Neural Language Models

Knowledge and Language Engineering Lab

POSTECH
POHANG UNIVERSITY OF SCIENCE AND TECHNOLOGY

목차

- 신경망 언어 모델 소개
- LSTMs 기반 언어 모델 실습

신경망 언어모델 소개

언어 모델

- 어떤 문장이 더 자연스러운가요?
  - Is the table on cup the.
    The cup is on the table.
  - 소녀는 꽃을 보았다 .
    소녀를 꽃이 보았다 .
  - 개가 짖는다.
    고양이가 짖는다.
- Language Model :
  - 컴퓨터를 통해 **문장의 정확도/유창성**을 판단하는 기술

언어 모델

- **언어 모델이란,**
  - 문장 또는 단어열에 대한 확률 분포
  - m개의 단어열이 주어졌을 때 m개의 단어열이 나타날 확률을 계산
  - P(I am a boy) = 0.7
  - P(I a am boy) = 0.02
- **적용 예**
  - 품사 태깅
    - $P(I_{noun}\ am_{verb}\ a_{article}\ boy_{noun})=?$
  - 기계 번역
    - P(<u>high</u> winds tonight) > P(<u>large</u> winds tonight)
  - 철자 교정
    - P(about fifteen <u>minutes</u> from) > P(about fifteen <u>minuets</u> from)
  - 기타 등등…

언어 모델

- Applications
  - 기계번역: Machine translation

나는 어제 고양이를 보았다 → I yesterday a cat saw / I saw a cat yesterday

Source Sentence | Target Sentence

  - 음성인식: Speech recognition

Speech Signal → rʌn tu ː ju ː → run to you / run two you

Speech Signal | Phonetic Sequence | Recognized Text

언어 모델

- 접근 방법
  - $P(\text{Today is Wednesday})$

    $= P(\text{Today})P(\text{is}|\text{Today})P(\text{Wednesday}|\text{is, Today})$

(a.k.a Auto-regressive)

  - $P(\text{W}) = P(w_1, w_2, w_3, \dots, w_n)$

    $= P(w_1)P(w_2|w_1)P(w_3|w_2, w_1) \dots P(w_n|w_{n-1}, w_{n-2}, \dots, w_1)$

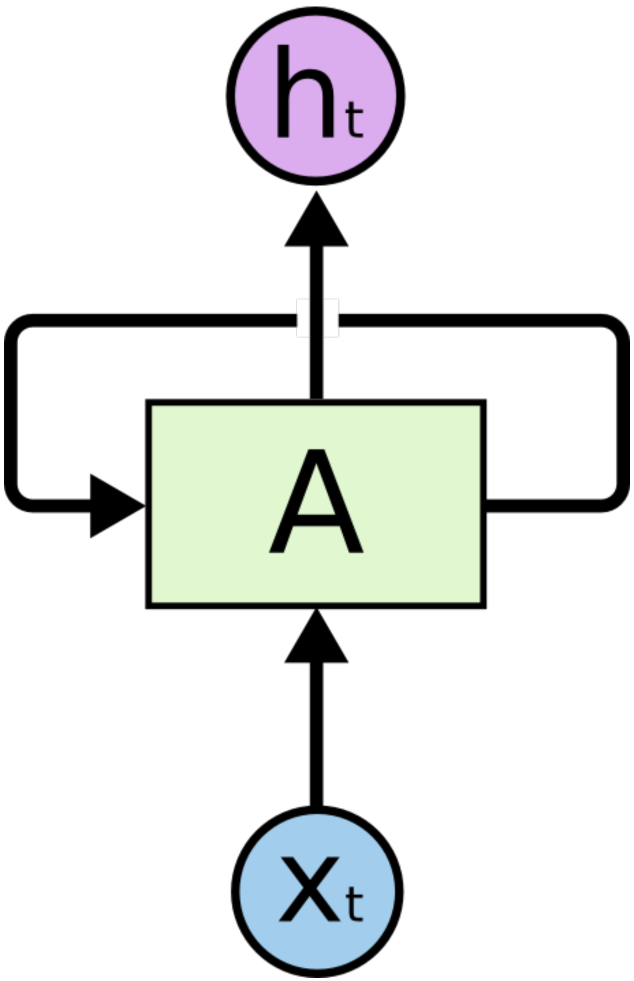
    $= \prod_{i=1}^{n} P(w_i|w_1^{i-1})$

언어 모델

- 통계기반 언어 모델
  - N-gram 언어 모델
- 신경망 기반 언어 모델 - Vector space model
  - Recurrent neural network 기반 언어 모델

Recurrent Neural Networks

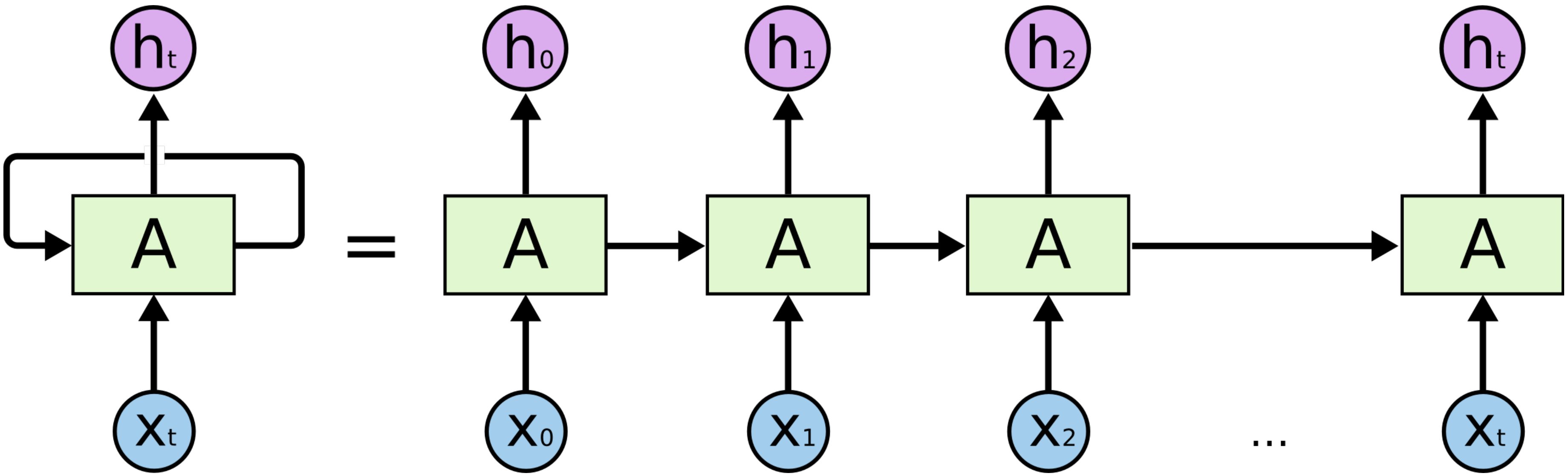
- 순환신경망 (Recurrent Neural Networks; RNNs)



x_t	Input
A	Cell
h_t	Output
t	Timestep

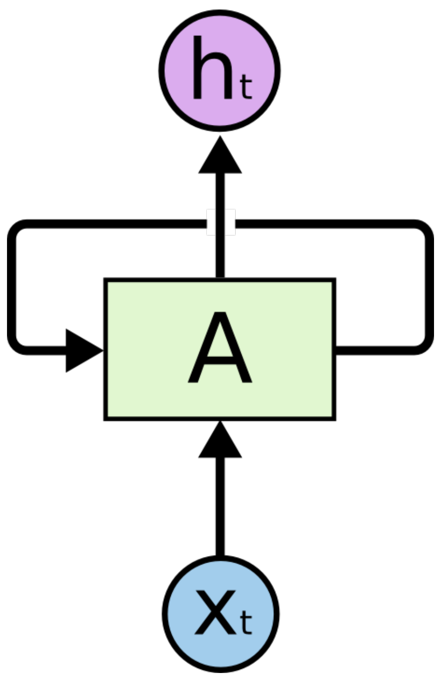
Recurrent Neural Networks

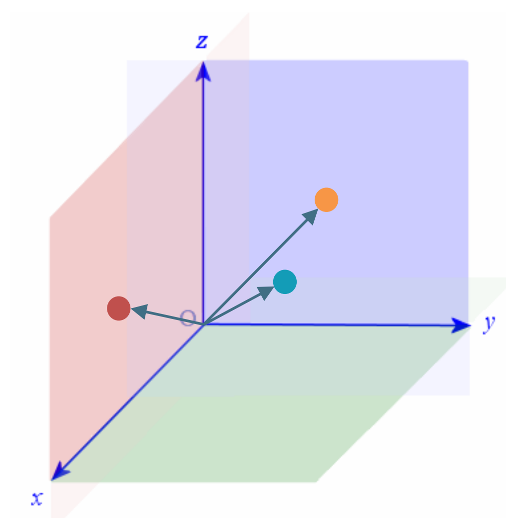
- 순환신경망 (Recurrent Neural Networks; RNNs)



Recurrent Neural Networks

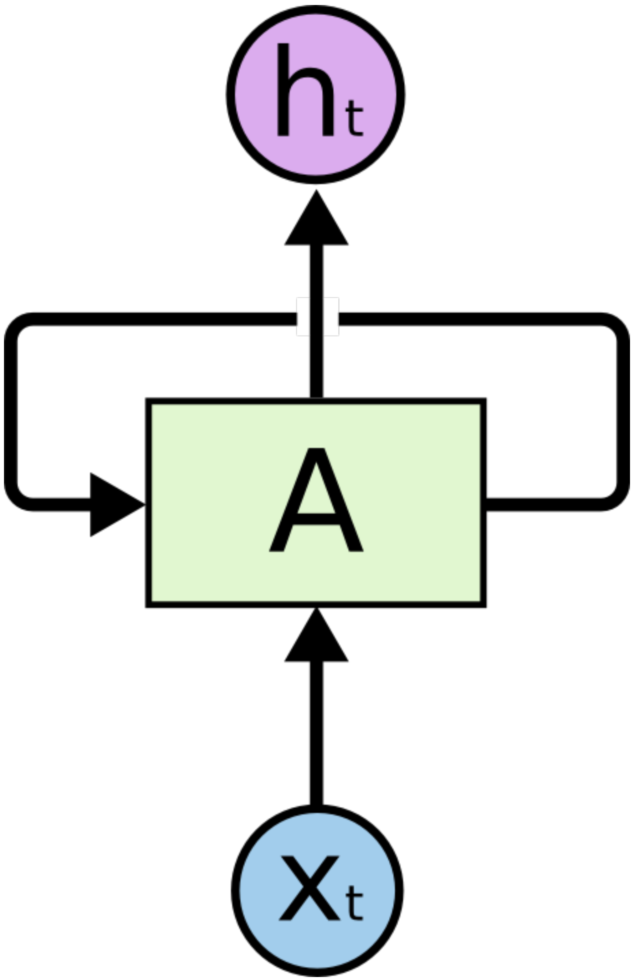
- 순환신경망 (Recurrent Neural Networks; RNNs)
  - 무작위 길이의 열 → 고정된 길이의 벡터 표현
    - I am a boy
    - Sometimes to understand a word's…
    - At your dictionary we try to gib…

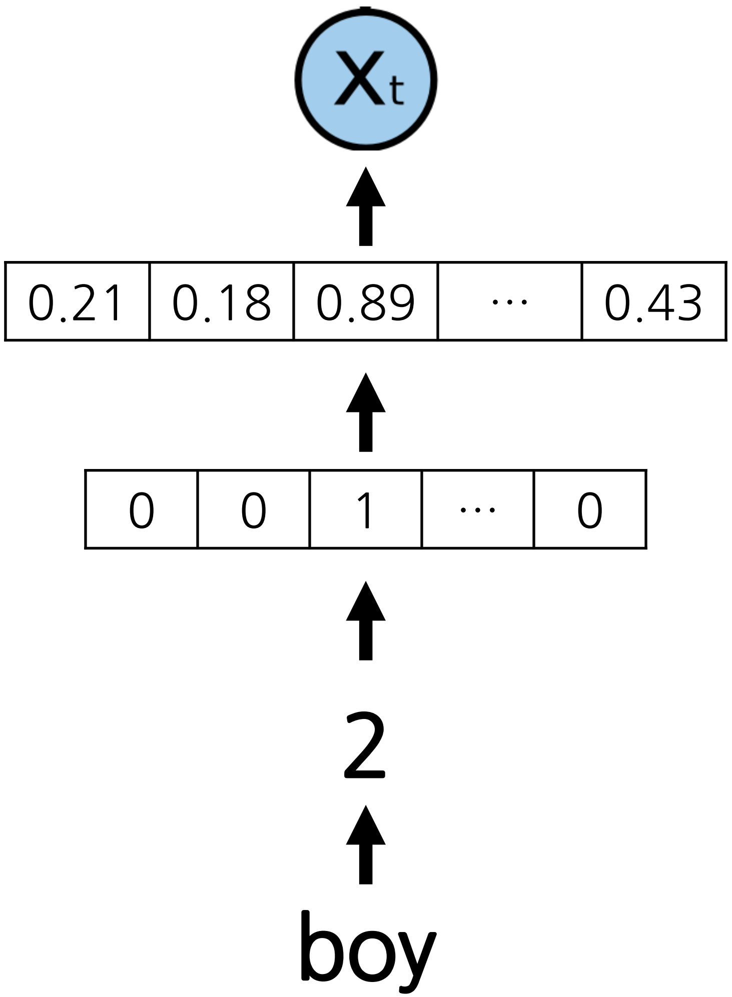




Recurrent Neural Networks

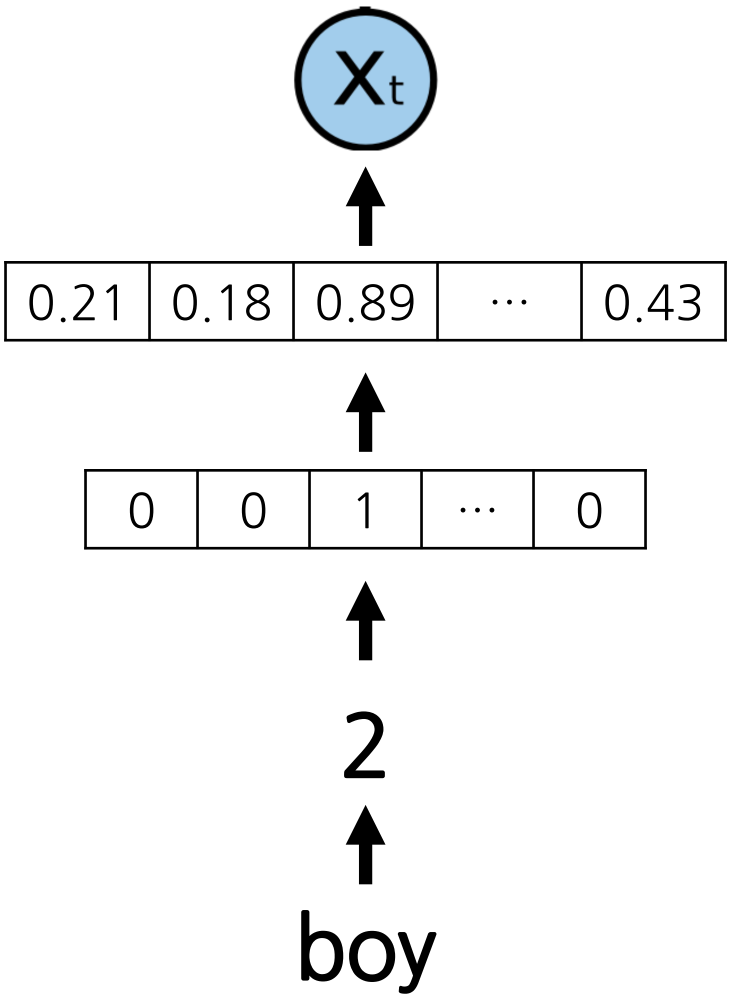
- RNN의 입력





Recurrent Neural Networks

- RNN의 입력: Embedding Layer (Word to Vector)



word index = 3

$$[0 \quad 0 \quad 0 \quad \boxed{1} \quad 0] \times \begin{bmatrix} 17 & 24 & 1 \\ 23 & 5 & 7 \\ 4 & 6 & 13 \\ 10 & 12 & 19 \\ 11 & 18 & 25 \end{bmatrix}$$

one-hot representation of word

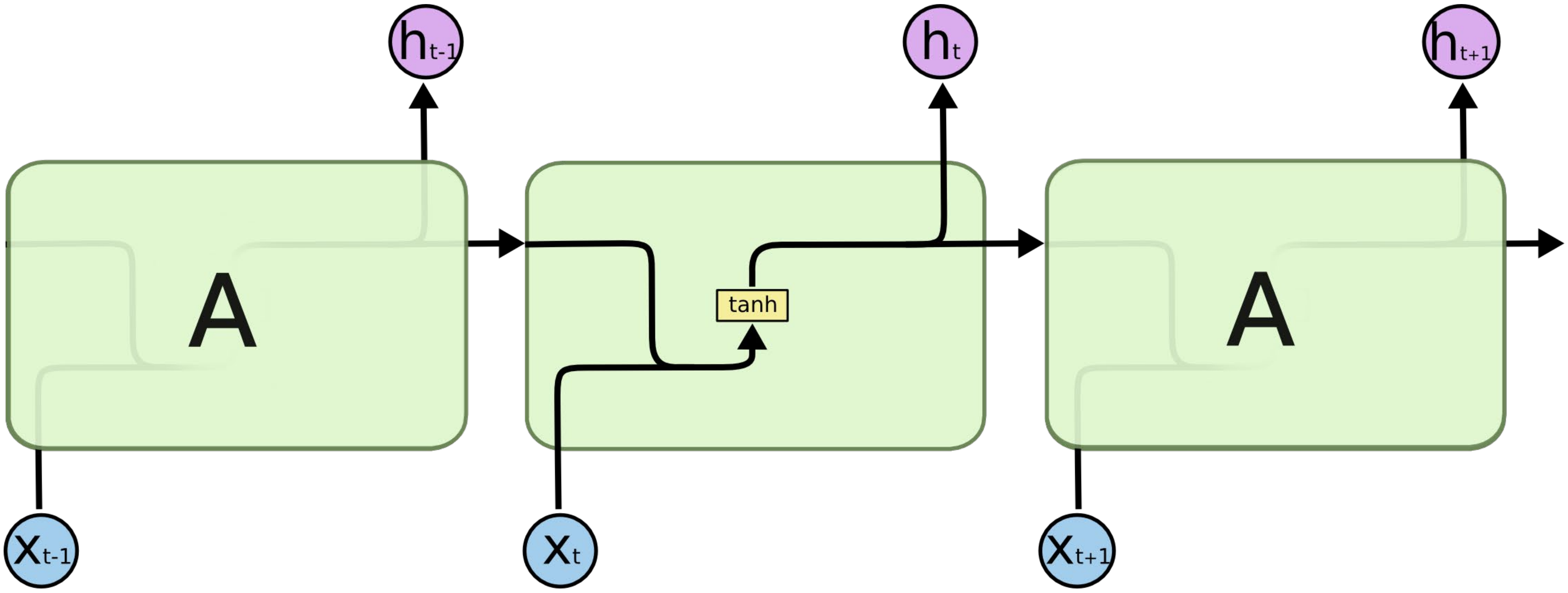
Word Embedding matrix
(Weight matrix)

$$= [10 \quad 12 \quad 19]$$

get embedding vector w.r.t word index

Recurrent Neural Networks

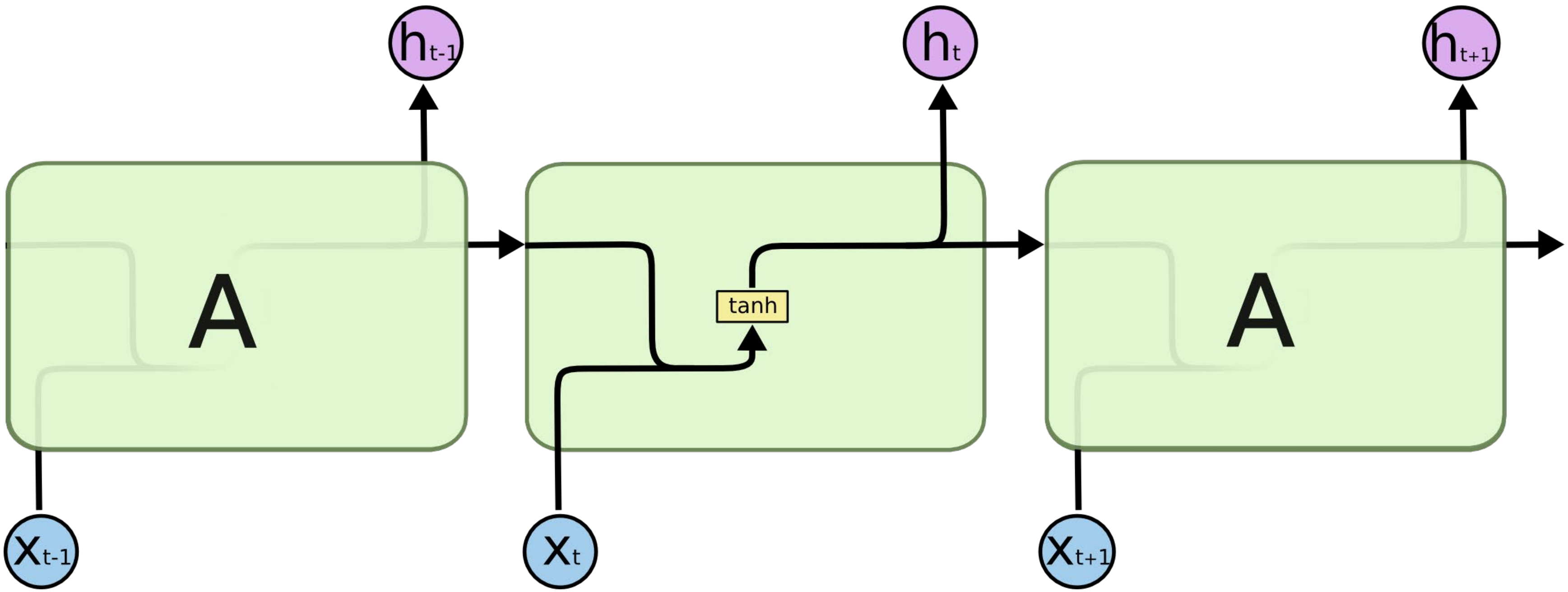
- RNN 출력



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

Recurrent Neural Networks

- Timestep마다 다른 Weight? Or weight sharing?



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

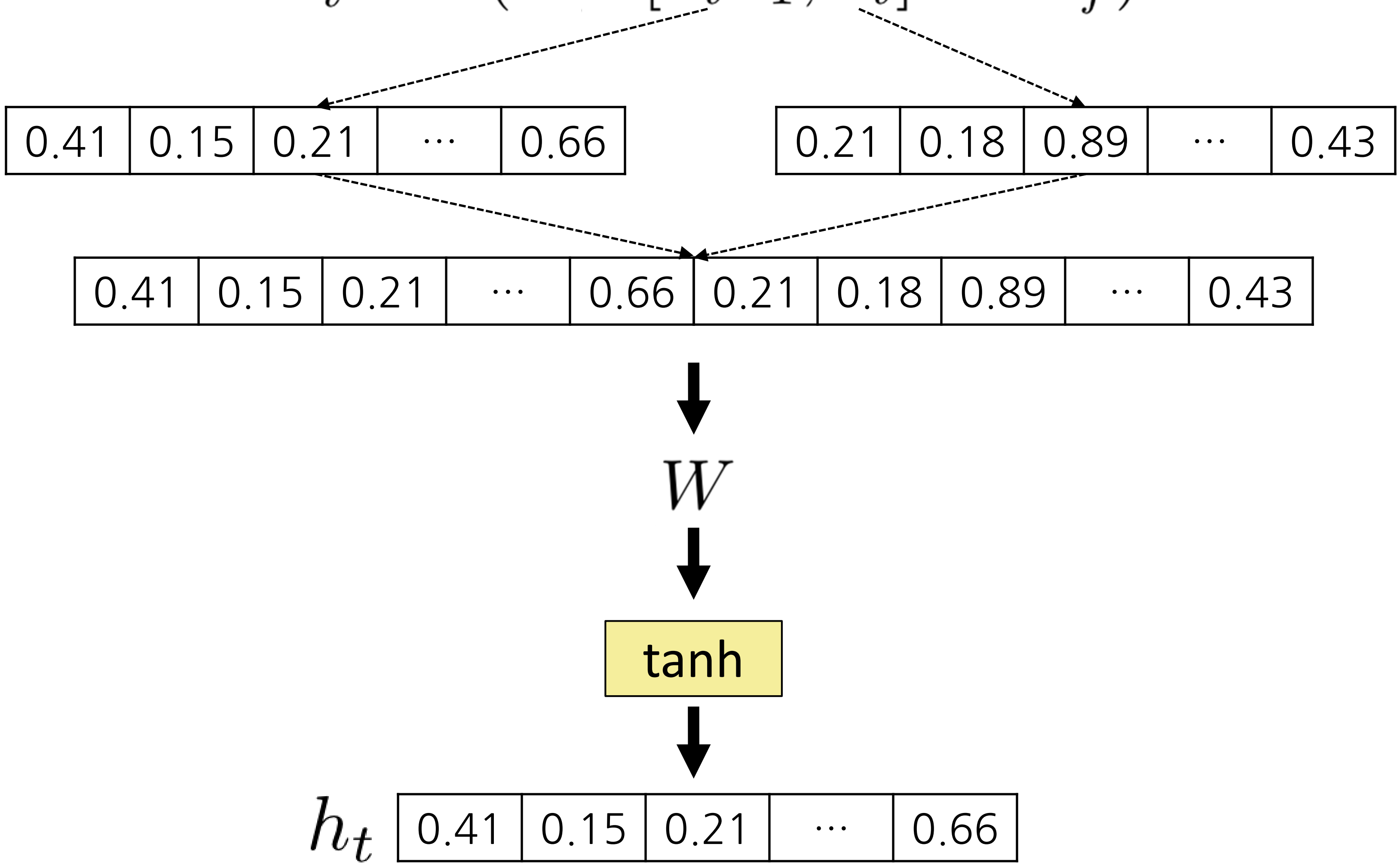
Recurrent Neural Networks

- Timestep마다 동일한 weight 공유
  - 학습 파라미터의 수 감소
  - 네트워크가 학습하지 못한 입력열에 대한 일반화 용이 (Overfitting 감소)
  - 가변길이 입력열에 대한 모델링 가능
  - on monday it was snowing ≈ it was snowing on Monday

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

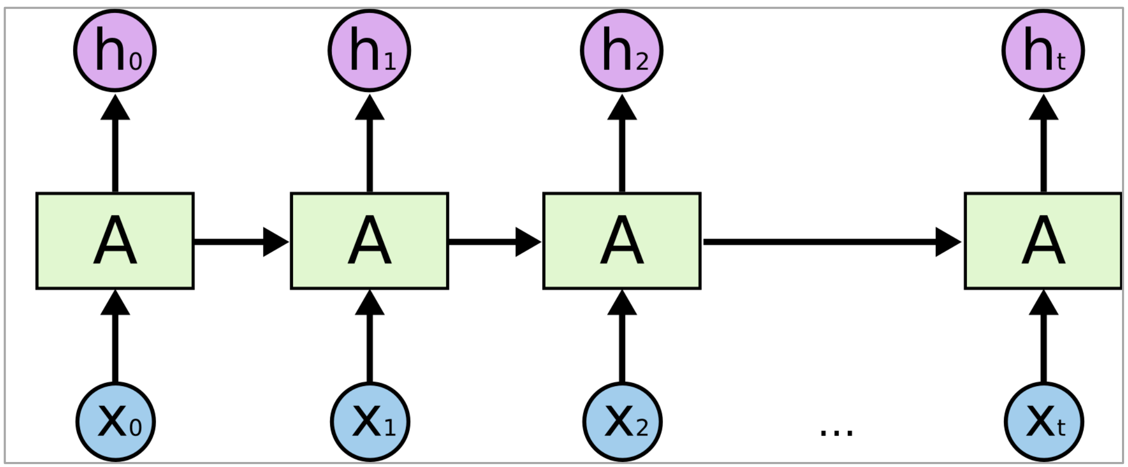
Recurrent Neural Networks

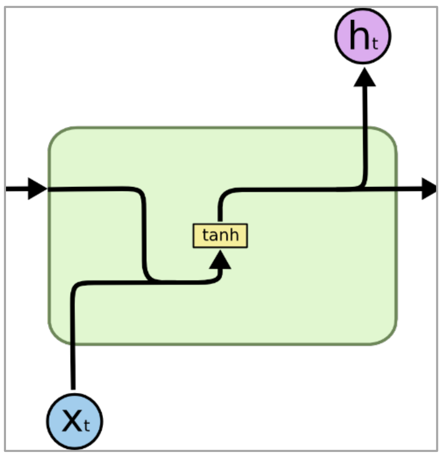
$$h_t = \sigma \left(W \cdot [h_{t-1}, x_t] + b_f \right)$$



Recurrent Neural Networks

- 불행하게도, **길이가 긴 열 학습** 어려움
  - Vanishing gradient problem
  - 장기 의존성 학습 어려움 (long-term dependency)



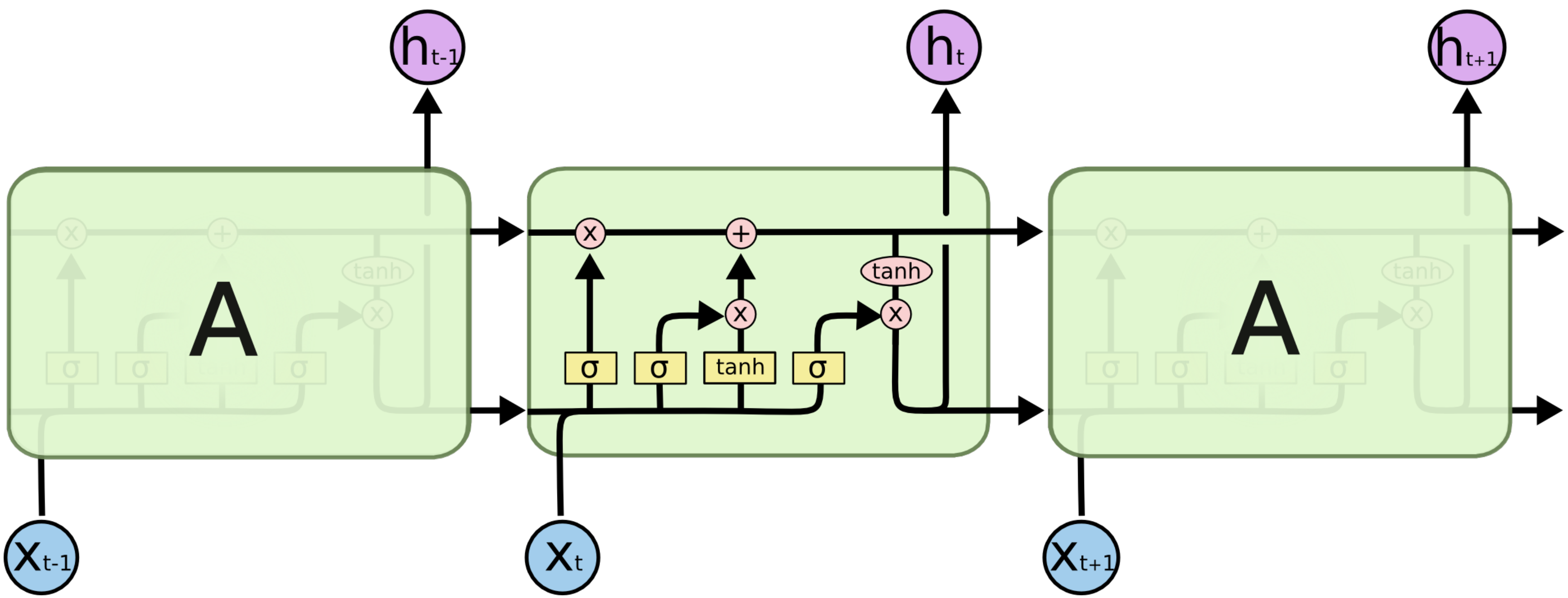


$* < 1 \quad * < 1 \quad * < 1 \quad \cdots \quad * < 1$

$= 0.000000000000000000001$

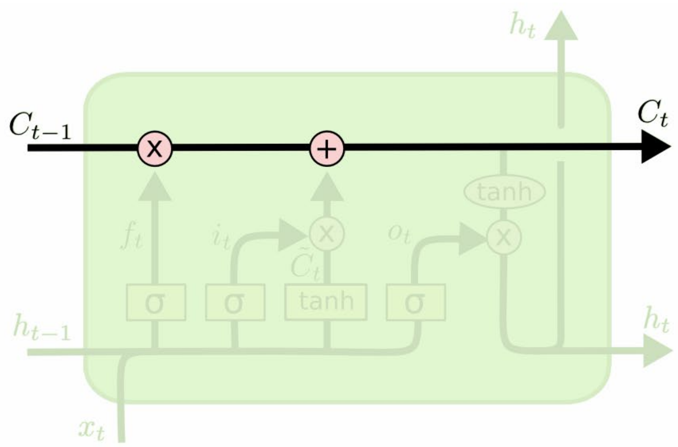
Recurrent Neural Networks

- Long Short-Term Memory networks (LSTMs)
  - Vanishing gradient problem 완화
  - 장기 의존성 학습문제 보완



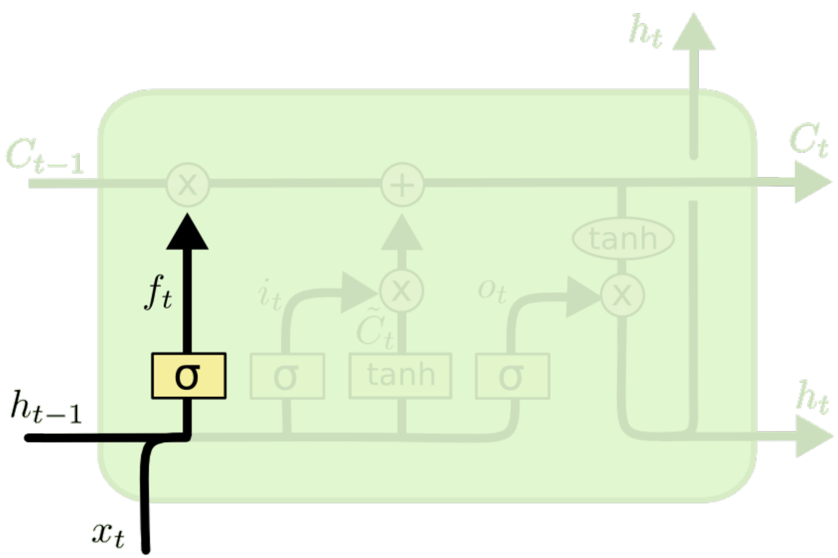
Recurrent Neural Networks

- LSTMs 핵심 아이디어
  - 셀 스테이트 (cell state) - 정보 전달 목적
    - 불필요한 정보 제거
    - 유용한 정보 추가



Recurrent Neural Networks

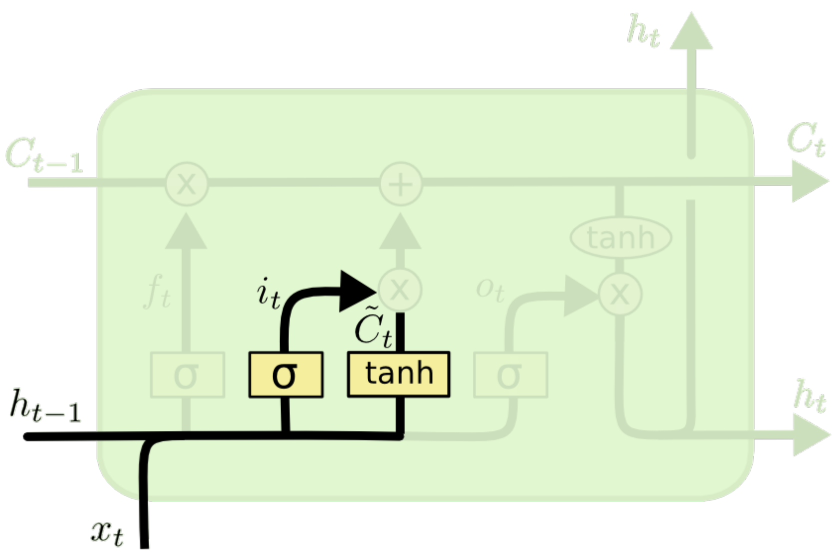
- LSTMs Step1
  - Forget gate layer
    - 어떤 정보를 셀 스테이트에서 제거할 것인지 결정



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

Recurrent Neural Networks

- LSTMs Step2
  - Input gate layer
    - 어떤 정보를 셀 스테이트에 더해 줄 것인지 결정

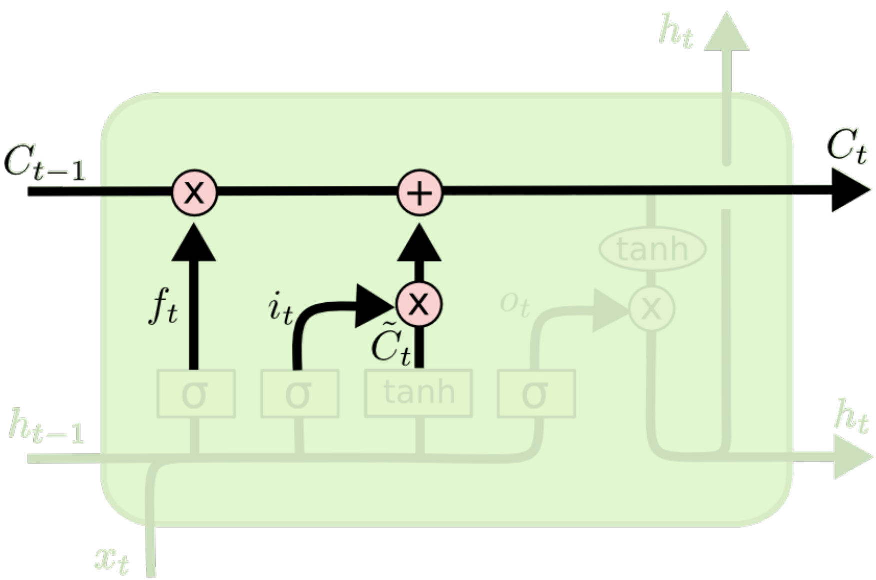


$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \; + \; b_i\right)$$

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

Recurrent Neural Networks

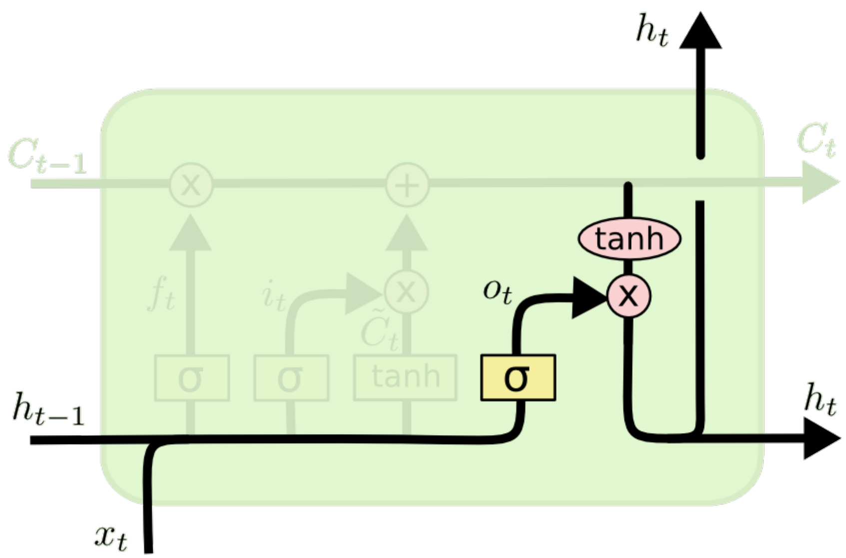
- LSTMs Step3
  - Update the cell state
    - 과거의 C_{t-1}을 새로운 C_t로 업데이트



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

Recurrent Neural Networks

- LSTMs Step4
  - Output gate layer
    - 셀 스테이트로부터 어떤 정보를 읽을 것인지 결정



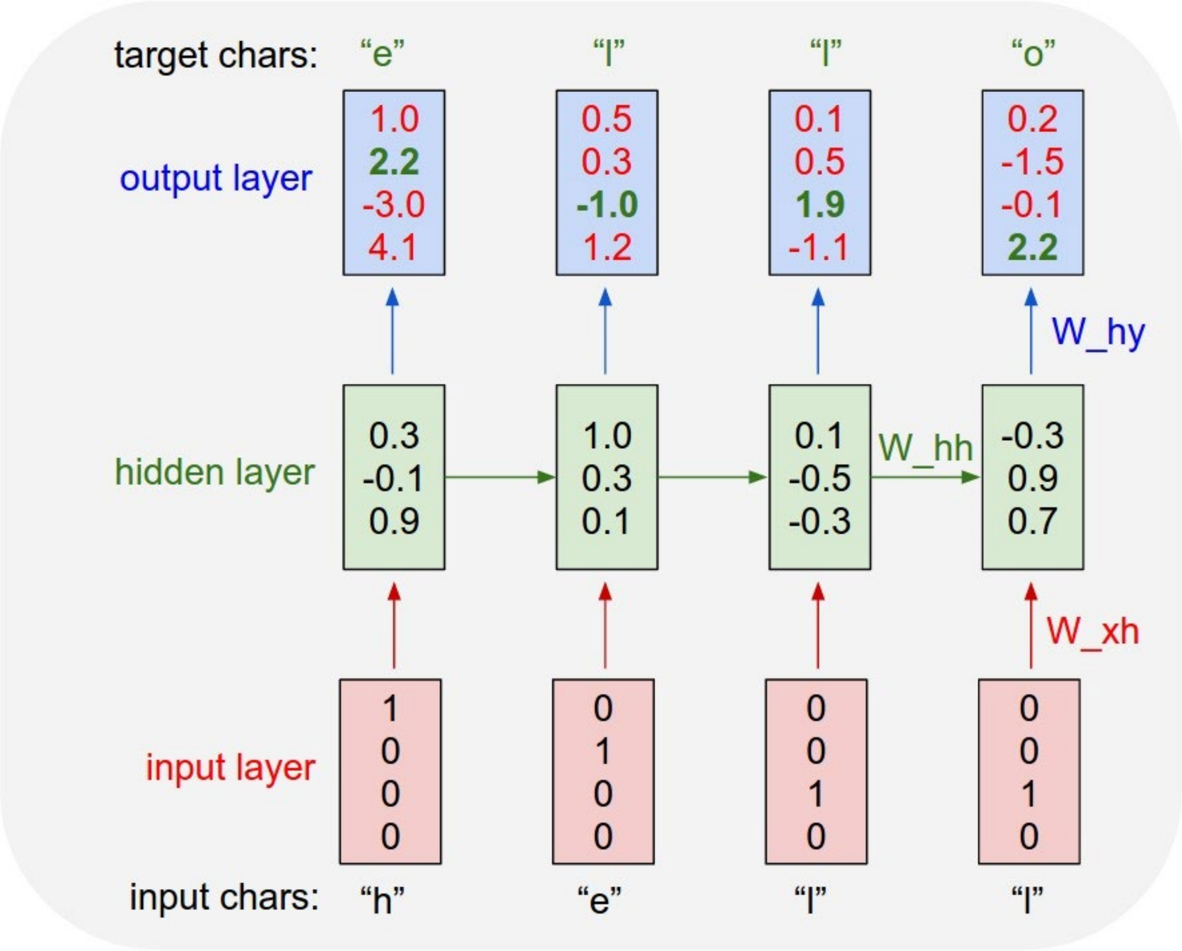
$$o_t = \sigma\left(W_o\ [h_{t-1}, x_t]\ +\ b_o\right)$$

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

Recurrent Neural Networks

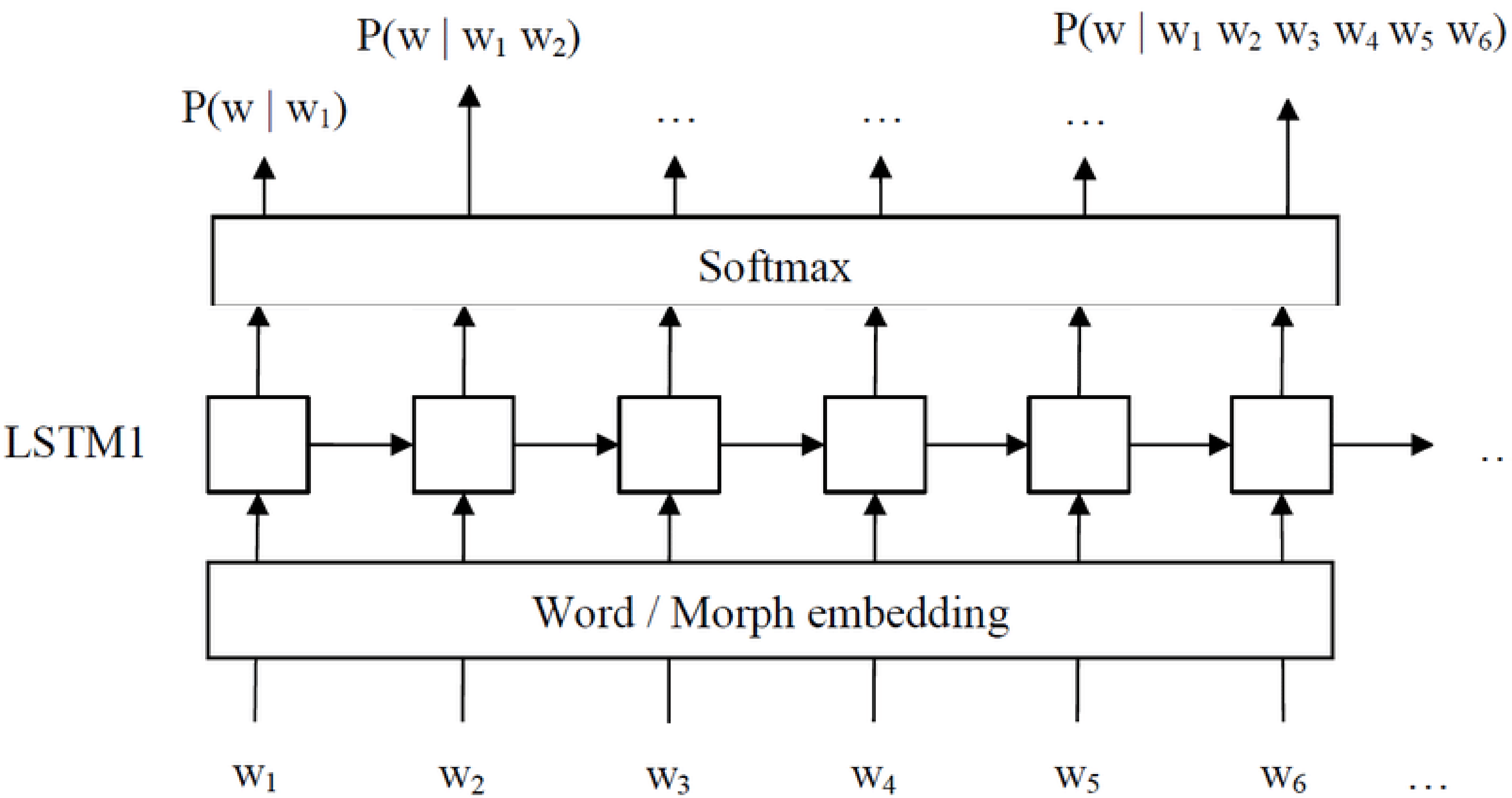
- LSTMs의 다양한 변형
  - Peep hole
  - Forget + Input gate
  - Gated Recurrent Unit (GRU)

RNN 기반의 언어모델



target chars:
“e”
“l”
“l”
“o”
output layer
1.0
2.2
-3.0
4.1
0.5
0.3
-1.0
1.2
0.1
0.5
1.9
-1.1
0.2
-1.5
-0.1
2.2
W_hy
hidden layer
0.3
-0.1
0.9
1.0
0.3
0.1
0.1
-0.5
-0.3
W_hh
-0.3
0.9
0.7
W_xh
input layer
1
0
0
0
0
1
0
0
0
0
1
0
0
0
1
0
input chars:
“h”
“e”
“l”
“l”

RNN 기반의 언어모델



P(w | w1 w2)
P(w | w1 w2 w3 w4 w5 w6)
P(w | w1)
…
…
…
Softmax
LSTM1
…
Word / Morph embedding
w1
w2
w3
w4
w5
w6
…

LSTM 기반 언어모델 실습

LSTM based language model

- Training 과정
  - 학습데이터 (수만 문장 이상)
    - i am a boy .
    - sometimes to understand a word's…
    - at your dictionary we try to gib…
    - …
  - 단어 사전 구축
    - {i=1, am=2, a=3, boy=4, .=5, sometimes=6, …}
  - 문장 속 단어들 → 숫자들로 변환
    - 1 2 3 4 5
    - 6 7 8 9 3 10 …
    - …

LSTM based language model

- Batching
- input

<S>	I	am	a	boy	.	<pad>	<pad>
<S>	sometimes	to	understand	a	word	.	<pad>
<S>	we	try	to	build	a	dictionary	.

- Output (Target)

I	am	a	boy	.	<E>	<pad>	<pad>
sometimes	to	understand	a	word	.	<E>	<pad>
we	try	to	build	a	dictionary	.	<E>

LSTM based language model

- Batching
- input

7	1	2	3	4	5	0	0
7	6	to	7	a	8	5	0
7	9	10	11	12	3	13	5

{〈pad〉=0, i=1, am=2, a=3, boy=4, .=5, sometimes=6, 〈S〉=7, 〈E〉=8, ⋯}

- Output (Target)

1	2	3	4	5	8	0	0
6	to	7	a	8	5	8	0
9	10	11	12	3	13	5	8

LSTM based language model

- Training 과정
  - One-hot representation 변환
    - word idx: 1 2 3 4 5 6

One-hot vector representation

Training sentence

<S>	1	0	0	0	0	0	0	…	0
i	0	1	0	0	0	0	0	…	0
am	0	0	1	0	0	0	0	…	0
a	0	0	0	1	0	0	0	…	0
boy	0	0	0	0	1	0	0	…	0
.	0	0	0	0	0	1	0	…	0
<pad>	0	0	0	0	0	0	1	…	0
<pad>	0	0	0	0	0	0	1	…	0
⋮	⋮								
<pad>	0	0	0	0	0	0	1	…	0

LSTM based language model

- Training 과정
  - Word-embedding 변환
    - word idx: 1 2 3 4 5 6

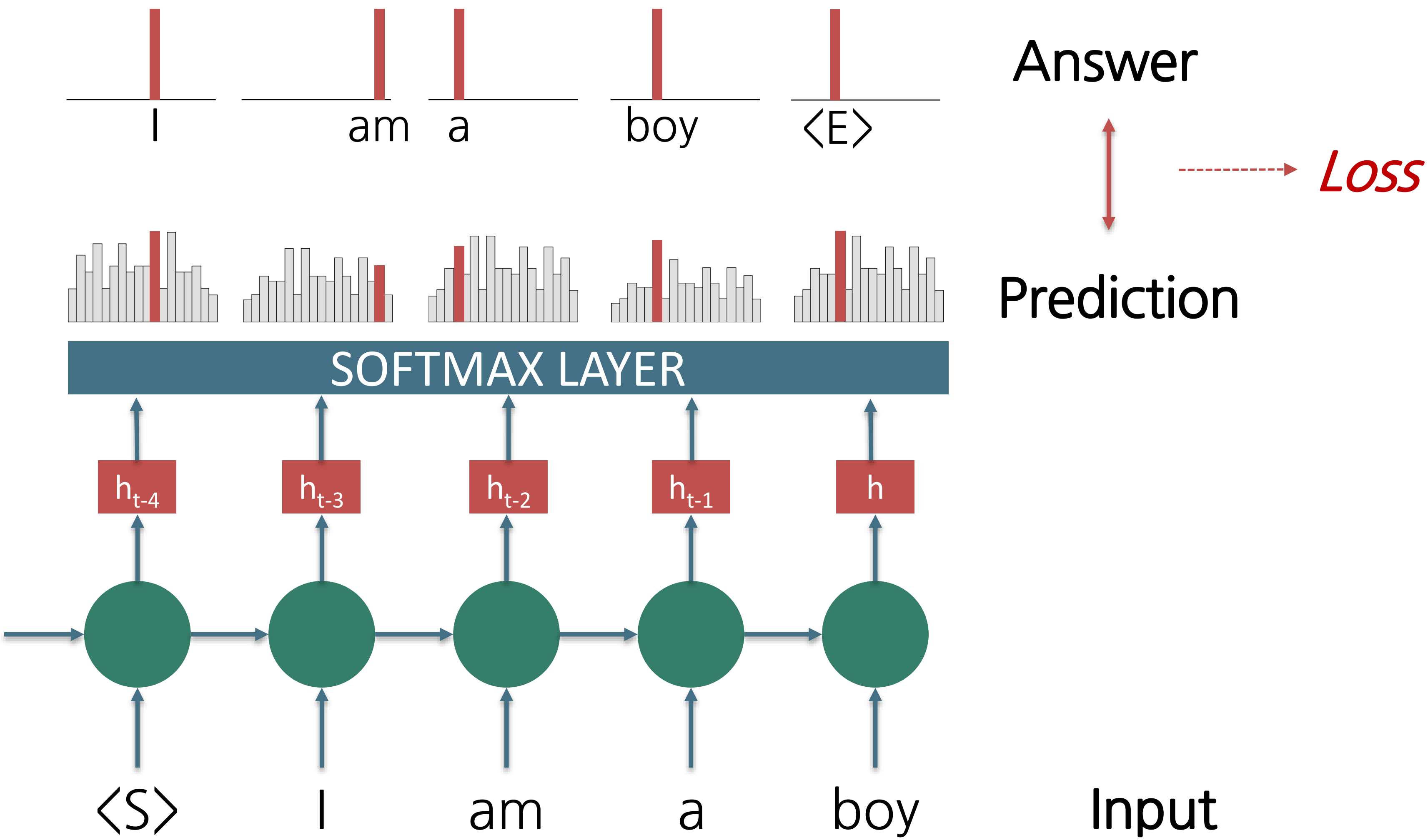
Word-embedding

Training sentence

<S>	0.24	0.15	0.58	0.94	0.14	0.25	0.33	0.85	0.15
i	0.11	0.78	0.91	0.17	0.64	0.75	0.64	0.87	0.36
am	0.78	0.91	0.33	0.87	0.36	0.87	0.36	0.25	0.33
a	0.85	0.15	0.36	0.64	0.78	0.64	0.75	0.87	0.36
boy	0.25	0.33	0.33	0.85	0.64	0.75	0.33	0.64	0.75
.	0.91	0.33	0.64	0.58	0.94	0.25	0.33	0.15	0.58
<pad>	0	0	0	0	0	0	0	0	0
<pad>	0	0	0	0	0	0	0	0	0
⋮	⋮								
<pad>	0	0	0	0	0	0	0	0	0

LSTM based language model

- Training 과정



Training

- Training objective

$$(y_0, y_1), (y_1, y_2), \dots, (y_{n-1}\,, y_n) \sim P(y_n | y_{0:n-1})$$

$$\mathcal{B} = \{(y_{i-1},\ y_i)\}_{i=1}^{n}$$

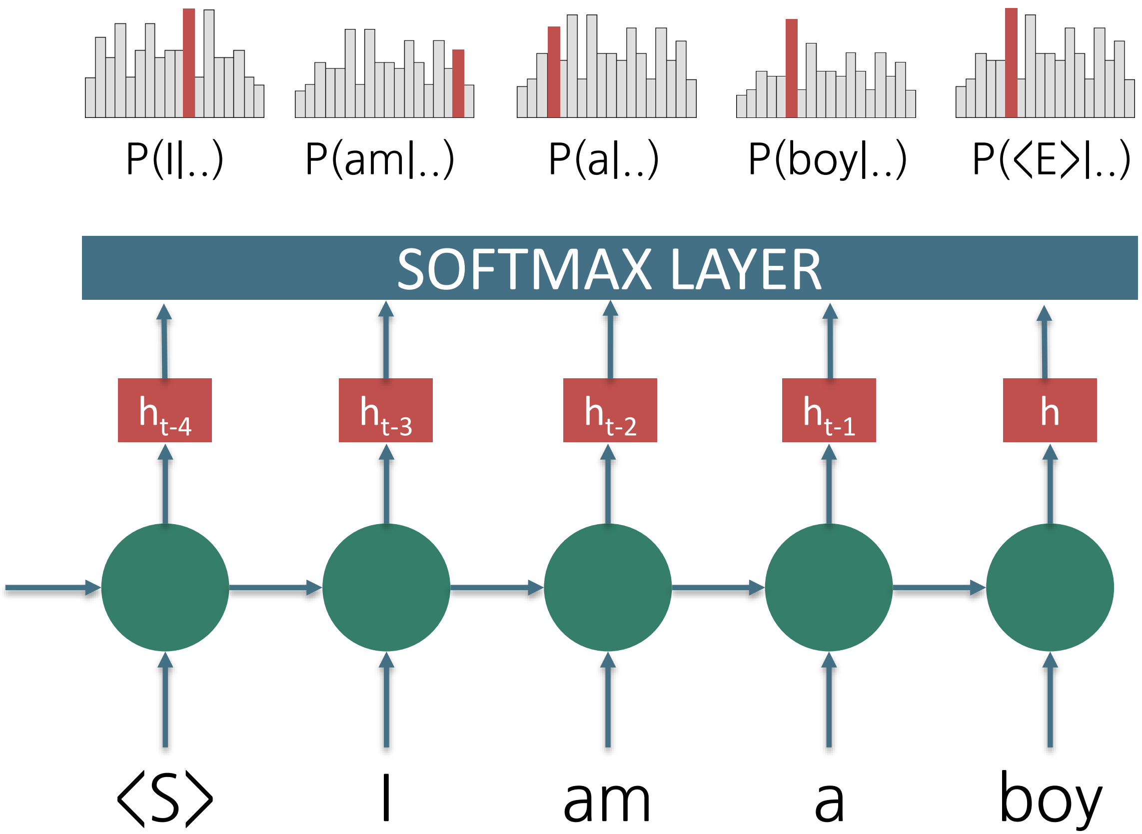
$$\mathcal{L}(\theta) \cong -\frac{1}{n}\sum_{i=1}^{n} \log p_\theta(\hat{y}_i = y_i | y_{0:i-1})$$

- Update

$$\hat{\theta} = \theta - \lambda \nabla_\theta \mathcal{L}(\theta)$$

LSTM based language model

- Testing 과정(1): 문장 확률 계산

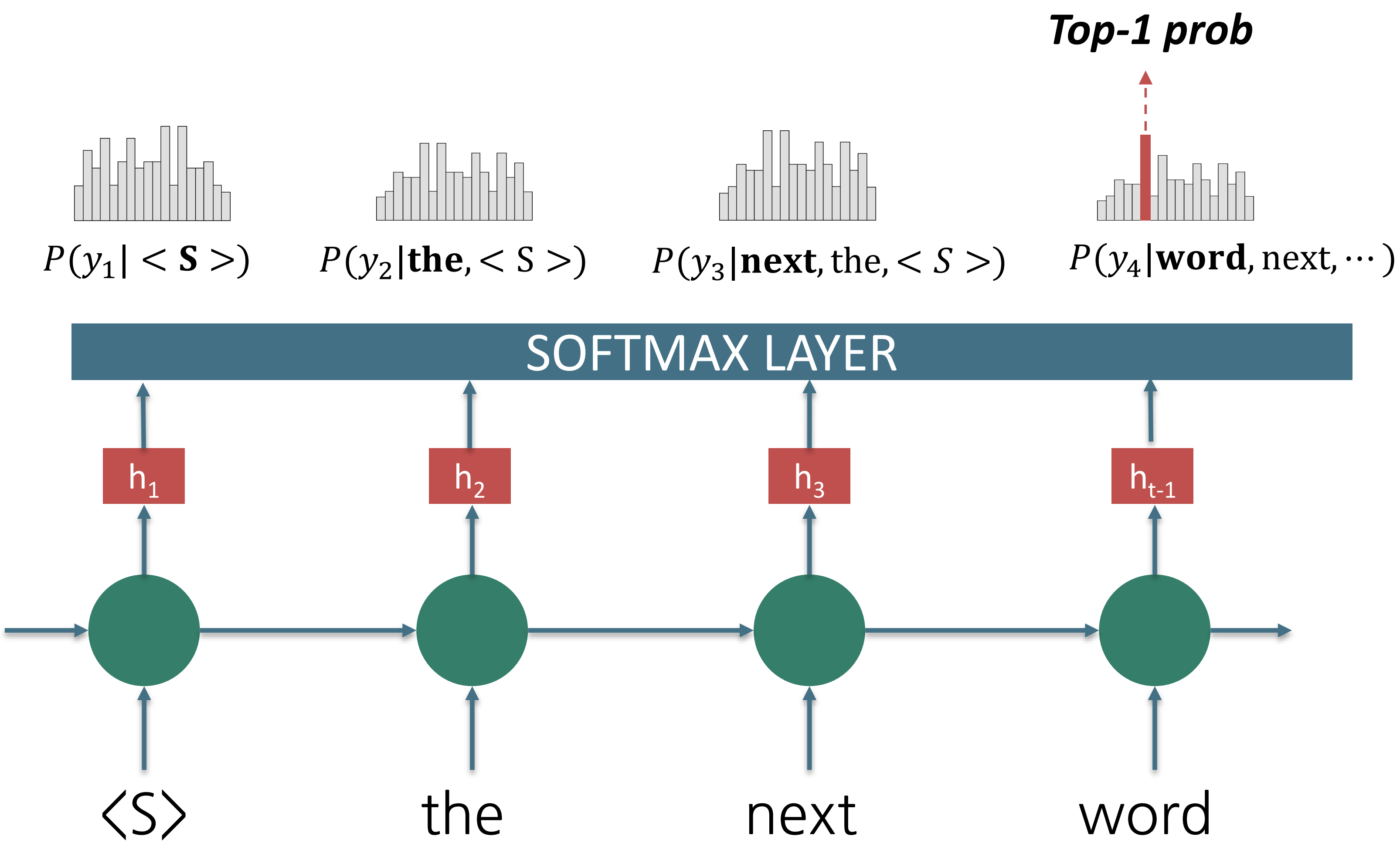


$$P(i, am, a, boy) = P(i|\langle S\rangle) * P(am|I,\langle S\rangle) * P(a|am,I,\langle S\rangle)$$

$$* P(boy|a,am,I\langle S\rangle) * P(\langle E\rangle|boy,a,am,I\langle S\rangle)$$

LSTM based language model

- Testing 과정(2): 다음 단어 예측



CODE REVIEW

언어모델 실습

- NLTK 설치
  - pip install nltk

언어모델 실습 (1)

1. 주어진 문장 Log 확률 분포 계산

- $\log(P(\mathrm{i}, \mathrm{am}, \mathrm{a}, \mathrm{boy}))$
  $= \log(p(\mathrm{i}| < \mathrm{S} >)) + \log(p(\mathrm{am}|\mathrm{i}, < \mathrm{S} >) + \log(p(\mathrm{a}|\mathrm{am}, \mathrm{i}, < \mathrm{S} >))$
  $+ \log(p(\mathrm{boy}|\mathrm{a}, \mathrm{am}, \mathrm{i}, < \mathrm{S} >)) + \log(p(< \mathrm{E} > |\mathrm{boy}, \mathrm{a}, \mathrm{am}, \mathrm{i}, < \mathrm{S} >))$

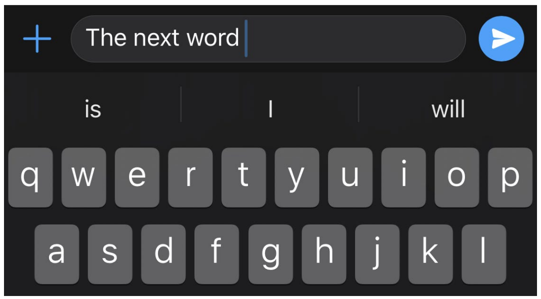
- **결과 출력** (아래 두 문장의 확률 비교)
  - pred_sent_prob([['i', 'am', 'a', 'boy']])
  - pred_sent_prob([['i', 'boy', 'am', 'a']])

In [23]:
```python
# load saved model
with open('./model.pt', 'rb') as f:
    print('load model from: ./model.pt')
    model = torch.load(f).to(device)

    print('log prob of [the dog bark .]: {:3.3f}'.format(pred_sent_prob([['the', 'dog', 'bark', '.']])))
    print('log prob of [the cat bark .]: {:3.3f}'.format(pred_sent_prob([['the', 'cat', 'bark', '.']])))

    print('log prob of [boy am a i .]: {:3.3f}'.format(pred_sent_prob([['boy', 'am', 'a', 'i', '.']])))
    print('log prob of [i am a boy .]: {:3.3f}'.format(pred_sent_prob([['i', 'am', 'a', 'boy', '.']])))
```

```
load model from: ./model.pt
log prob of [the dog bark .]: -38.792
log prob of [the cat bark .]: -42.303
log prob of [boy am a i .]: -45.705
log prob of [i am a boy .]: -19.975
```

언어모델 실습 (2)

2. 다음에 등장할 단어 예측



➔ $argmax\left(\log\left(P\left(\mathrm{y}_4|\mathrm{word, next, the}\right)\right)\right)$

- **결과 출력**
  - pred_next_word([['the', 'next', 'word']], topN=3)

In [25]:
```
partial_sent = [['the', 'next', 'word']]
N=3
candidates = pred_next_word(partial_sent, topN=N)

# print
partial_sent = ' '.join(partial_sent[0])
print('Top {0} next words for a partial sentence [{1}] is: '.format(N, partial_sent))
print('===>', candidates)
```

```
Top 3 next words for a partial sentence [the next word] is:
===> [',', 'of', 'was']
```

Q & A