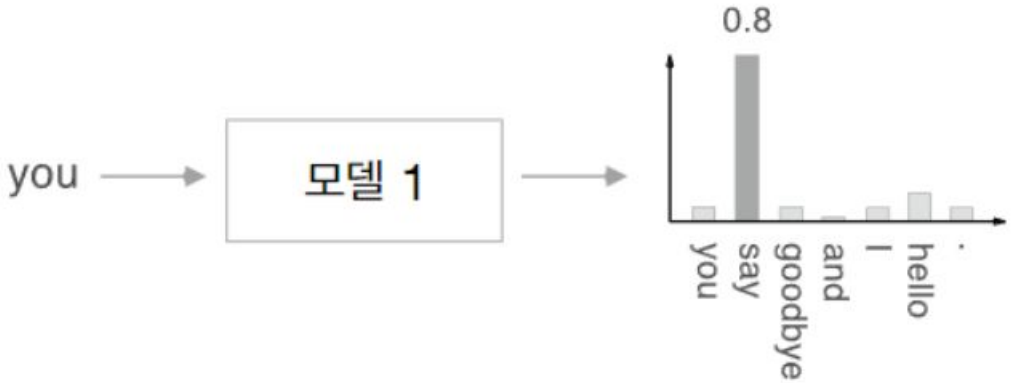


‘모델 2’의 perplexity = **5**

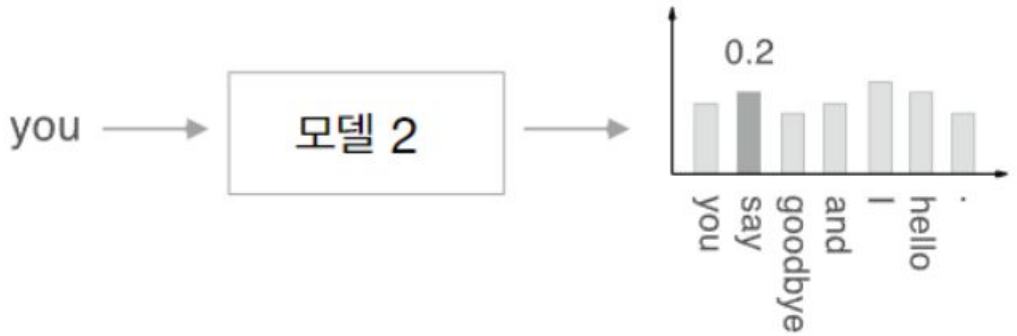
1.3 RNNLM

- **언어 모델의 평가**
  - **Perplexity :** 언어 모델의 예측 성능을 측정하는 지표
  - **분기 수 :** 다음에 출현할 수 있는 단어의 후보 수

Ex) "**You** say goodbye and I say hello."



'모델 1'의 perplexity = **1.25**

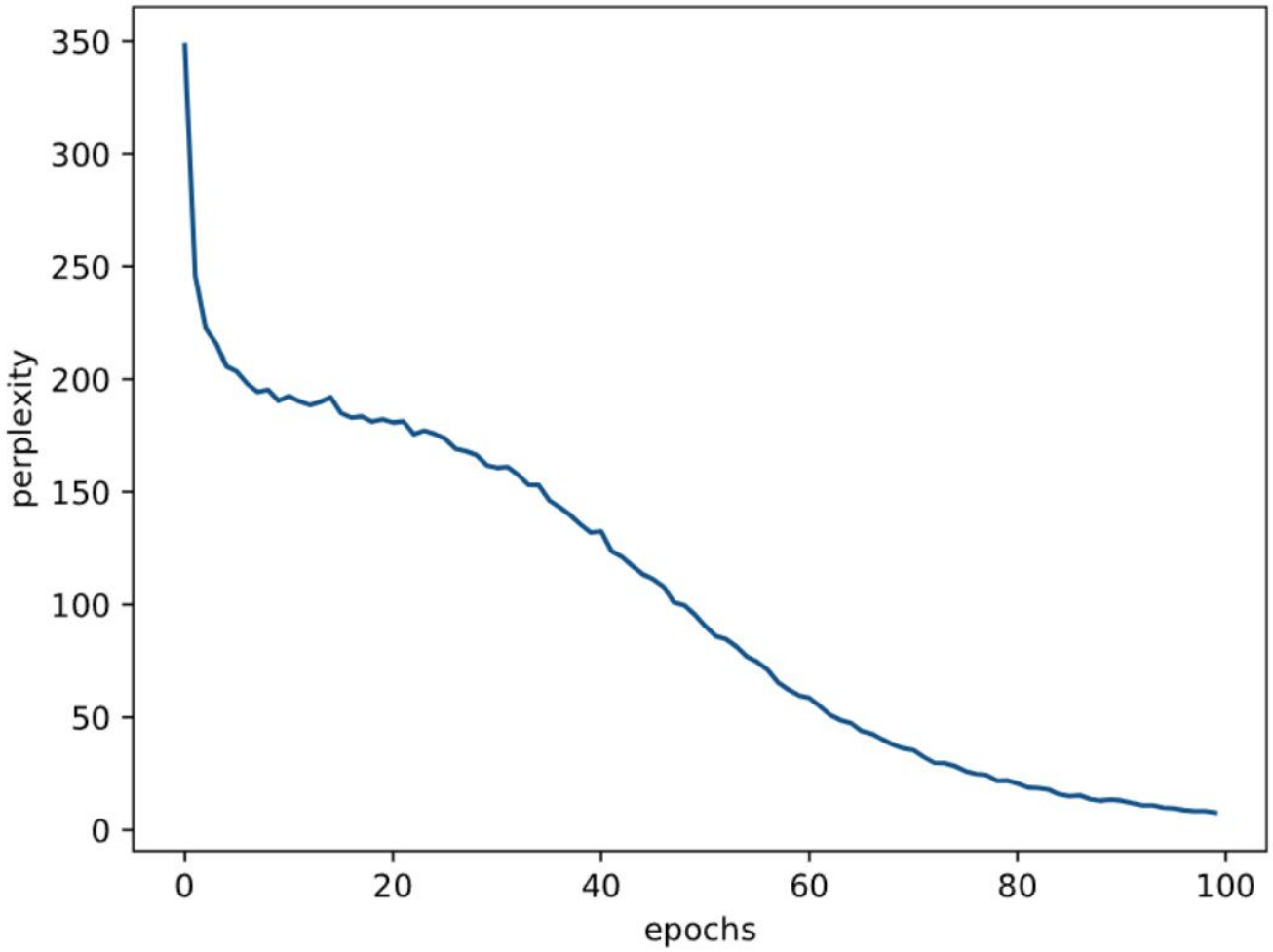


'모델 2'의 perplexity = **5**

∴ '모델 1'이 보다 좋은 성능의 모델

1.3 RNNLM

- **언어 모델의 평가**
  - PTB dataset
  - **처음 1,000개의 단어**로 학습 및 시각화



2. LSTM

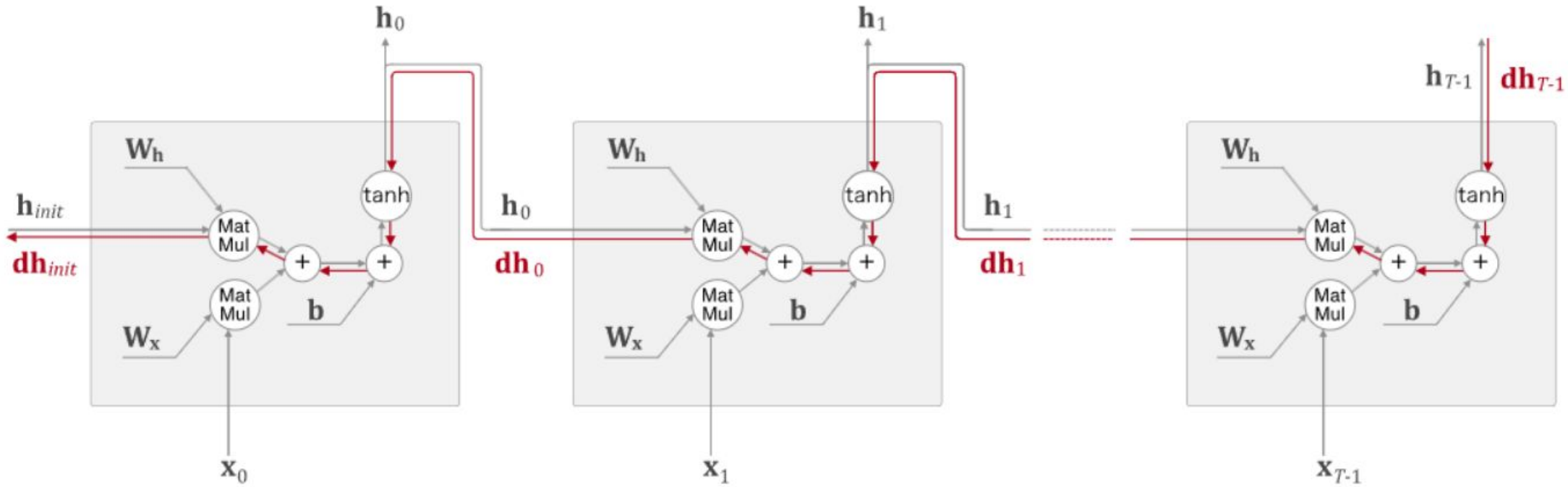
2.1 RNN의 한계

- **장기 의존 관계 학습의 어려움**
  - 기울기 폭발 : 역전파 과정에서 기울기가 기하급수적으로 커지는 현상
  - 기울기 소실 : 역전파 과정에서 기울기가 점차 작아져서 거의 0에 가까워지는 현상

⇒ **가중치 갱신에 문제** 가 생겨서 **모델의 학습** 이 제대로 되지 않음

2.1 RNN의 한계

- 기울기가 전파되며 거치는 노드
  - tanh 노드
  - MatMul 노드



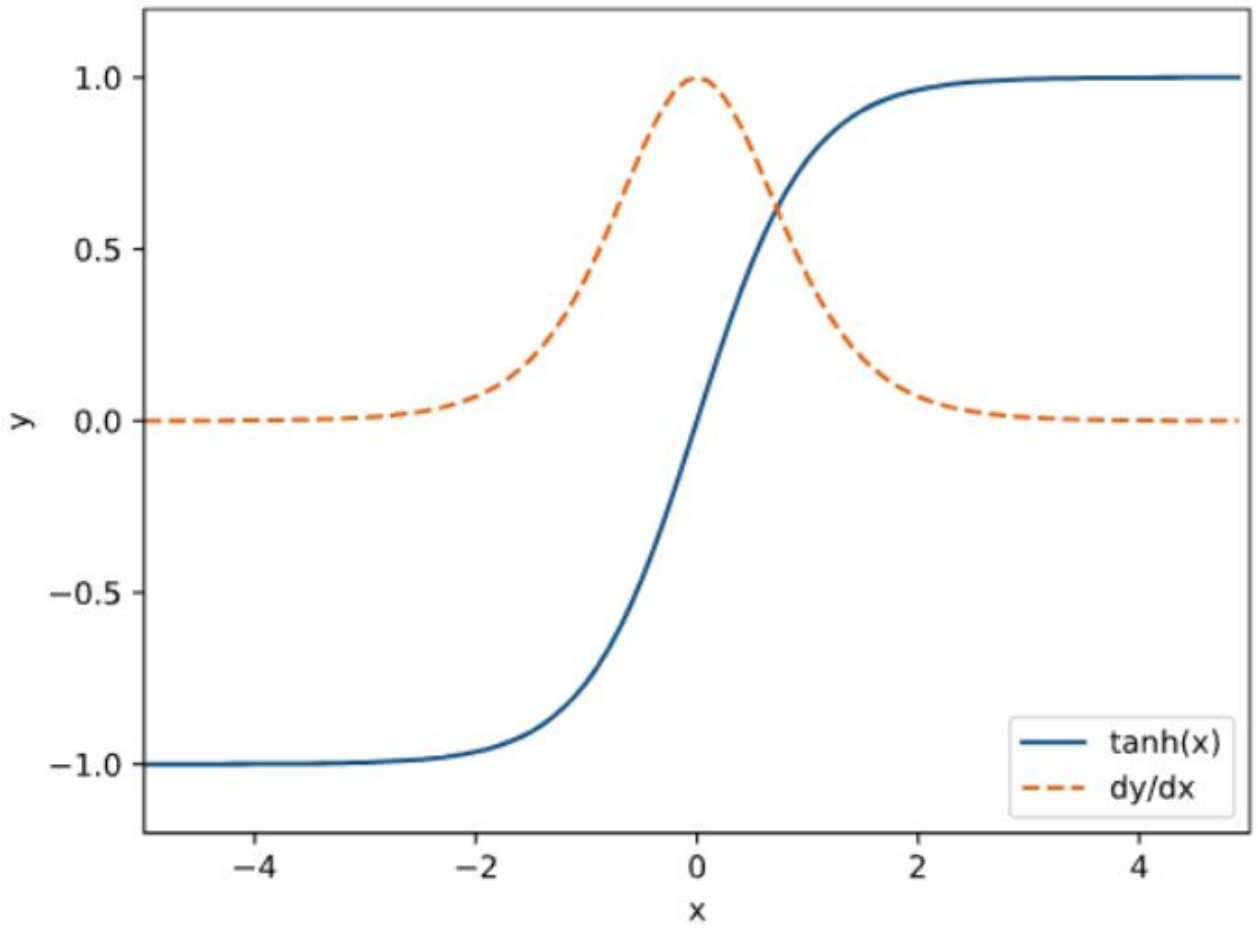
2.1 RNN의 한계

- tanh 노드
  - x가 커질수록 기울기가 0에 수렴

∴ **tanh 노드를 지날 때 마다 기울기 감소**

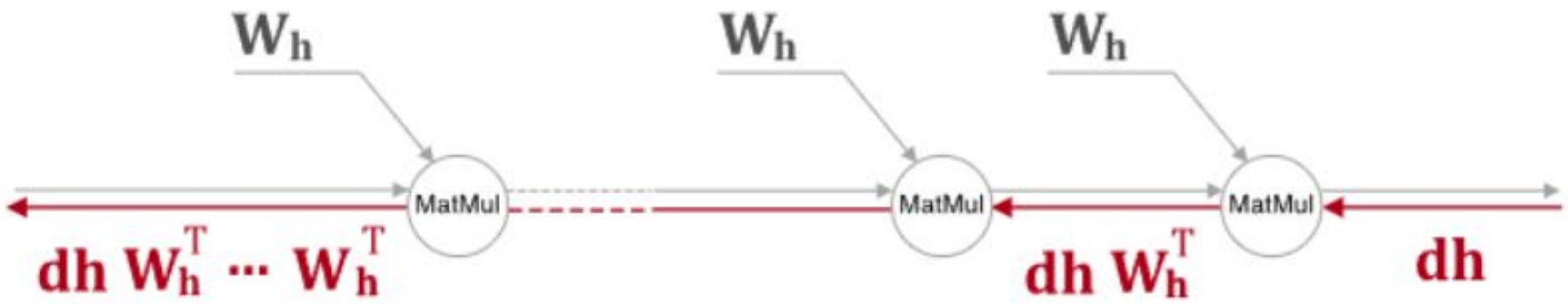
$$y = \tanh(x)$$

$$\frac{dy}{dx} = 1 - y^2$$



2.1 RNN의 한계

- MatMul 노드
  - 동일한 가중치의 반복적인 행렬곱 계산

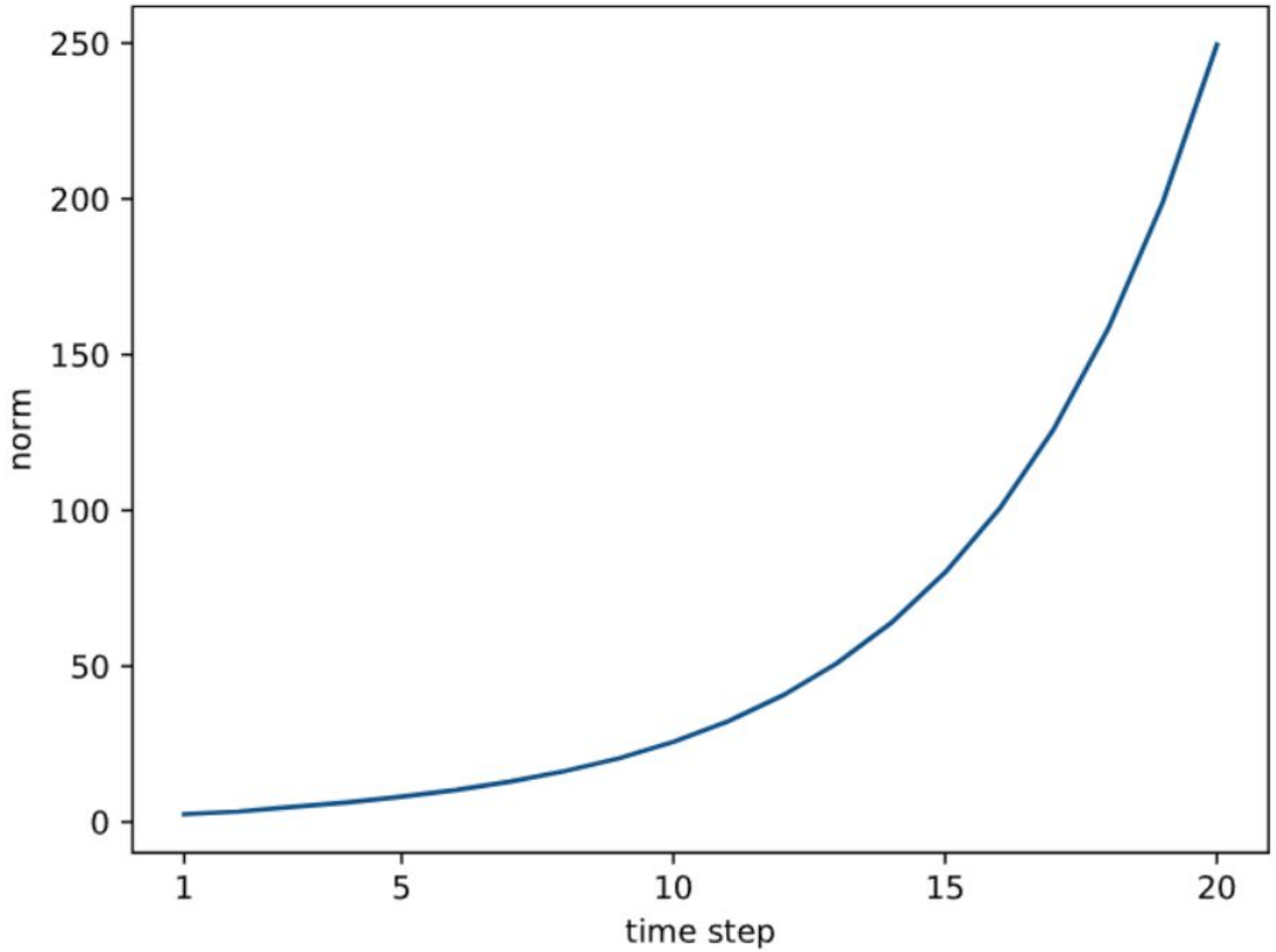


2.1 RNN의 한계

- MatMul 노드
  - 반복적인 행렬곱으로 인한 가중치 폭발

```
dh = np.ones((N, H))
np.random.seed(3)
Wh = np.random.randn(H, H)
# Wh = np.random.randn(H, H) * 0.5

norm_list = []
for t in range(T):
    dh = np.matmul(dh, Wh.T)
    norm = np.sqrt(np.sum(dh**2)) / N
```

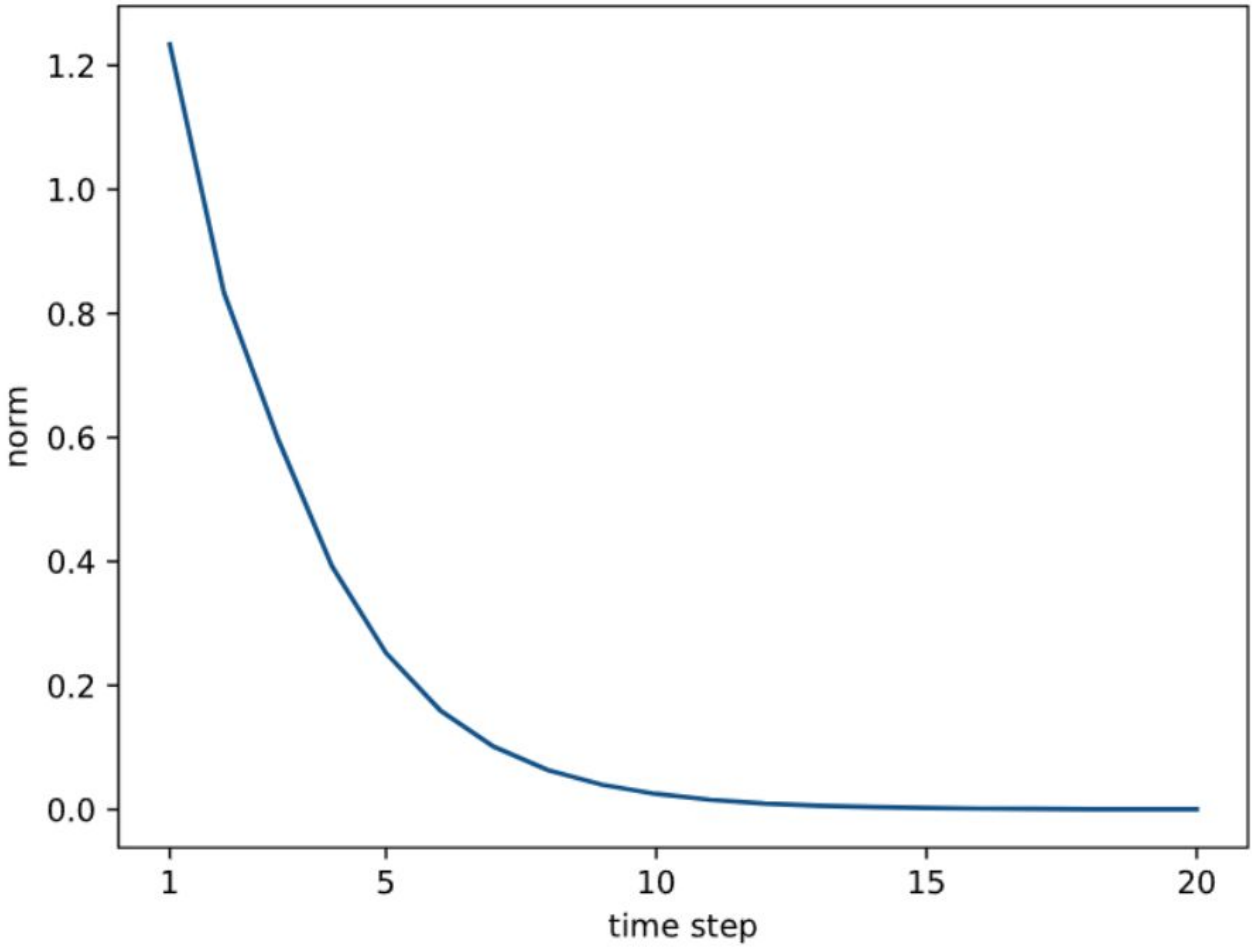


2.1 RNN의 한계

- MatMul 노드
  - 반복적인 행렬곱으로 인한 가중치 소실

```
dh = np.ones((N, H))
np.random.seed(3)
# Wh = np.random.randn(H, H)
Wh = np.random.randn(H, H) * 0.5

norm_list = []
for t in range(T):
    dh = np.matmul(dh, Wh.T)
    norm = np.sqrt(np.sum(dh**2)) / N
```



2.1 RNN의 한계

1. 기울기 폭발 대책

- 기울기 클리핑

: 기울기가 threshold 값을

초과하지 못하도록 하는 알고리즘

$$\text{if } \|\hat{\mathbf{g}}\| \geq threshold:$$

$$\hat{\mathbf{g}} = \frac{threshold}{\|\hat{\mathbf{g}}\|}\hat{\mathbf{g}}$$

2.1 RNN의 한계

1. 기울기 폭발 대책

– 기울기 클리핑

: 기울기가 threshold 값을

초과하지 못하도록 하는 알고리즘

$$\text{if } \|\hat{\mathbf{g}}\| \geq threshold:$$

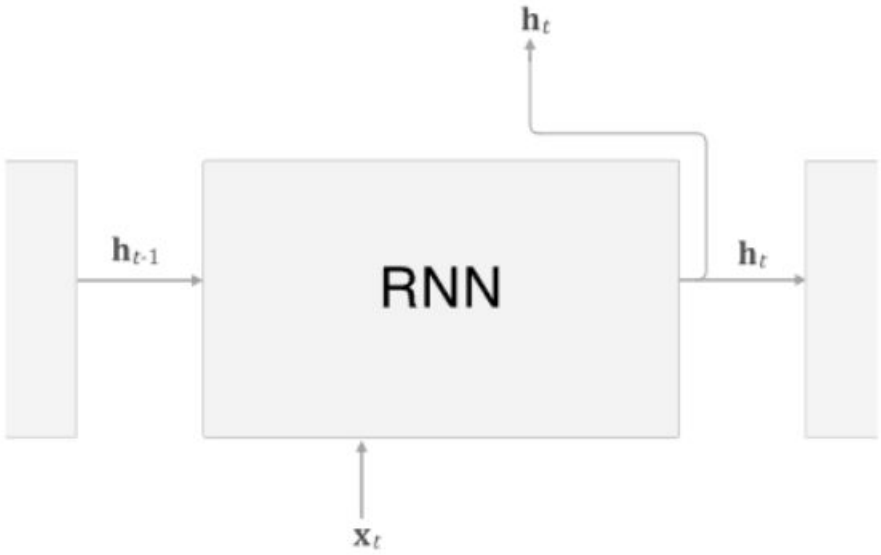
$$\hat{\mathbf{g}} = \frac{threshold}{\|\hat{\mathbf{g}}\|}\hat{\mathbf{g}}$$

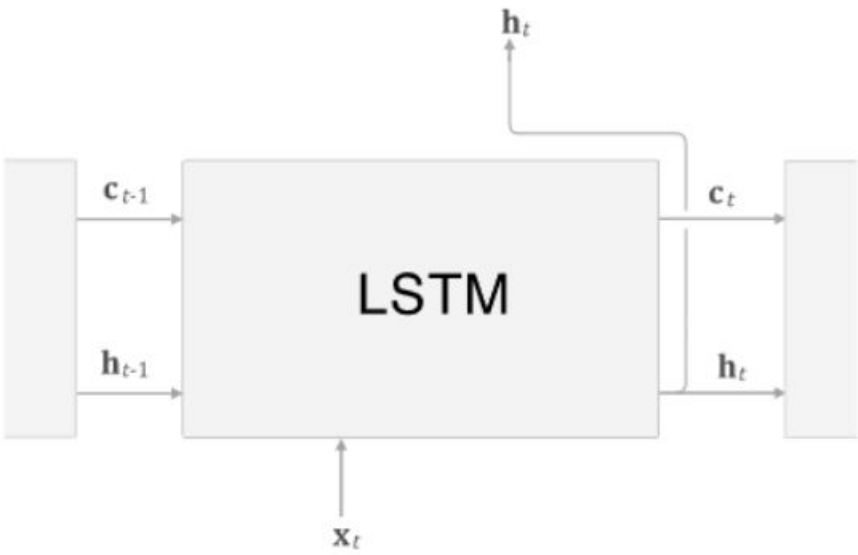
2. 기울기 소실 대책

- 게이트가 추가된 RNN
  - **LSTM(Long Short-Term Memory)**
  - GRU(Gated Recurrent Unit)

2.2 LSTM이란

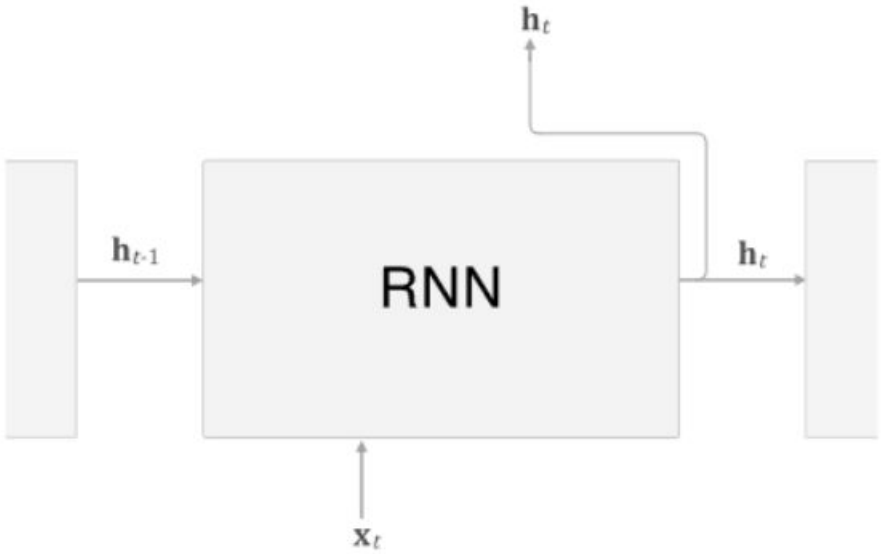
- RNN과 LSTM 인터페이스 비교

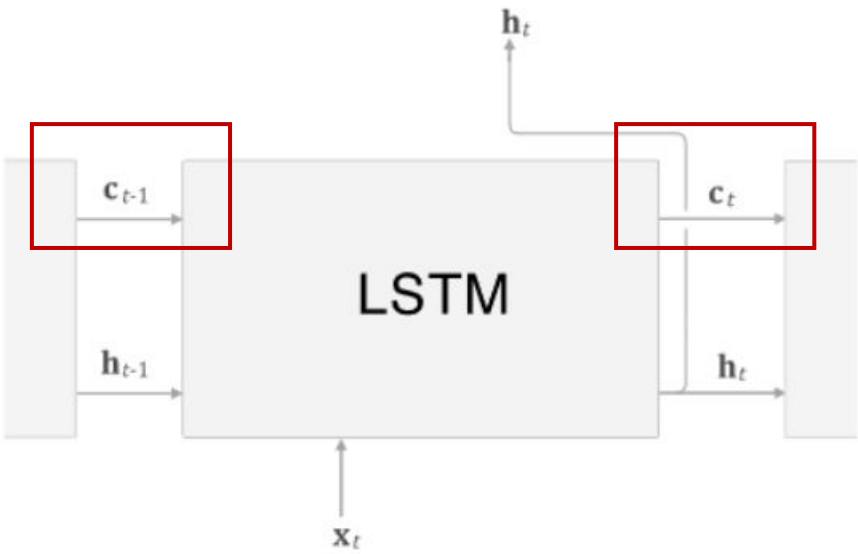




2.2 LSTM이란

- **RNN과 LSTM 인터페이스 비교**
  - 기억 셀(c) : 과거 정보를 일정 기간 동안 저장하고 필요할 때 이를 사용할 수 있도록 하는 메모리 장치

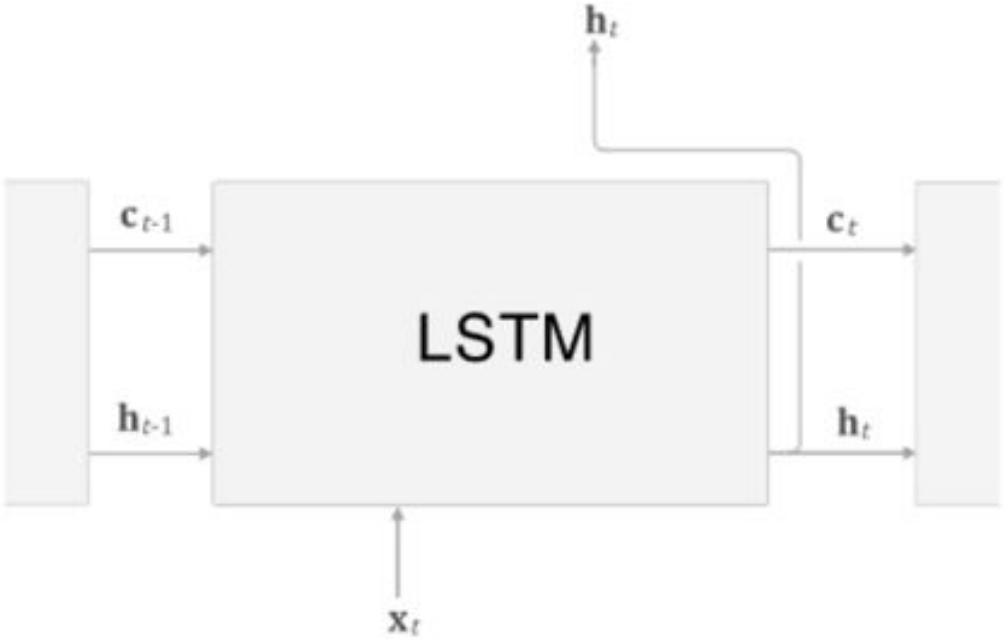




2.2 LSTM이란

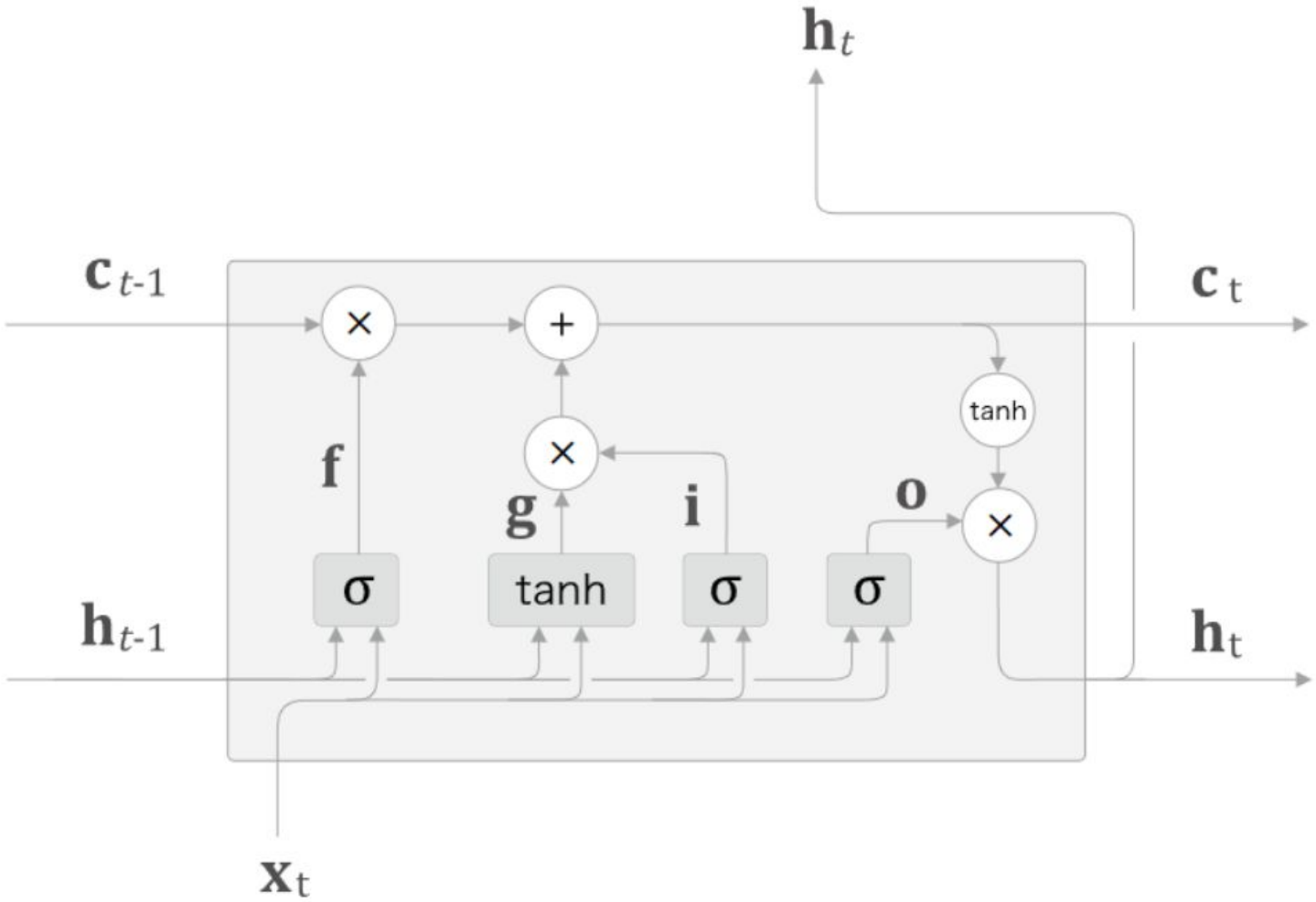
Q1. **단기 기억**을 어떻게 **장기적으로 기억** 할 것인가?

Q2. 어떻게 **기울기 소실을 해결** 할 것인가?



2.2 LSTM이란

- **게이트가 추가된 LSTM 계층**
  - Output 게이트
  - Forget 게이트
  - Input 게이트



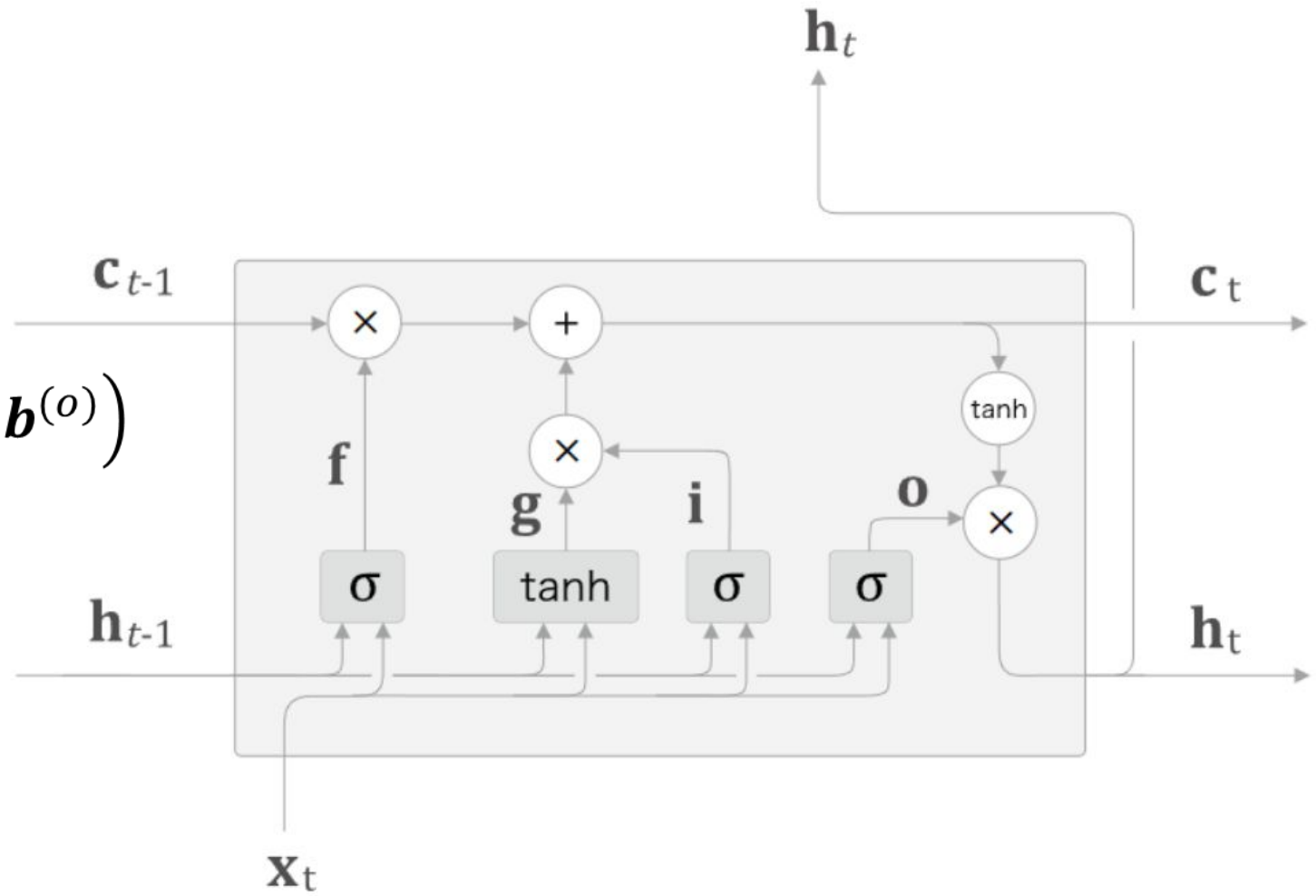
2.2 LSTM이란

- **게이트가 추가된 LSTM 계층**
  - Output 게이트

  : 현재 상태의 정보(c_t)를 얼마나 출력할지 결정

$$\boldsymbol{o} = \sigma\left(\boldsymbol{x}_t \boldsymbol{W}_x^{(o)} + \boldsymbol{h}_{t-1} \boldsymbol{W}_x^{(o)} + \boldsymbol{b}^{(o)}\right)$$

$$\boldsymbol{h}_t = \boldsymbol{o} \odot tanh(\boldsymbol{c}_t)$$

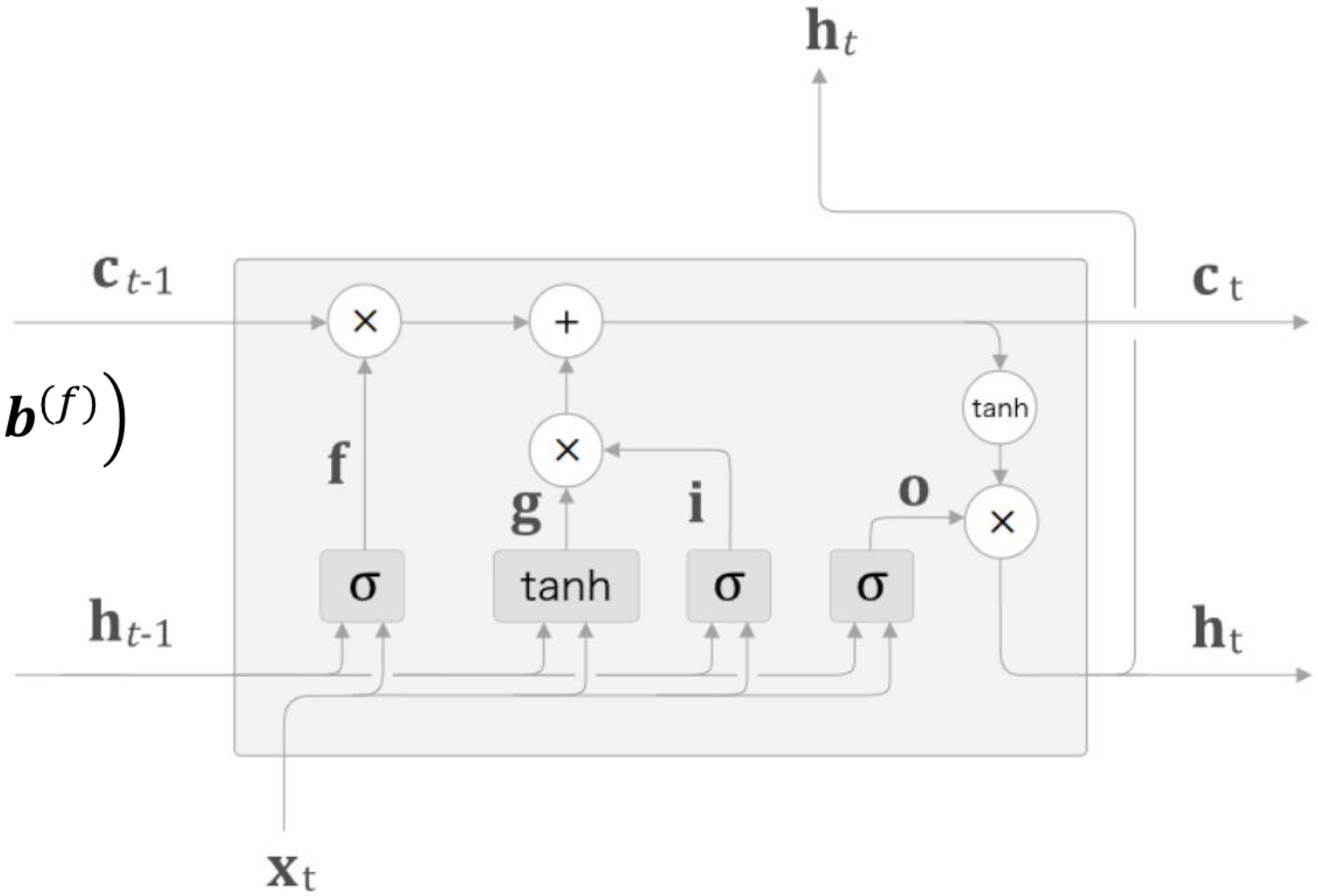


2.2 LSTM이란

- **게이트가 추가된 LSTM 계층**
  - Forget 게이트

: 이전 상태의 정보(c_{t-1})를 얼마나 잊을지 결정

$$\boldsymbol{f} = \sigma\left(\boldsymbol{x}_t \boldsymbol{W}_x^{(f)} + \boldsymbol{h}_{t-1} \boldsymbol{W}_x^{(f)} + \boldsymbol{b}^{(f)}\right)$$



2.2 LSTM이란

- **게이트가 추가된 LSTM 계층**
  - Input 게이트

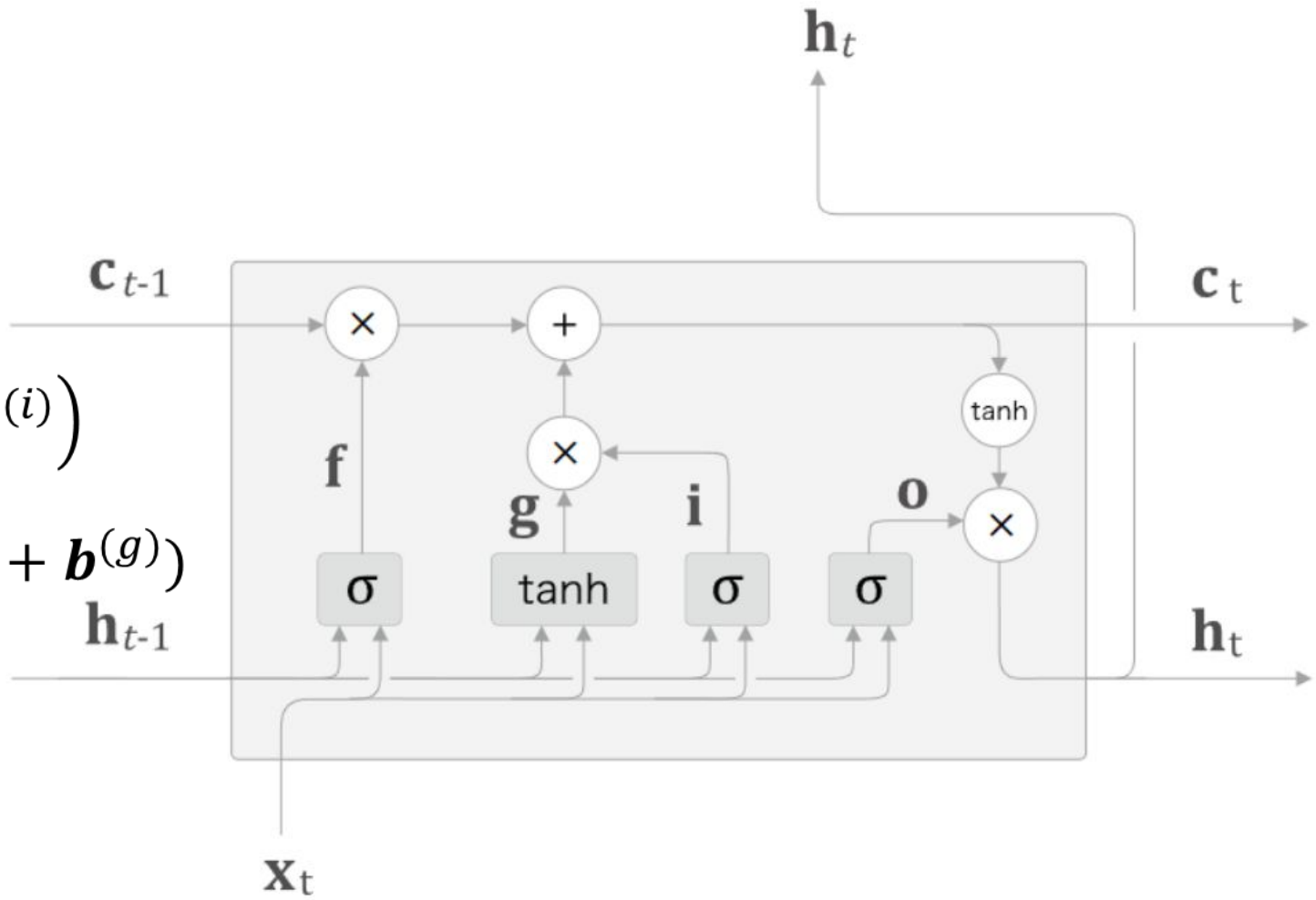
: 새로 추가되는 정보를

얼마나 기억할지 결정

$$\boldsymbol{i} = \sigma\left(\boldsymbol{x}_t \boldsymbol{W}_x^{(i)} + \boldsymbol{h}_{t-1} \boldsymbol{W}_x^{(i)} + \boldsymbol{b}^{(i)}\right)$$

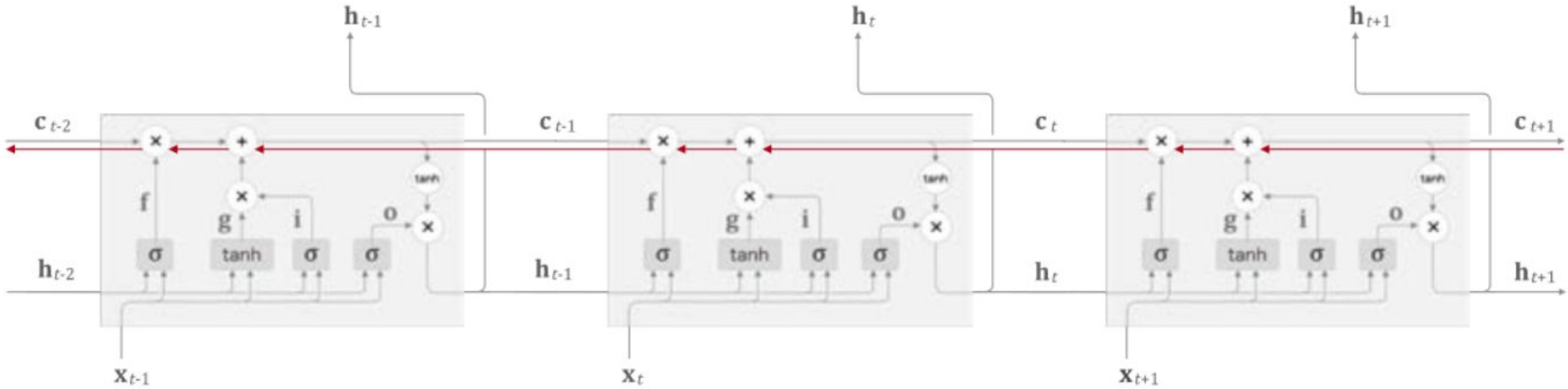
$$\boldsymbol{g} = \tanh\left(\boldsymbol{x}_t \boldsymbol{W}_x^{(g)} + \boldsymbol{h}_{t-1} \boldsymbol{W}_x^{(g)} + \boldsymbol{b}^{(g)}\right)$$

$$\boldsymbol{c}_t = \boldsymbol{f} \odot \boldsymbol{c}_{t-1} + \mathbf{g} \odot \boldsymbol{i}$$



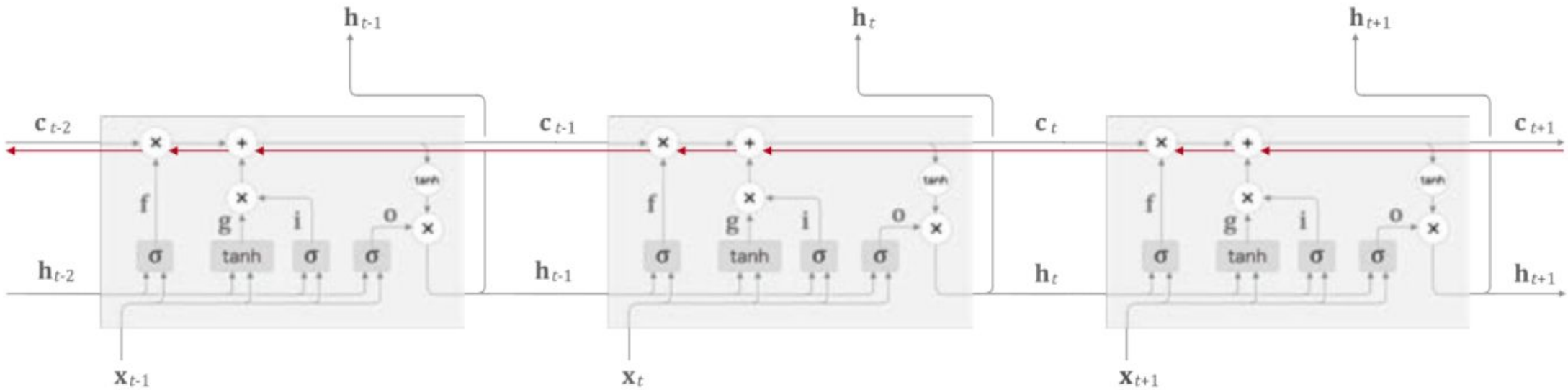
2.2 LSTM이란

- LSTM의 기울기 흐름
  - 기억 셀의 역전파 : 곱하기 노드를 지나며 **원소별 곱**으로 계산



2.2 LSTM이란

- **LSTM의 기울기 흐름**
  - 기억 셀의 역전파 : 곱하기 노드를 지나며 **원소별 곱**으로 계산



∴ 기울기 소실 완화

2.2 LSTM이란

- **Time LSTM 클래스 구현**
  - 아핀 변환 계산

  : 공통적으로 계산하는 네 번의 아핀 변환을 한 번의 아핀 변환으로 계산

$$
\begin{aligned}
&\mathbf{x}_t\ \mathbf{W}_{\mathrm{x}}^{(\mathrm{f})} + \mathbf{h}_{t-1}\ \mathbf{W}_{\mathrm{h}}^{(\mathrm{f})} + \mathbf{b}^{(\mathrm{f})}\\
&\mathbf{x}_t\ \mathbf{W}_{\mathrm{x}}^{(\mathrm{g})} + \mathbf{h}_{t-1}\ \mathbf{W}_{\mathrm{h}}^{(\mathrm{g})} + \mathbf{b}^{(\mathrm{g})}\\
&\mathbf{x}_t\ \mathbf{W}_{\mathrm{x}}^{(\mathrm{i})} + \mathbf{h}_{t-1}\ \mathbf{W}_{\mathrm{h}}^{(\mathrm{i})} + \mathbf{b}^{(\mathrm{i})}\\
&\mathbf{x}_t\ \mathbf{W}_{\mathrm{x}}^{(\mathrm{o})} + \mathbf{h}_{t-1}\ \mathbf{W}_{\mathrm{h}}^{(\mathrm{o})} + \mathbf{b}^{(\mathrm{o})}
\end{aligned}
$$

↓

$$
\mathbf{x}_t\ \left[\mathbf{W}_{\mathrm{x}}^{(\mathrm{f})}\ \mathbf{W}_{\mathrm{x}}^{(\mathrm{g})}\ \mathbf{W}_{\mathrm{x}}^{(\mathrm{i})}\ \mathbf{W}_{\mathrm{x}}^{(\mathrm{o})}\right] + \mathbf{h}_{t-1}\left[\mathbf{W}_{\mathrm{h}}^{(\mathrm{f})}\ \mathbf{W}_{\mathrm{h}}^{(\mathrm{g})}\ \mathbf{W}_{\mathrm{h}}^{(\mathrm{i})}\ \mathbf{W}_{\mathrm{h}}^{(\mathrm{o})}\right] + \left[\mathbf{b}^{(\mathrm{f})}\ \mathbf{b}^{(\mathrm{g})}\ \mathbf{b}^{(\mathrm{i})}\ \mathbf{b}^{(\mathrm{o})}\right]
$$

↓

$$
\mathbf{x}_t\ \mathbf{W}_{\mathrm{x}} + \mathbf{h}_{t-1}\ \mathbf{W}_{\mathrm{h}} = \mathbf{A}
$$

$N \times D$ $D \times 4H$ $N \times H$ $H \times 4H$ $N \times 4H$

일치 일치

2.2 LSTM이란

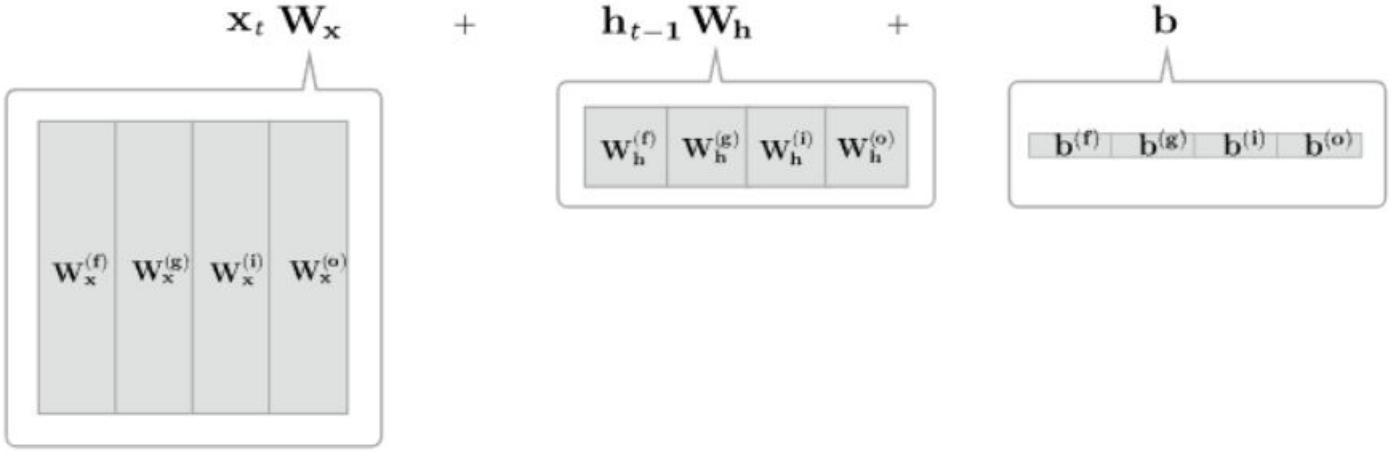
- Time LSTM 클래스 구현
  - 아핀 변환 계산

  : 공통적으로 계산하는 네 번의 아핀 변환을 한 번의 아핀 변환으로 계산

$$
\begin{aligned}
&\mathbf{x}_t\ \mathbf{W}_{\mathbf{x}}^{(\mathrm{f})} + \mathbf{h}_{t-1}\ \mathbf{W}_{\mathbf{h}}^{(\mathrm{f})} + \mathbf{b}^{(\mathrm{f})} \\
&\mathbf{x}_t\ \mathbf{W}_{\mathbf{x}}^{(\mathrm{g})} + \mathbf{h}_{t-1}\ \mathbf{W}_{\mathbf{h}}^{(\mathrm{g})} + \mathbf{b}^{(\mathrm{g})} \\
&\mathbf{x}_t\ \mathbf{W}_{\mathbf{x}}^{(\mathrm{i})} + \mathbf{h}_{t-1}\ \mathbf{W}_{\mathbf{h}}^{(\mathrm{i})} + \mathbf{b}^{(\mathrm{i})} \\
&\mathbf{x}_t\ \mathbf{W}_{\mathbf{x}}^{(\mathrm{o})} + \mathbf{h}_{t-1}\ \mathbf{W}_{\mathbf{h}}^{(\mathrm{o})} + \mathbf{b}^{(\mathrm{o})}
\end{aligned}
$$

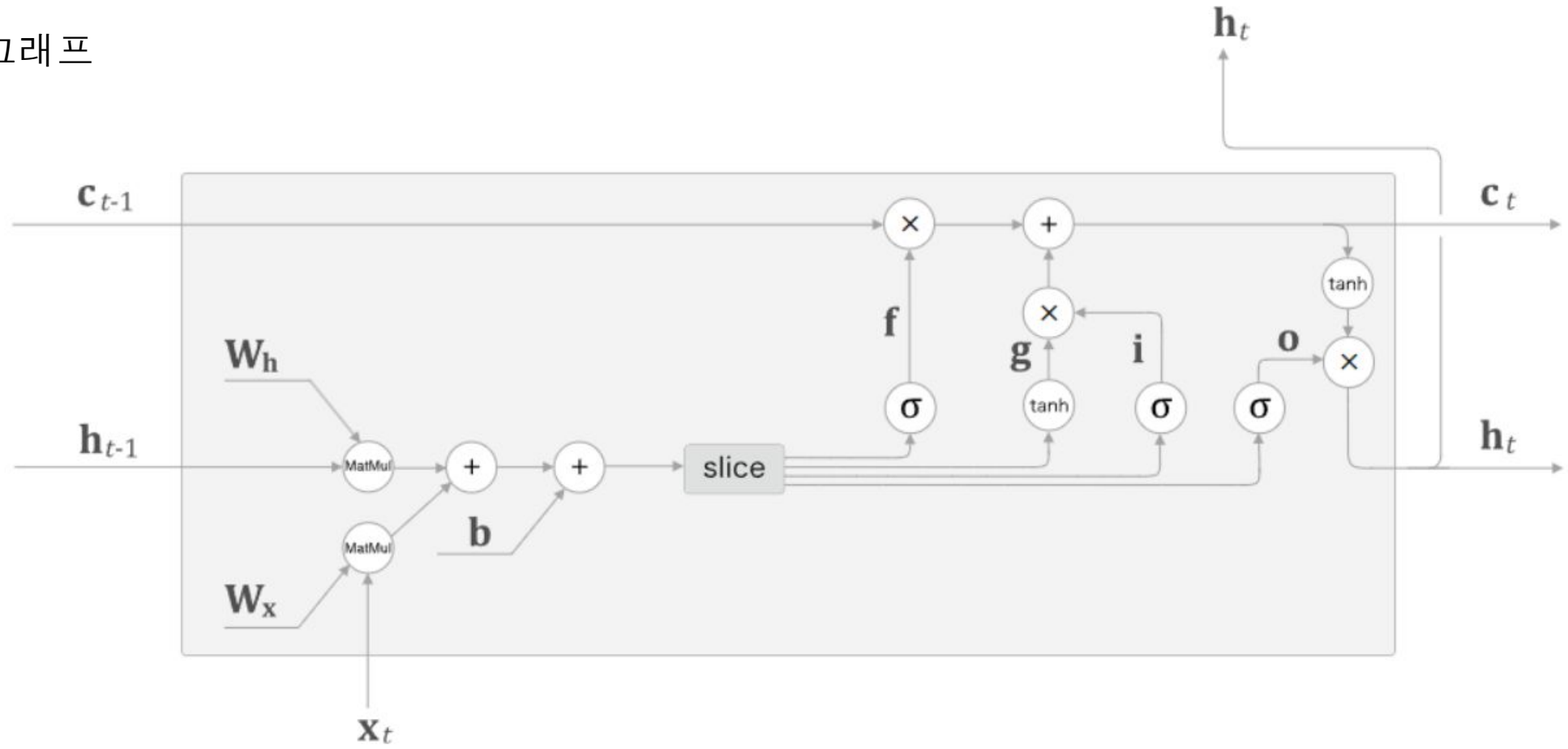
$$
\mathbf{x}_t \left[\mathbf{W}_{\mathbf{x}}^{(\mathrm{f})}\ \mathbf{W}_{\mathbf{x}}^{(\mathrm{g})}\ \mathbf{W}_{\mathbf{x}}^{(\mathrm{i})}\ \mathbf{W}_{\mathbf{x}}^{(\mathrm{o})}\right] + \mathbf{h}_{t-1} \left[\mathbf{W}_{\mathbf{h}}^{(\mathrm{f})}\ \mathbf{W}_{\mathbf{h}}^{(\mathrm{g})}\ \mathbf{W}_{\mathbf{h}}^{(\mathrm{i})}\ \mathbf{W}_{\mathbf{h}}^{(\mathrm{o})}\right] + \left[\mathbf{b}^{(\mathrm{f})}\ \mathbf{b}^{(\mathrm{g})}\ \mathbf{b}^{(\mathrm{i})}\ \mathbf{b}^{(\mathrm{o})}\right]
$$

$$
\mathbf{x}_t\,\mathbf{W}_{\mathbf{x}} + \mathbf{h}_{t-1}\,\mathbf{W}_{\mathbf{h}} + \mathbf{b}
$$



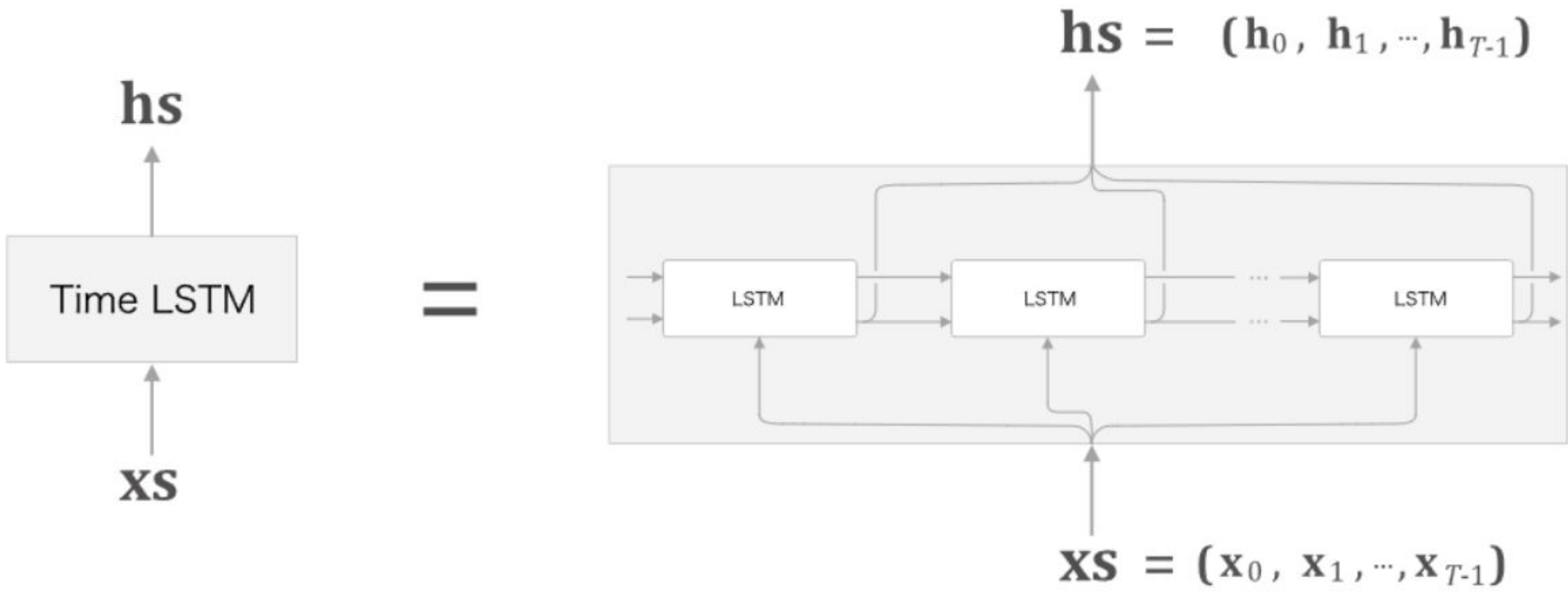
2.2 LSTM이란

- Time LSTM 클래스 구현
  - 계산 그래프



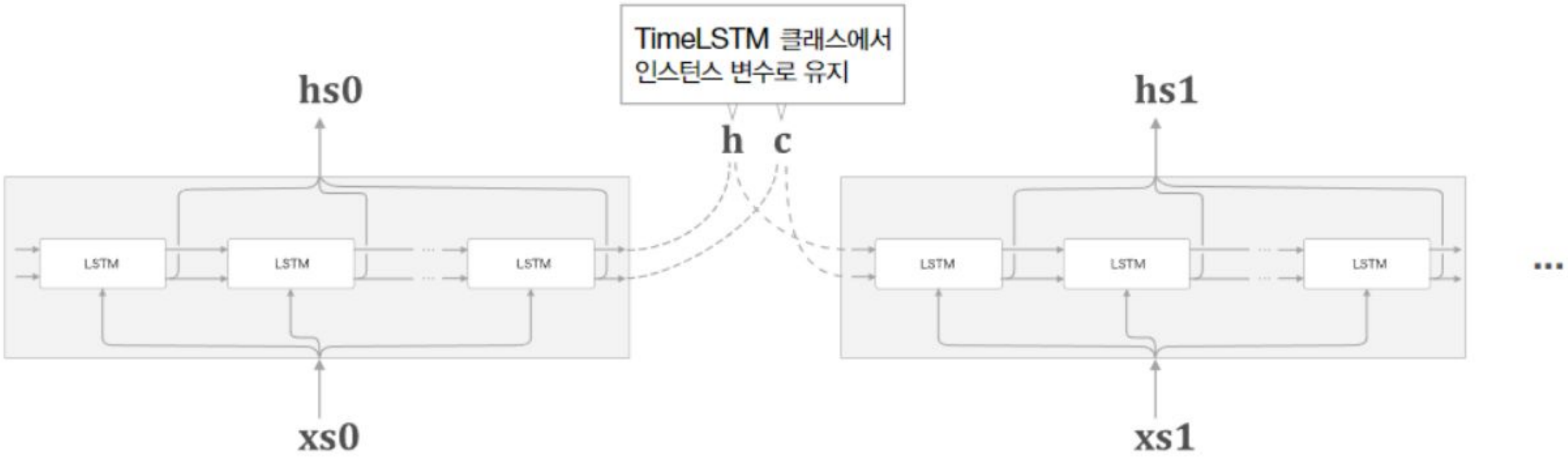
2.2 LSTM이란

- **Time LSTM 클래스 구현**
  - T개의 LSTM 계층



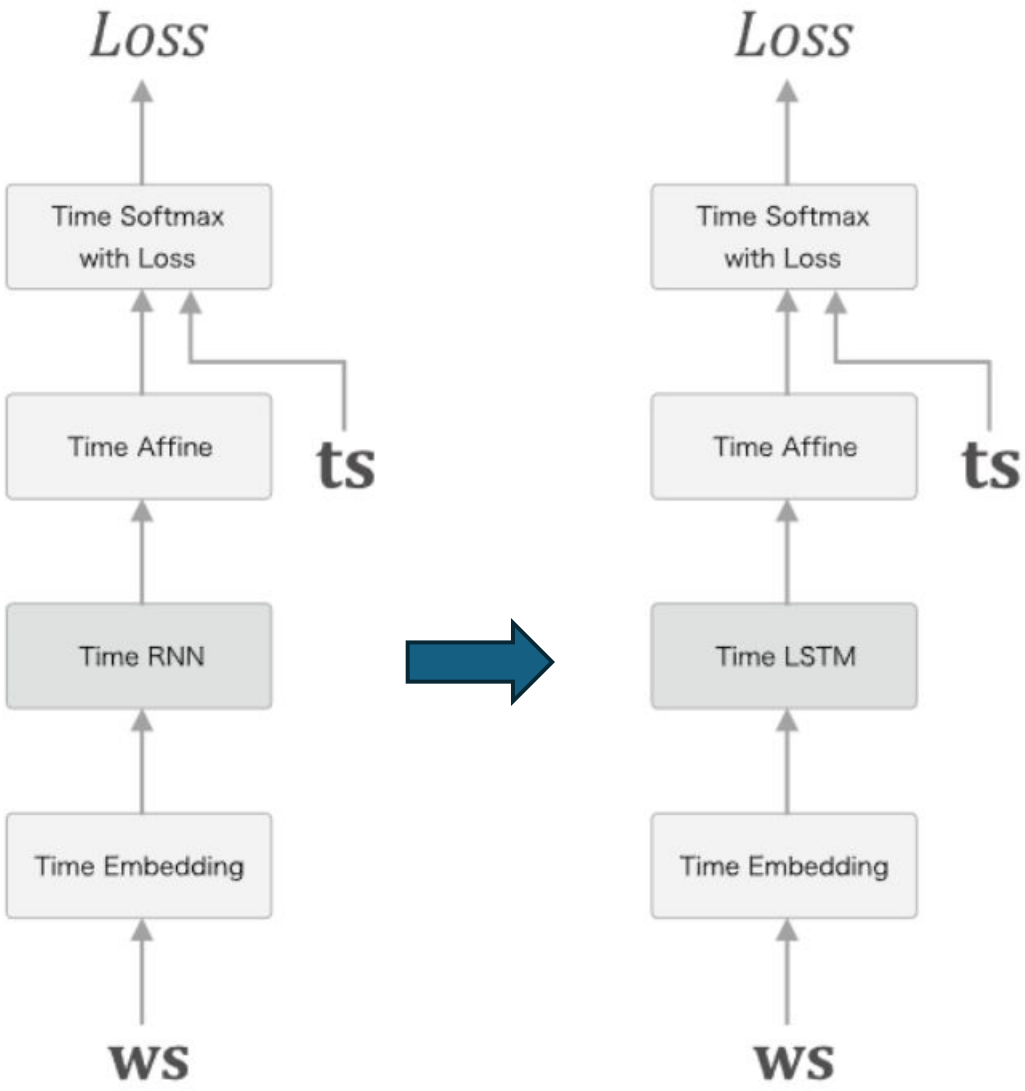
2.2 LSTM이란

- **Time LSTM 클래스 구현**
  - 은닉 상태 벡터(h)와 기억 셀(c)을 통한 장기 의존 관계 학습



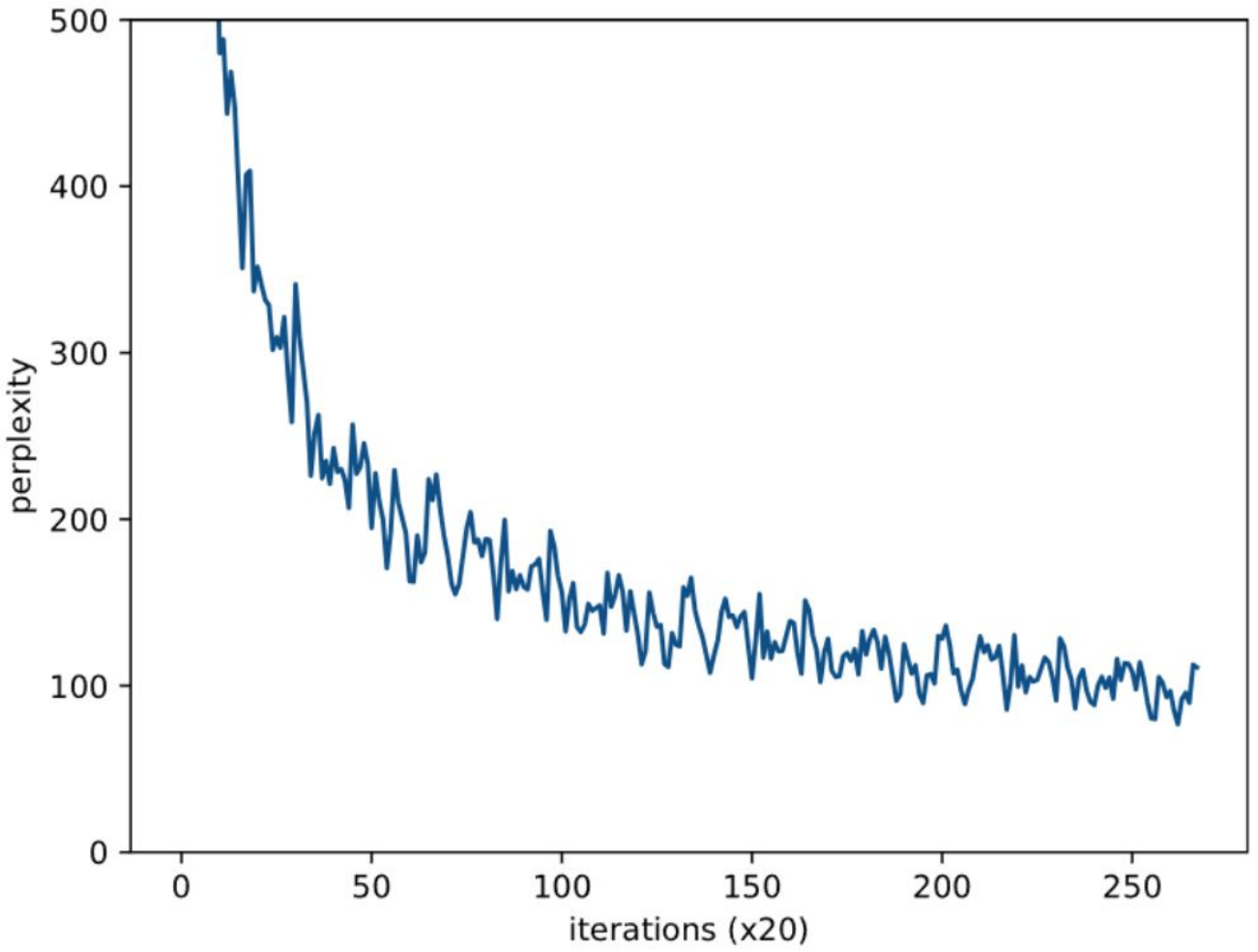
2.2 LSTM이란

- **LSTM을 사용한 언어 모델**
  - Time LSTM 계층으로 변경



2.2 LSTM이란

- **언어 모델의 평가**
  - PTB dataset
  - **훈련 데이터셋 전부** 를 사용한 학습 및 시각화

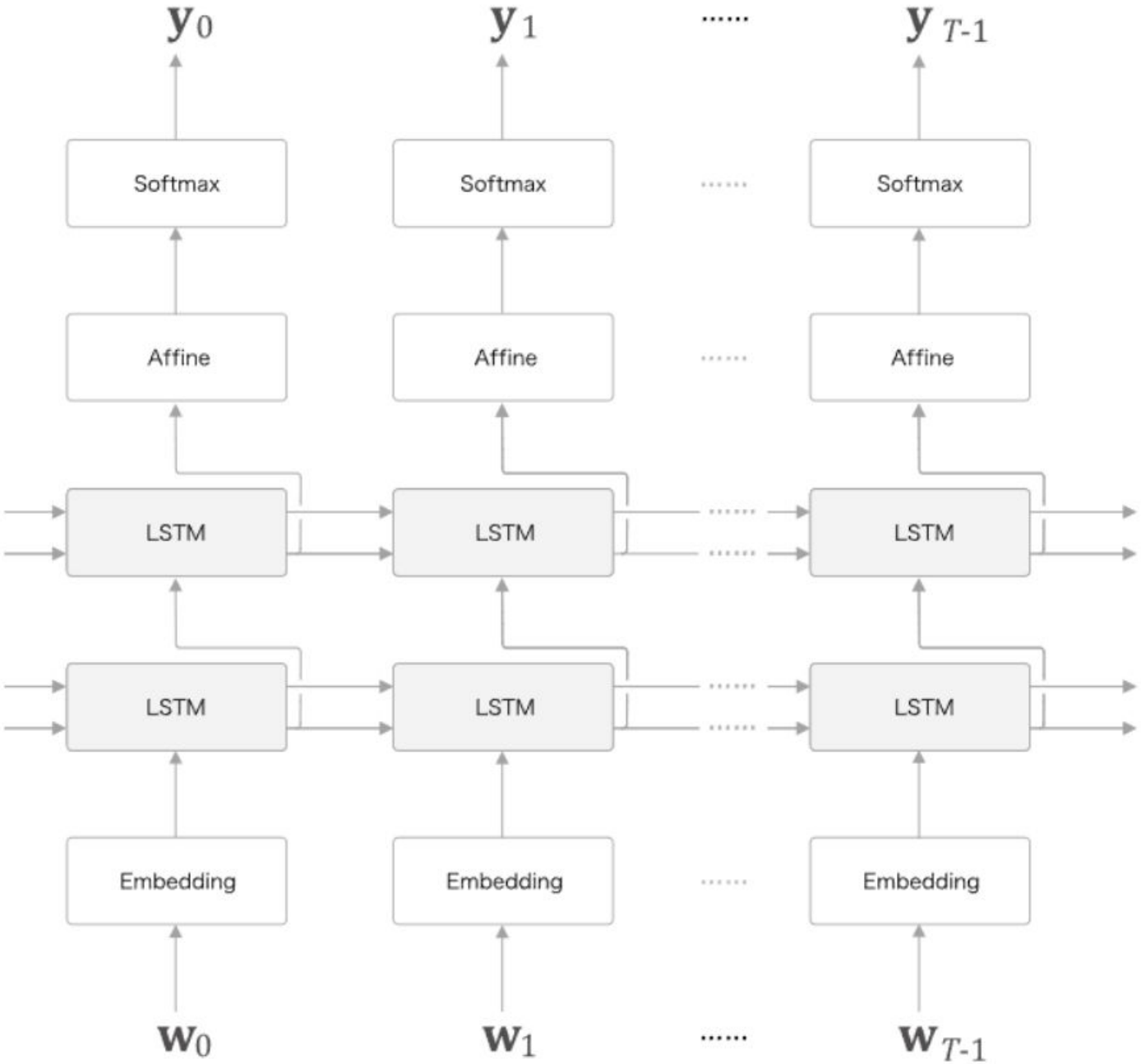


2.3 개선 방안

1. LSTM 계층 다층화
2. 드롭아웃
3. 가중치 공유
4. GRU

2.3 개선 방안

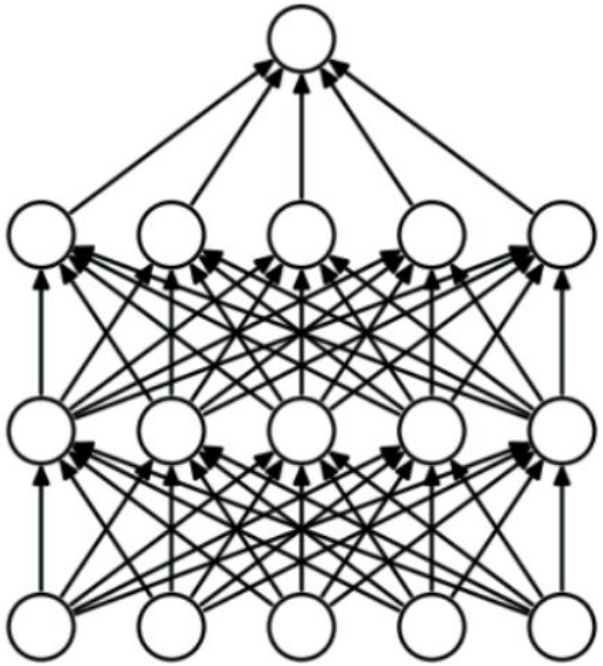
1. **LSTM 계층 다층화**
   - LSTM 계층을 여러 겹 쌓아서 **복잡한 의존 관계 학습** 가능



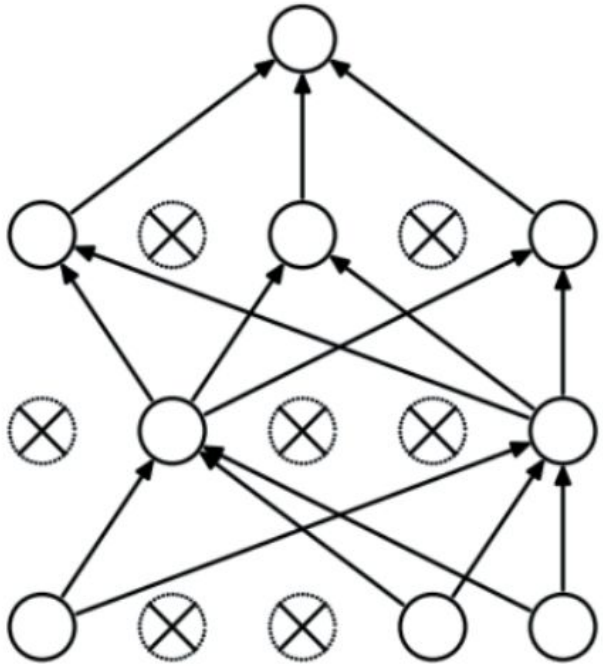
2.3 개선 방안

2. 드롭아웃

- 학습 과정에서 뉴런의 일부를 무작위로 무시



(a) 일반적인 신경망

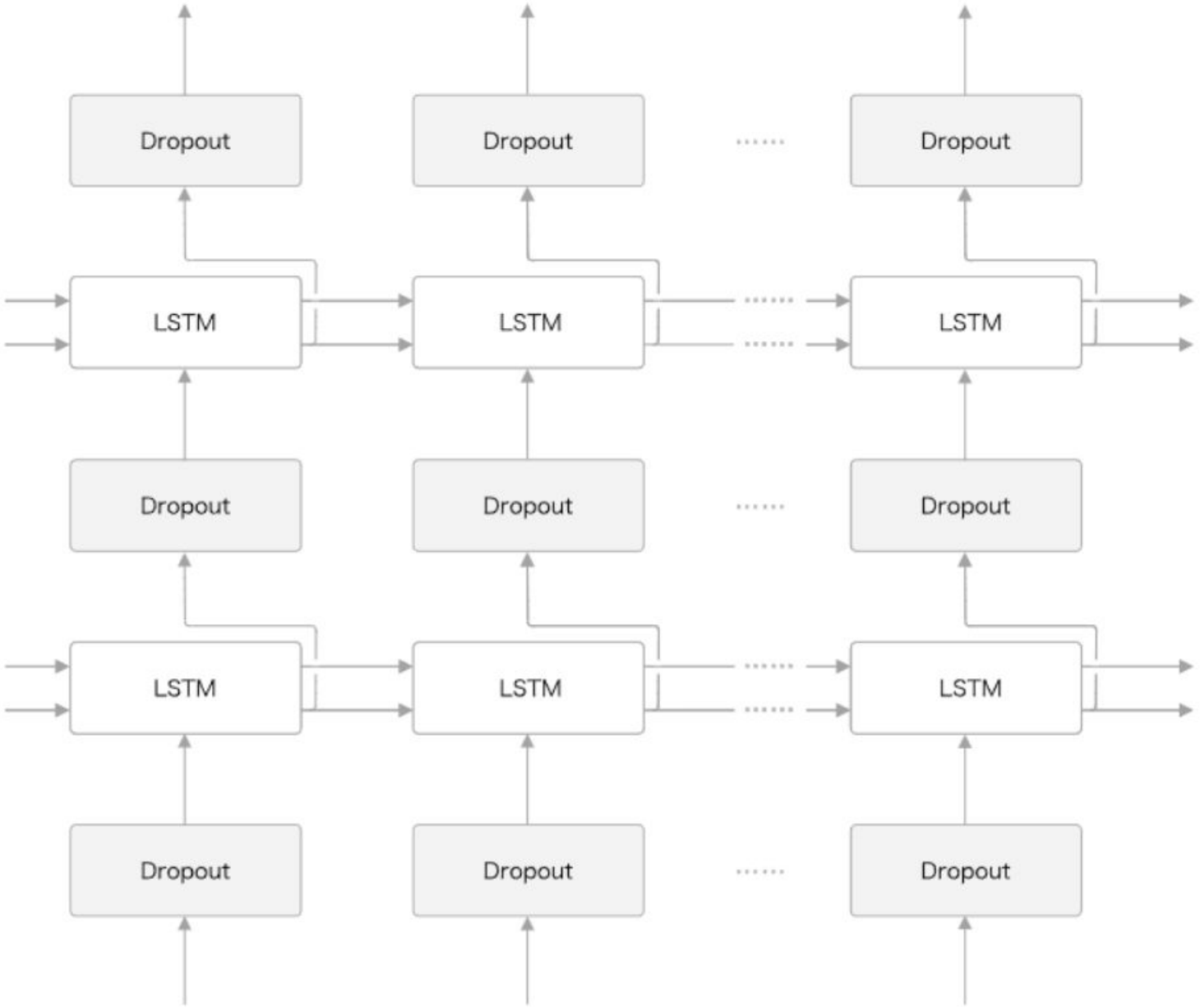


(b) 드롭아웃을 적용한 모습

2.3 개선 방안

2. 드롭아웃

⇒ 과적합 억제

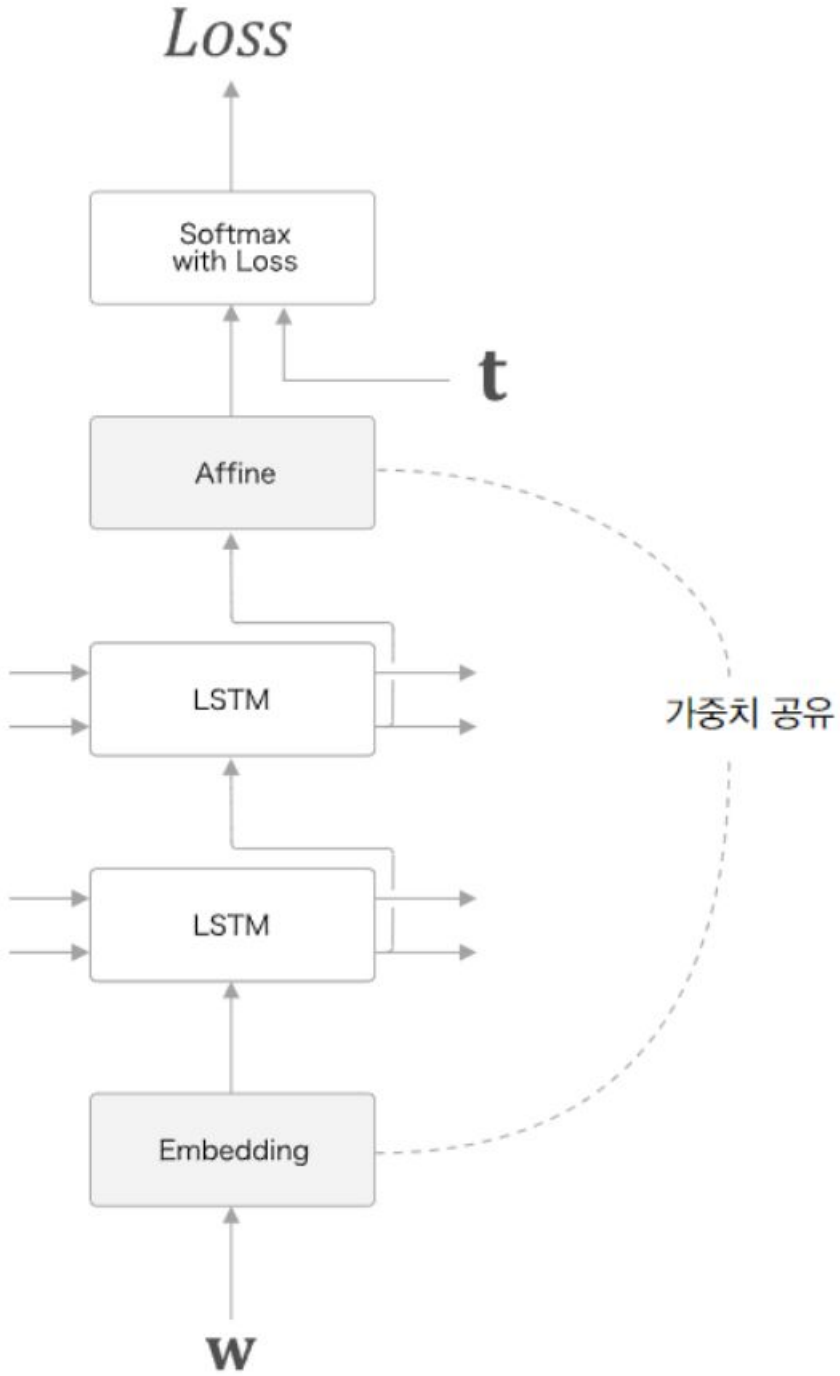


2.3 개선 방안

3. 가중치 공유

- Embedding 계층과 Affine 계층의 가중치 공유

⇒ **학습해야 할 매개변수 감소**



2.3 개선 방안

- Perplexity 개선 결과

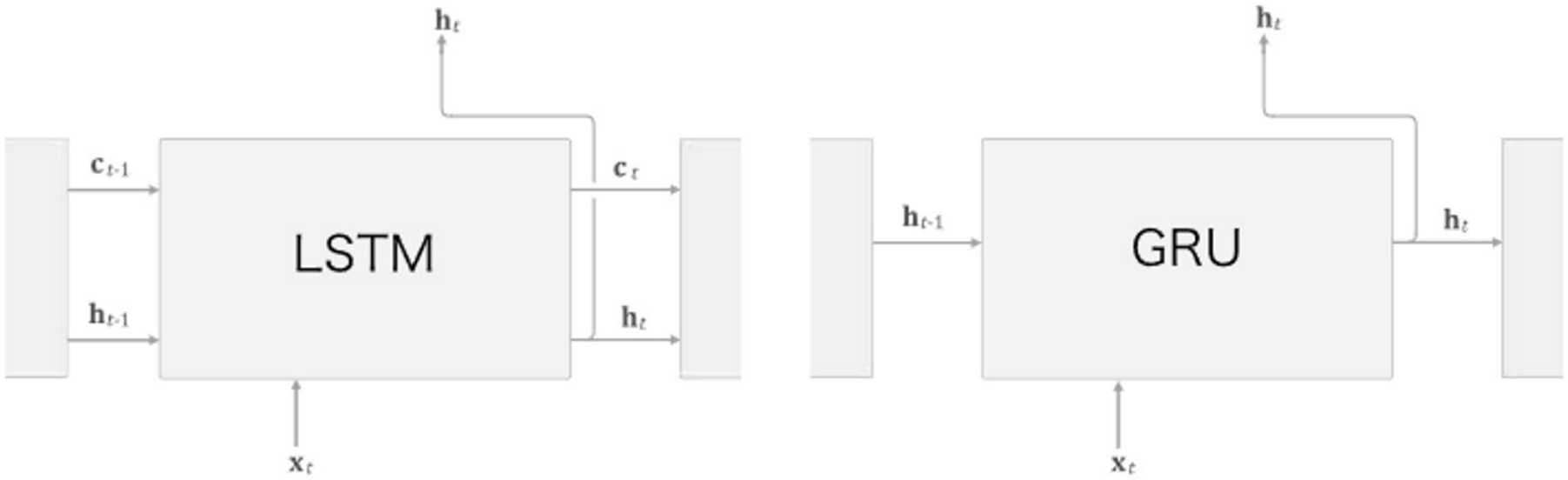
```
Humane0@DESKTOP-D9DE7UT MINGW64 ~/Desktop/NLP_study/ch06 (main)
$ C:/Users/Humane0/AppData/Local/Programs/Python/Python312/pytho
perplexity 평가 중 ...
234 / 235
test perplexity:  136.07759032293416
```

```
Humane0@DESKTOP-D9DE7UT MINGW64 ~/Desktop/NLP_study/ch06 (main)
$ C:/Users/Humane0/AppData/Local/Programs/Python/Python312/pytho
perplexity 평가 중 ...
234 / 235
test perplexity:  75.76414156396132
```

2.3 개선 방안

4. GRU

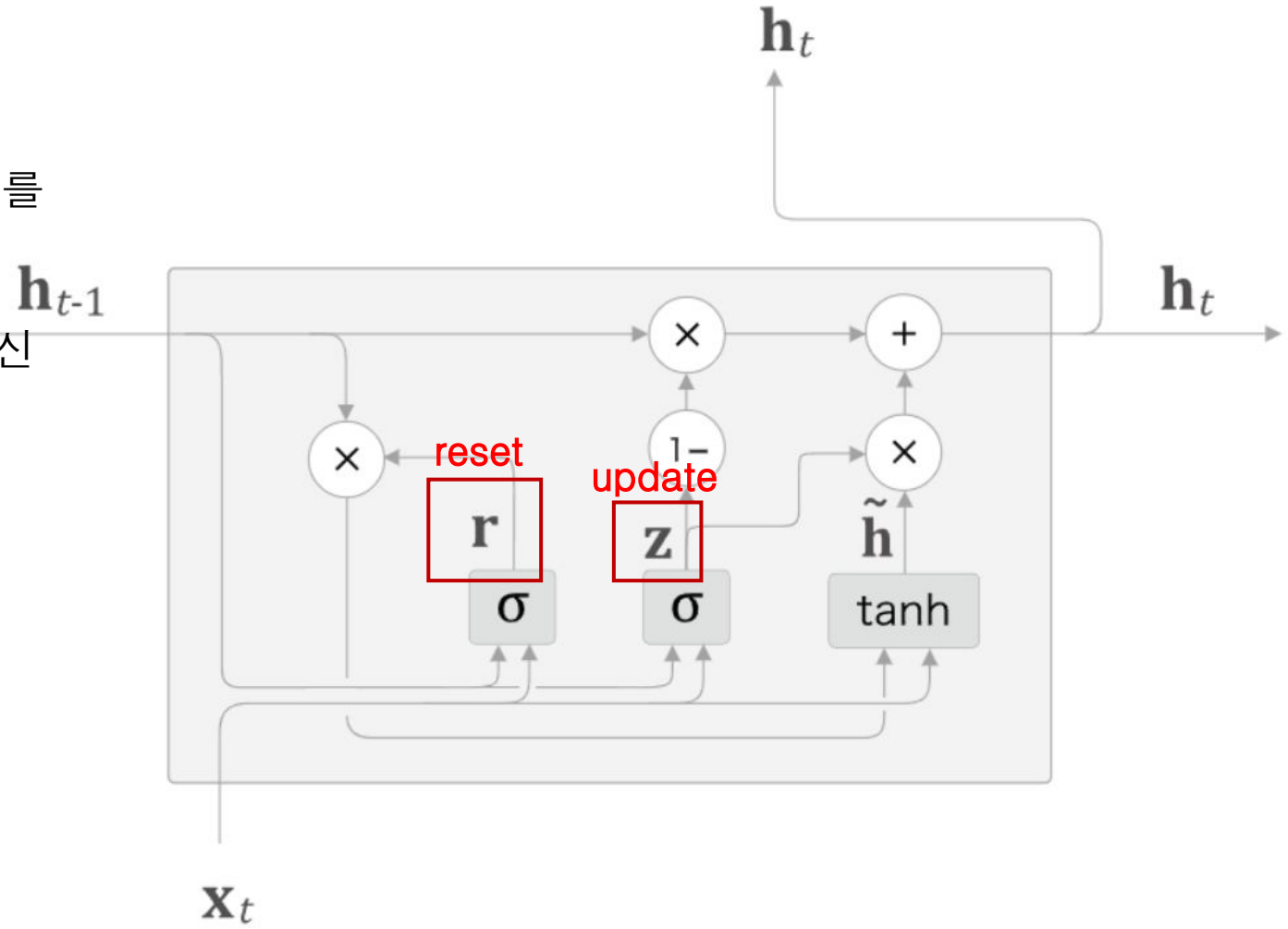
- LSTM과 인터페이스 비교
  - RNN 계층처럼 **은닉 상태 벡터만 사용**



2.3 개선 방안

4. GRU

- 계산 그래프
  - **Reset 게이트** : 이전 상태의 정보를 얼마나 잊을지 결정
  - **Update 게이트** : 은닉 상태를 갱신



2.3 개선 방안

4. GRU

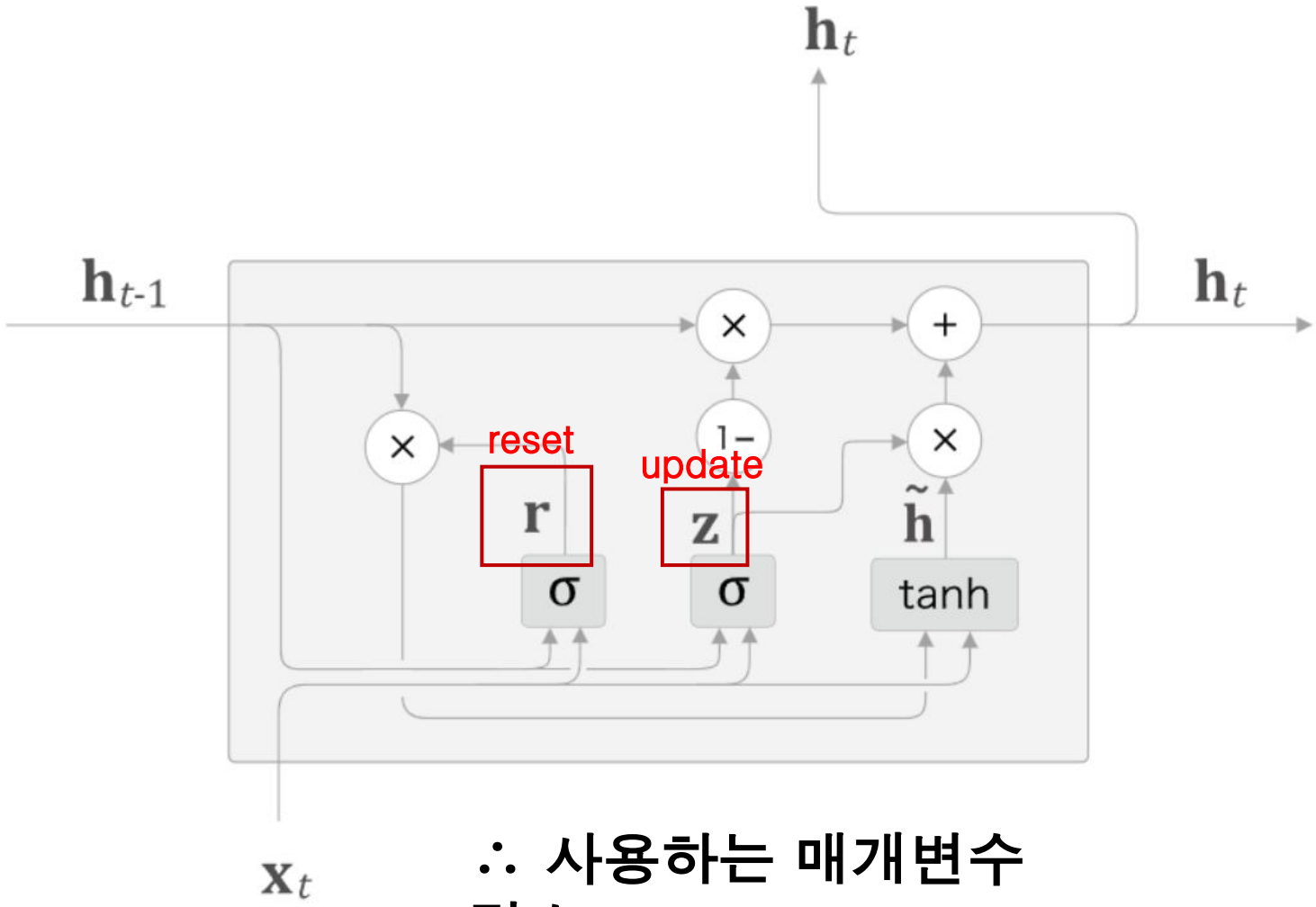
- 계산 그래프

$$\mathbf{z} = \sigma(\mathbf{x}_t \mathbf{W}_{\mathbf{x}}^{(\mathbf{z})} + \mathbf{h}_{t-1} \mathbf{W}_{\mathbf{h}}^{(\mathbf{z})} + \mathbf{b}^{(\mathbf{z})})$$

$$\mathbf{r} = \sigma(\mathbf{x}_t \mathbf{W}_{\mathbf{x}}^{(\mathbf{r})} + \mathbf{h}_{t-1} \mathbf{W}_{\mathbf{h}}^{(\mathbf{r})} + \mathbf{b}^{(\mathbf{r})})$$

$$\tilde{\mathbf{h}} = \tanh(\mathbf{x}_t \mathbf{W}_{\mathbf{x}} + (\mathbf{r} \odot \mathbf{h}_{t-1}) \mathbf{W}_{\mathbf{h}} + \mathbf{b})$$

$$\mathbf{h}_t = (1 - \mathbf{z}) \odot \mathbf{h}_{t-1} + \mathbf{z} \odot \tilde{\mathbf{h}}$$



∴ 사용하는 매개변수 감소

정리

- RNN 개념 및 구현
- RNN을 활용한 언어 모델
- LSTM을 통한 기울기 소실 완화
- LSTM을 활용한 언어 모델과 개선 방안

QnA