

Natural Language Processing

자연어 처리

목차

- 자연어 처리란?
- 단어의 분산 표현
- word2vec
- word2vec의 개선
- 순환 신경망 (RNN)
- 게이트가 추가된 RNN

자연어 처리란?



NAVER SPORTS 뉴스 날씨 TV연예

구글플레이 99+

스포츠홈 야구 해외야구 축구 해외축구 농구 배구 골프 일반 e스포츠&게임 | 오늘의 경기 연재 랭킹

최신뉴스 영상 생생회보 일정/결과 기록/순위

이 시각 많이 본 뉴스 ⓘ

1 토트넘 82행에 베일 환호, 레알에서 볼 수 있었던 '한미오음'
2 "FXXK 지역의 페파드" 무리뉴 폭발, 경기 중 터치라인에서...
3 9000원 파운드로는 부족! 트로트문트, 엔딩 제작 또 거절
4 무리뉴 농담, "다이어 목요일엔 뭘 할 거야...내가 차단(?)할..."
5 김지민 시리즈니 다이어 트로트는 화장실... '소변을 왜 안 봐'
6 손흥민 복귀 10월 17일 웨스트 헤ン 유리, 약 3주 이월 전망
7 카라바오와 헨더리와 페파드 등장! 토트넘, 월시 승부차기...
8 이강인 발렌체-소시간 평균 6.3... 아느자리 최고점
9 [eSports] 디아스까지 양팀... 팀의 수리진 소집 총력 6194억
10 적은 출전 시간에 끝난 판더비크 에이전트 '본선 동안 무언...

선발 이강인 평점 6.71 레전드 디비드 실바와 맞대결 성사
창작자 영상

리버풀TV의 분석, 아스널전 역전승의 원동력은?
프리미어리그

무리뉴, "손흥민, 놀랄만큼 많은 시간 뛰었진!"... A매치 휴식기 후 복귀
OSN

"FXXK 지역의 페파드" 무리뉴 폭발, 경기 중 터치라인에서 말싸움
스포츠조선

'벌각소' 밀란원 원맨소 해우개, 일주일 만에 밀란 유니폼 입게 될 사연
골닷컴

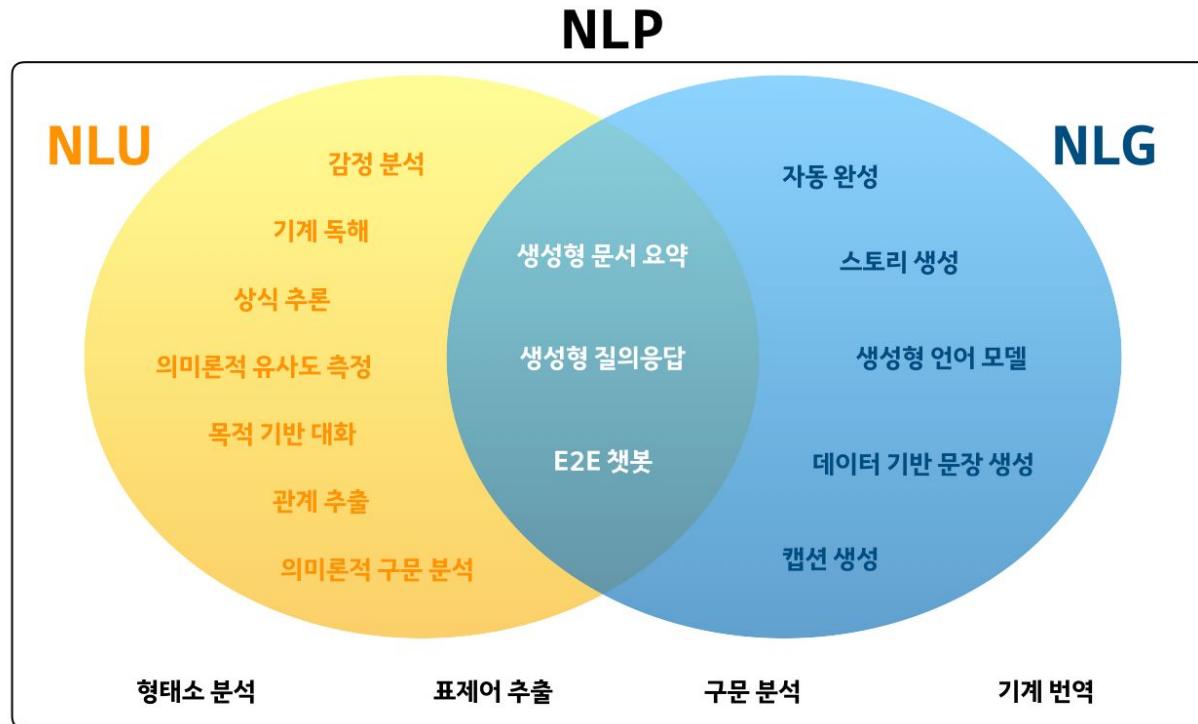
적은 출전 시간에 끝난 판더비크 에이전트 "4분 동안 무엇을 할 수 있나?"
스포츠서울

다시 보고싶은 리버풀 명장면
네이버스포츠 — UFC TV

일상 생활에서 사용 되는 언어를 자연어(Natural Language) 라고 함



자연어 처리란?



자연어 처리 적용 사례 소개



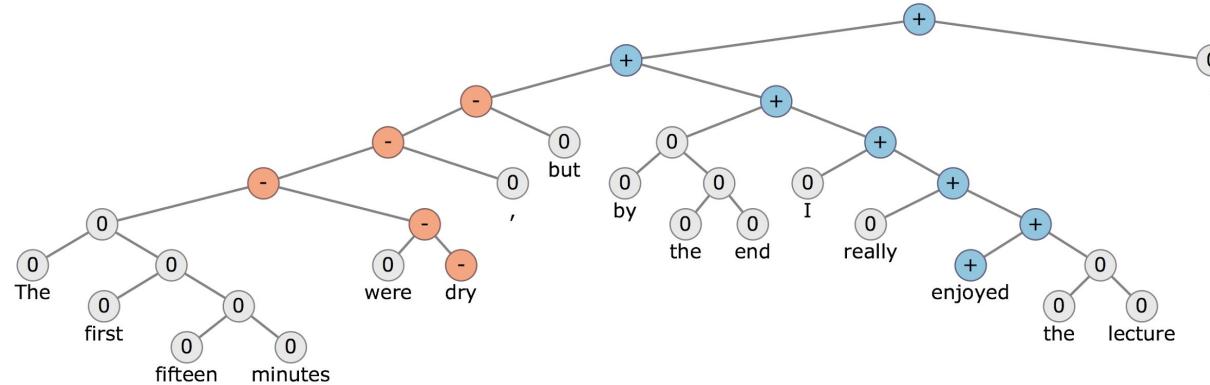
IBM사의 Watson은 2011년 퀴즈쇼에서 사람을 상대로 압도적인 차이로 우승을 차지함

자연어 처리 적용 사례 소개

Sentiment Analysis



Discovering people opinions, emotions and feelings about a product or service



자연어 처리 적용 사례 소개

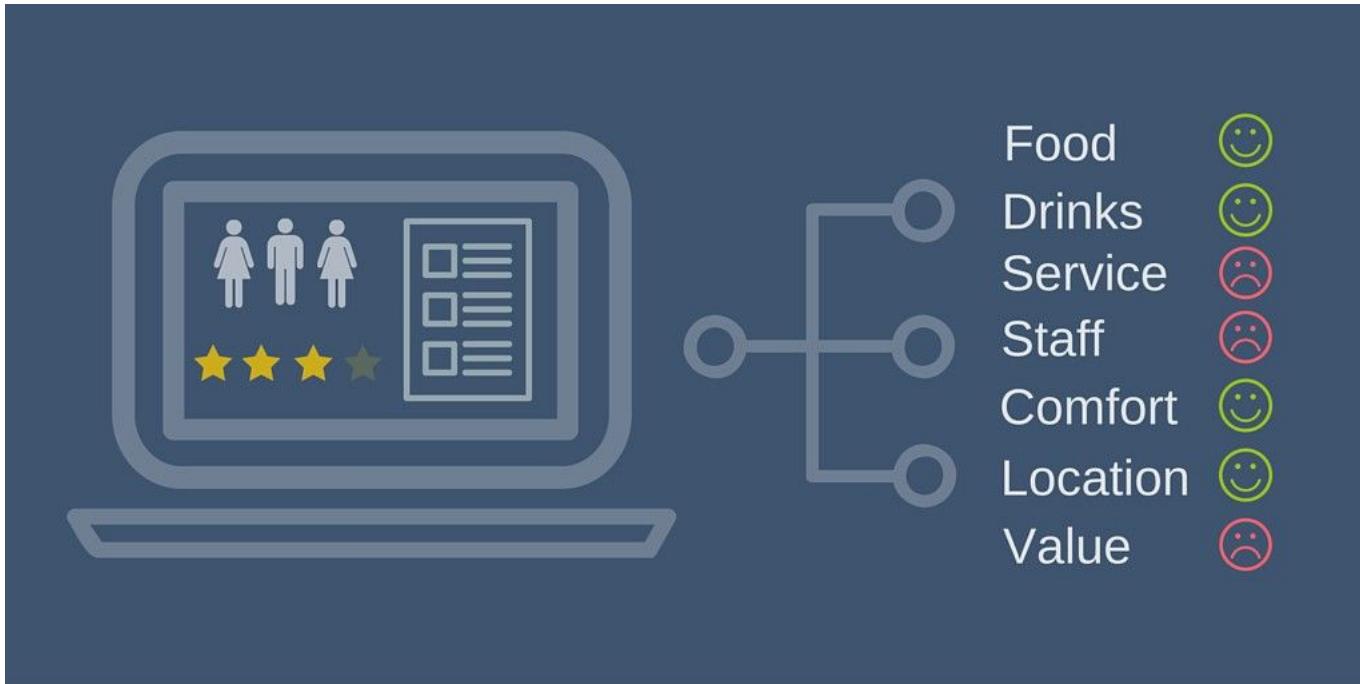
Sentiment Analysis

- 영화 평점, 코멘트 정보를 수집하여 긍/부정 평가
- 온라인 쇼핑몰 후기를 수집하여 상품에 대한 긍/부정 정도 평가
- 뉴스 기사 제목으로 긍정 / 부정 / 중립 분류하는 모델

위 예제들처럼 주어진 텍스트 정보에서 감성 분석이 필요한 경우에 적용할 수 있음

자연어 처리 적용 사례 소개

Aspect-Based Sentiment Analysis



자연어 처리 적용 사례 소개

Text Classification



텍스트 데이터에 있는 단어들과 분류하려는 **label**의 연관성을 학습한다고 볼 수 있음

자연어 처리 적용 사례 소개

Text Summarization

news 1

"80만원대 프리미엄 5G폰 온다"...갤S20 FE, 고급사양에 몸값 낮춰

기사입력 2020.09.23. 오후 11:00 기사등록 스크랩 본문듣기 · 설정

36 36

전면 3200만 화소...후면 카메라도 역대최대 듀얼 이미지센서
추석 이후 10월6일부터 예판...공식 출시는 10월 중순



삼성전자가 23일 온라인으로 열린 세번째 인텍 행사에서 갤럭시S20의 보급형 모델인 '갤럭시S20 팬에디션(FE)'을 공개했다. <삼성전자 제공>

© 뉴스1

(서울=뉴스1) 이창규 기자 = 최근 출시되는 5세대(5G) 스마트폰 출고가가 대부분 120만원대를 웃도는 가운데, 삼성전자가 프리미엄 브랜드와 사양을 그대로 유지하면서 80만원대로 몸값을 '획' 낮춘 갤럭시S20 팬에디션을 공개해 주목된다.

본문 요약봇 ?

자동 추출 기술로 요약된 내용입니다. 요약 기술의 특성상 본문의 주요 내용이 제외될 수 있어, 전체 맥락을 이해하기 위해서는 기사 본문 전체보기를 권장합니다.

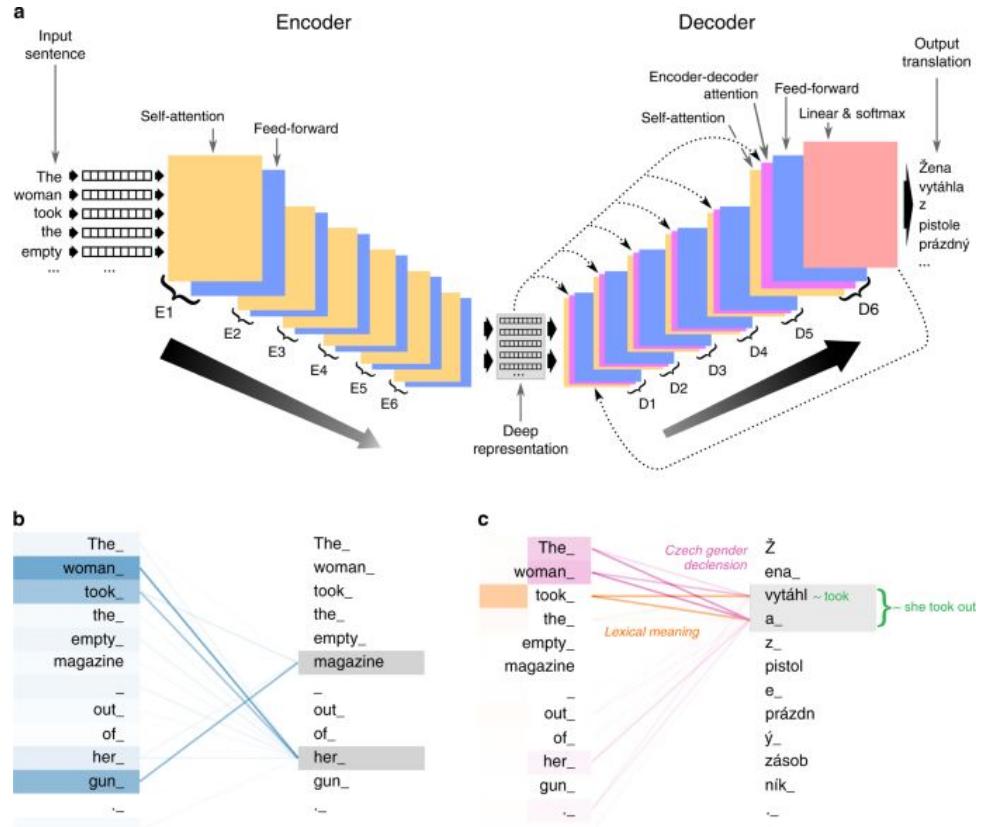
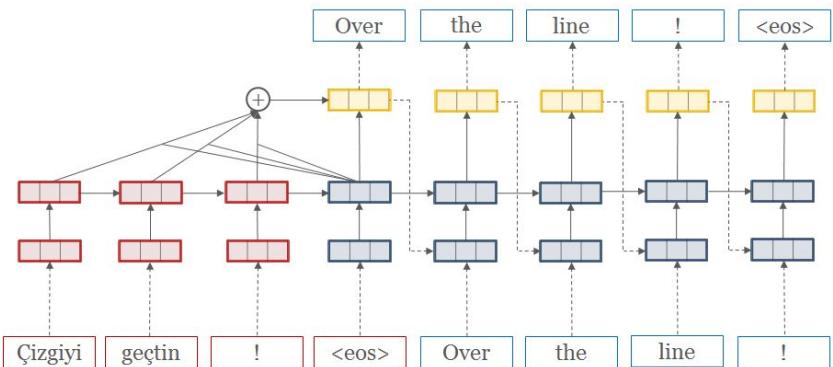
최근 출시되는 5세대 스마트폰 출고가가 대부분 120만원대를 웃도는 가운데, 삼성전자가 프리미엄 브랜드와 사양을 그대로 유지하면서 80만원대로 몸값을 '획' 낮춘 갤럭시S20 팬에디션을 공개해 주목된다.

또한 갤럭시 S20 FE는 5G 이동통신과 엑스박스 게임 패스 얼티밋을 지원, Δ마인크래프트 던전 Δ포르자 호라이즌4 등 100여 개의 엑스박스 인기 게임을 즐길 수 있다.

삼성전자 무선사업부장 노태문 사장은 "삼성전자는 지속적으로 팬들의 피드백을 듣고 소통하고 있으며, 갤럭시 S20 출시 후, 가장 선호하는 부분과 가장 자주 사용하는 기능, 새 스마트폰에 기대하고 있는 점 등에 귀를 기울였다"며 "갤럭시 S20 FE는 의미 있는 혁신이 담긴 갤럭시 S20 시리즈의 확장 모델로 최고의 갤럭시 스마트폰 경험을 더 많은 소비자들이 누릴 수 있게 할 것"이라고 말했다.

자연어 처리 적용 사례 소개

Machine Translation



자연어 처리

단어의 의미

컴퓨터에게 단어의 의미를 전달하기 위해서는 어떤 방법을 사용할까?

- 시소러스를 활용한 기법
- 통계 기반 기법
- 추론 기반 기법

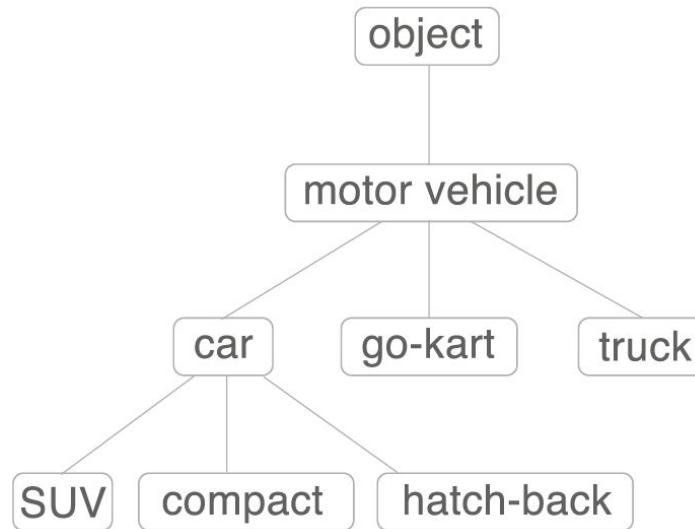
자연어 처리

시소러스 (thesaurus : 유의어 사전)

그림 2-1 동의어의 예: “car”, “auto”, “automobile” 등은 “자동차”를 뜻하는 동의어다.



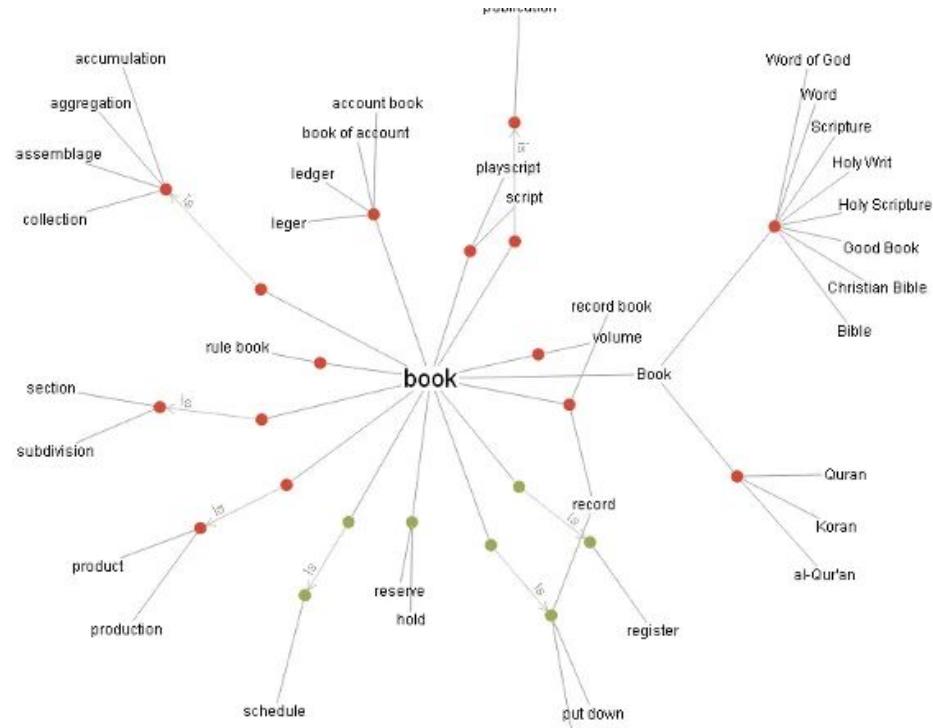
그림 2-2 단어들을 의미의 상 · 하위 관계에 기초해 그래프로 표현한다(문헌 [14]를 참고하여 그림).



자연어 처리

WordNet

자연어 처리 분야에서 가장 유명한 시소러스



자연어 처리

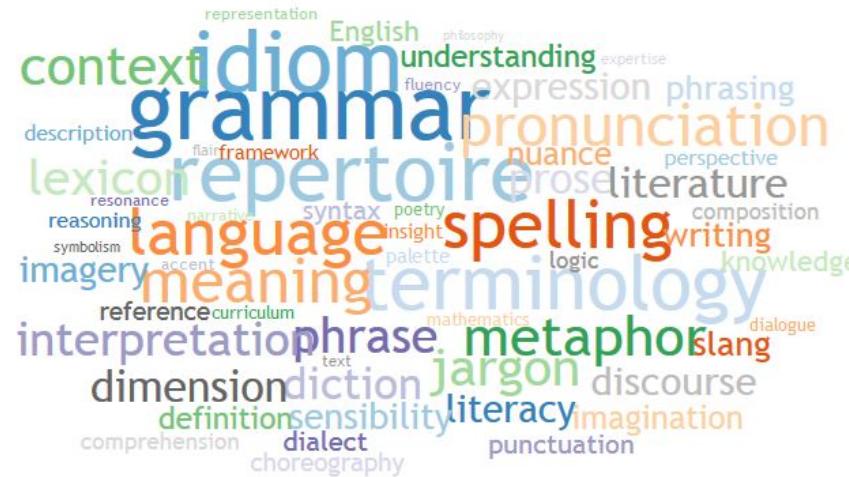
시소러스의 단점

- 시대 변화에 대응하기 어렵다
- 사람을 쓰는 비용이 크다
- 단어의 미묘한 차이를 표현할 수 없다

자연어 처리

통계 기반 기법

말뭉치 (corpus)



자연어 처리를 목적으로 수집한 대량의 텍스트 데이터를 뜻함

자연어 처리

말뭉치 전처리 맛보기

```
말뭉치 . text = 'You say goodbye and I say hello.'
```

```
1 text = text.lower()
2 text = text.replace('. ', '.')
3 text
```

전처리 & 단어 분할 'you say goodbye and i say hello .'

```
1 words = text.split(' ')
2 words
['you', 'say', 'goodbye', 'and', 'i', 'say', 'hello', '.']
```

```
1 word_to_id
{'.': 6, 'and': 3, 'goodbye': 2, 'hello': 5, 'i': 4, 'say': 1, 'you': 0}
```

ID 할당

```
1 id_to_word
{0: 'you', 1: 'say', 2: 'goodbye', 3: 'and', 4: 'i', 5: 'hello', 6: '.'}
```

통계 기반 기법

단어의 분산 표현

- 분산 표현 (Distributional representation)

단어를 고정 길이의 밀집벡터(Dense vector)로 표현함

단어끼리의 관련성 파악에 유리함

- 분포 가설 (Distributional Hypothesis)

“ 단어의 의미는 주변 단어에 의해 형성된다 ”

- 동시발생 행렬

주변에 어떤 단어가 몇번이나 등장하는지 집계하는 방법

통계 기반 기법

분포 가설

단어 자체의 의미보다 그 단어가 사용된 ‘**맥락(Context)**’이 의미를 형성한다는 개념

그림 2-3 원도우 크기가 2인 ‘맥락’의 예. 단어 “goodbye”에 주목한다면, 그 좌우의 두 단어(총 네 단어)를 맥락으로 이용한다.



맥락은 특정 단어를 중심에 둔 그 주변 단어를 말함
주변의 몇 단어를 볼지를 원도우 크기로 정의함

통계 기반 기법

동시발생 행렬 (Co-occurrence matrix)

그림 2-5 단어 “you”의 맥락에 포함되는 단어의 빈도를 표로 정리한다.

	you	say	goodbye	and	i	hello	.
you	0	1	0	0	0	0	0

그림 2-6 단어 “say”의 맥락에 포함되는 단어의 빈도를 표로 정리한다.



	you	say	goodbye	and	i	hello	.
say	1	0	1	0	1	1	0

통계 기반 기법

동시발생 행렬 (Co-occurrence matrix)

그림 2-7 모든 단어 각각의 맥락에 해당하는 단어의 빈도를 세어 표로 정리한다.

	you	say	goodbye	and	i	hello	.
you	0	1	0	0	0	0	0
say	1	0	1	0	1	1	0
goodbye	0	1	0	1	0	0	0
and	0	0	1	0	1	0	0
i	0	1	0	1	0	0	0
hello	0	1	0	0	0	0	1
.	0	0	0	0	0	1	0

문장에 등장하는 모든 단어에 대한 동시발생 행렬

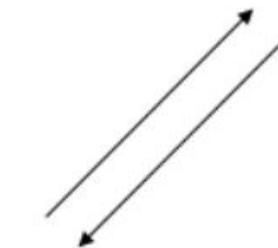
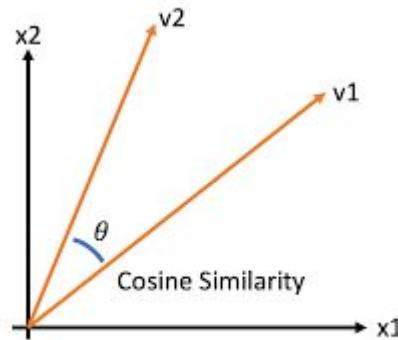
동시발생 행렬을 통해 두 단어의 유사도를 측정할 수 있음

통계 기반 기법

벡터 간 유사도

코사인
유사도

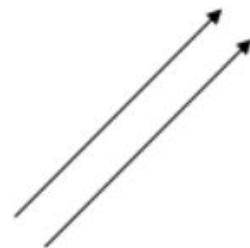
$$\text{similarity}(\mathbf{x}, \mathbf{y}) = \frac{\mathbf{x} \cdot \mathbf{y}}{\|\mathbf{x}\| \|\mathbf{y}\|} = \frac{x_1 y_1 + \cdots + x_n y_n}{\sqrt{x_1^2 + \cdots + x_n^2} \sqrt{y_1^2 + \cdots + y_n^2}}$$



코사인 유사도 : -1



코사인 유사도 : 0



코사인 유사도 : 1

두 단어가 얼마나 가까운 거리에 있는지 (비슷한 특징을 가지고 있는지) 판단할 수 있는 척도가 될 수 있음

통계 기반 기법 개선하기

상호 정보량

문장에서 ‘the car’ 두 단어가 동시발생하는 횟수 자체는 많겠지만,
car는 drive와 더 관련이 깊기 때문에 등장 횟수만 본다면 the와 더 관련성이 높다고 측정될 것

점별 상호정보량 (Pointwise Mutual Information)

$$\text{PMI}(x, y) = \log_2 \frac{P(x, y)}{P(x)P(y)}$$

단어 x가 말뭉치에 등장할
확률

통계 기반 기법 개선하기

상호 정보량

점별 상호정보량 (**Pointwise Mutual Information**)

식을 동시발생 행렬로

정리하면 ...

$$\text{PMI}(x,y) = \log_2 \frac{P(x,y)}{P(x)P(y)} = \log_2 \frac{\frac{C(x,y)}{N}}{\frac{C(x)}{N} \frac{C(y)}{N}} = \boxed{\log_2 \frac{C(x,y) \cdot N}{C(x)C(y)}}$$

통계 기반 기법 개선하기

상호 정보량

말뭉치 단어 수 (N) : 10,000

‘the’, ‘car’, ‘drive’가 각각 1,000, 20, 10 번씩

‘the car’, ‘car drive’는 각각 10회, 5회 등장했다고 가정

$$\text{PMI}(\text{"the"}, \text{"car"}) = \log_2 \frac{10 \cdot 10000}{1000 \cdot 20} \approx 2.32$$

$$\text{PMI}(\text{"car"}, \text{"drive"}) = \log_2 \frac{5 \cdot 10000}{20 \cdot 10} \approx 7.97$$

동시 발생은 the car가 많지만, 말뭉치에서 the의 발생 회수가 훨씬 많기 때문에 PMI 지수가 작아지는
결과를 확인할 때 동시 발생 횟수가 0이면 계산할 불가능하기 때문에 **PPMI 지수**를 사용함

통계 기반 기법 개선하기

상호 정보량

양의 상호정보량 (**Positive Pointwise Mutual Information**)

$$\text{PPMI}(x, y) = \max(0, \text{PMI}(x, y))$$

단어 유사도에 대한 정보는 개선했지만...

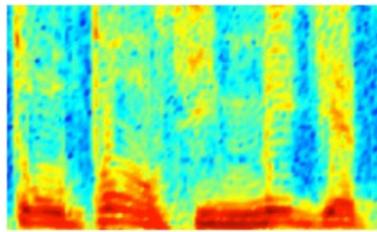
단어 수가 많아지면 행렬이 지나치게 커지는 단점이 존재함

단어 수가 많아질수록 행렬의 내용에 0이 많아지게 됨

통계 기반 기법 개선하기

상호 정보량

AUDIO



Audio Spectrogram

DENSE

IMAGES

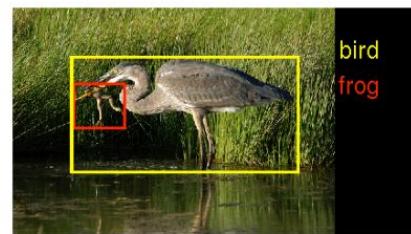
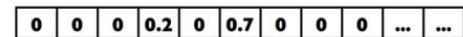


Image pixels

DENSE

TEXT



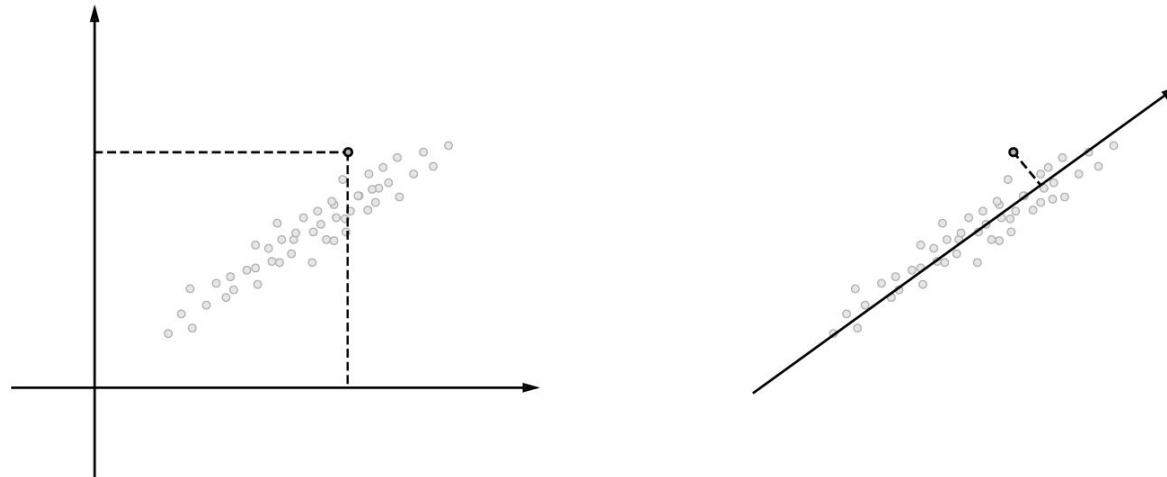
Word, context, or document vectors

SPARSE

통계 기반 기법 개선하기

차원 감소 (Dimensionality reduction)

그림 2-8 그림으로 이해하는 차원 감소: 2차원 데이터를 1차원으로 표현하기 위해 중요한 축(데이터를 넓게 분포시키는 축)을 찾는다.



축을 새로 정의하여 더 낮은 차원에서 유의미한 정보를 나타내는 것이 목적

통계 기반 기법 개선하기

차원 감소 (Dimensionality reduction)

특잇값분해 (Singular Value Decomposition)

$$\mathbf{X} = \mathbf{U}\mathbf{S}\mathbf{V}^T$$

그림 2-9 SVD에 의한 행렬의 변환(행렬의 '흰 부분'은 원소가 0임을 뜻함)

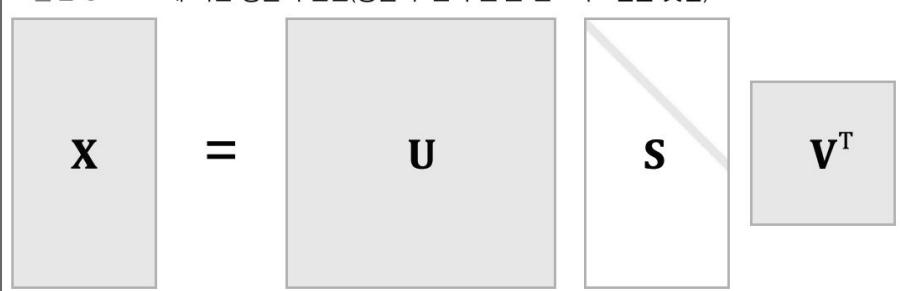


그림 2-10 SVD에 의한 차원 감소



하지만 SVD도 행렬이 커지면 계산량이 매우 커지므로..

Truncated SVD같은 기법을 활용함 (특잇값이 작은 것은 버리는 방식)

단어의 분산 표현

지금까지 배운 내용은...

- WordNet 등의 시소러스를 이용하면 단어의 유사도를 측정하는 작업을 할 수 있음
- 시소러스는 여러 단점이 있음 (새로운 단어 등장 등)
- 말뭉치를 이용해 단어를 벡터화 함
- 분포 가설 : ‘단어의 의미는 주변 단어에 의해 형성된다’
- 동시발생 행렬
- PPMI, 차원 감소를 통해 희소벡터를 밀집벡터로 변환 가능
- 단어의 벡터 공간에서 비슷한 의미의 단어들은 서로 거리가 가까울 것

단어의 분산 표현

PTB (Penn Tree Bank) dataset

- PTB 데이터셋은 상대적으로 크기가 작고, 빠르게 학습이 가능함
- 주로 월스트리트 저널의 문장들로 이루어짐
- 330만 어절 이상

[PTB데이터셋 다운로드 코드](#)

추론 기반 기법

word2vec

- 통계 기법 기반보다 강력한 ‘**추론 기반 기법**’
- 단순한 **word2vec**을 구현하려함
- SVD 등 통계 기반 기법은 거대 행렬을 다루어야하는 치명적 단점이 존재함

그림 3-1 통계 기반 기법과 추론 기반 기법 비교



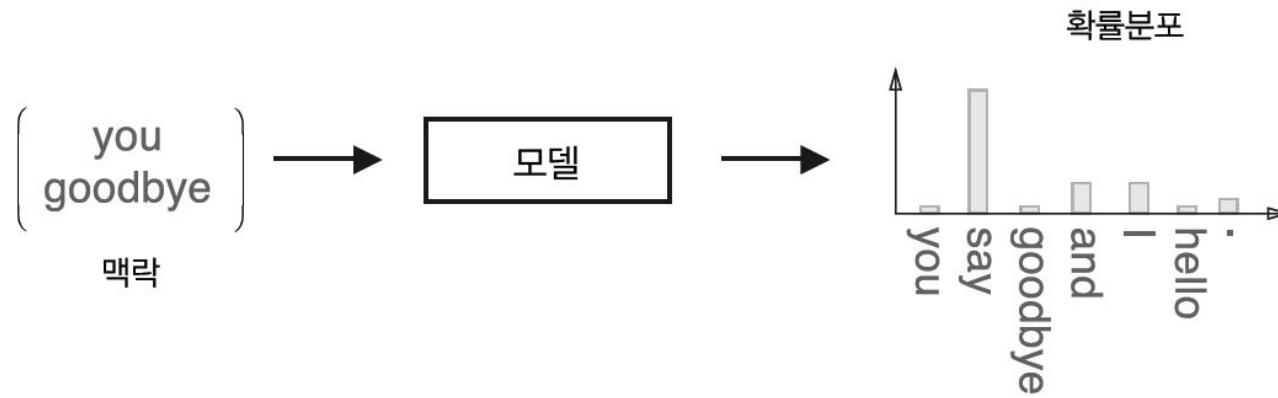
추론 기반 기법

추론 기반 기법 개요

그림 3-2 주변 단어들을 맥락으로 사용해 “?”에 들어갈 단어를 추측한다.

you **?** goodbye and I say hello.


그림 3-3 추론 기반 기법: 맥락을 입력하면 모델은 각 단어의 출현 확률을 출력한다.



추론 기반 기법

추론 기반 기법 개요

그림 3-2 주변 단어들을 맥락으로 사용해 “?”에 들어갈 단어를 추측한다.

you  goodbye and I say hello.



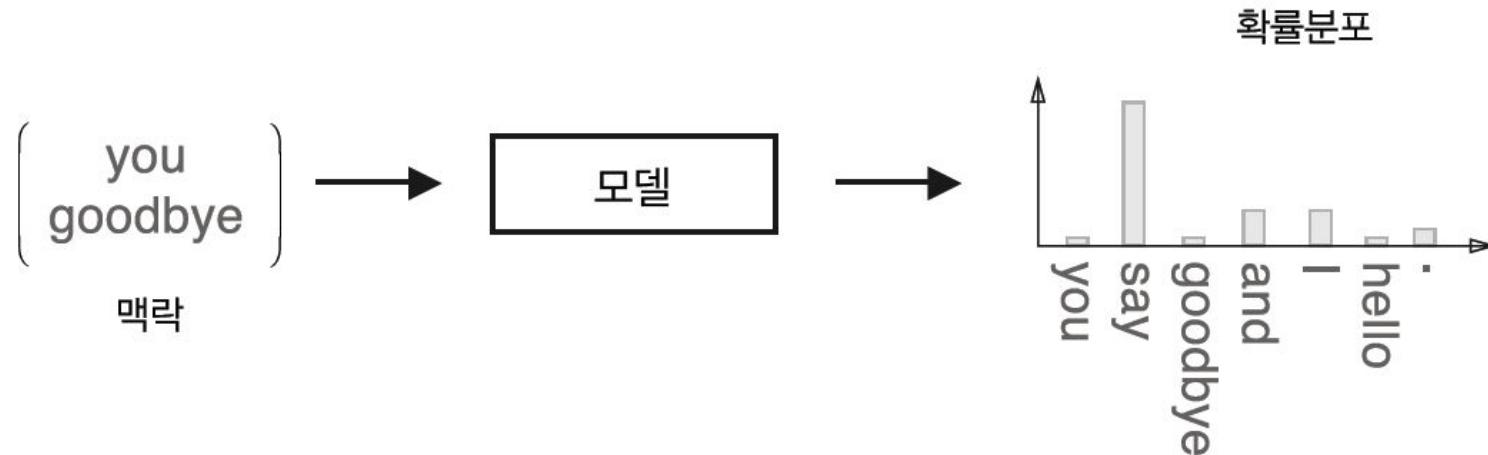
맥락이 주어졌을 때 “?”에 어떤 단어가 들어가는지 추측

추론 기반 기법

추론 기반 기법 개요

모델 관점에서 본다면?

그림 3-3 추론 기반 기법: 맥락을 입력하면 모델은 각 단어의 출현 확률을 출력한다.



추론 기반 기법

신경망에서의 단어 처리

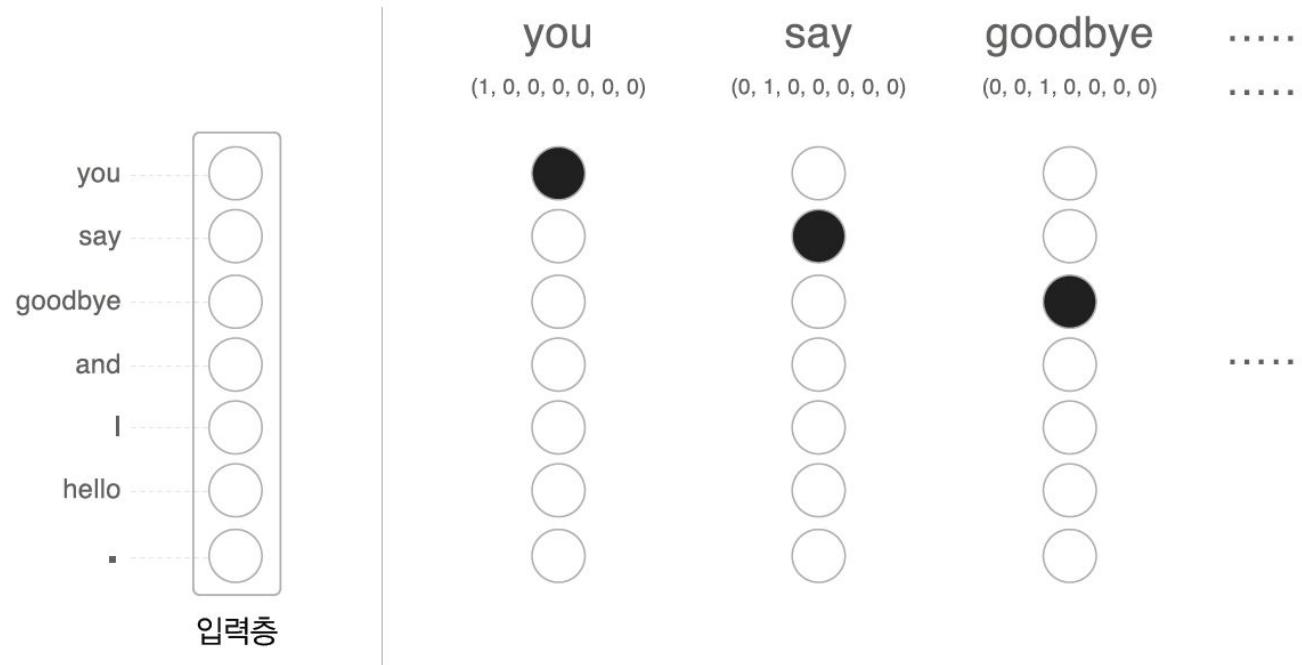
그림 3-4 단어, 단어 ID, 원핫 표현

단어(텍스트)	단어 ID	원핫 표현
$\begin{pmatrix} \text{you} \\ \text{goodbye} \end{pmatrix}$	$\begin{pmatrix} 0 \\ 2 \end{pmatrix}$	$\begin{pmatrix} (1, 0, 0, 0, 0, 0, 0) \\ (0, 0, 1, 0, 0, 0, 0) \end{pmatrix}$

추론 기반 기법

신경망에서의 단어 처리

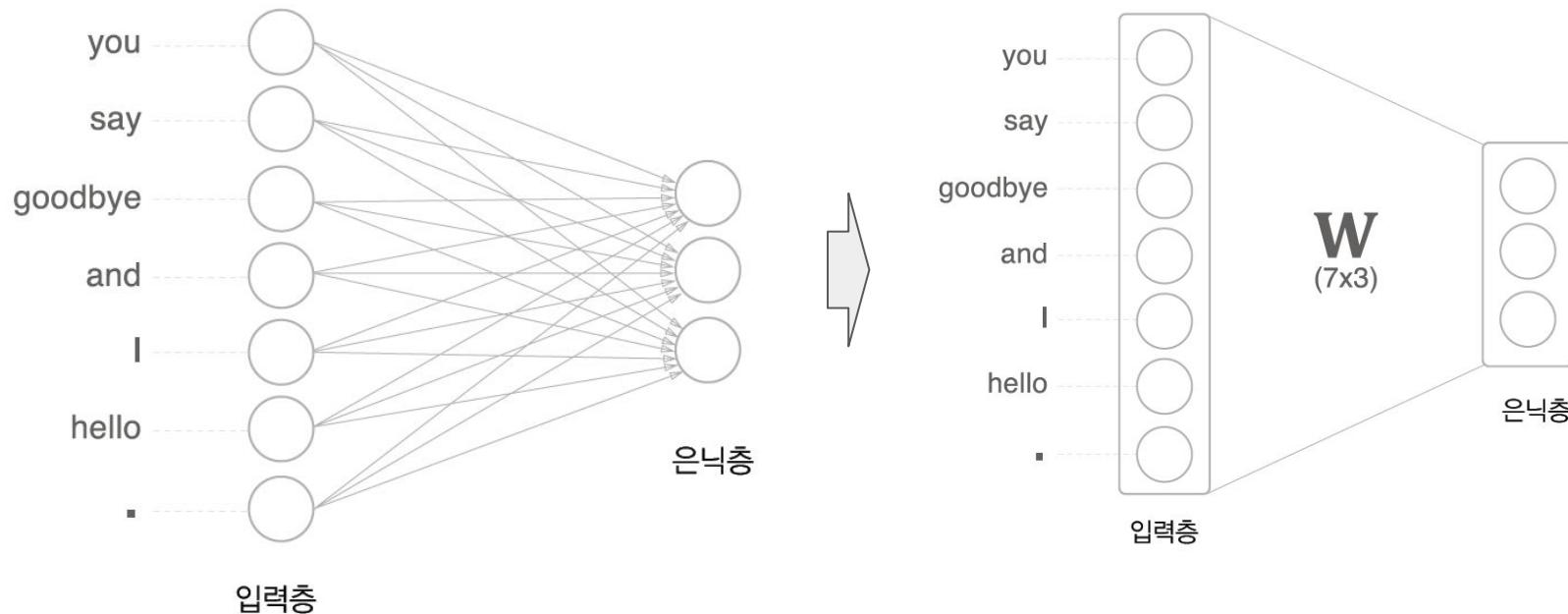
그림 3-5 입력층의 뉴런: 각 뉴런이 각 단어에 대응(해당 뉴런이 1이면 검은색, 0이면 흰색)



추론 기반 기법

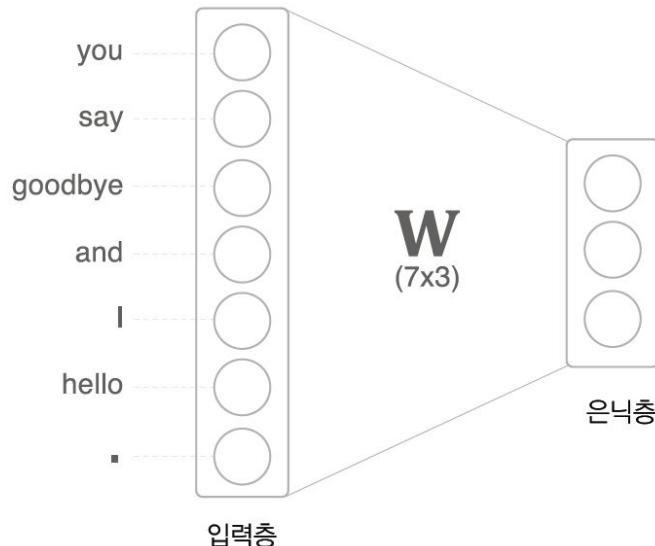
신경망에서의 단어 처리

그림 3-6 완전연결계층에 의한 변환: 입력층의 각 뉴런은 7개의 단어 각각에 대응(은닉층 뉴런은 3개를 준비함)



추론 기반 기법

신경망에서의 단어 처리



```
1 import numpy as np
```

```
1 c = np.array([[1,0,0,0,0,0,0]])  
2 W = np.random.randn(7,3)  
3 h = np.matmul(c, W)
```

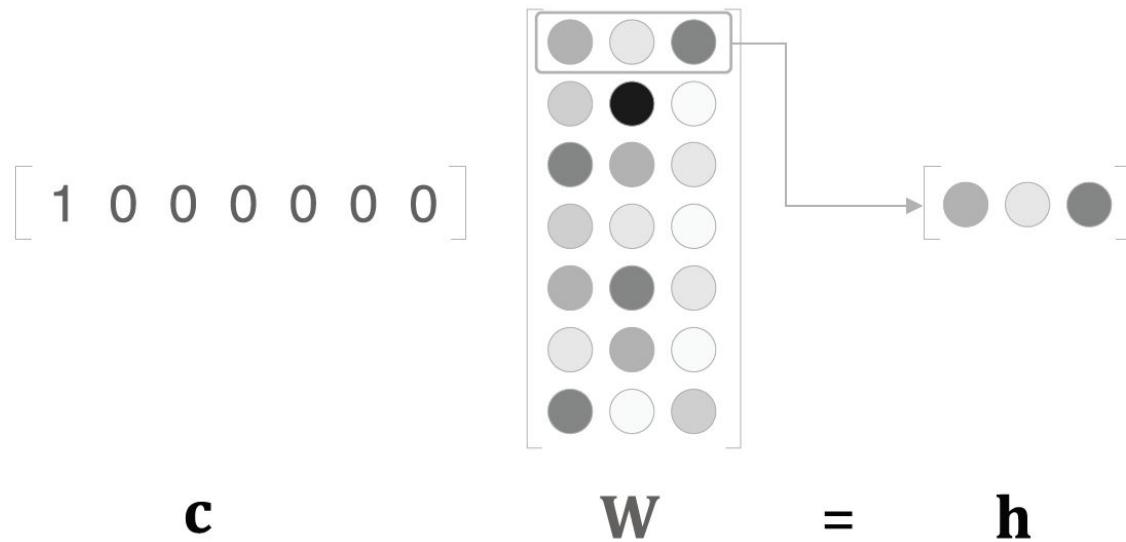
```
1 print(h)
```

```
[-0.6051838  0.96355666  0.13499427]
```

추론 기반 기법

신경망에서의 단어 처리

그림 3-8 맥락 \mathbf{c} 와 가중치 \mathbf{W} 의 곱으로 해당 위치의 행벡터가 추출된다(각 요소의 가중치 크기는 흑백의 진하기로 표현).



단순한 word2vec

word2vec은 프로그램이나 도구를 가리키는 용어였지만, 용어의 활용이 늘어나면서 신경망 모델을 지칭하는 경우가 많아짐

대표적으로 두 가지 모델이 사용됨

- CBOW 모델
- skip-gram 모델

단순한 word2vec

CBOW (Continuous Bag-Of-Words)

그림 3-9 CBOW 모델의 신경망 구조

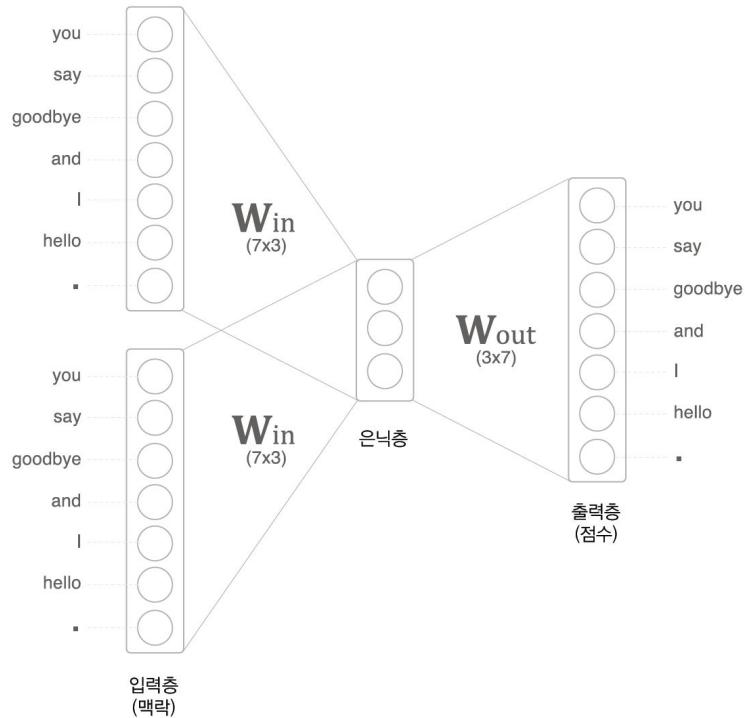
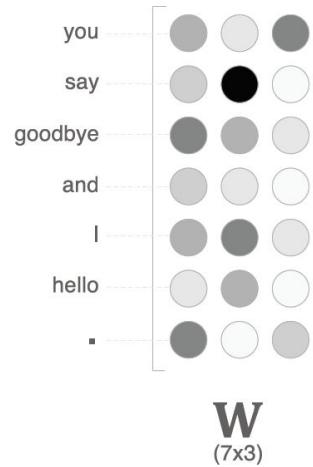
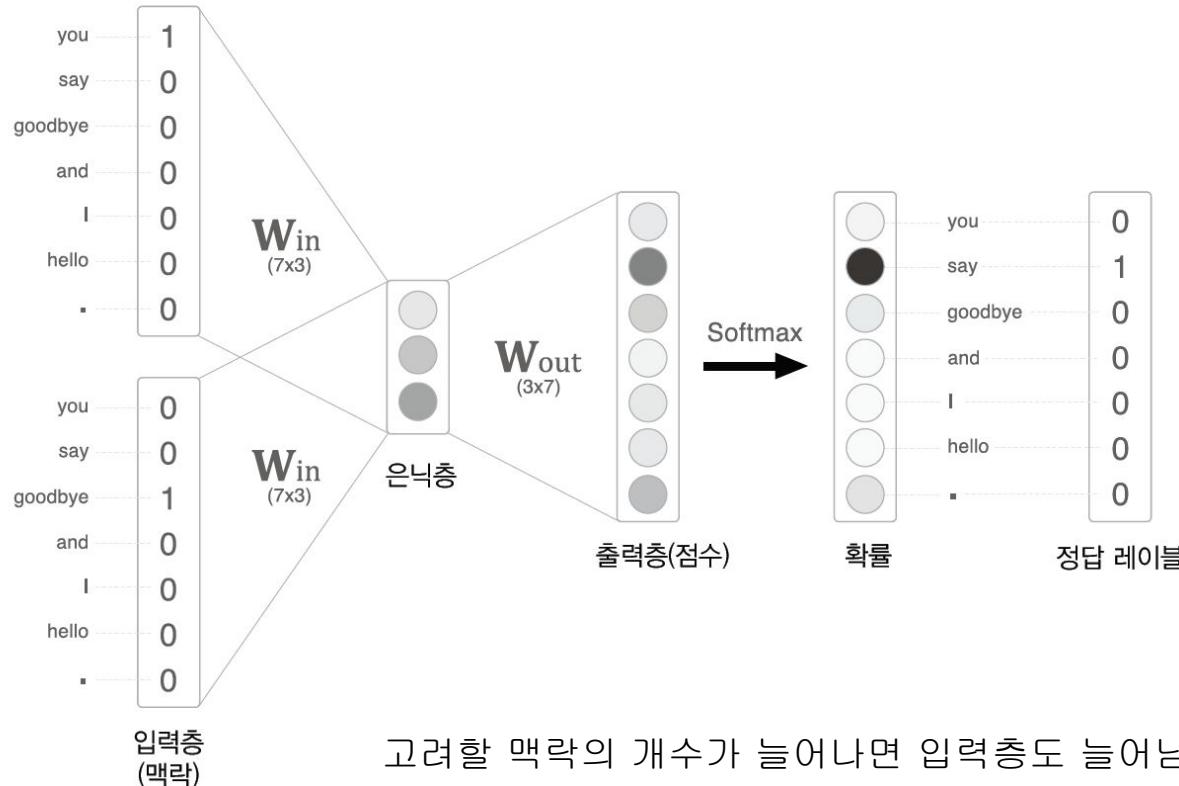


그림 3-10 가중치의 각 행이 해당 단어의 분산 표현이다.



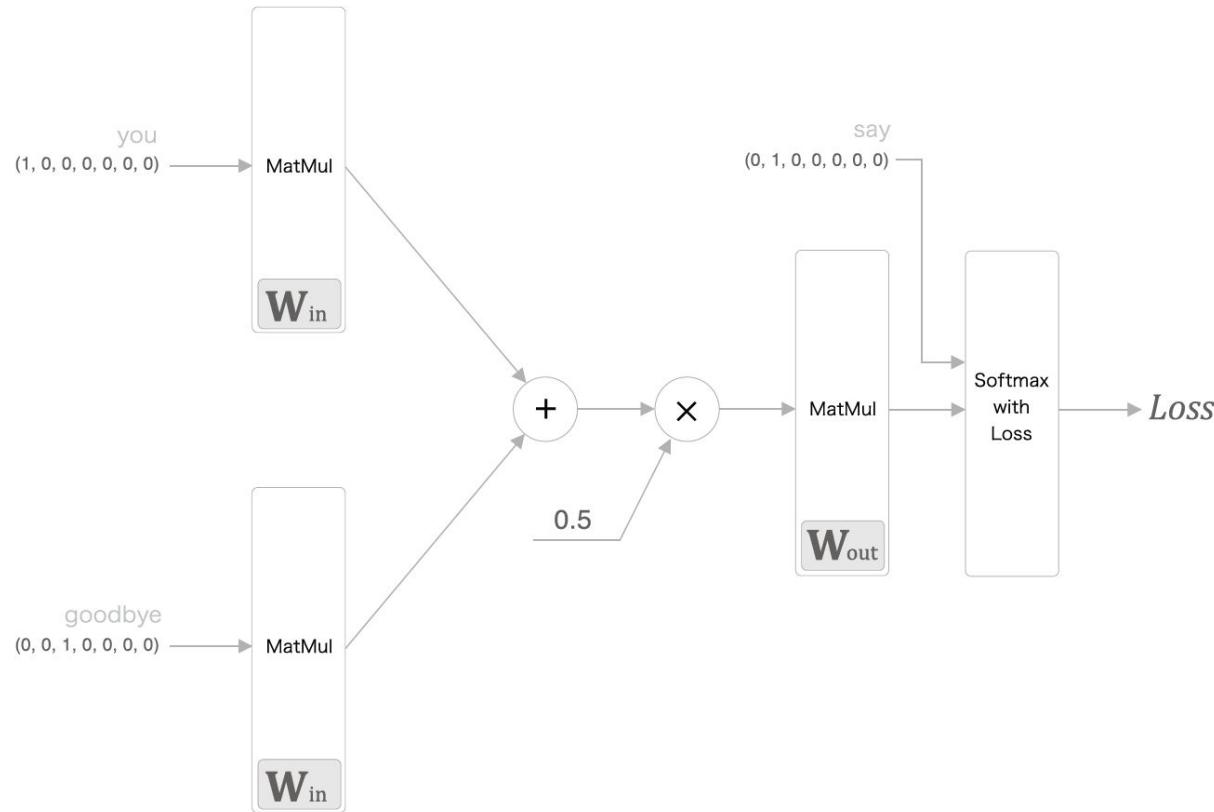
단순한 word2vec

CBOW (Continuous Bag-Of-Words)



단순한 word2vec

CBOW (Continuous Bag-Of-Words)



단순한 word2vec

CBOW (Continuous Bag-Of-Words)

```
# 샘플 맥락 데이터
c0 = np.array([[1, 0, 0, 0, 0, 0, 0]])
c1 = np.array([[0, 0, 1, 0, 0, 0, 0]])

# 가중치
W_in = np.random.randn(7,3)
W_out = np.random.randn(3,7)

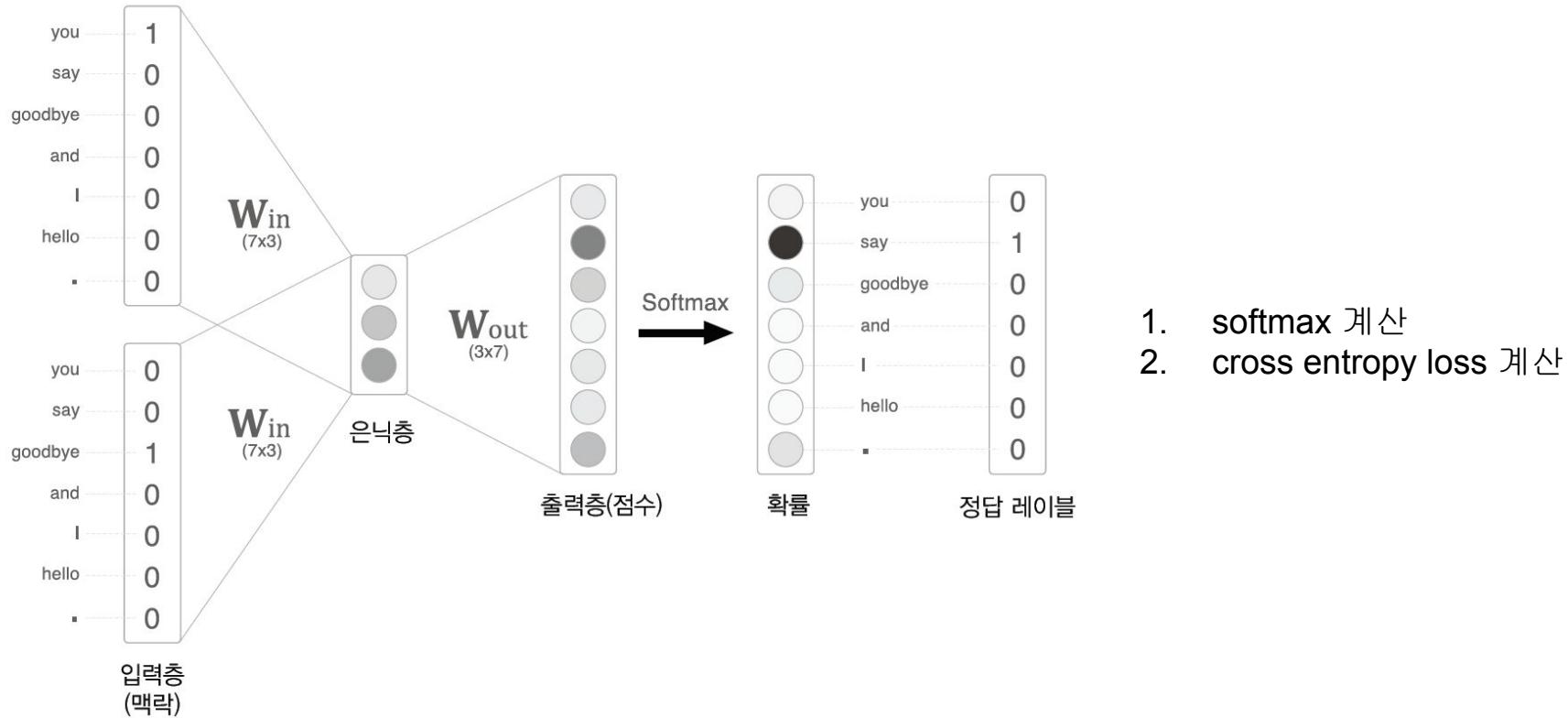
# 계층 생성
in_layer0 = MatMul(W_in)
in_layer1 = MatMul(W_in)
out_layer = MatMul(W_out)
```

```
# 순전파
h0 = in_layer0.forward(c0)
h1 = in_layer1.forward(c1)
h = 0.5 * (h0 + h1)
s = out_layer.forward(h)

print(s)
0.69066274 1.04000766 0.09605814 -0.23178727 0.20771755 0.34685519
0.32418914]
```

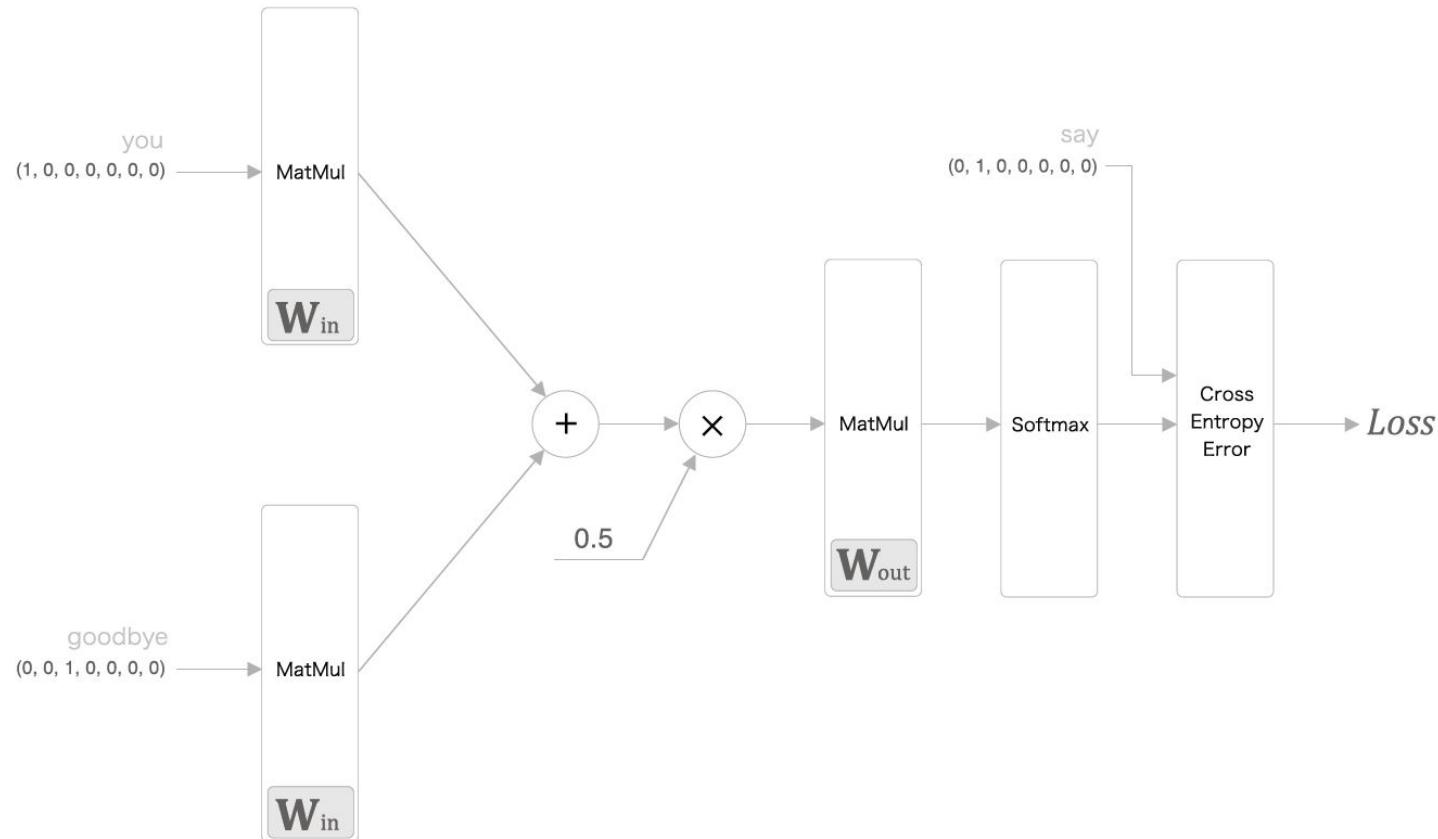
단순한 word2vec

CBOW 모델의 학습



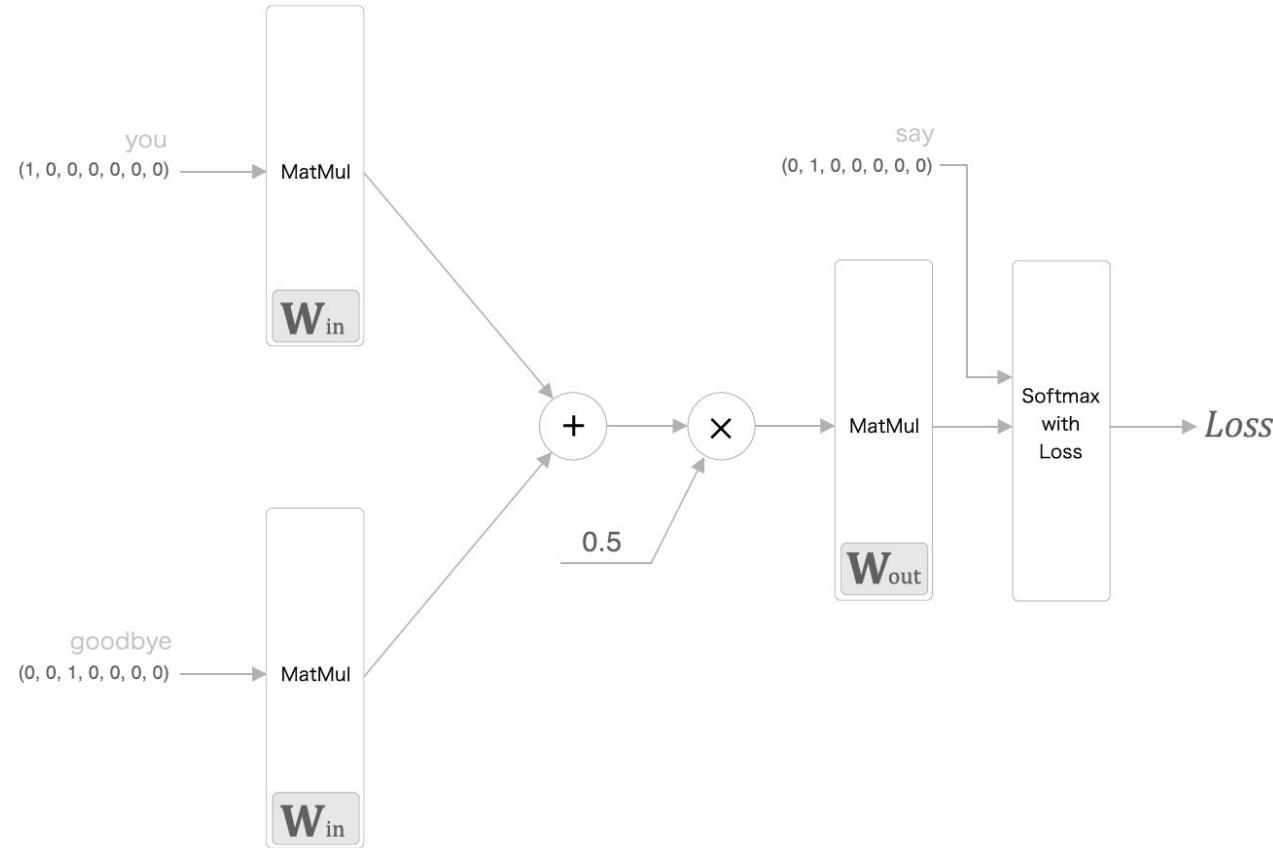
단순한 word2vec

CBOW 모델의 학습



단순한 word2vec

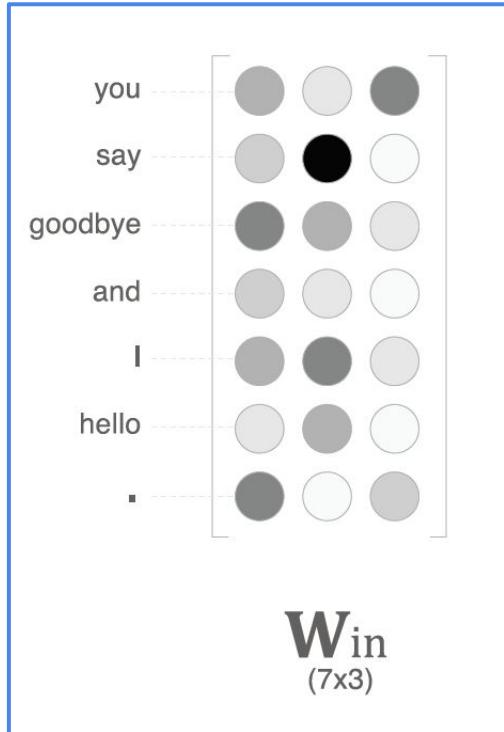
CBOW 모델의 학습



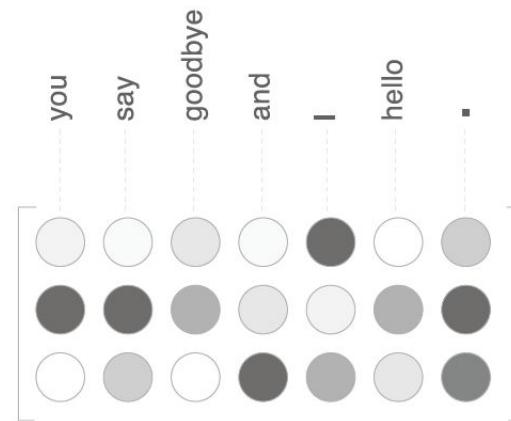
단순한 word2vec

word2vec의 가중치와 분산 표현

그림 3-15 각 단어의 분산 표현은 입력 측과 출력 측 모두의 가중치에서 확인할 수 있다.



W_{in}
 (7×3)



W_{out}
 (3×7)

word2vec을 위한 학습 데이터

맥락과 타깃

그림 3-16 말뭉치에서 맥락과 타깃을 만드는 예

말뭉치	맥락(contexts)	타깃
you say goodbye and I say hello .	you, goodbye	say
you say goodbye and I say hello .	say, and	goodbye
you say goodbye and I say hello .	goodbye, I	and
you say goodbye and I say hello .	and, say	I
you say goodbye and I say hello .	I, hello	say
you say goodbye and I say hello .	say, .	hello

word2vec을 위한 학습 데이터

맥락과 타깃

그림 3-17 단어 ID의 배열인 corpus로부터 맥락과 타깃을 작성하는 예(맥락의 원도우 크기는 1)

말뭉치	맥락(contexts)	타깃
[0 1 2 3 4 1 5 6]	[[0 2]	[1
	[1 3]	2
→	[2 4]	3
	[3 1]	4
	[4 5]	1
	[1 6]]	5]

형상: (8,)

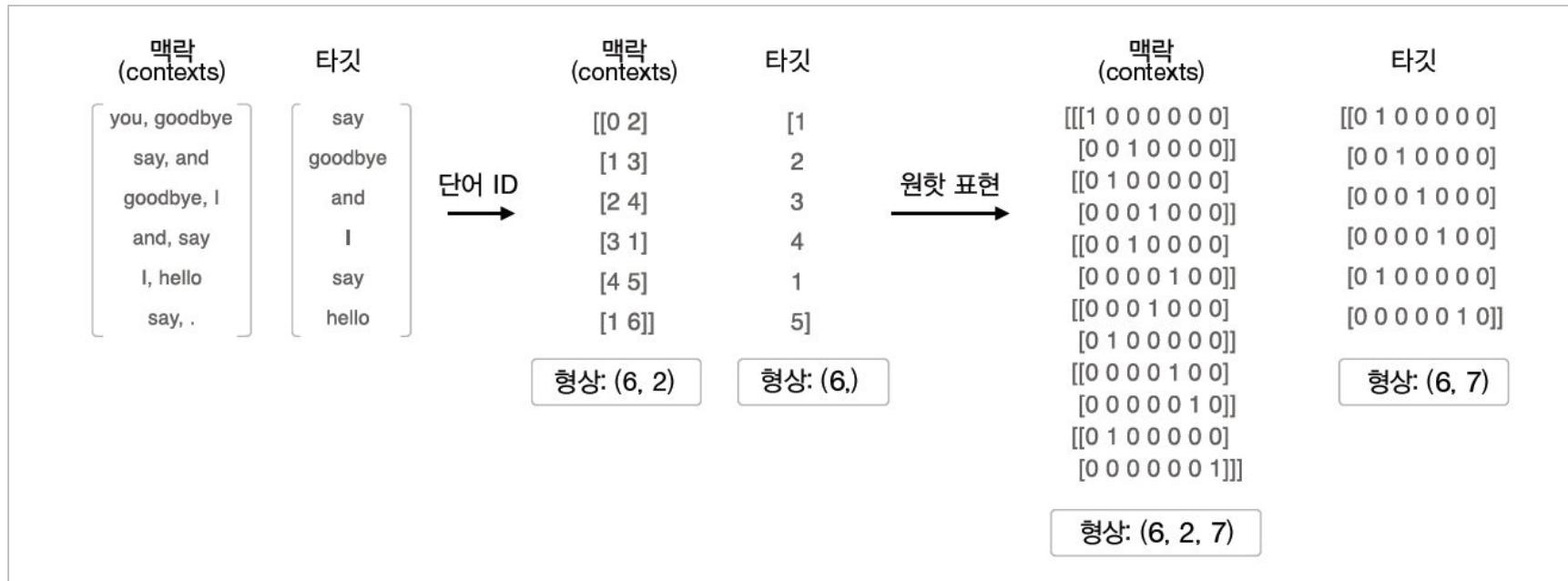
형상: (6, 2)

형상: (6,)

word2vec을 위한 학습 데이터

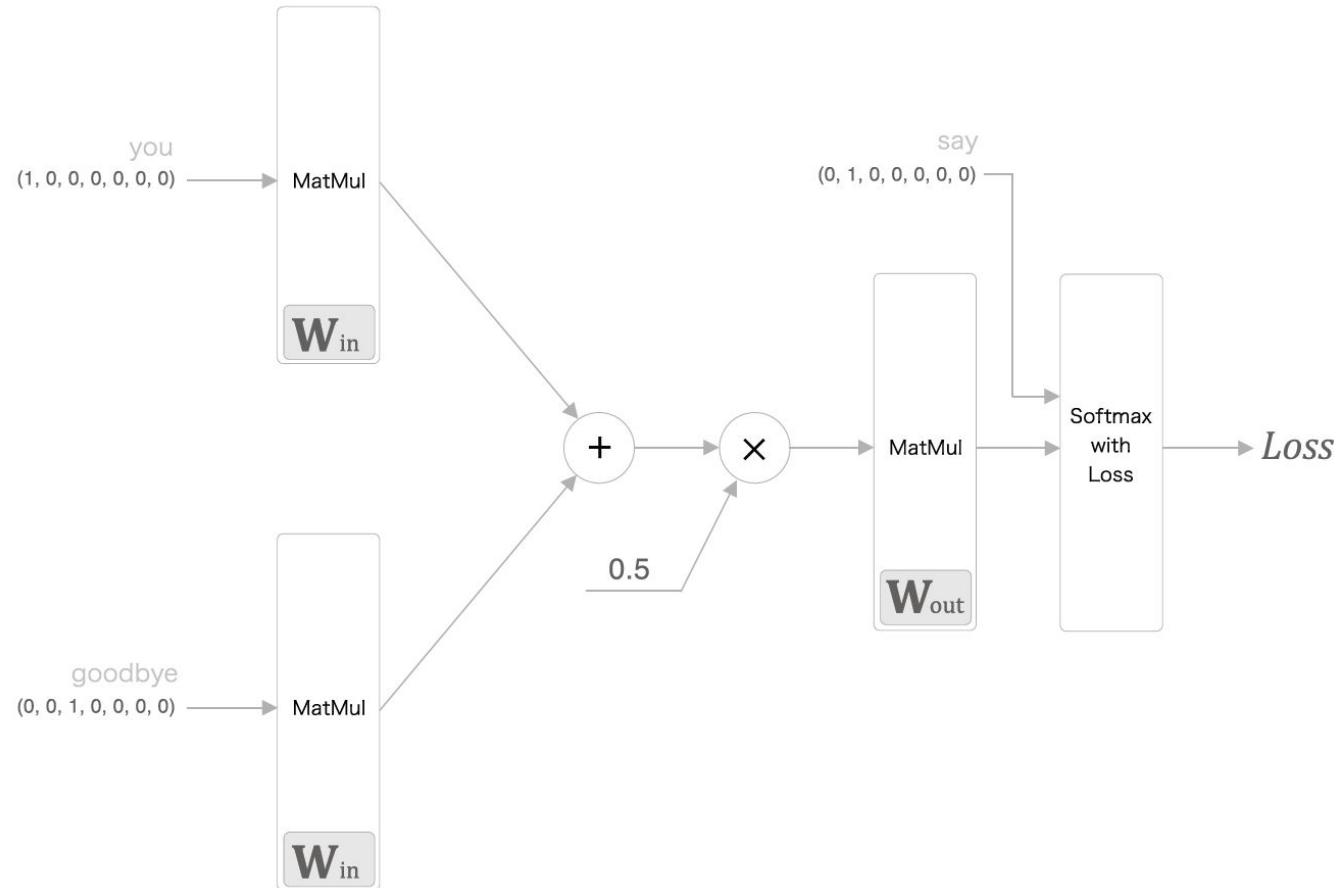
맥락과 타깃

그림 3-18 ‘맥락’과 ‘타깃’을 원핫 표현으로 변환하는 예



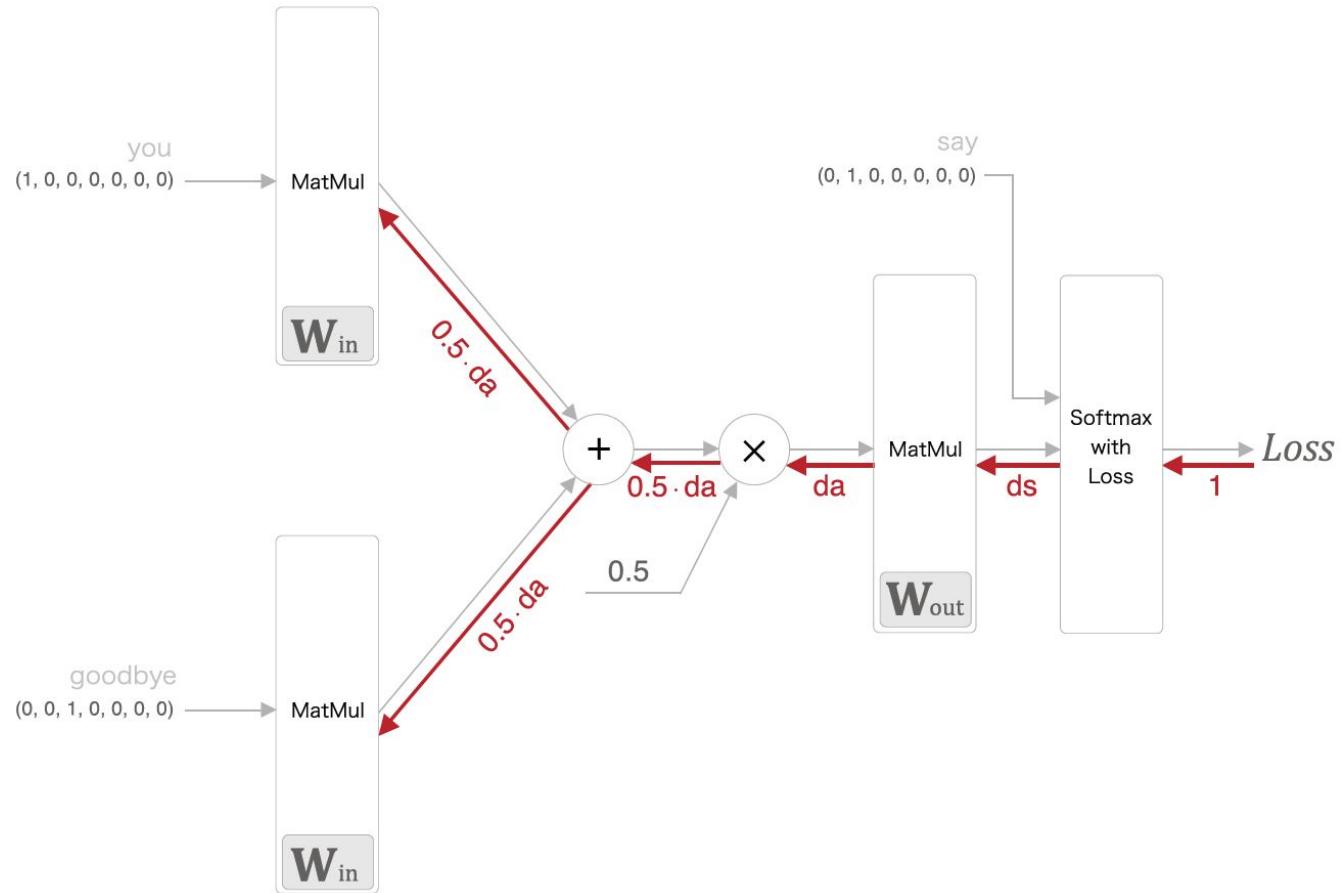
word2vec을 위한 학습 데이터

CBOW 모델 구현



word2vec을 위한 학습 데이터

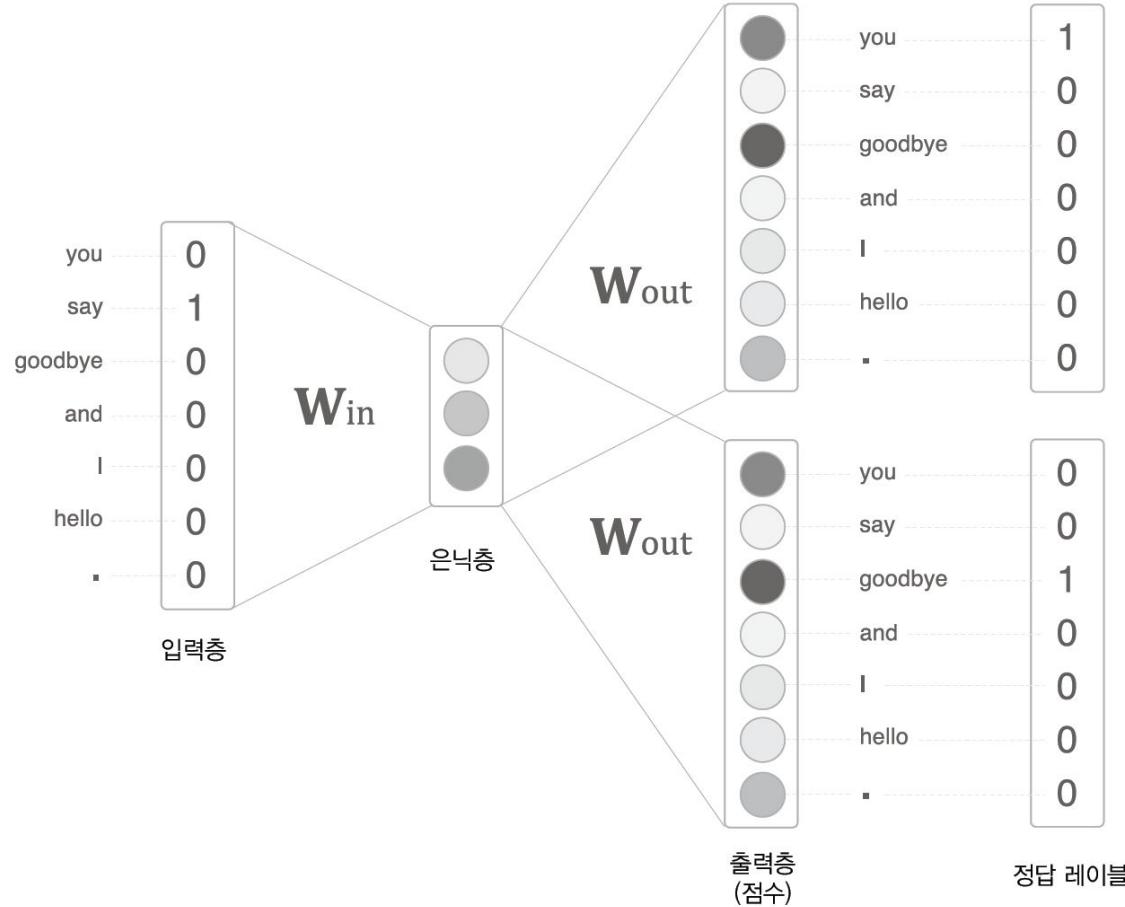
CBOW 모델 구현



word2vec

skip-gram 모델

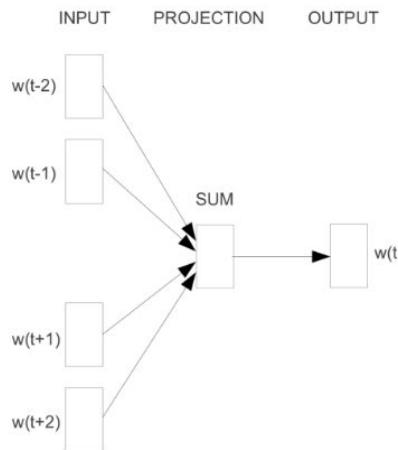
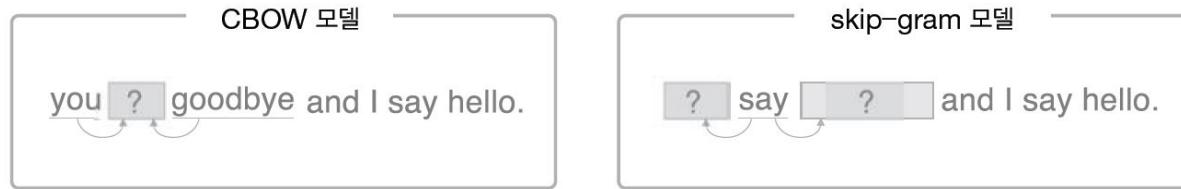
그림 3-24 skip-gram 모델의 신경망 구성 예



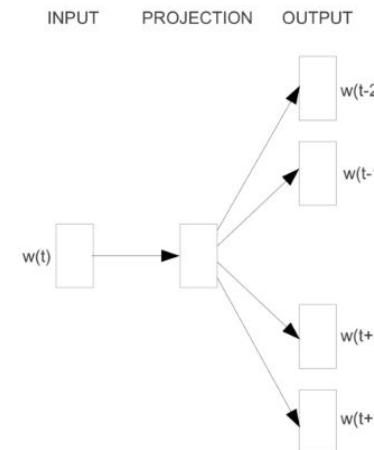
word2vec

skip-gram 모델

그림 3-23 CBOW 모델과 skip-gram 모델이 다루는 문제



CBOW



Skip-gram

word2vec

지금까지 배운 내용

- 추론 기반 방법은 추측하는 것이 목적, 그 과정에서 단어의 분산 표현을 얻음
- word2vec은 추론 기반 기법이며, 단순한 2층 신경망
- word2vec은 CBOW 모델, skip-gram 모델로 나뉨
- CBOW 모델은 맥락으로 부터 단어를 추측
- skip-gram 모델은 단어로부터 맥락을 추측
- word2vec은 데이터가 추가되더라도 가중치 학습을 추가로 진행하면 되기 때문에 효율적임

실습

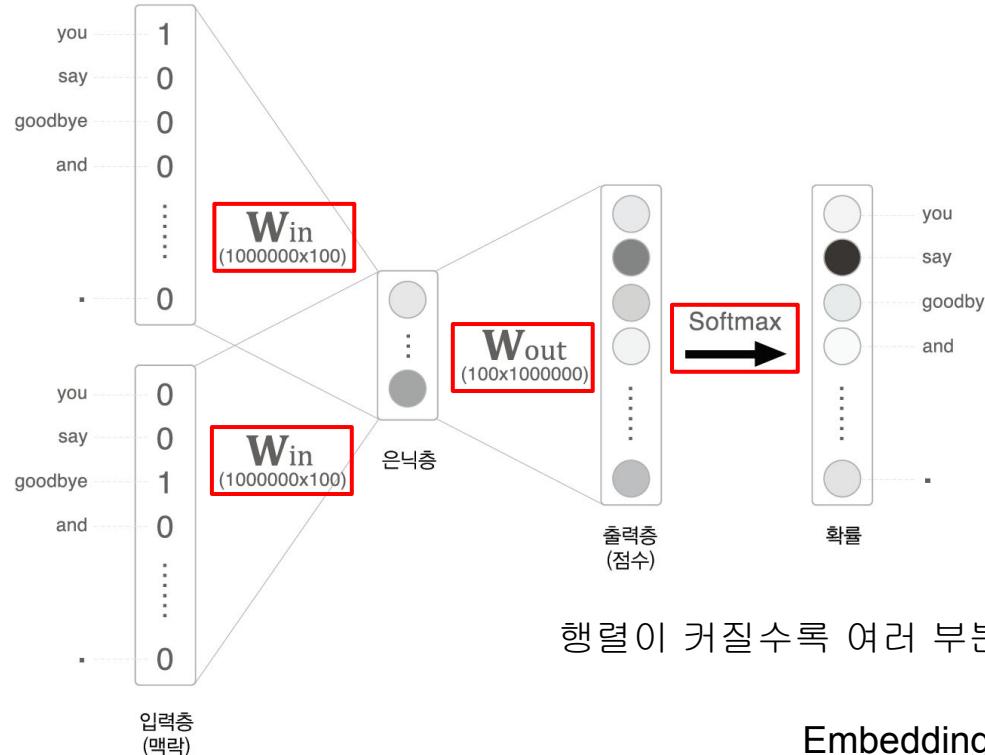
중간점검 실습

- 아무 TEXT 데이터나 복사해와서 분석해보자
- Text preprocessing 구현하기 (가능하면 본인 아이디어 추가해보기)
- 동시발생 행렬 계산해보기
- CBOW 모델 구현을 위해 context, target 분리해보기
- model 학습까지

word2vec 속도 개선

CBOW 모델 다시 보기

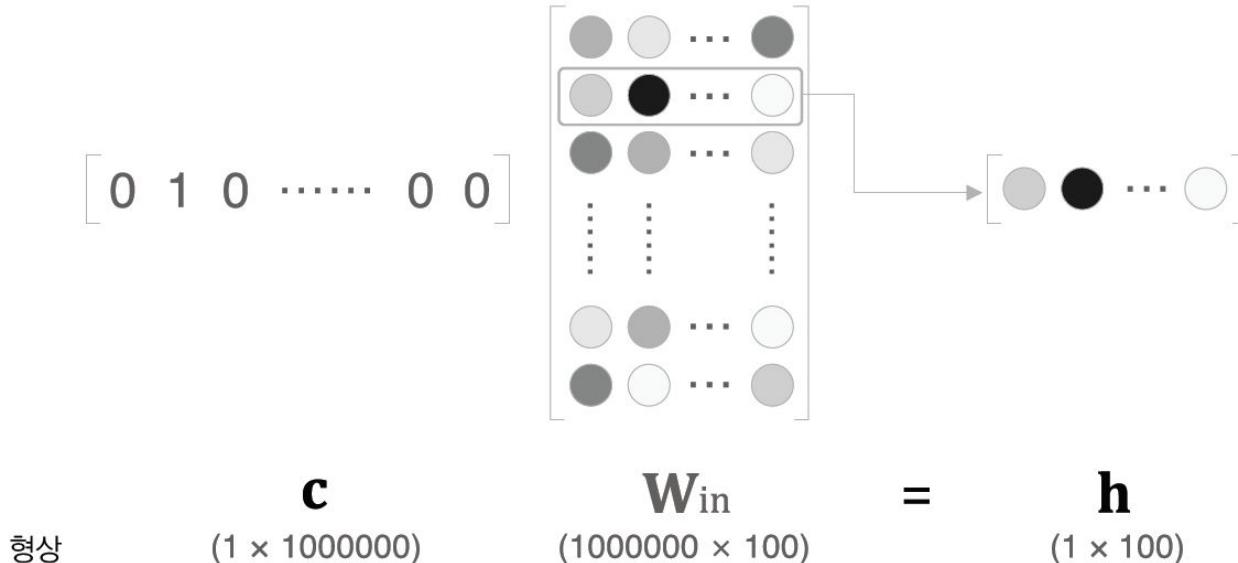
그림 4-2 어휘가 100만 개일 때를 가정한 CBOW 모델



word2vec 속도 개선

Embedding 계층

그림 4-3 맥락(원핫 표현)과 MatMul 계층의 가중치를 곱한다.



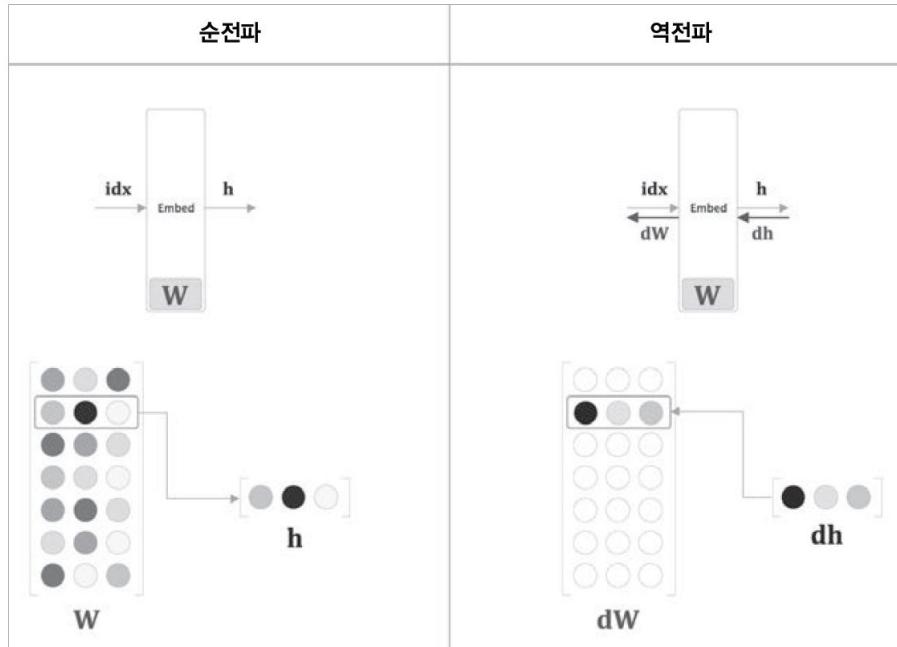
입력층에 가중치가 곱해진 결과를 보면, 특정 행을 추출한 것일 뿐임

word2vec 속도 개선

Embedding 계층

행렬 계산 없이 단어 ID에 해당하는 행을 추출하는 계층 => “Embedding 계층”

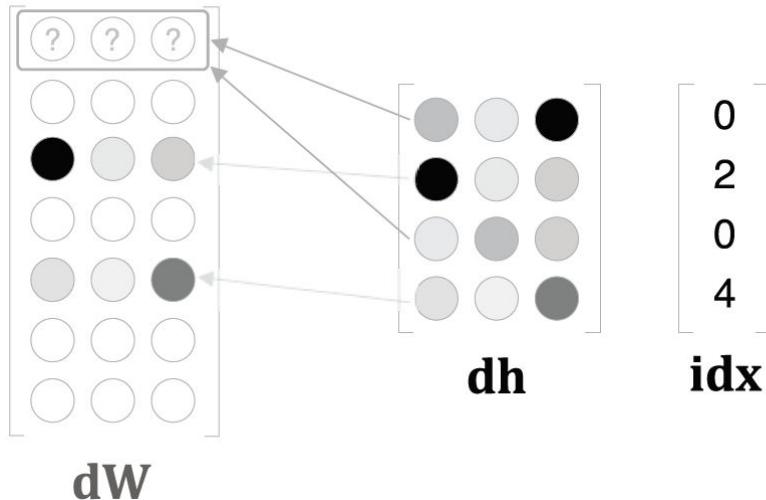
그림 4-4 Embedding 계층의 forward와 backward 처리(Embedding 계층은 Embed로 표기)



word2vec 속도 개선

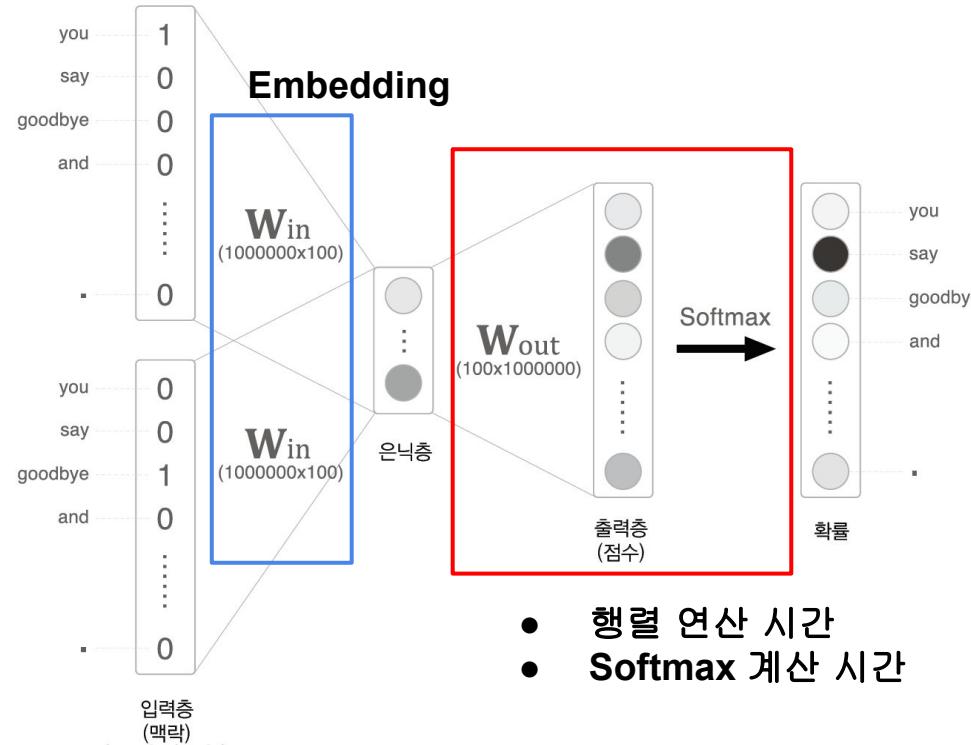
Embedding 계층

그림 4-5 idx 배열의 원소 중 값(행 번호)이 같은 원소가 있다면, $d\mathbf{h}$ 를 해당 행에 할당할 때 문제가 생긴다.



word2vec 속도 개선

은닉층 이후 계산의 문제점

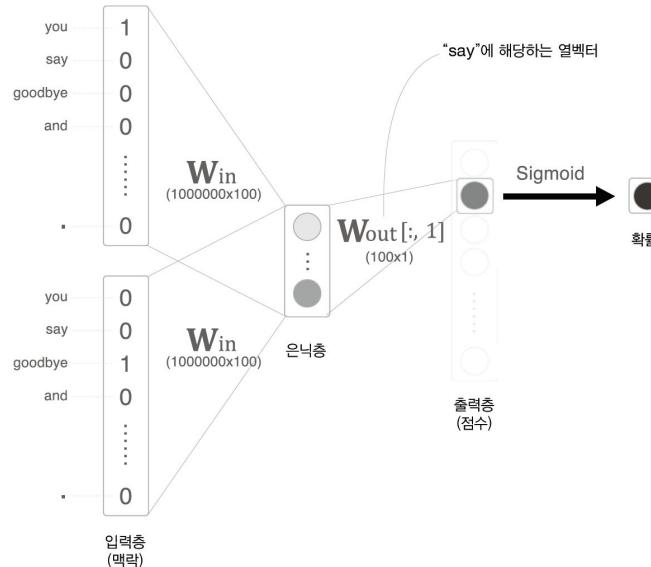


word2vec 속도 개선

은닉층 이후 계산의 문제점

지금까지는 말뭉치에 존재하는 단어 중 가장 확률이 높은 단어를 고르는 ‘다중 분류 문제’를 풀고 있었음
단어가 너무 많아지는 경우 계산량이 늘어나는 단점이 있음

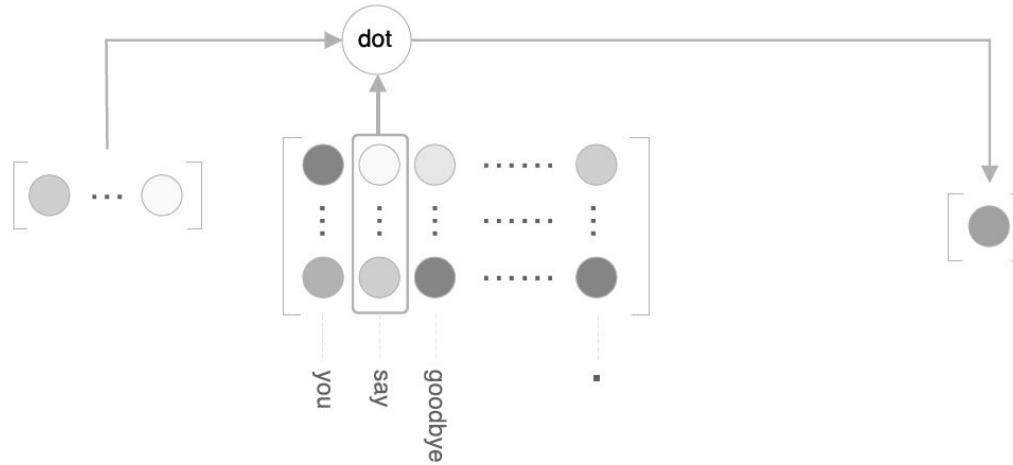
각 단어의 **출현확률**을 구하는 ‘**이진 분류**’ 문제를 고려하면 계산량을 줄일 수 있음



word2vec 속도 개선

은닉층 이후 계산의 문제점

그림 4-8 “say”에 해당하는 열벡터와 은닉층 뉴런의 내적을 계산한다(‘dot’ 노드가 내적을 계산함).



h

형상 : (1×100)

W_{out}

(100×1000000)

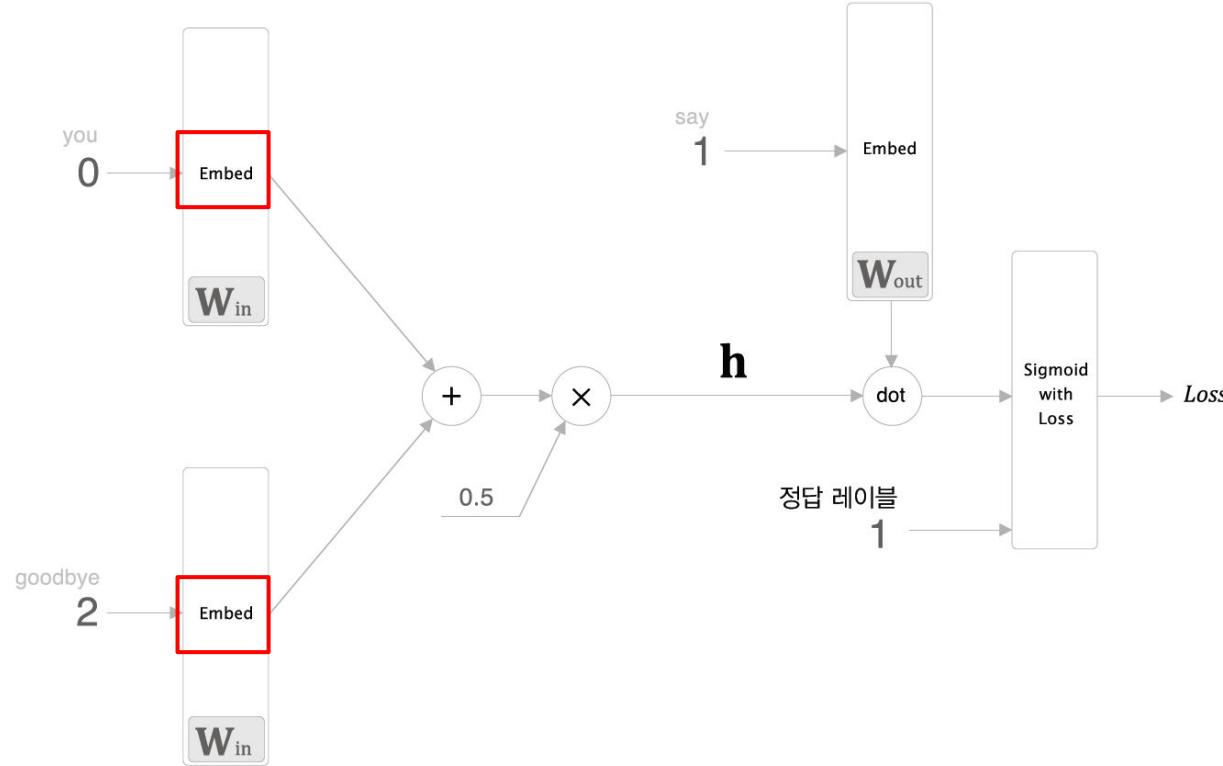
s

(1×1)

word2vec 속도 개선

이진 분류를 수행하는 word2vec(CBOW 모델) 전체 그림

그림 4-12 이진 분류를 수행하는 word2vec(CBOW 모델)의 전체 그림



word2vec 속도 개선

이진 분류를 수행하는 word2vec(CBOW 모델) 전체 그림

그림 4-14 Embedding Dot 계층의 각 변수의 구체적인 값

```
embed = Embedding(W)
target_W = embed.forward(idx)

out = np.sum(target_W * h, axis=1)
```

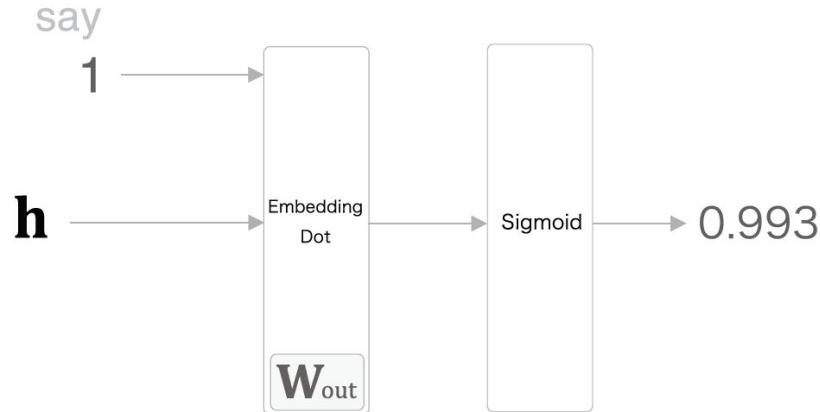
W	idx	target_W	h	target_W * h	out
[[0 1 2]	[0 3 1]	[[0 1 2]	[[0 1 2]	[[0 1 4]	[5 122 86]
[3 4 5]		[9 10 11]	[3 4 5]	[27 40 55]	
[6 7 8]		[3 4 5]]	[6 7 8]]	[18 28 40]]	
[9 10 11]					
[12 13 14]					
[15 16 17]					
[18 19 20]]					

word2vec 속도 개선

네거티브 샘플링

이진 분류 문제로 바꾸더라도 여전히 정답 타깃에 대한 정보만 학습하게 되는 문제가 남아 있음

그림 4-15 CBOW 모델의 은닉층 이후의 처리 예: 맥락은 “you”와 “goodbye”이고, 타깃이 “say”일 확률은 0.993(99.3%)이다.

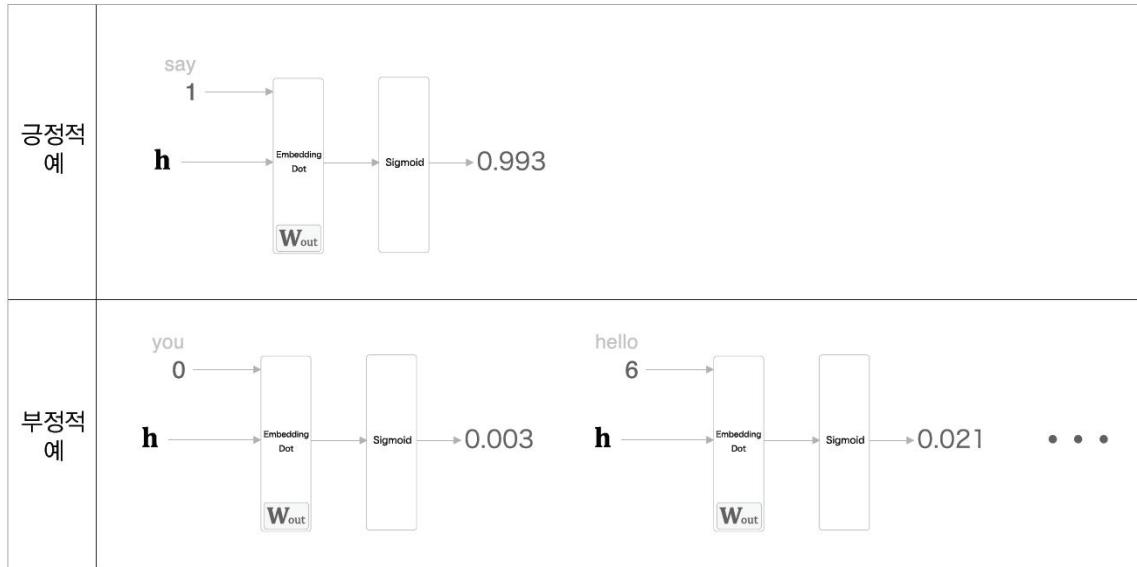


정답이 아닌 단어를 임의로 추출하여 학습에 반영하면 부정적인 예에 대해서도 학습이 가능해짐

word2vec 속도 개선

네거티브 샘플링

그림 4-16 긍정적 예(정답)를 “say”라고 가정하면, “say”를 입력했을 때의 Sigmoid 계층 출력은 1에 가깝고, “say” 이외의 단어를 입력했을 때의 출력은 0에 가까워야 한다. 이런 결과를 내어주는 가중치가 필요하다.



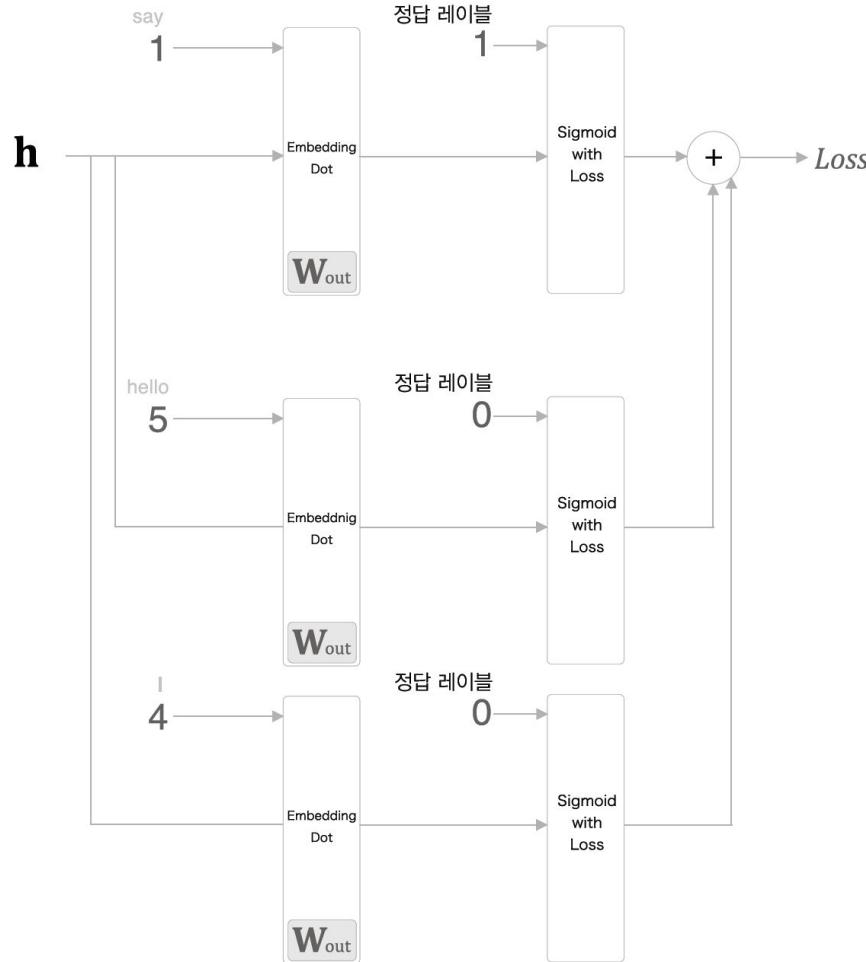
“Negative Sampling”

word2vec 속도 개선

네거티브 샘플링

정답이 아닌 모든 단어를 학습시키는 것은
비효율적이므로 **임의로 샘플링**하는 것이 핵심

그림 4-17 네거티브 샘플링의 예(은닉층 이후의 처리에 주목하여 그린 계산 그래프)

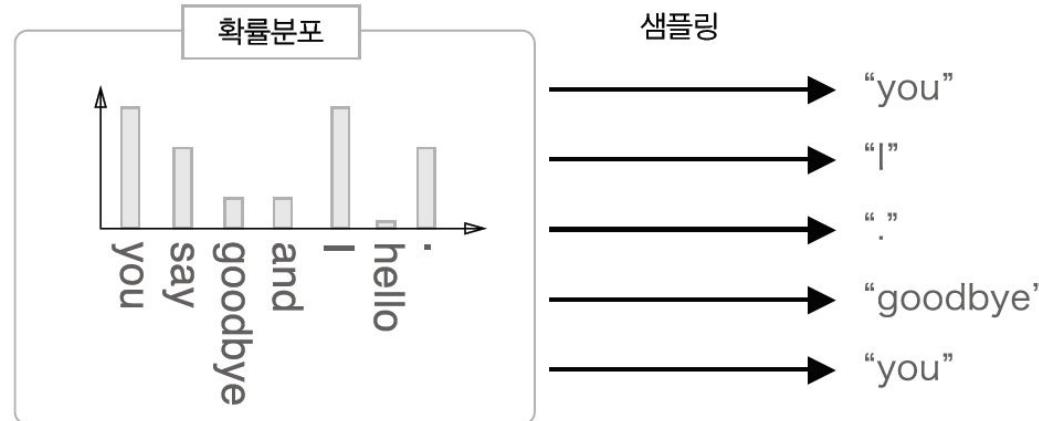


word2vec 속도 개선

네거티브 샘플링의 샘플링 기법

무작위 샘플링보다 말뭉치 통계 기반으로 등장 빈도에 따라 샘플링 확률을 높이는 방식으로 샘플링 했을 때 효율이 더 좋았다고 함

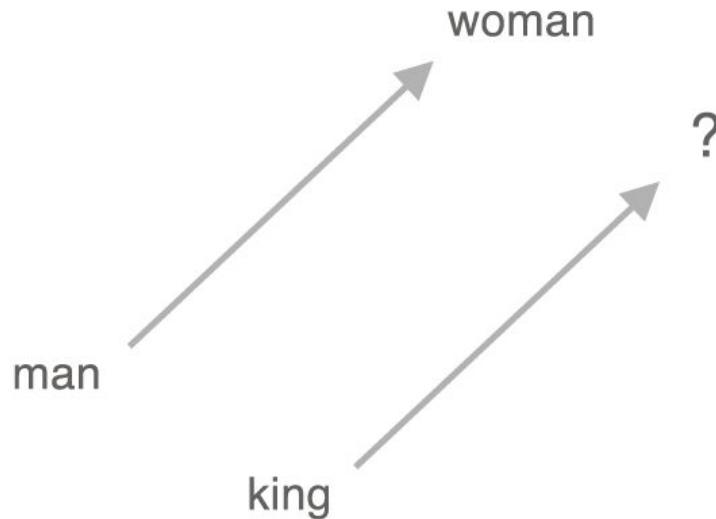
그림 4-18 확률분포에 따라 샘플링을 여러 번 수행한다.



word2vec 속도 개선

CBOW 모델 학습

그림 4-20 “man : woman = king : ?” 유추 문제 풀기(단어 벡터 공간에서 각 단어의 관계성)



학습이 잘 되었다면 ‘queen’을 유추할 수 있다!

word2vec 의 다른 주제

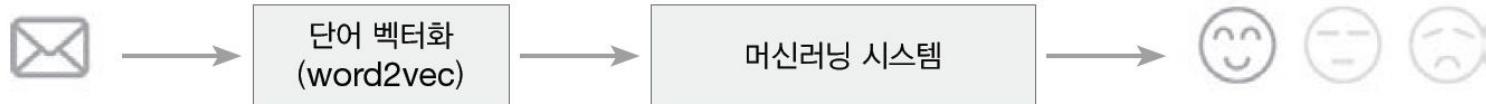
word2vec을 사용한 예시들

- 방대한 데이터로 학습한 word2vec 단어 분산 표현을 가져와서 다른 분야에 활용하는 ‘전이 학습’
(텍스트 분류, 문서 클러스터링, 품사 태그 달기, 감정 분석 등..)
- 단어나 문장을 고정 길이의 벡터로 변환할 수 있음
=> 일반적인 Machine Learning 방법론으로 분석이 가능해짐

그림 4-21 단어의 분산 표현을 이용한 시스템의 처리 흐름



그림 4-22 메일 자동 분류 시스템의 예(감정 분석)



word2vec 의 다른 주제

단어 벡터 평가 방법

모델	차수	말뭉치 크기	의미(semantics)	구문(syntax)	종합
CBOW	300	16억	16.1	52.6	36.1
skip_gram	300	10억	61	61	61
CBOW	300	60억	63.6	67.4	65.7
skip_gram	300	60억	73.0	66.0	69.1
CBOW	1000	60억	57.3	68.9	63.7
skip_gram	1000	60억	66.1	65.1	65.6

king : queen = actor : **actress**

같이 단어의 의미를 유추

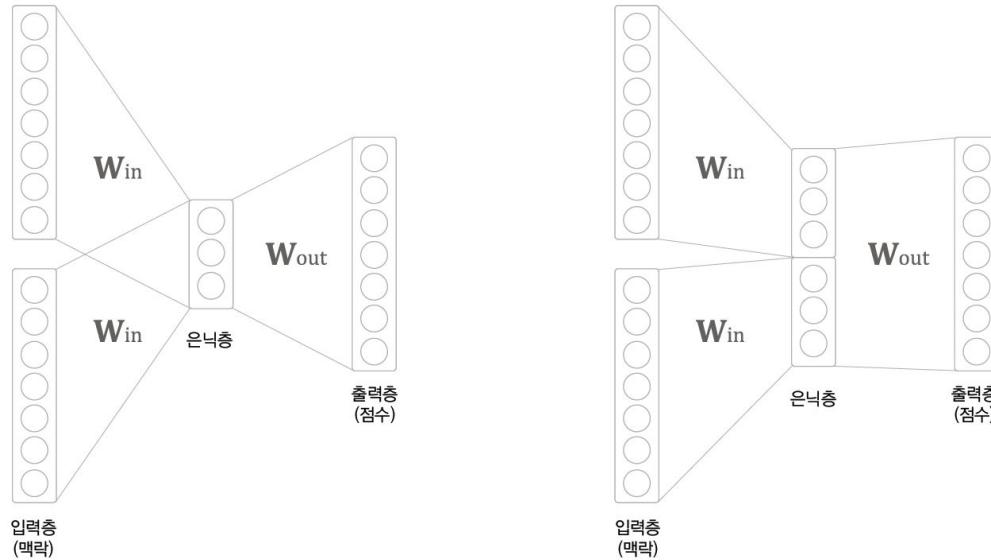
bad : worst = good : **best**

같은 단어의 형태 정보 유추

순환 신경망 (RNN)

확률과 언어 모델

그림 5-5 왼쪽이 일반 CBOW 모델을, 오른쪽은 은닉층에서 각 맥락의 단어 벡터를 '연결'한 모델을 나타낸다(입력층은 원핫 벡터).



입력된 맥락 데이터의 순서가 고려되지 못한다는 단점이 있음

순환 신경망 (RNN)

CBOW 같은 모델은 문장의 길이가 길어질 경우 window 사이즈를 아주 크게 늘리지 않으면 문장 전체의 정보를 고려할 수 없는 단점이 있음

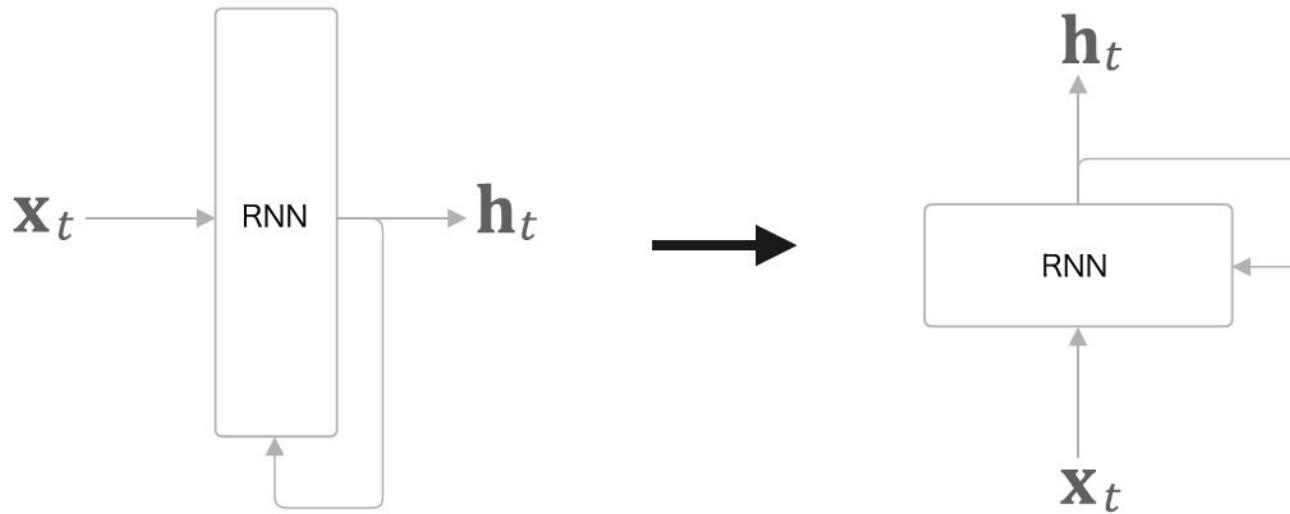
그림 5-4 “?”에 들어갈 단어는 무엇일까? (긴 맥락이 필요한 문제의 예)

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

예문의 ‘?’로 부터 ‘Tom’까지는 18개의 간격이 있음

맥락의 크기 (window size)를 키운다 하더라도, 순서가 무시되므로 한계가 있음

순환 신경망 (RNN)

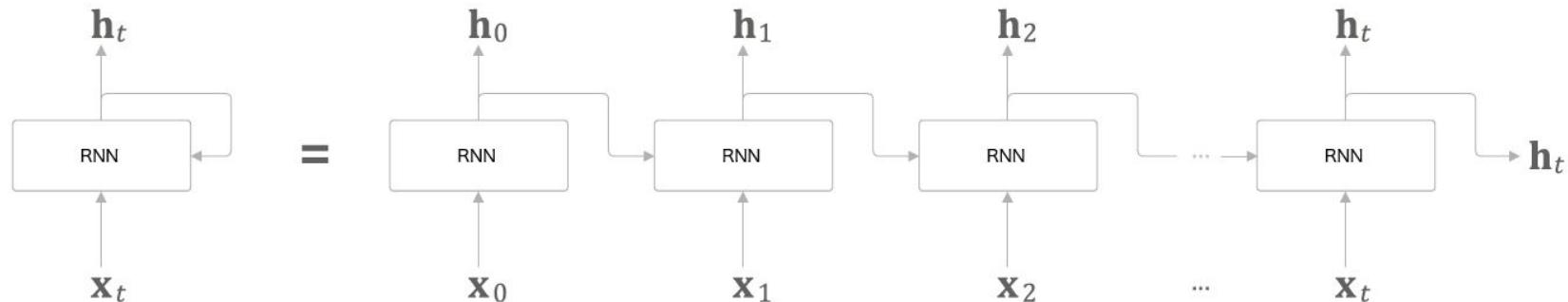


Recurrent Neural Network (순환 신경망)

순환 신경망 (RNN)

순환구조 펼쳐보기

그림 5-8 RNN 계층의 순환 구조 펼치기

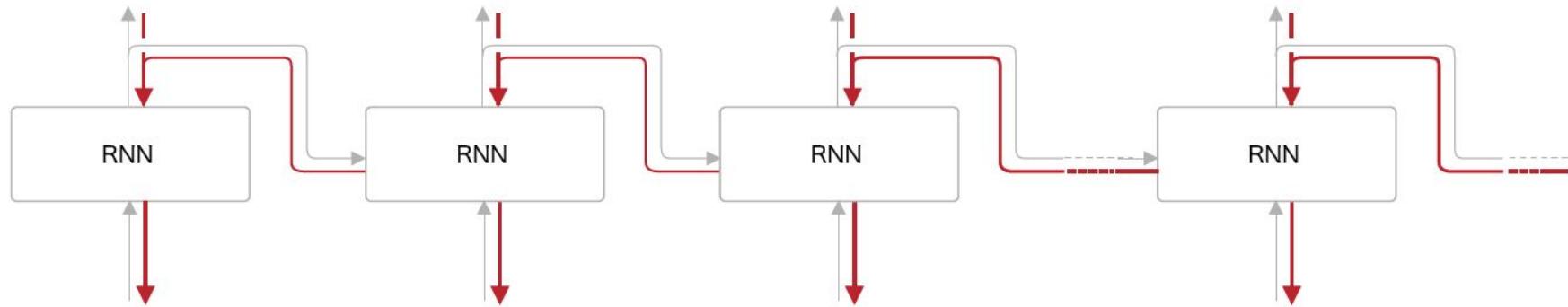


$$\mathbf{h}_t = \tanh(\mathbf{h}_{t-1} \mathbf{W}_h + \mathbf{x}_t \mathbf{W}_x + \mathbf{b})$$

순환 신경망 (RNN)

RNN의 오차역전파법

그림 5-10 순환 구조를 펼친 RNN 계층에서의 오차역전파법



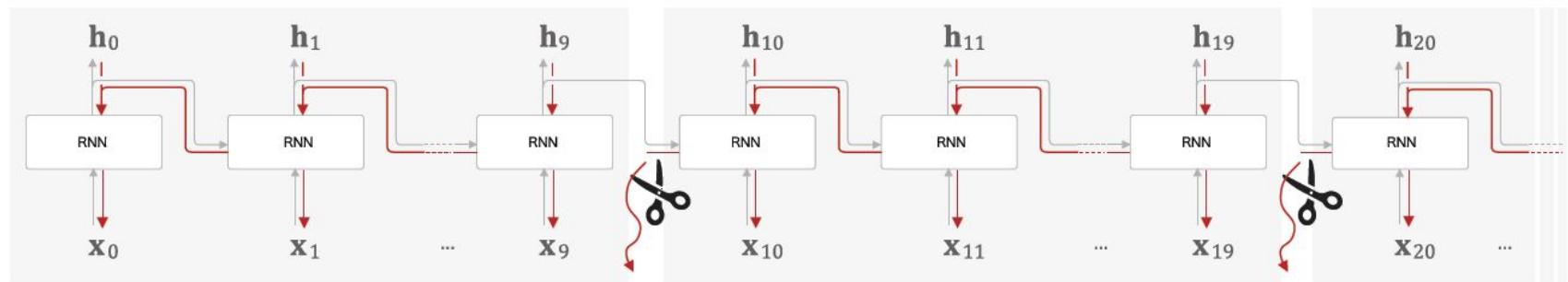
시간 방향으로 펼친 신경망의 오차역전파법
BPTT(BackPropagation Through Time)

순환 신경망 (RNN)

Truncated BPTT

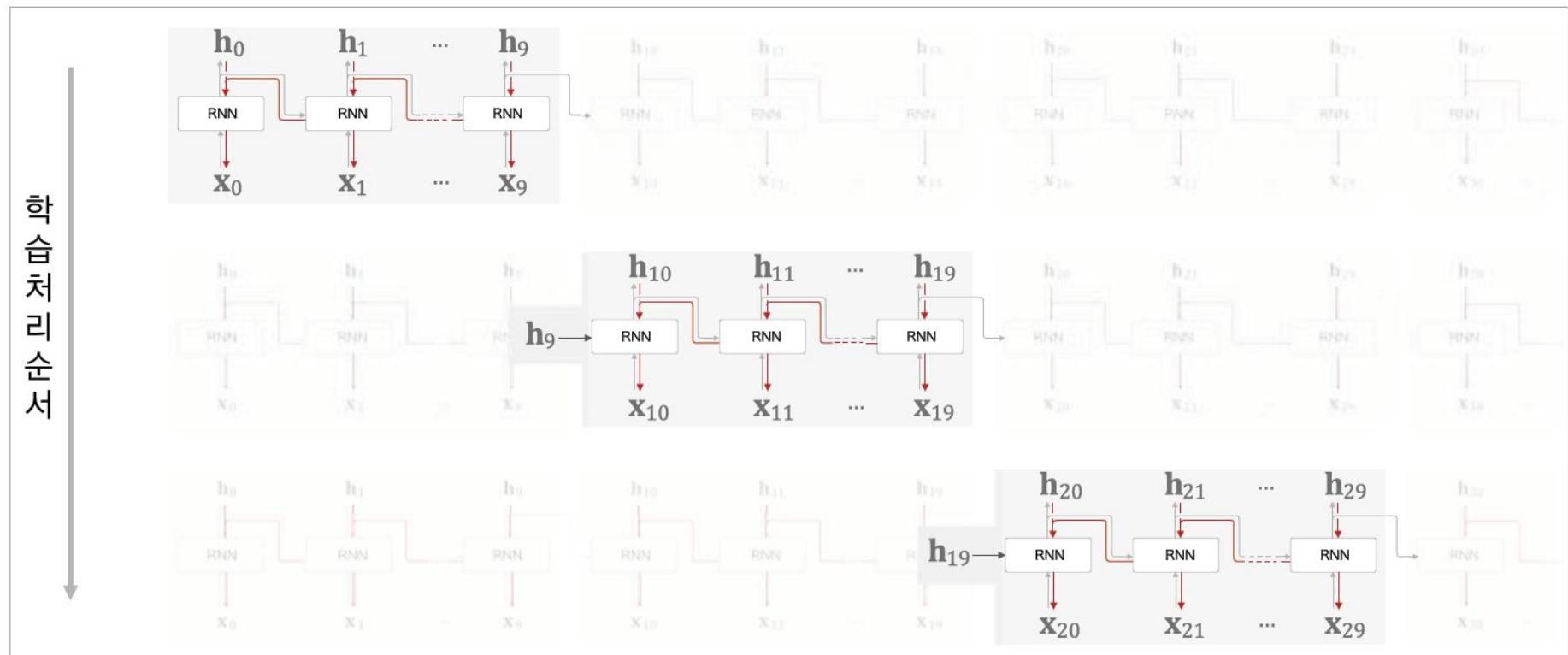
시계열 데이터가 너무 커지면 계산량이 증가하는 문제가 있음

그림 5-11 역전파의 연결을 적당한 지점에서 끊는다. (역전파가 연결되는 일련의 RNN 계층을 ‘블록’이라 하고, 배경을 회색으로 칠해 다른 블록과 구분함)



순환 신경망 (RNN)

Truncated BPTT



순환 신경망 (RNN)

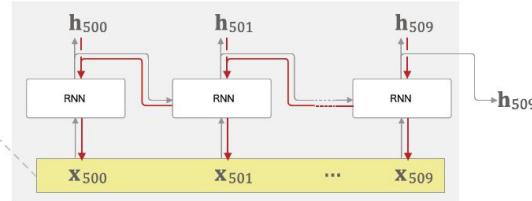
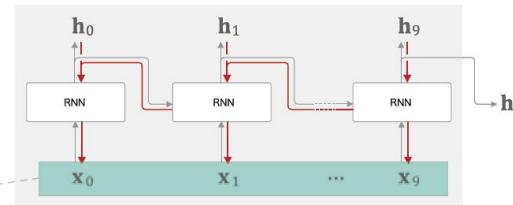
Truncated BPTT

학습
처리
순서

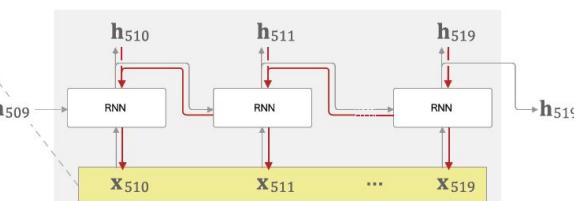
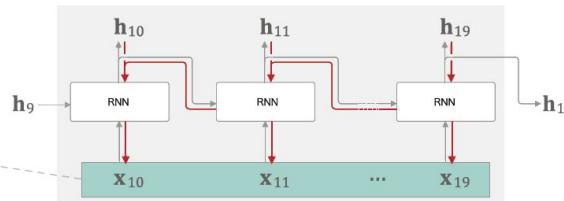
첫 번째 미니배치의 원소

$$\begin{pmatrix} x_0, x_1, \dots, x_9 \\ x_{500}, x_{501}, \dots, x_{509} \end{pmatrix}$$

두 번째 미니배치의 원소



$$\begin{pmatrix} x_{10}, x_{11}, \dots, x_{19} \\ x_{510}, x_{511}, \dots, x_{519} \end{pmatrix}$$



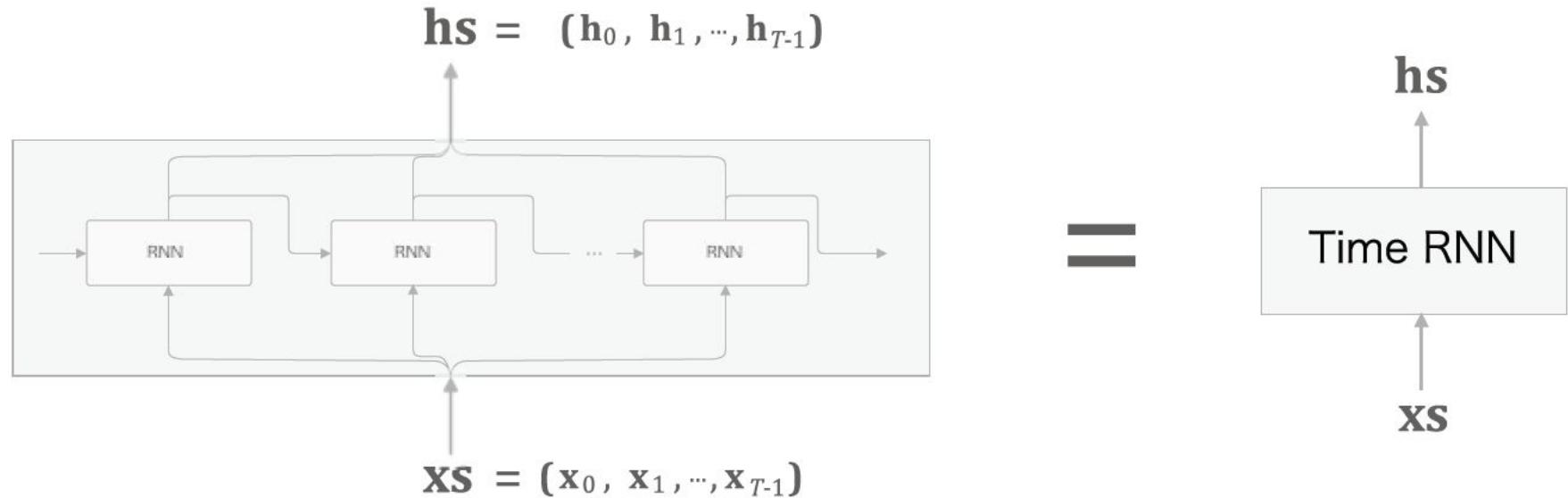
•

•

•

순환 신경망 (RNN)

RNN 구현



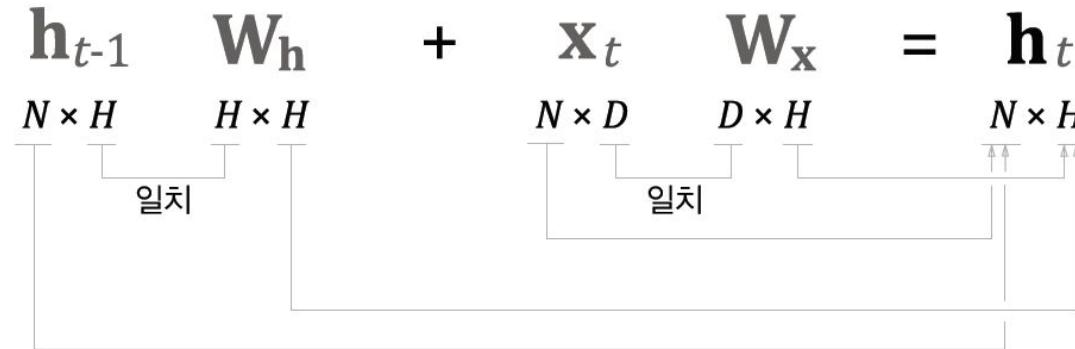
T 개의 단계를 한꺼번에 처리하는 계층을 'Time RNN 계층' 이라 하자

순환 신경망 (RNN)

RNN 구현

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

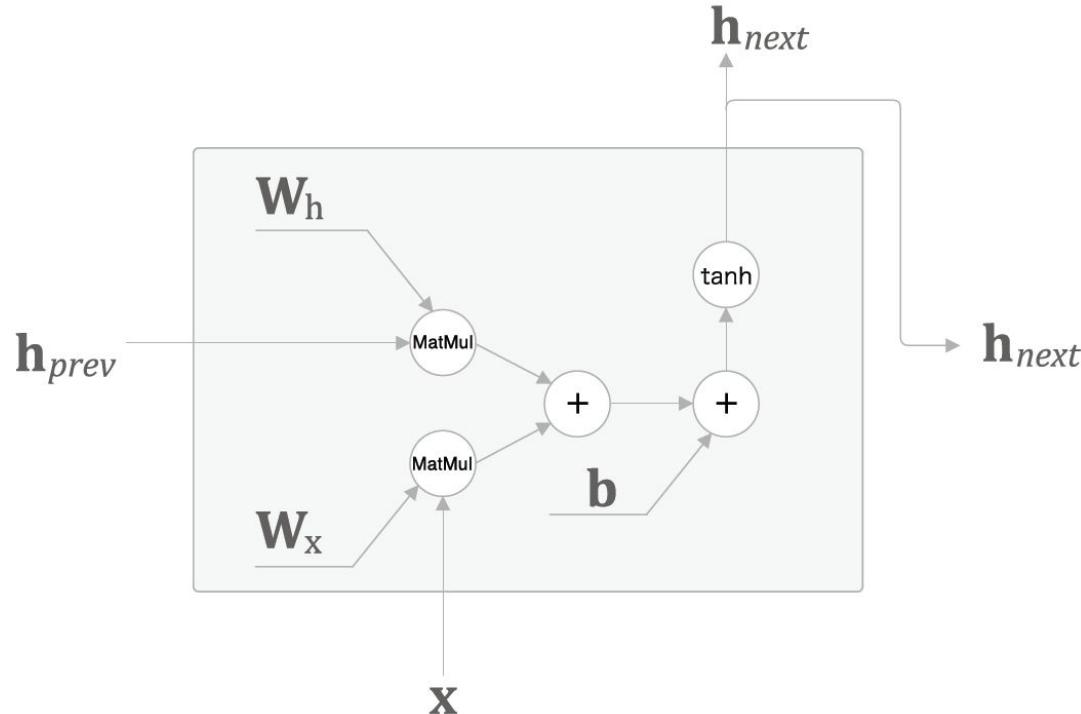
그림 5-18 형상 확인: 행렬 곱에서는 대응하는 차원의 원소 수를 일치시킨다(편향은 생략).



순환 신경망 (RNN)

RNN 구현

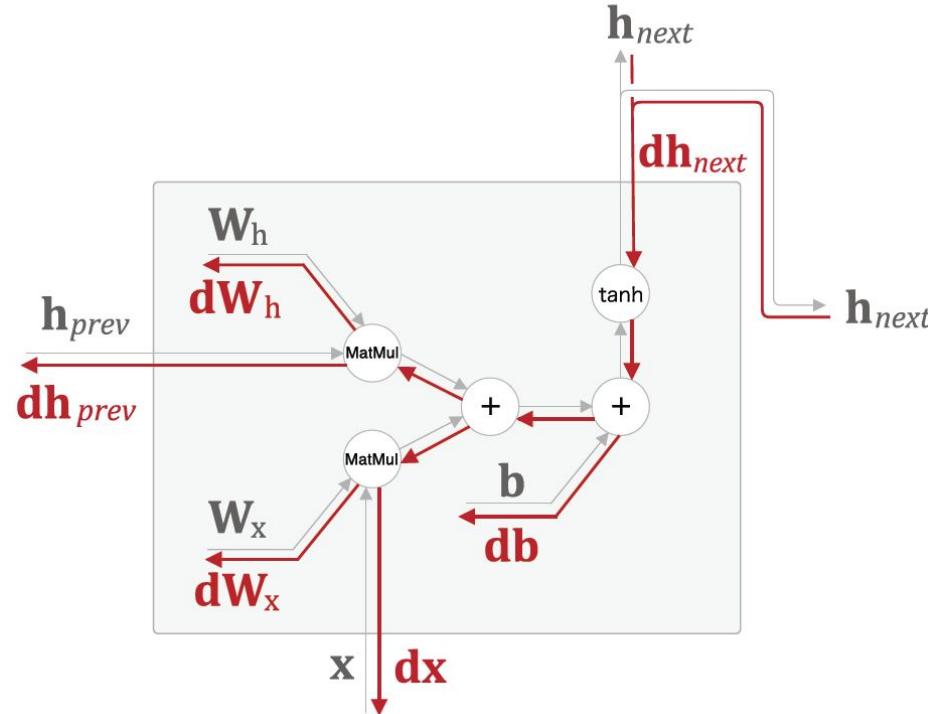
그림 5-19 RNN 계층의 계산 그래프(MatMul 노드는 행렬의 곱셈을 나타냄)



순환 신경망 (RNN)

RNN 구현

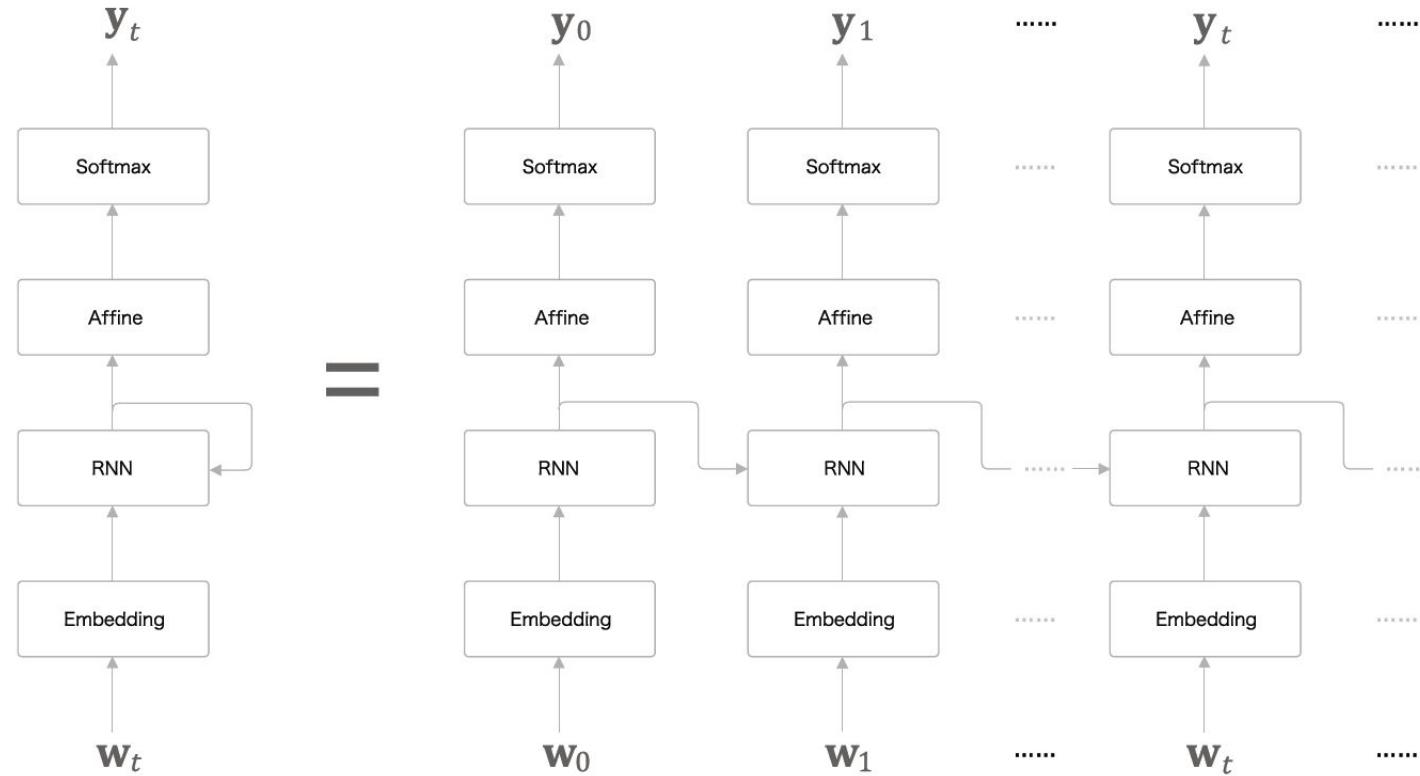
그림 5-20 RNN 계층의 계산 그래프(역전파 포함)



순환 신경망 (RNN)

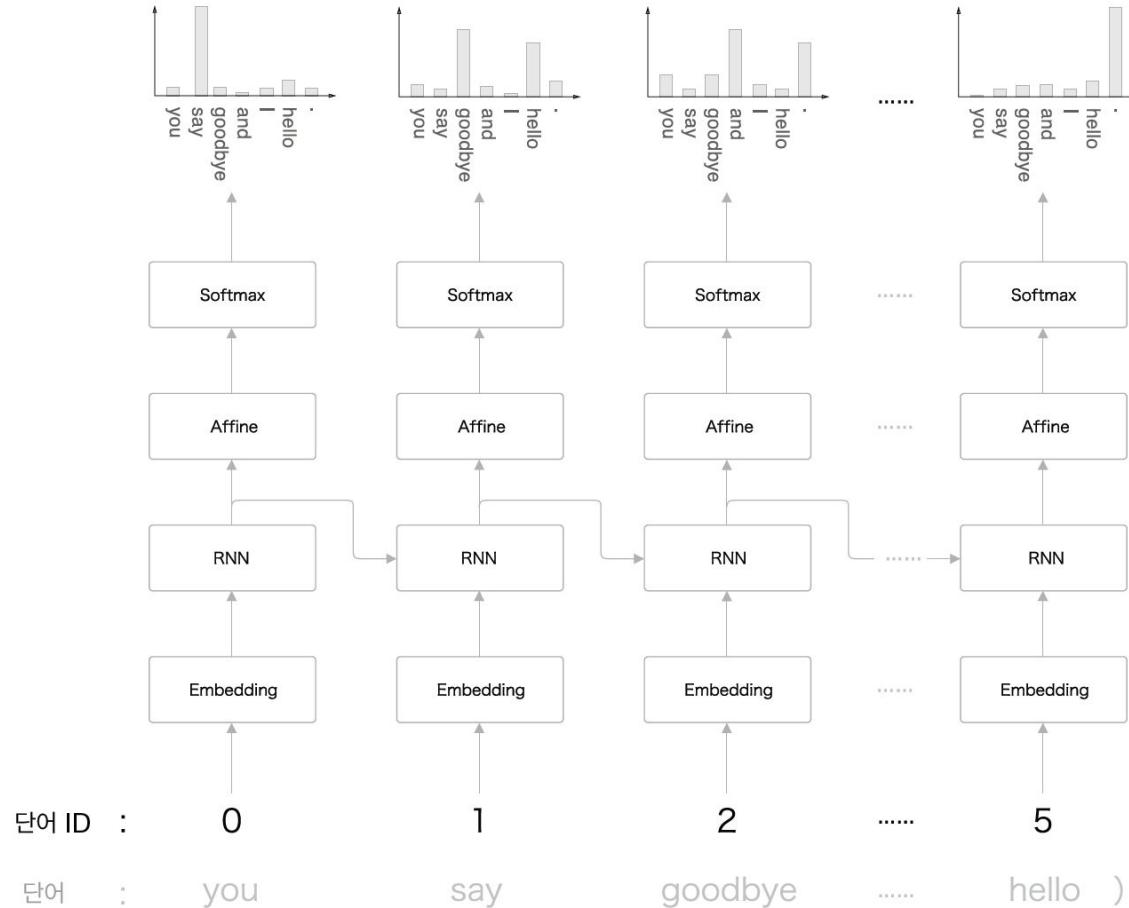
RNNLM (언어모델) 구현

그림 5-25 RNNLM의 신경망(왼쪽이 펼치기 전, 오른쪽은 펼친 후)



순환 신경망 (RNN)

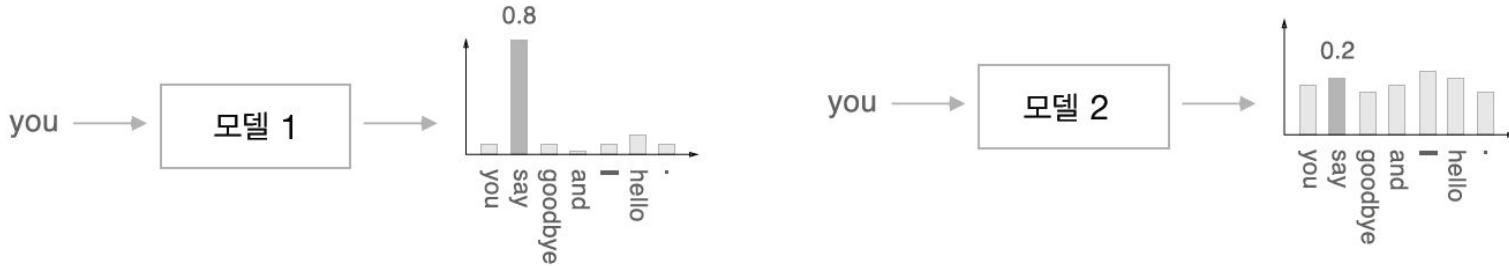
그림 5-26 샘플 말뭉치로 “you say goodbye and I say hello .”를 처리하는 RNNLM의 예



순환 신경망 (RNN)

언어 모델의 평가

그림 5-32 단어 “you”를 입력하여 다음에 출현할 단어의 확률분포를 출력하는 모델의 예



Perplexity

$$L = -\frac{1}{N} \sum_n \sum_k t_{nk} \log y_{nk}$$

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



확률의 역수로 구함

확률이 0.8 이면 => $1/0.8 = 1.25$
0.2 이면 => $1/0.2 = 5$

‘다음에 출현할 수 있는 단어의 수’ 정도로 해석할 수 있음
(작을수록 좋은 모델)

순환 신경망 (RNN)

지금까지 배운 내용

- RNN은 순환하는 경로가 있고, 이를 통해 내부의 ‘은닉 상태’를 기억함
- RNN 순환 경로를 펼쳐 다수의 RNN 계층이 연결된 것으로 해석 할 수 있음
- Truncated BPTT를 사용하여 긴 시계열을 학습함
- 언어 모델은 단어 시퀀스를 확률로 해석함

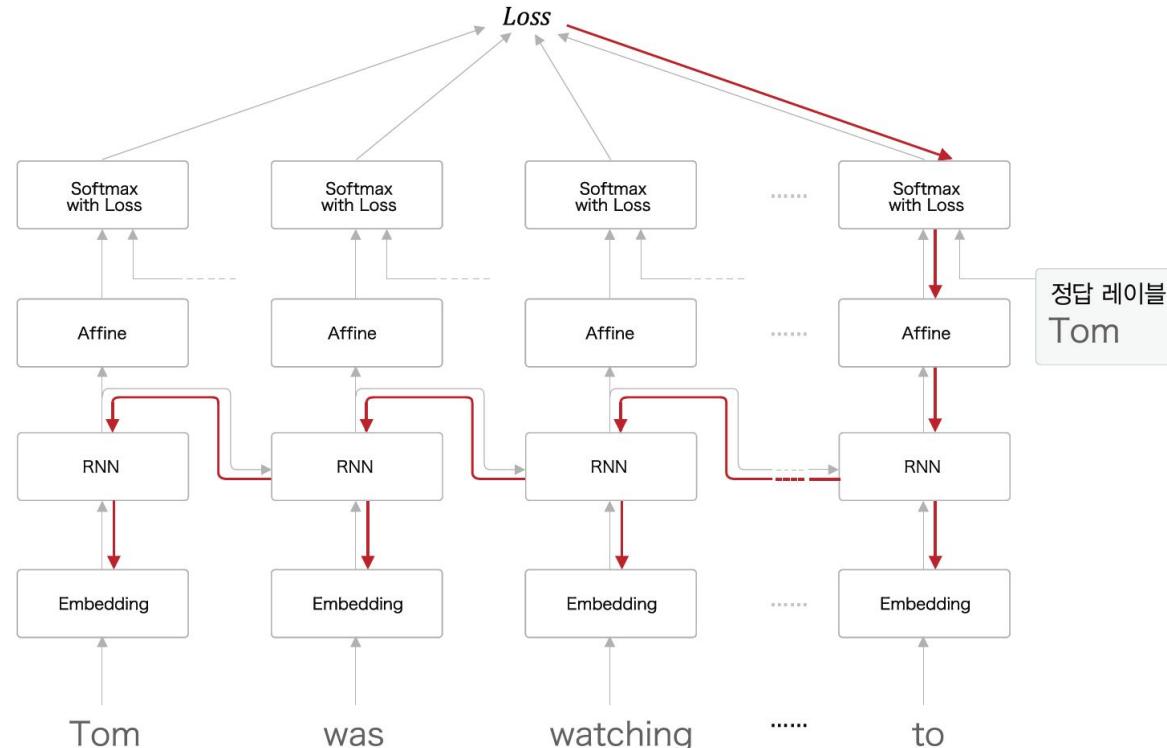
순환 신경망 (RNN)

- RNN은 순환하는 경로가 있음
- 순환 경로를 펼침으로써 다수의 RNN 계층이 연결된 신경망으로 해석할 수 있음
- 언어 모델은 단어 시퀀스를 확률로 해석함
- RNN 계층을 이용한 조건부 언어모델은 등장한 단어의 정보를 기억함

순환 신경망 (RNN)

RNN

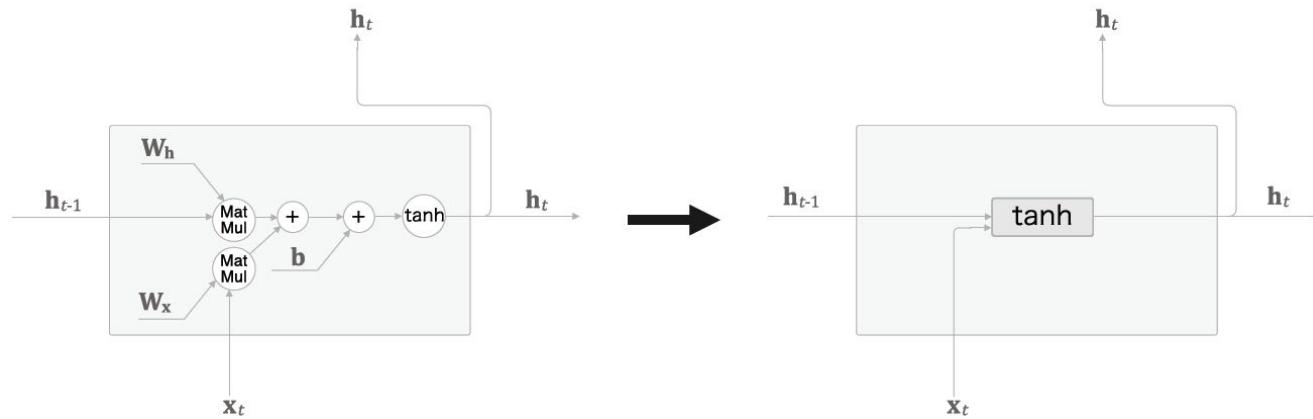
그림 6-4 정답 레이블이 “Tom”임을 학습할 때의 기울기 흐름



순환 신경망 (RNN)

RNN

그림 6-10 단순화한 도법을 적용한 RNN 계층: 이번 절에서는 가독성을 고려해 단순화한 도법을 사용한다.



순환 신경망 (RNN)

Long Short-Term Memory (LSTM)

그림 6-10 단순화한 도법을 적용한 RNN 계층: 이번 절에서는 가독성을 고려해 단순화한 도법을 사용한다.

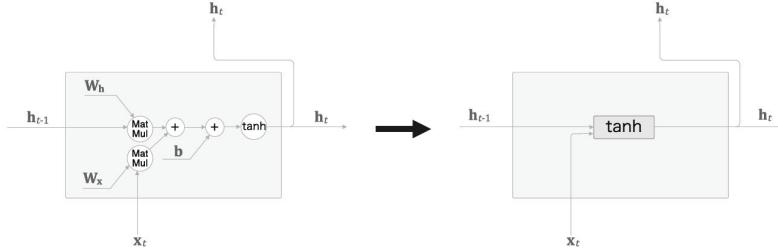
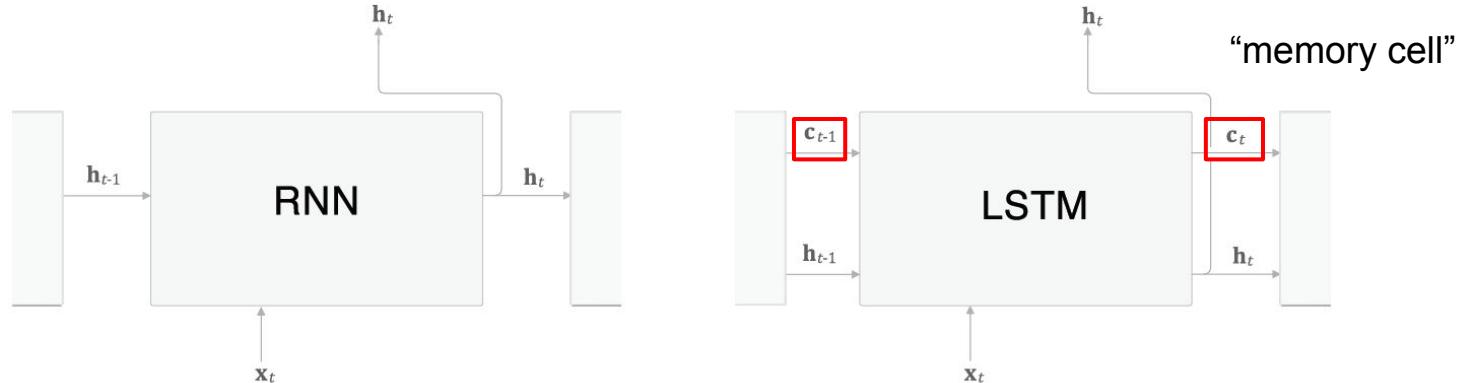


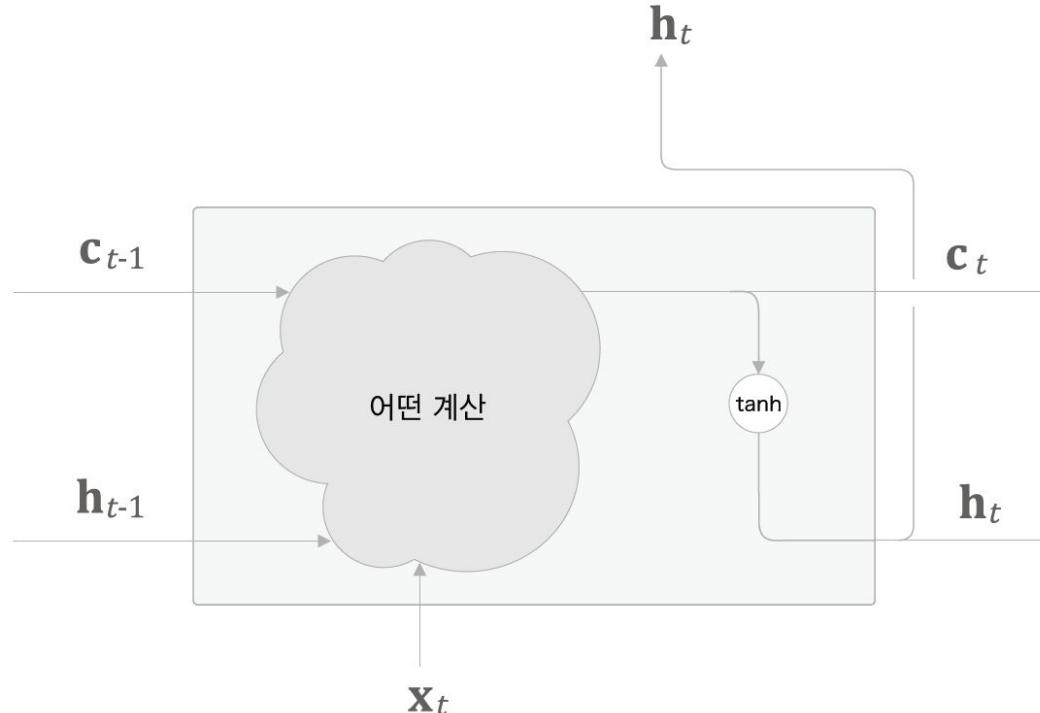
그림 6-11 RNN 계층과 LSTM 계층 비교



순환 신경망 (RNN)

Long Short-Term Memory (LSTM)

그림 6-12 기억 셀 c_t 를 바탕으로 은닉 상태 h_t 를 계산하는 LSTM 계층



순환 신경망 (RNN)

게이트란?

그림 6-13 비유하자면 게이트는 물의 흐름을 제어한다.

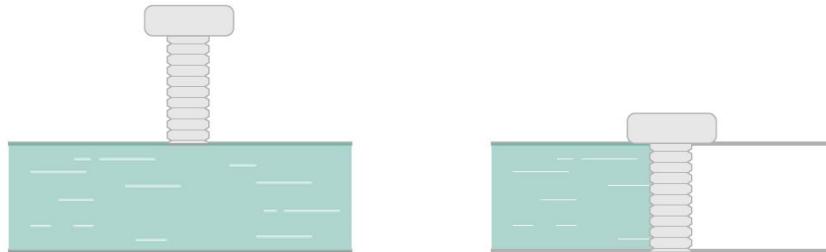
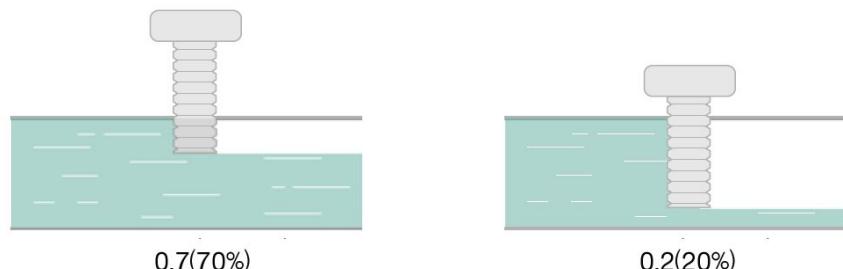


그림 6-14 물이 흐르는 양을 0.0~1.0 범위에서 제어한다.

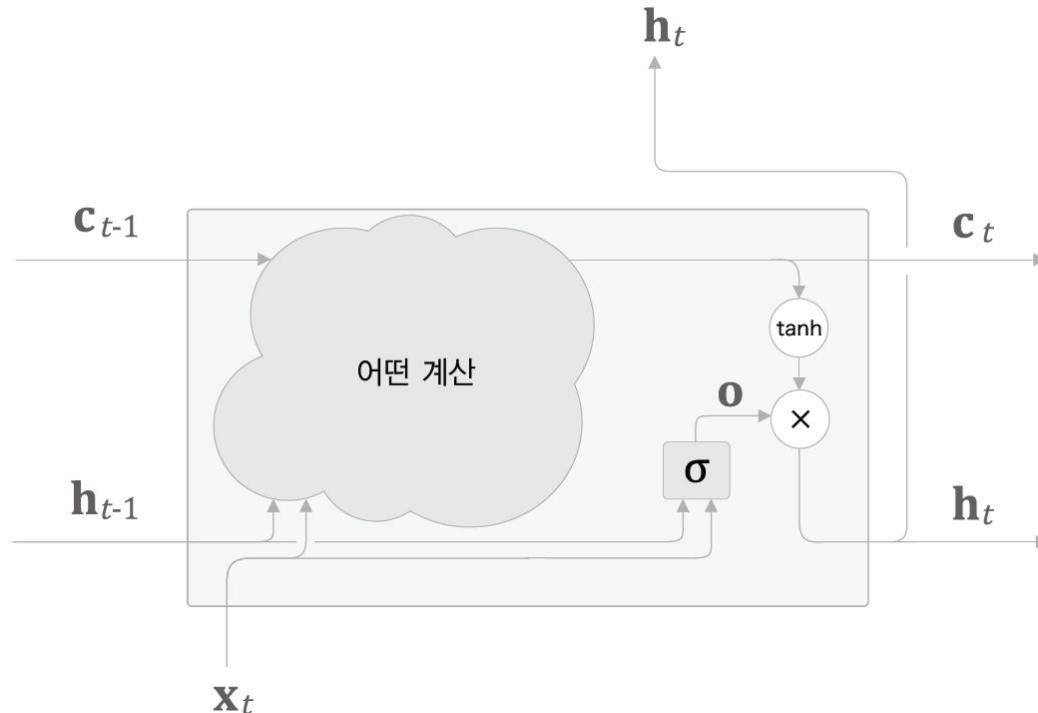


LSTM에서 사용하는 게이트는 열기/닫기 뿐 아니라, 어느 정도 열지 조절할 수 있고, 이 수치 또한 학습함

순환 신경망 (RNN)

Long Short-Term Memory (LSTM)

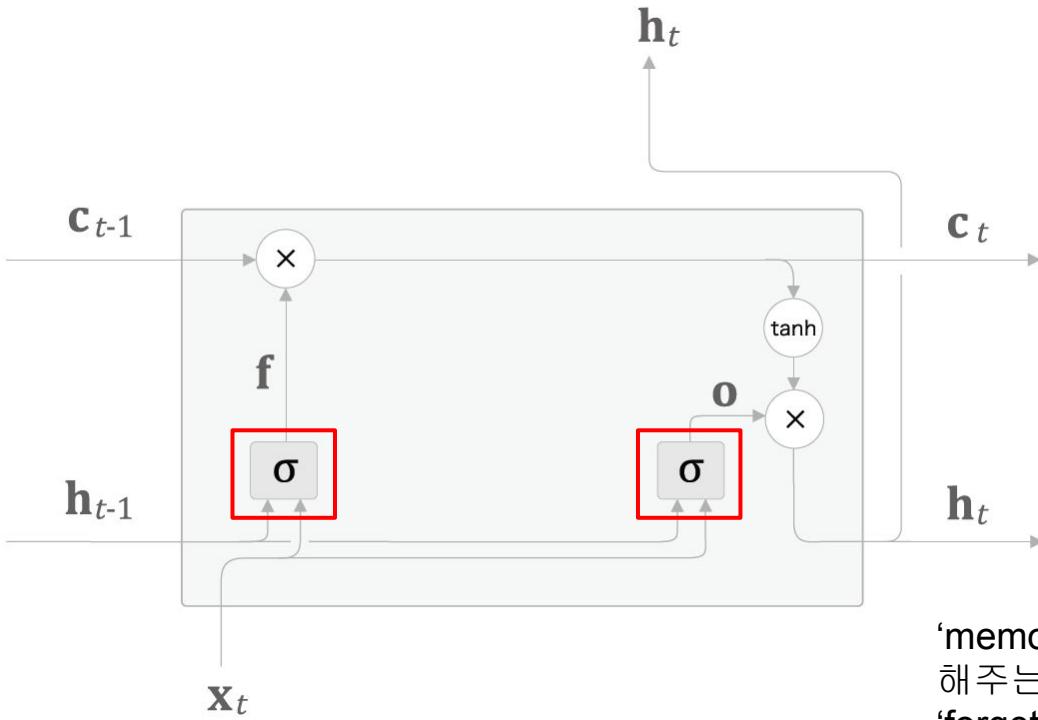
그림 6-15 output 게이트 추가



순환 신경망 (RNN)

forget 게이트

그림 6-16 forget 게이트 추가

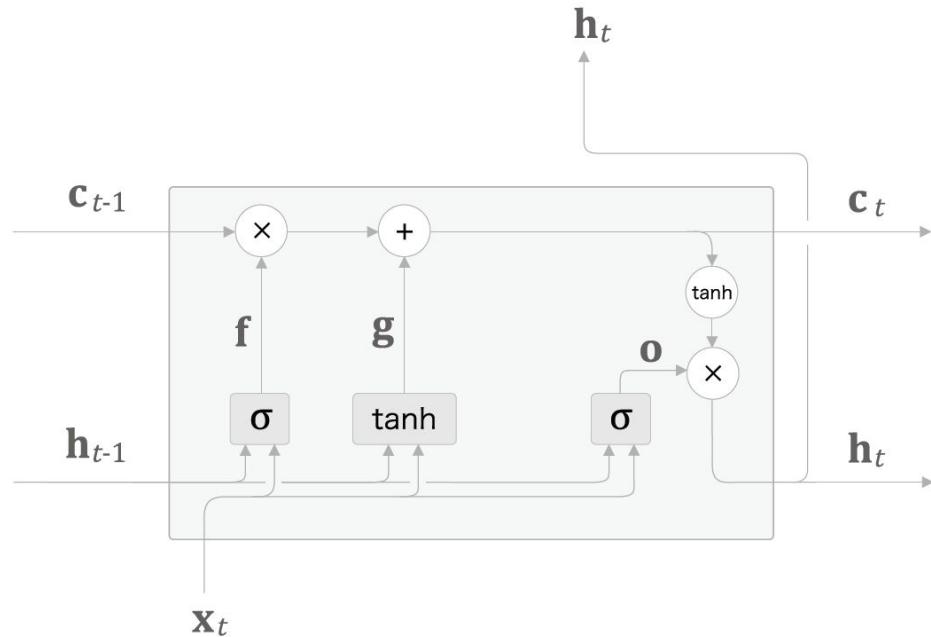


‘memory cell’의 정보 중 불필요한 기억을 잊게
해주는
‘forget gate’를 추가함

순환 신경망 (RNN)

새로운 기억 셀

그림 6-17 새로운 기억 셀에 필요한 정보를 추가

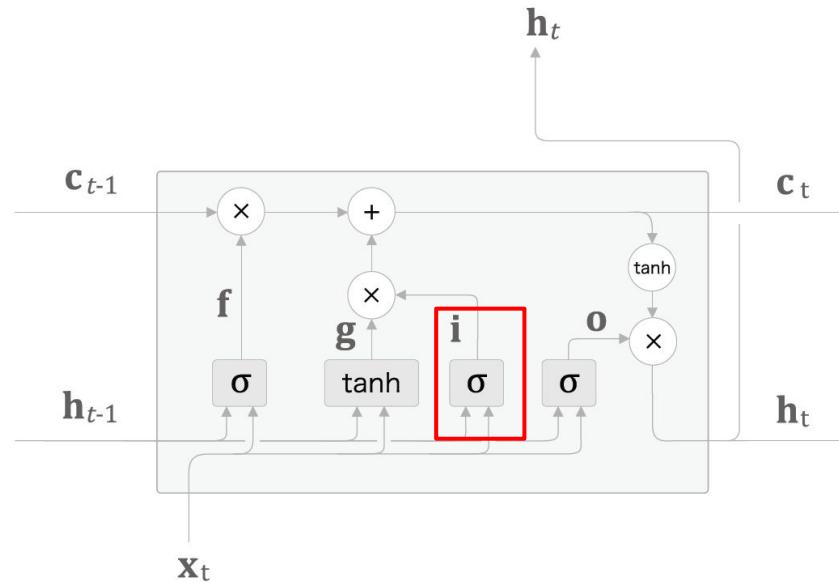


forget 게이트만 지나면 기억할 정보에 대한게 사라지기 때문에..

순환 신경망 (RNN)

input 게이트

그림 6-18 input 게이트 추가



g 의 각 원소가 새로 추가되는 정보로써 얼마나 가치가 큰가?

순환 신경망 (RNN)

LSTM

$$\mathbf{f} = \sigma(\mathbf{x}_t \mathbf{W}_{\mathbf{x}}^{(\mathbf{f})} + \mathbf{h}_{t-1} \mathbf{W}_{\mathbf{h}}^{(\mathbf{f})} + \mathbf{b}^{(\mathbf{f})}) \quad \mathbf{c}_t = \mathbf{f} \odot \mathbf{c}_{t-1} + \mathbf{g} \odot \mathbf{i}$$

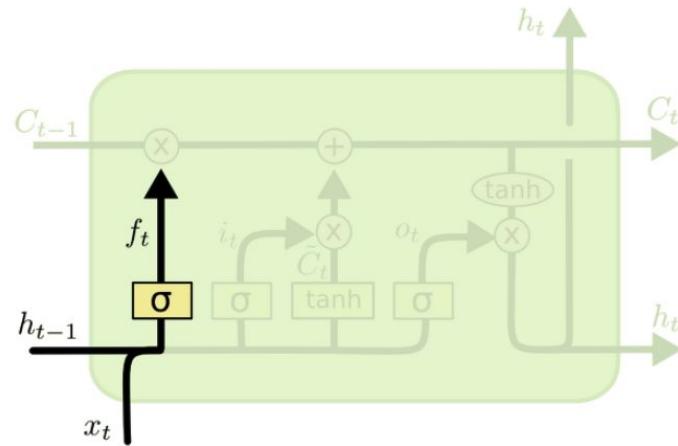
$$\mathbf{g} = \tanh(\mathbf{x}_t \mathbf{W}_{\mathbf{x}}^{(\mathbf{g})} + \mathbf{h}_{t-1} \mathbf{W}_{\mathbf{h}}^{(\mathbf{g})} + \mathbf{b}^{(\mathbf{g})}) \quad \mathbf{h}_t = \mathbf{o} \odot \tanh(\mathbf{c}_t)$$

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

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

순환 신경망 (RNN)

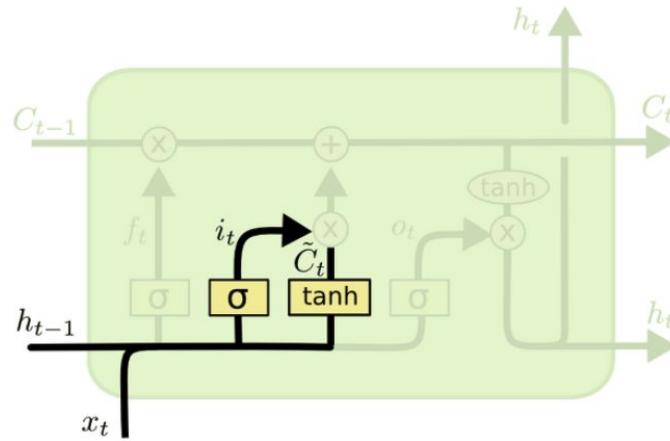
LSTM



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

순환 신경망 (RNN)

LSTM

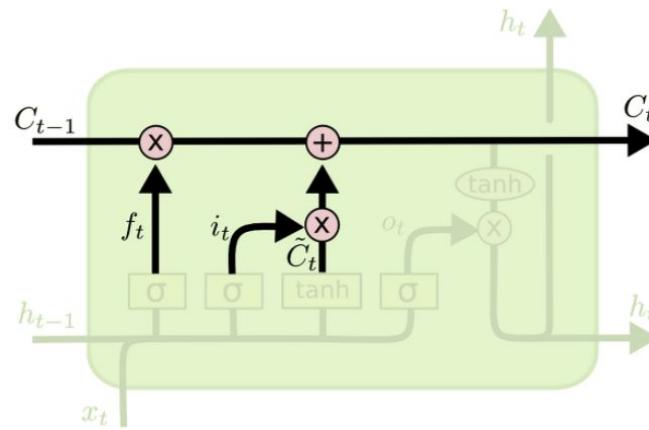


$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$$

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

순환 신경망 (RNN)

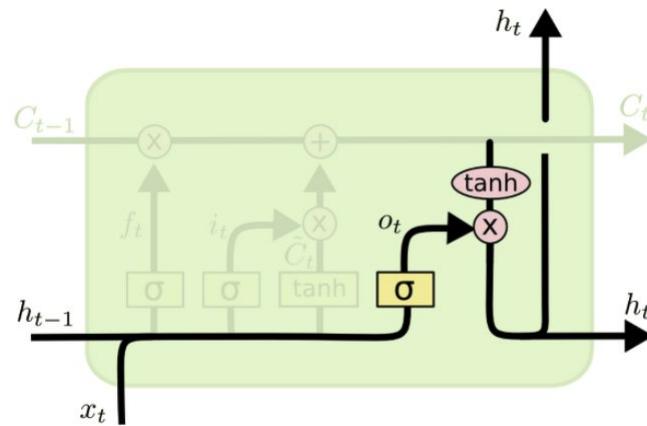
LSTM



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

순환 신경망 (RNN)

LSTM



$$o_t = \sigma (W_o [h_{t-1}, x_t] + b_o)$$
$$h_t = o_t * \tanh (C_t)$$

순환 신경망 (RNN)

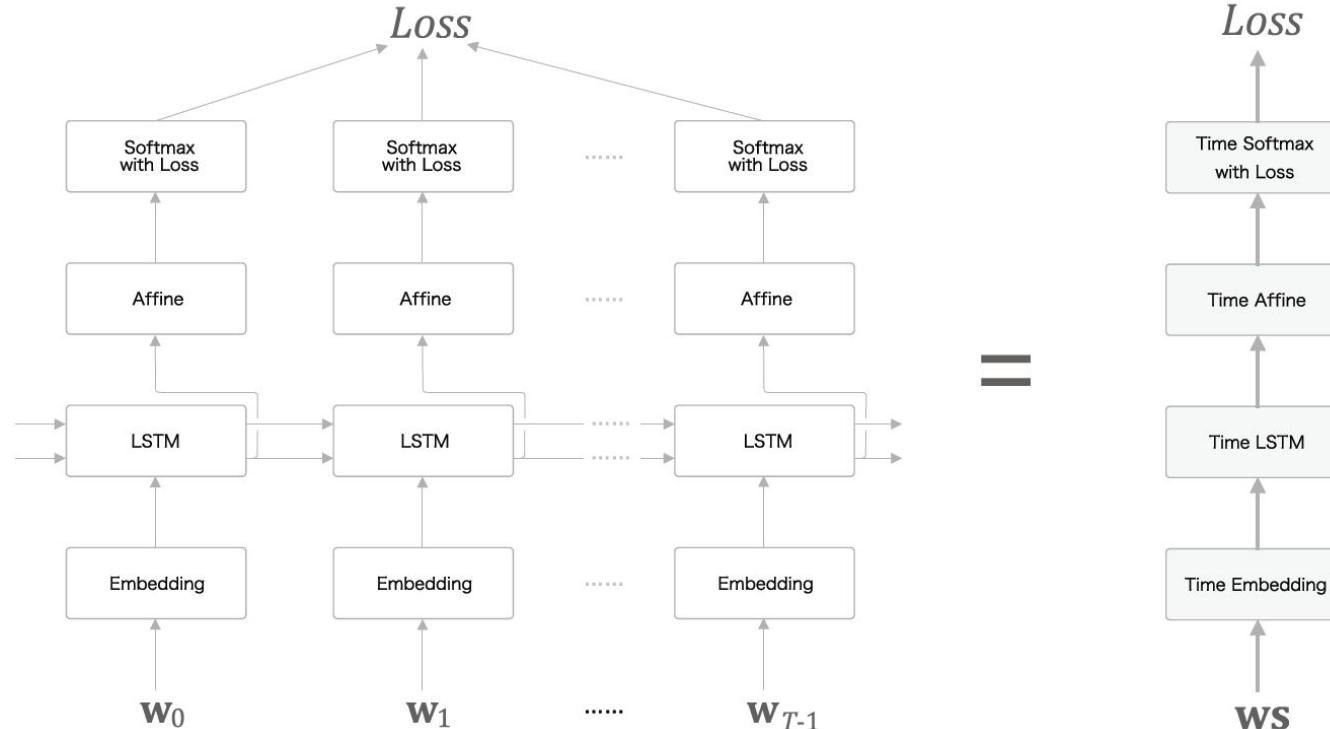
지금까지 배운 내용..

- 단순한 RNN 학습과정에서는 기울기 소실과 폭발이 문제가 됨
- 게이트가 추가된 RNN을 활용하면 이를 보완할 수 있음
- LSTM에는 input, output, forget 총 3개의 게이트가 있음

RNN을 사용한 문장 생성

언어 모델을 사용한 문장 생성

그림 7-1 앞 장에서 구현한 언어 모델: 오른쪽은 시계열 데이터를 한꺼번에 처리하는 Time 계층을 사용했고, 왼쪽은 같은 구성을 펼친 모습



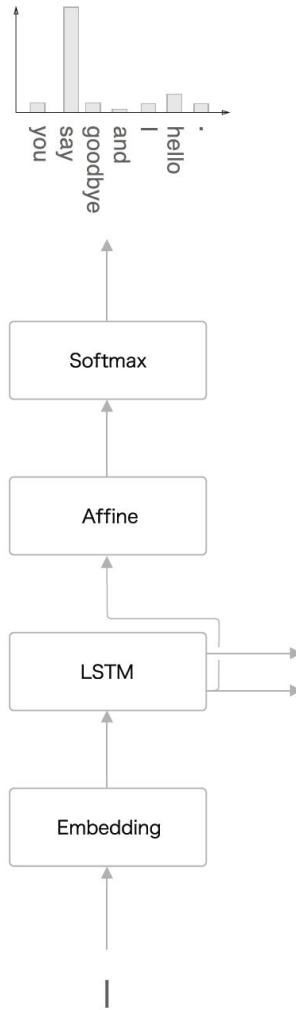
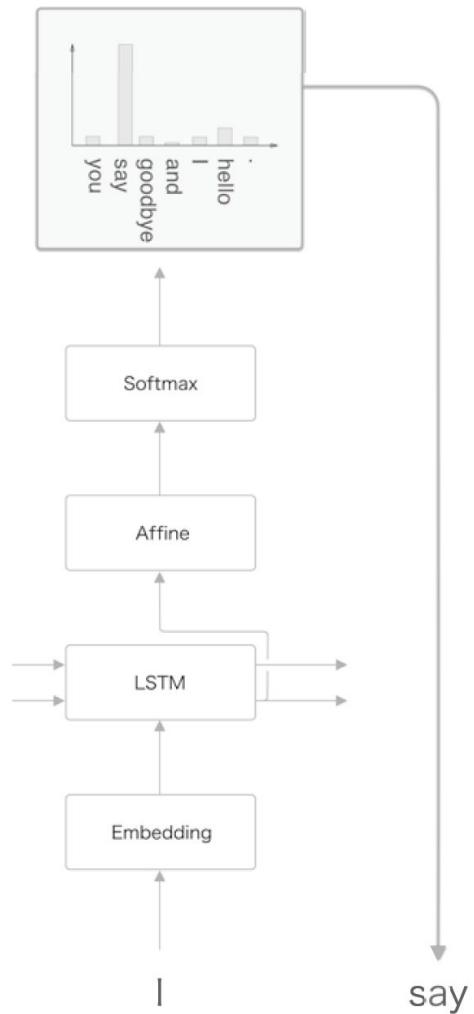
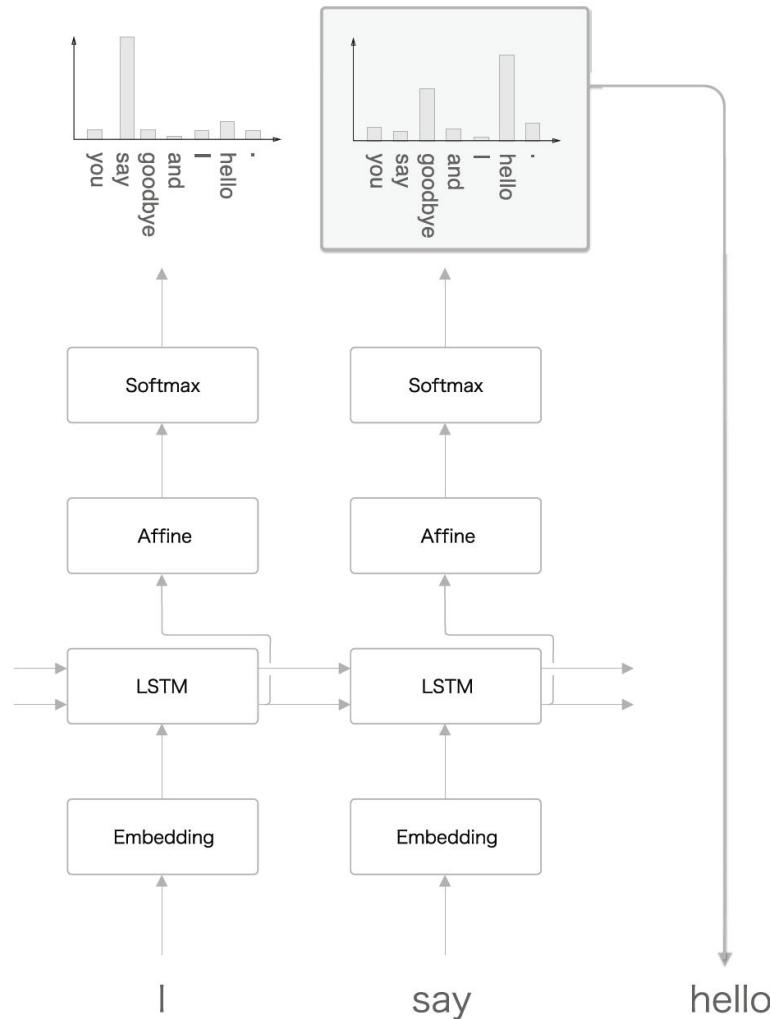


그림 7-3 확률분포대로 단어를 하나 샘플링한다.





RNN을 사용한 문장 생성

문장 생성 구현 실습

1. RNN 구조 실습
2. RNN을 활용한 문장 생성 실습

RNN을 사용한 문장 생성

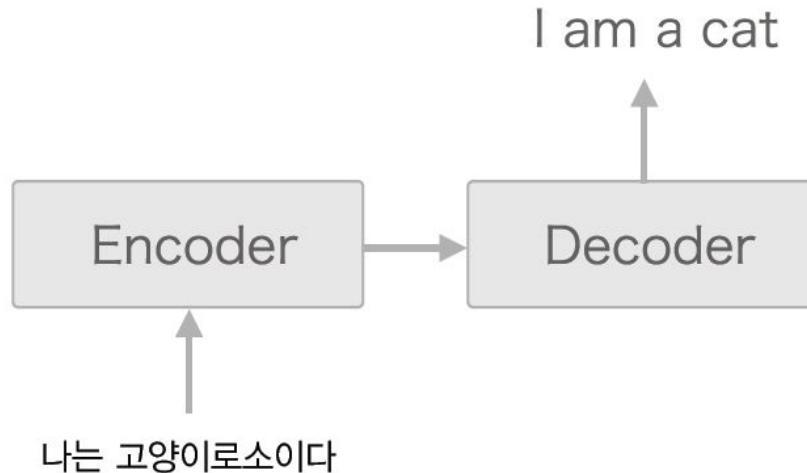
seq2seq



RNN을 사용한 문장 생성

seq2seq

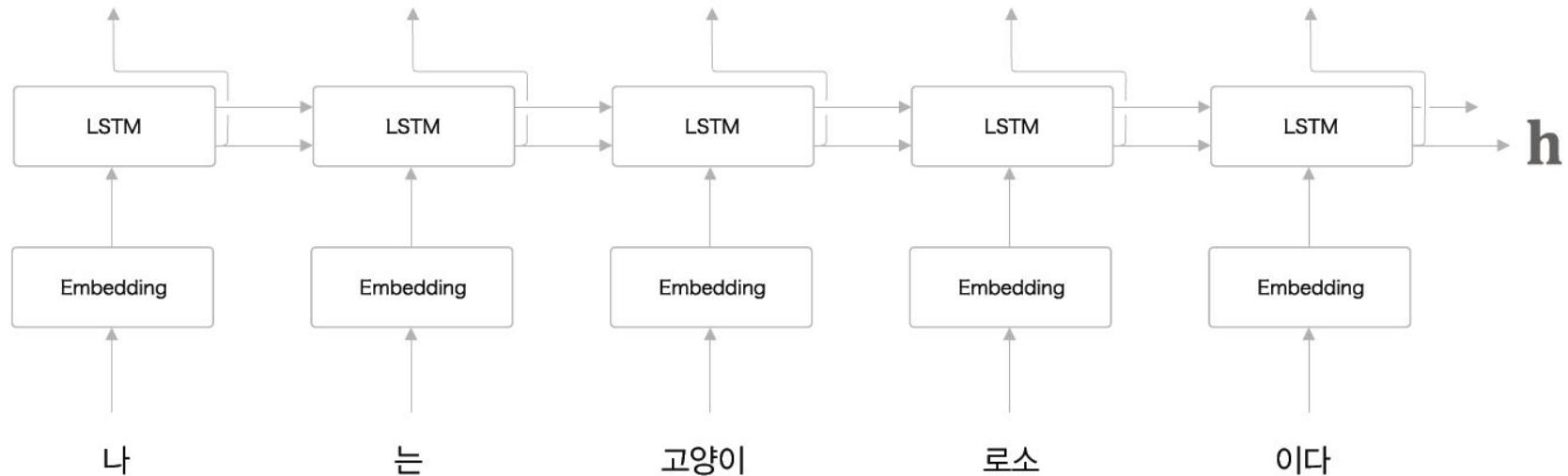
seq2seq는 Encoder-Decoder 모델임



RNN을 사용한 문장 생성

seq2seq

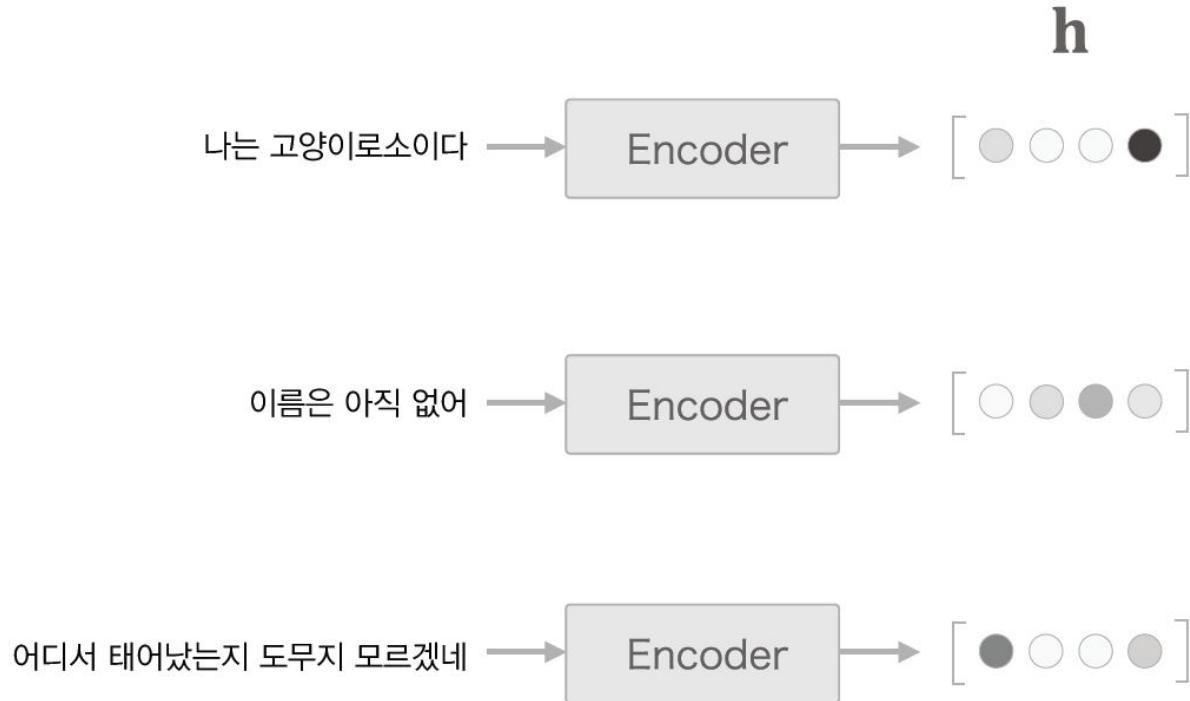
그림 7-6 Encoder를 구성하는 계층



RNN을 사용한 문장 생성

seq2seq

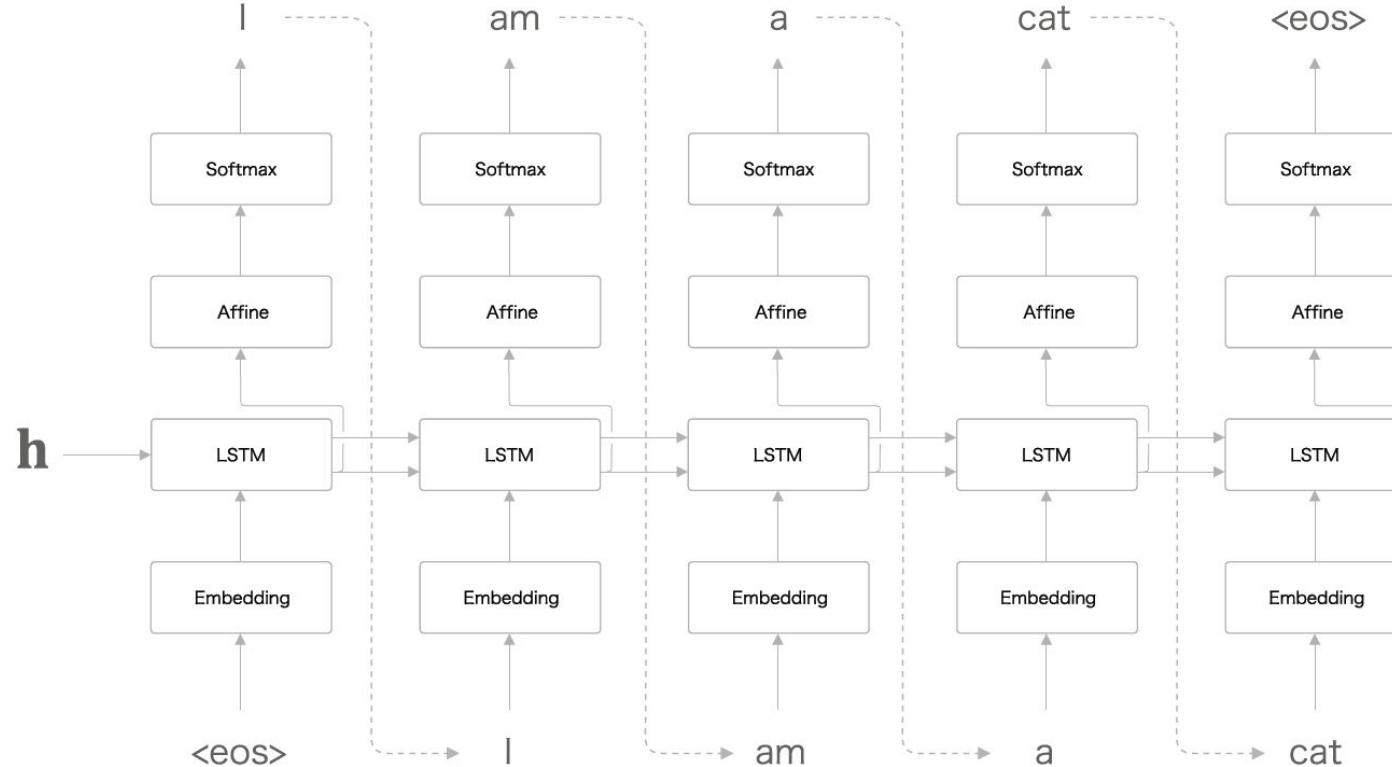
그림 7-7 Encoder는 문장을 고정 길이 벡터로 인코딩한다.



RNN을 사용한 문장 생성

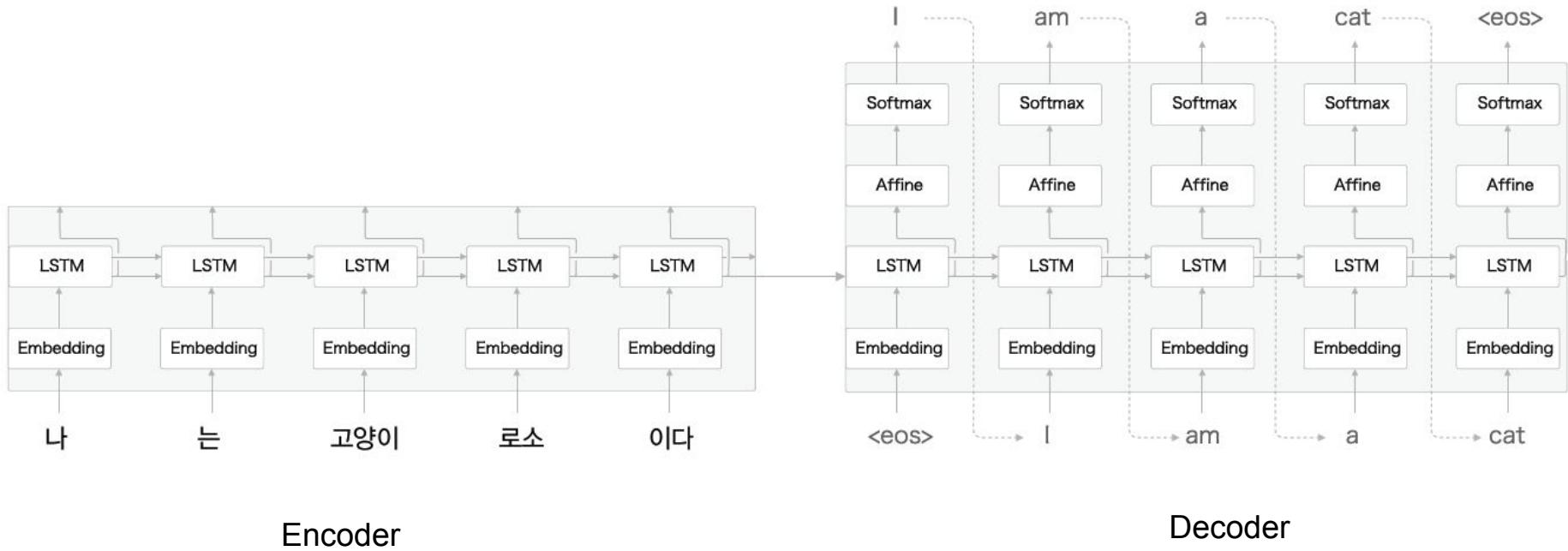
seq2seq

그림 7-8 Decoder를 구성하는 계층



RNN을 사용한 문장 생성

seq2seq



RNN을 사용한 문장 생성

seq2seq - toy problem



덧셈 예제를 학습시켜본다

RNN을 사용한 문장 생성

seq2seq - toy problem

그림 7-11 미니배치 학습을 위해 ‘공백 문자’로 패딩을 수행하여 입력 · 출력 데이터의 크기를 통일한다.

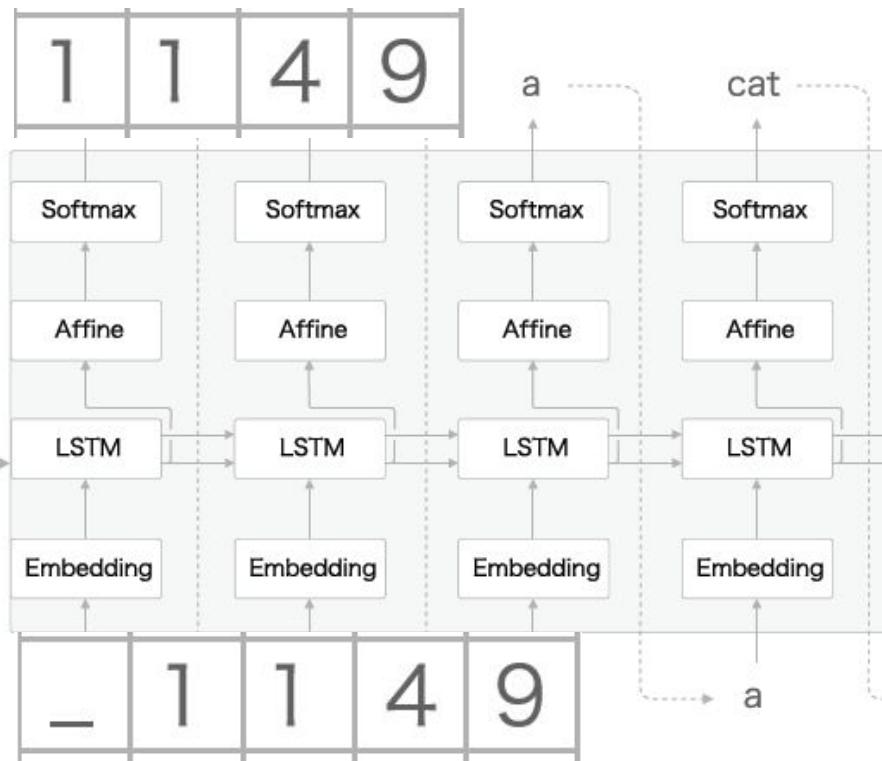
입력

5	7	+	5			
6	2	8	+	5	2	1
2	2	0	+	8		

출력

-	6	2			
-	1	1	4	9	
-	2	2	8		

최대 세자리 숫자끼리 더하는 문제를 가정한다면, 입력의 최대 길이는 7 (덧셈 부호 포함)
출력의 최대 길이는 구분자 까지 포함하여 5



RNN을 사용한 문장 생성

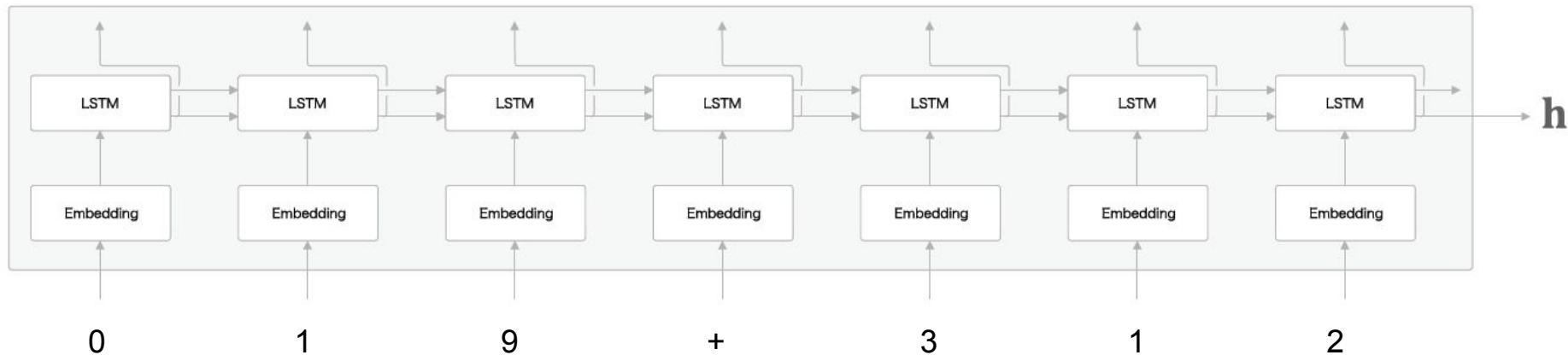
seq2seq - toy problem

덧셈 문제 데이터셋 구현 실습

RNN을 사용한 문장 생성

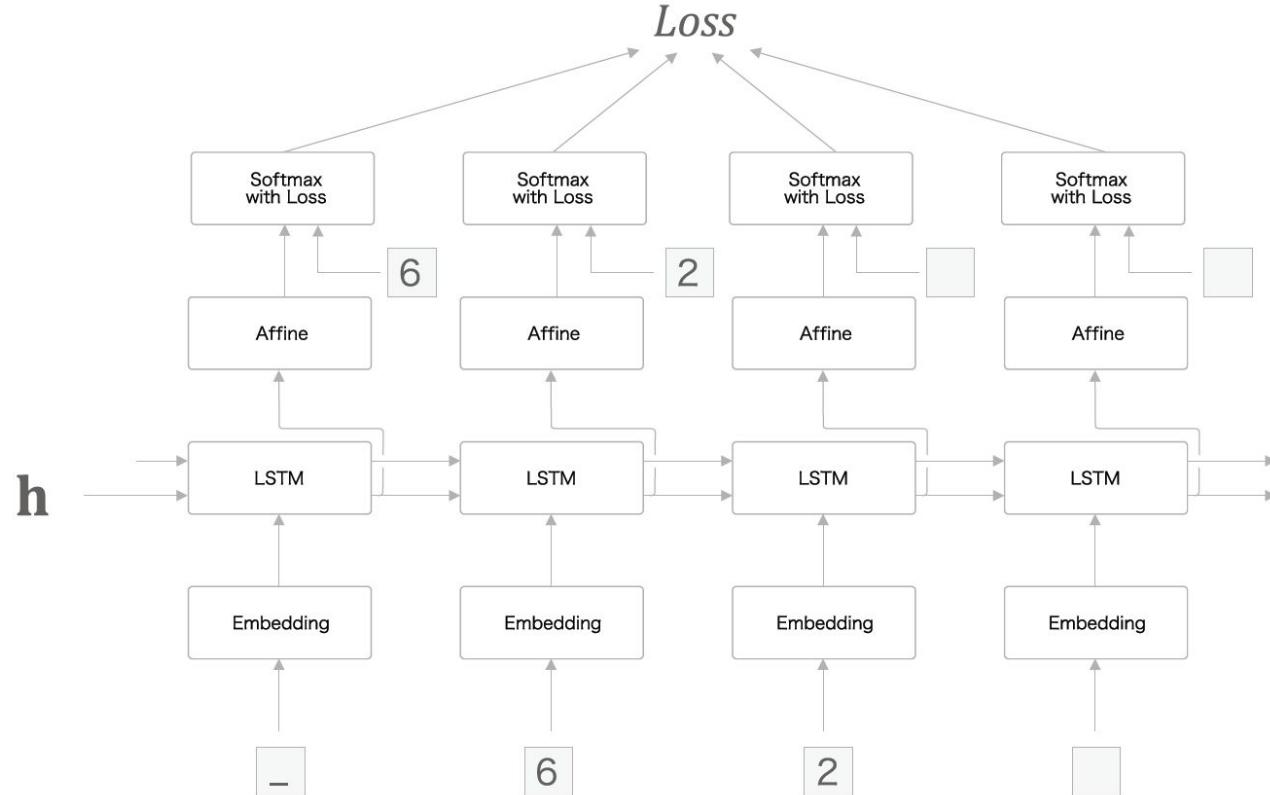
seq2seq - toy problem

그림 7-14 Encoder의 계층 구성



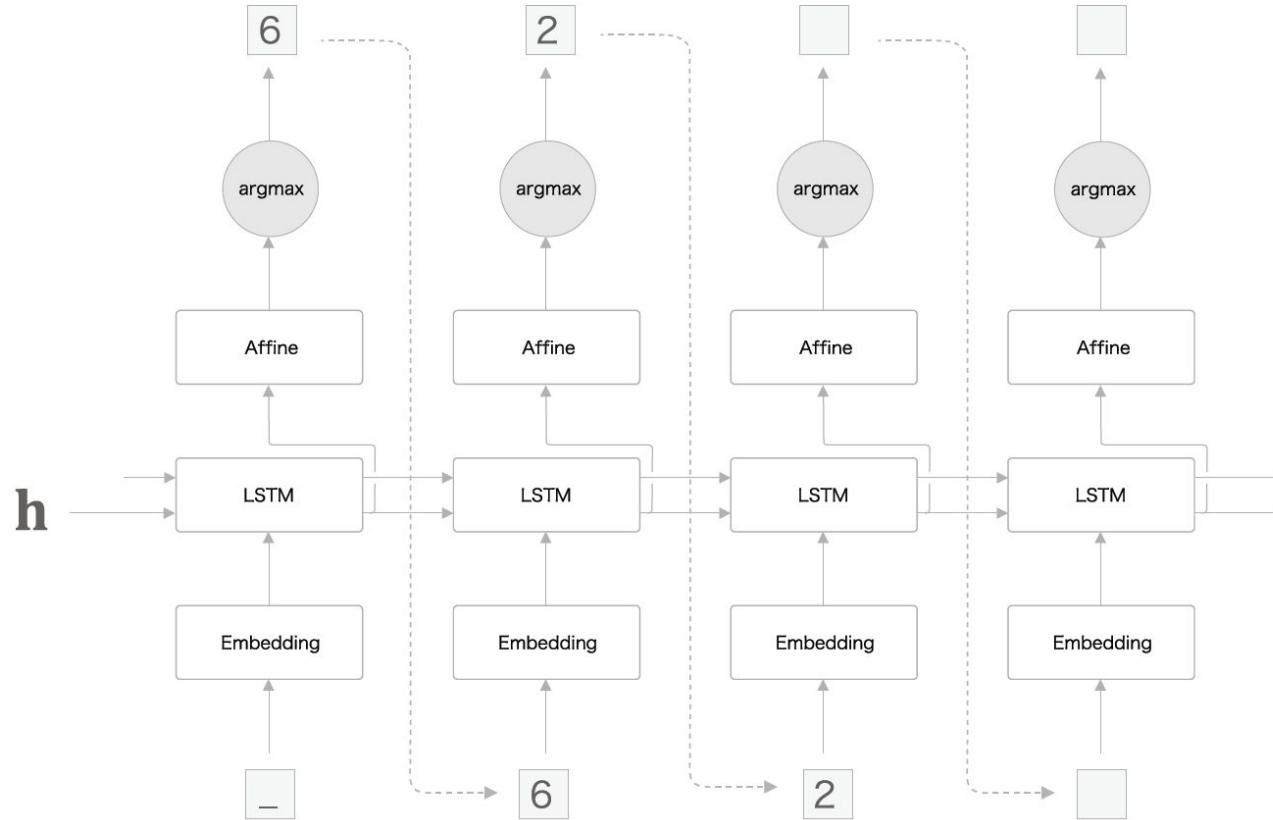
RNN을 사용한 문장 생성

seq2seq - toy problem



RNN을 사용한 문장 생성

seq2seq - toy problem



RNN을 사용한 문장 생성

seq2seq - toy problem

덧셈 문제 toy problem 학습 구현 실습

RNN을 사용한 문장 생성

seq2seq를 이용하는 애플리케이션 - 챗봇

기계: 안녕하세요.

사람: 안녕하세요.

기계: 어떤 OS를 사용하시죠?

사람: 리눅스요.

.....

기계: 실행할 때 어떤 오류가 발생하죠?

사람: VPN 어쩌고 하는 오류가 나옵니다.

기계: VPN을 사용하여 네트워크에 연결할 때 오류 메시지가 뭐라고 뜨나요?

사람: 연결이 되지 않거나 혹은 그와 비슷한 상태...

.....

기계: 지금 집인가요, 아니면 회사인가요?

사람: 집이요.

기계: 그럼, <URL>에 가서 선생님의 이름과 암호를 입력해보세요.

사람: 했어요.

기계: <NAME>에서 로그인하여 잘 되는지 확인해보세요.

사람: 오, 잘 되네요!

기계: 좋습니다. 다른 문제는 없나요?

사람: 없어요, 매우 잘 됩니다.

RNN을 사용한 문장 생성

seq2seq를 이용하는 애플리케이션 - 알고리즘 학습

Input:

```
j=8584  
for x in range(8):  
    j+=920  
    b=(1500+j)  
    print((b+7567))
```

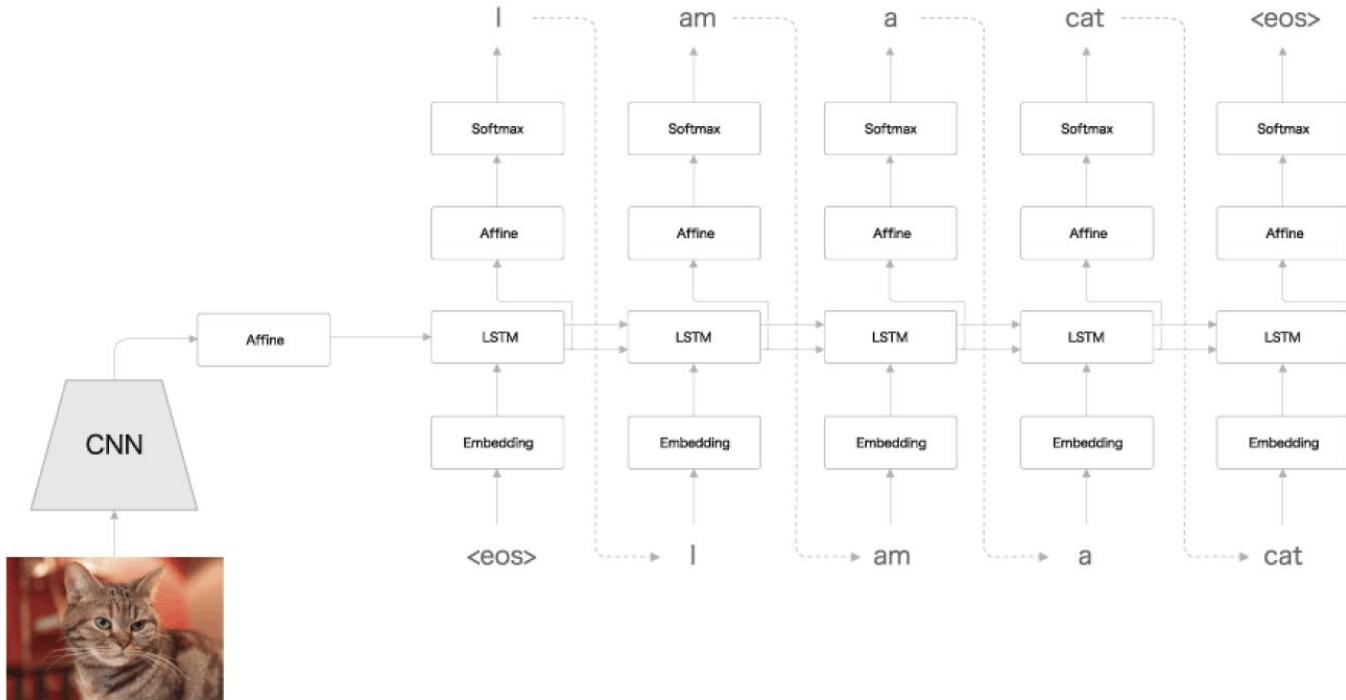
Target: 25011.**Input:**

```
i=8827  
c=(i-5347)  
print((c+8704) if 2641<8500 else 5308)
```

Target: 12184.

RNN을 사용한 문장 생성

seq2seq를 이용하는 애플리케이션 - 이미지 캡셔닝



RNN을 사용한 문장 생성

seq2seq를 이용하는 애플리케이션 - 이미지 캡셔닝

A person on a beach flying
a kite(해변에서 연 날리는 사람)



A black and white photo of a train
on a train track
(선로 위의 열차를 찍은 흑백 사진)



A person skiing down a snow
covered slope
(눈 덮인 슬로프에서 스키 타는 사람)



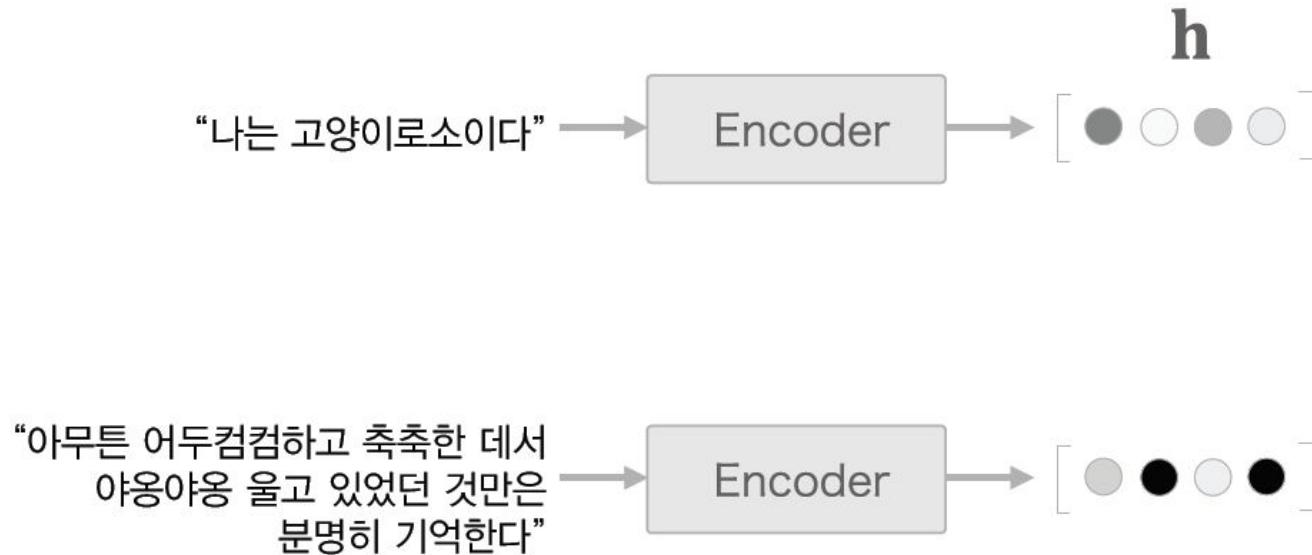
A group of giraffe standing next
to each other
(횡대로 줄지어 서 있는 기린 떼)



어텐션 (Attention)

seq2seq의 문제점

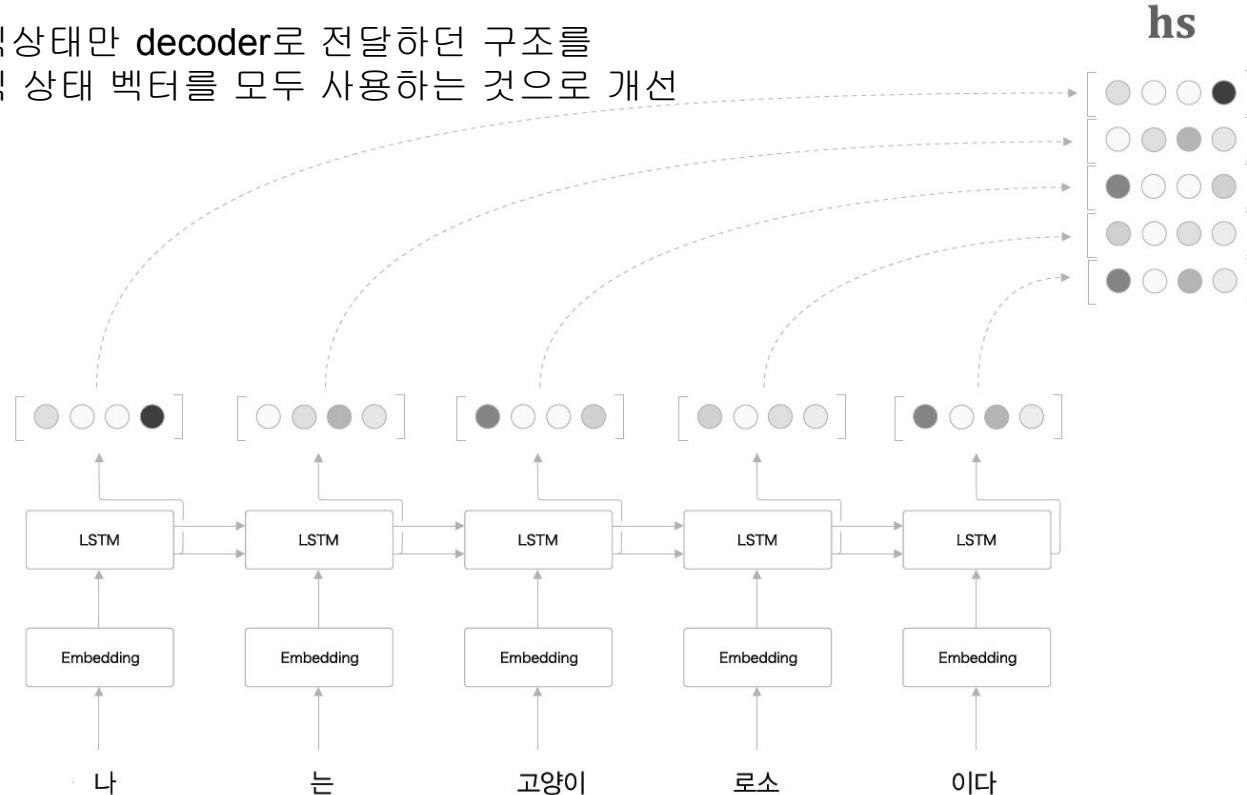
그림 8-1 입력 문장의 길이에 관계없이, Encoder는 정보를 고정 길이의 벡터로 밀어 넣는다.



어텐션 (Attention)

Encoder 개선

마지막 은닉상태만 **decoder**로 전달하던 구조를
계층의 은닉 상태 벡터를 모두 사용하는 것으로 개선

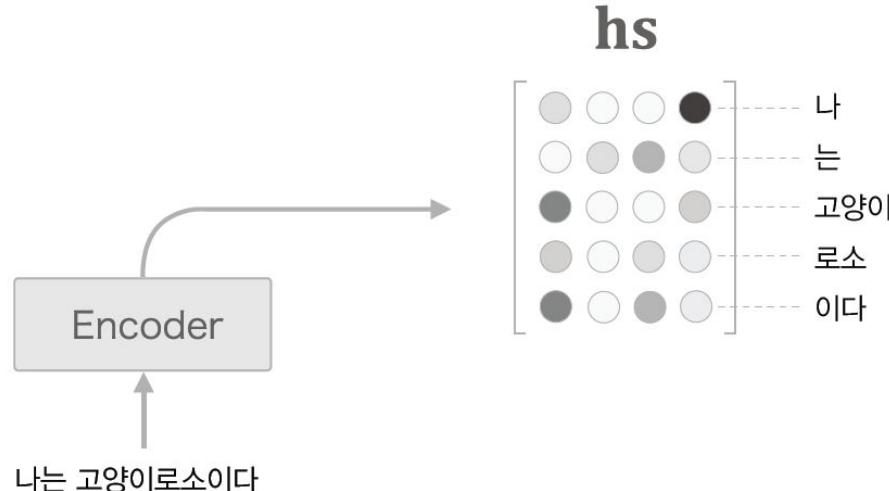


어텐션 (Attention)

Encoder 개선

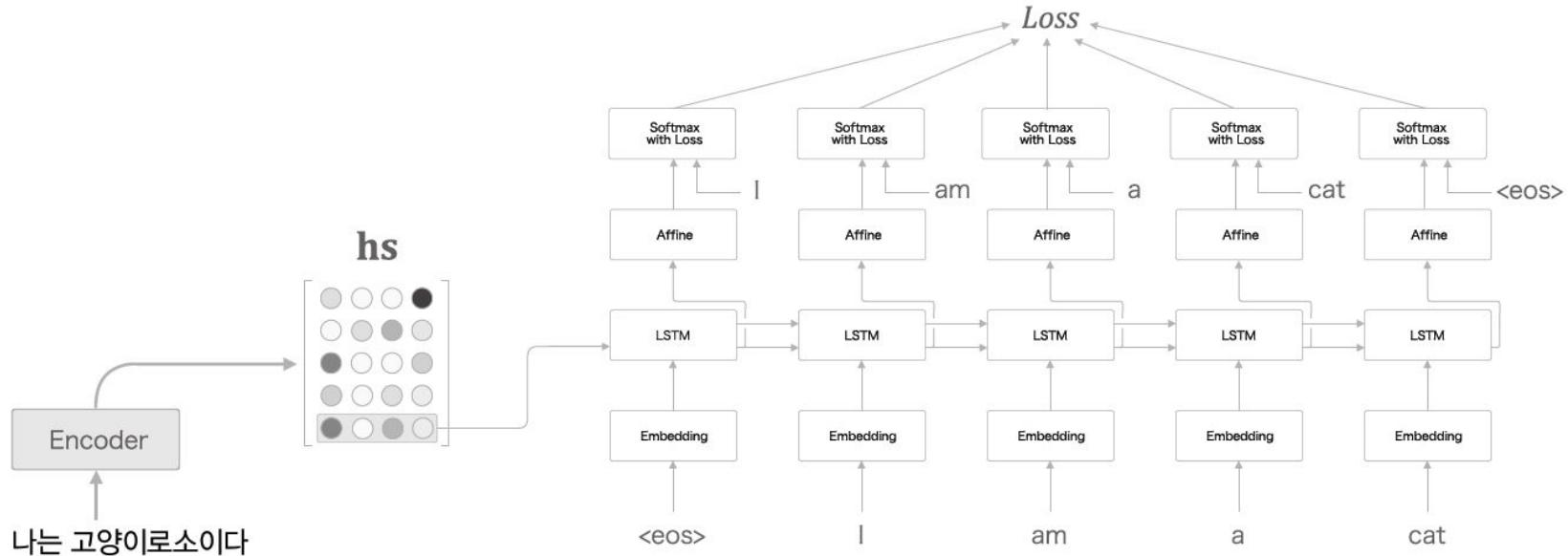
마지막 은닉상태만 **decoder**로 전달하던 구조를
계층의 은닉 상태 벡터를 모두 사용하는 것으로 개선

그림 8-3 Encoder의 출력 **hs**는 단어 수만큼의 벡터를 포함하며, 각각의 벡터는 해당 단어에 대한 정보를 많이 포함 한다.



어텐션 (Attention)

Decoder 개선

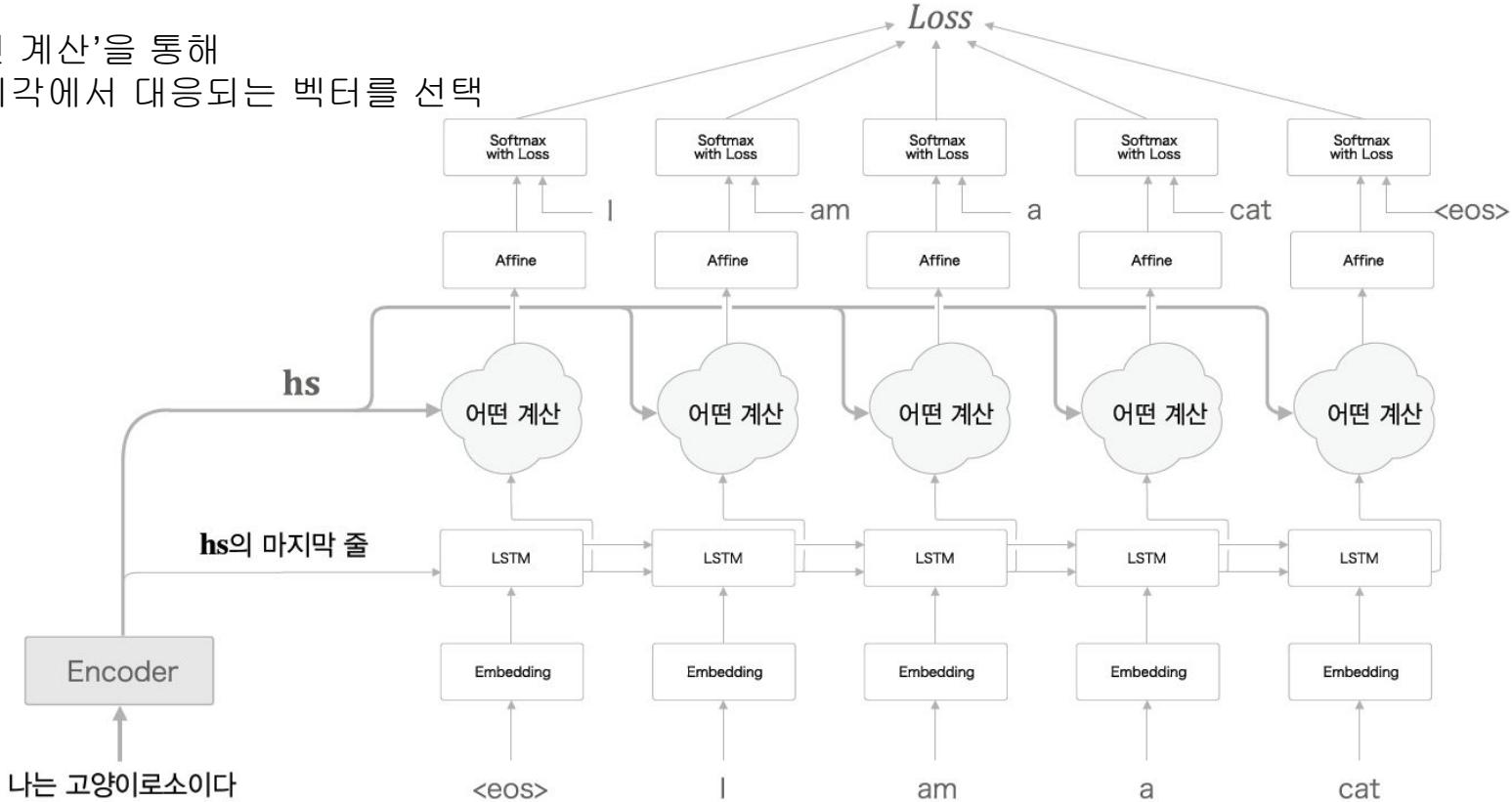


기존 방법에서는 마지막 은닉상태 정보만을 **decoder**로 넘겨주는 구조였음

어텐션 (Attention)

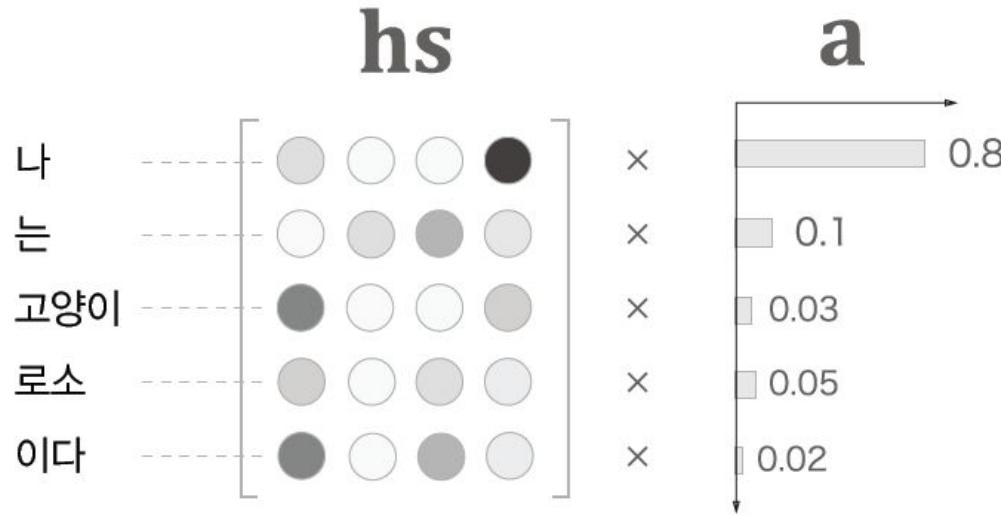
Decoder 개선

'어떤 계산'을 통해
각 시각에서 대응되는 벡터를 선택



어텐션 (Attention)

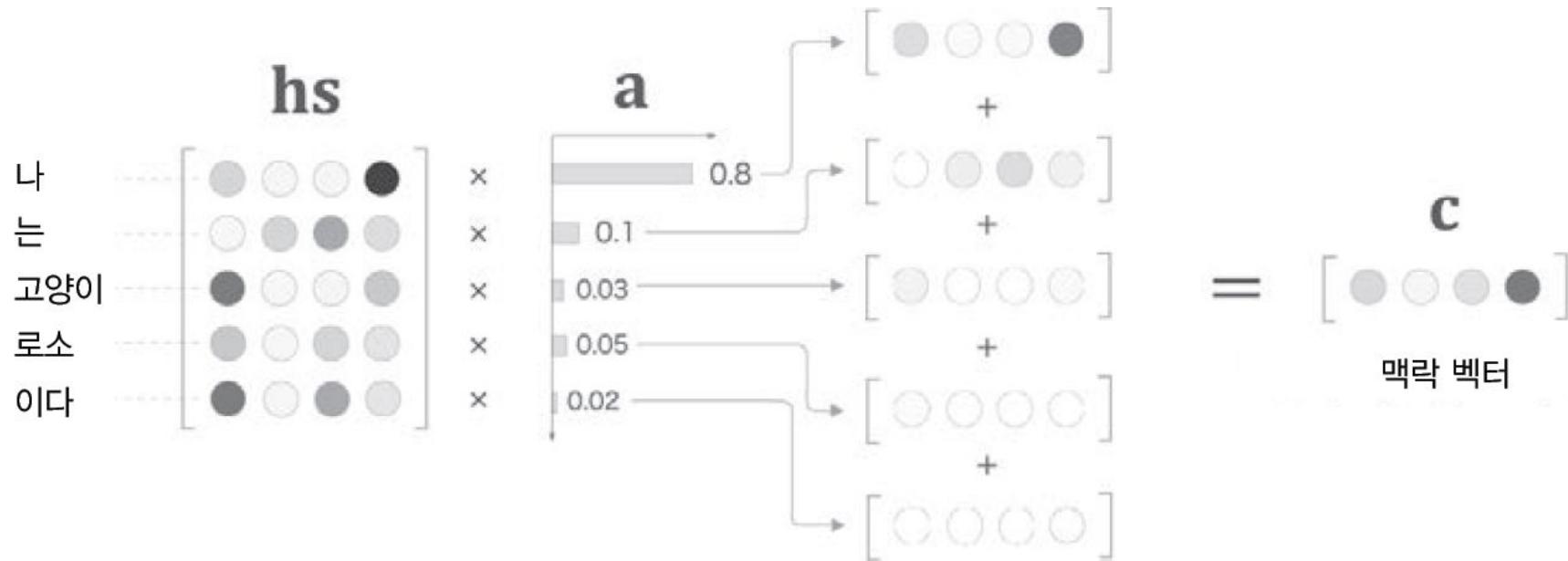
Decoder 개선



‘선택 한다’는 개념은 미분가능하게 표현이 불가능하므로
단어에 가중치를 부여하는 방식으로 해결

어텐션 (Attention)

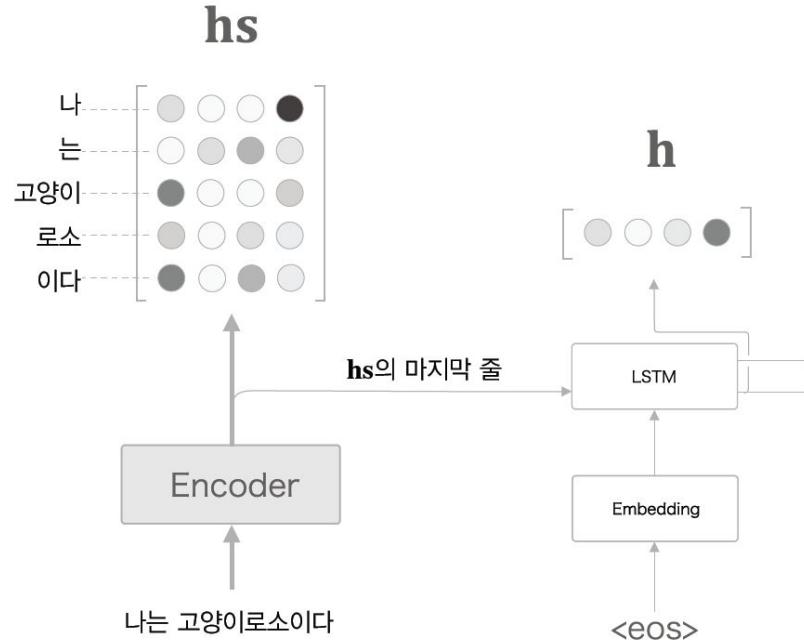
Decoder 개선



어텐션 (Attention)

Decoder 개선

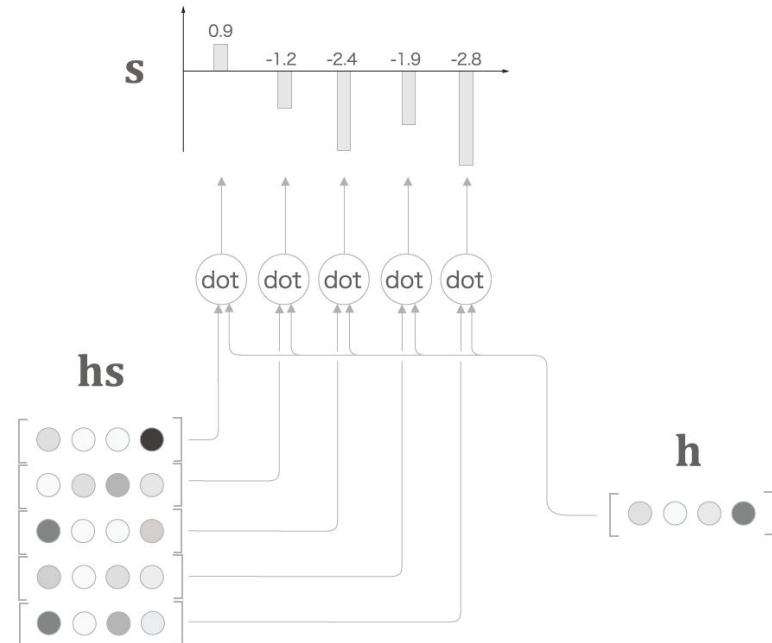
그렇다면 가중치는 어떻게 구해야 하는가?



어텐션 (Attention)

Decoder 개선

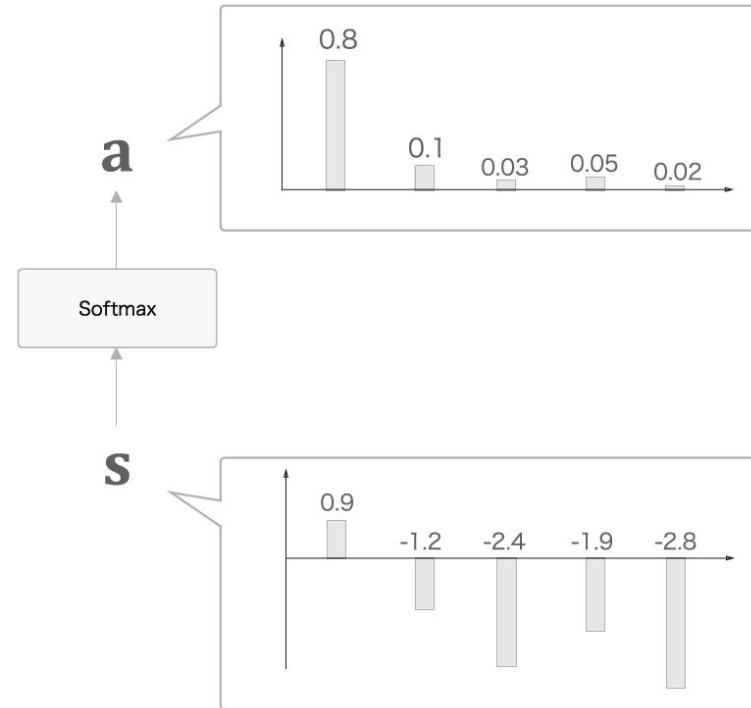
그렇다면 가중치는 어떻게 구해야 하는가? => 벡터의 유사도를 산출



어텐션 (Attention)

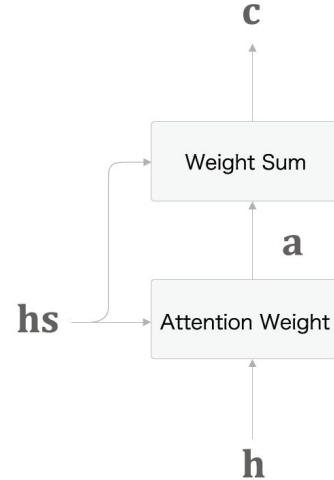
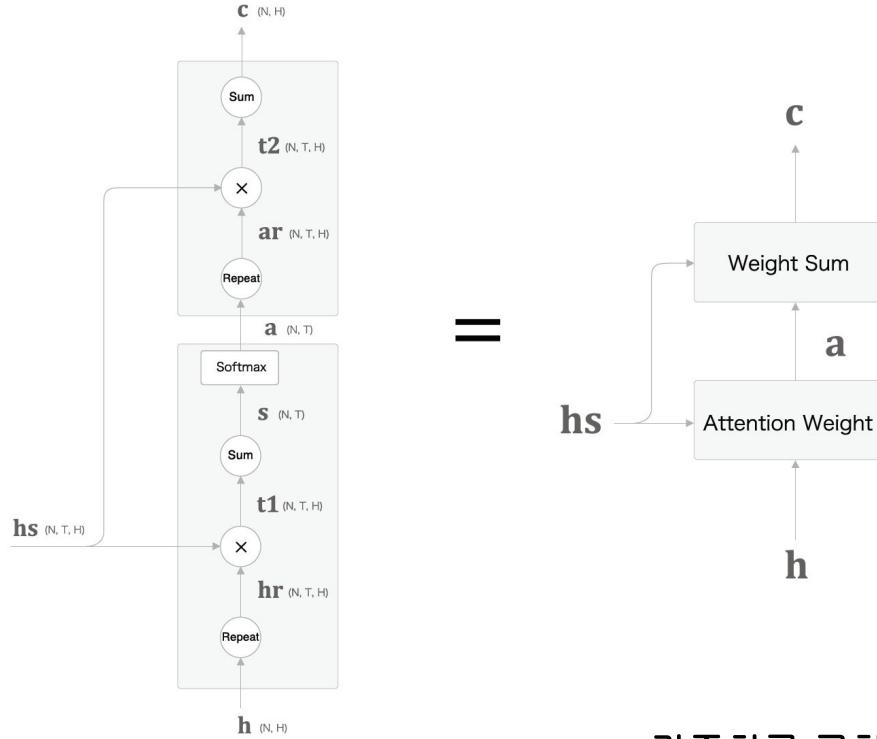
Decoder 개선

그림 8-14 Softmax를 통한 정규화



어텐션 (Attention)

Decoder 개선

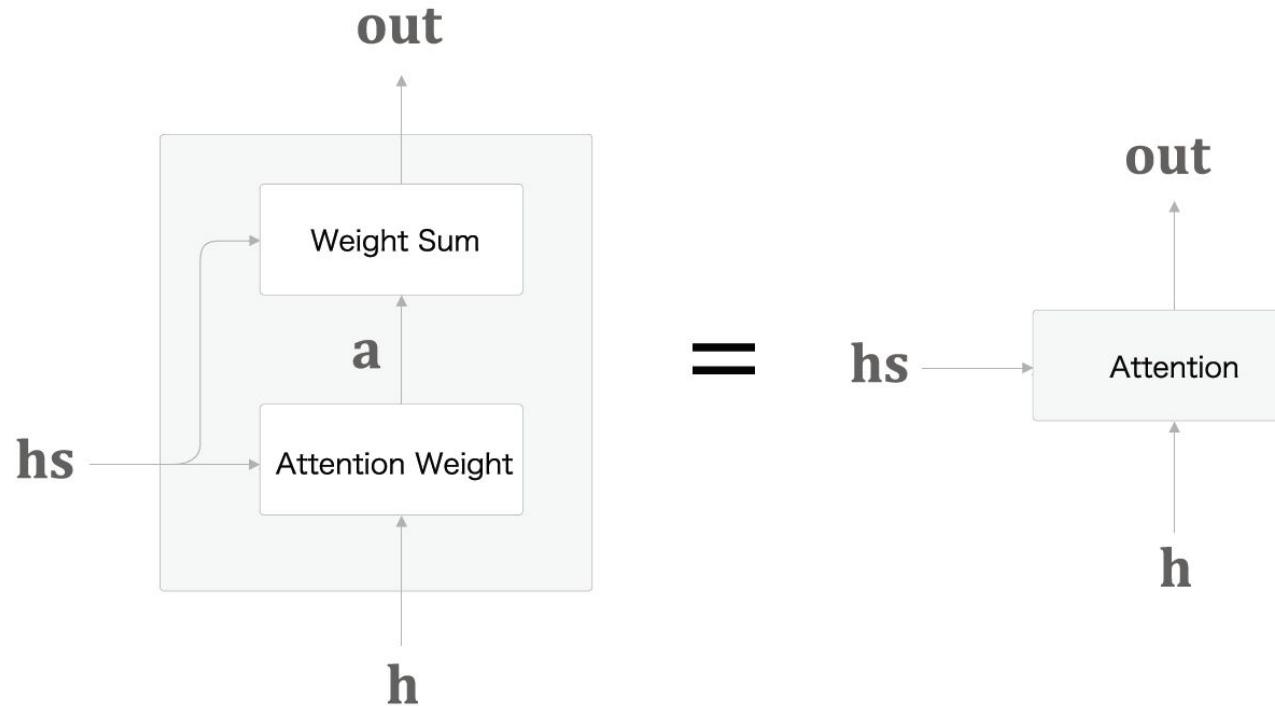


가중치를 곱한 후 가중합을 구하는 계층 : Attention 계층

어텐션 (Attention)

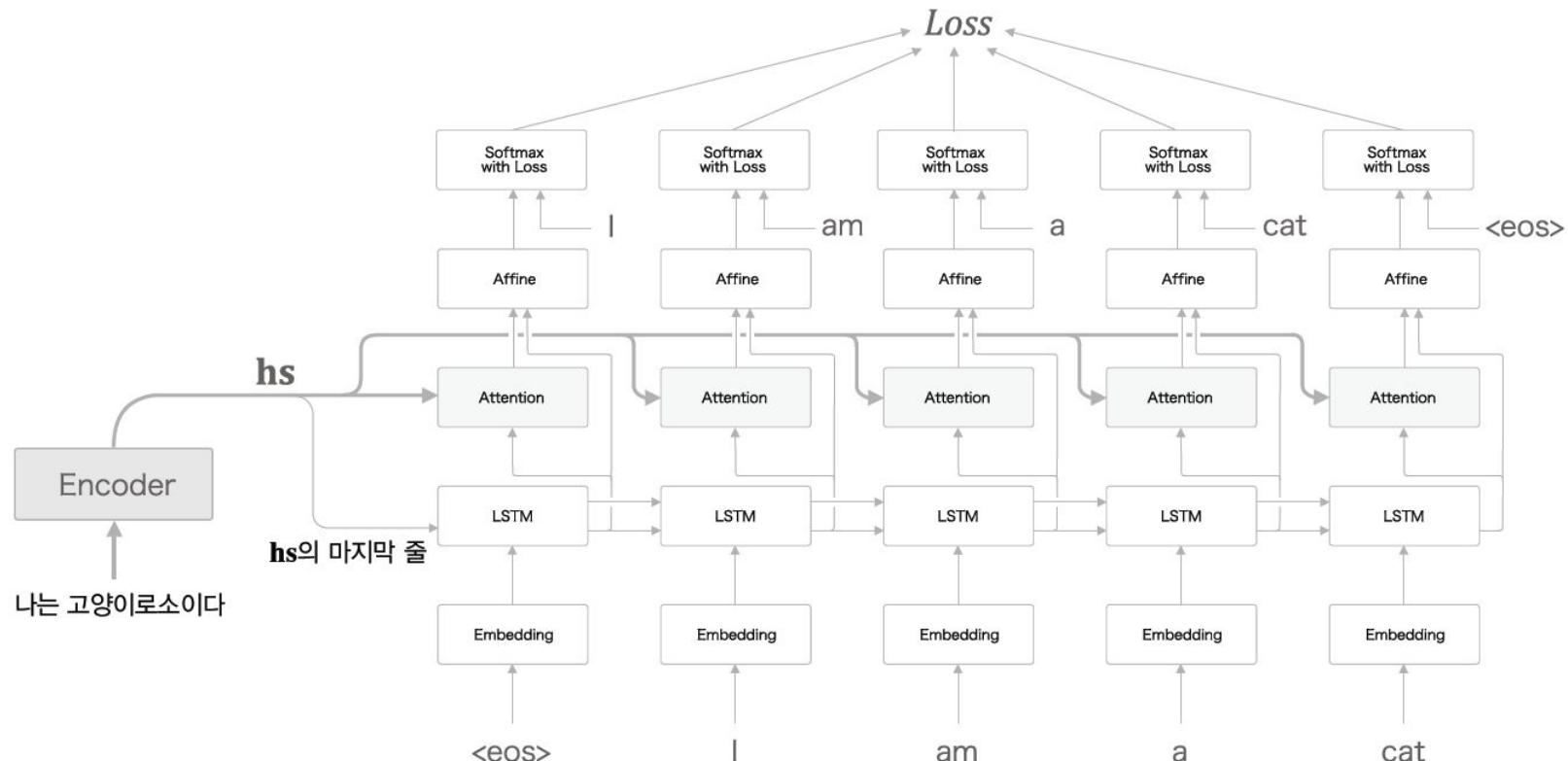
Decoder 개선

그림 8-17 왼쪽 계산 그래프를 Attention 계층으로 정리



어텐션 (Attention)

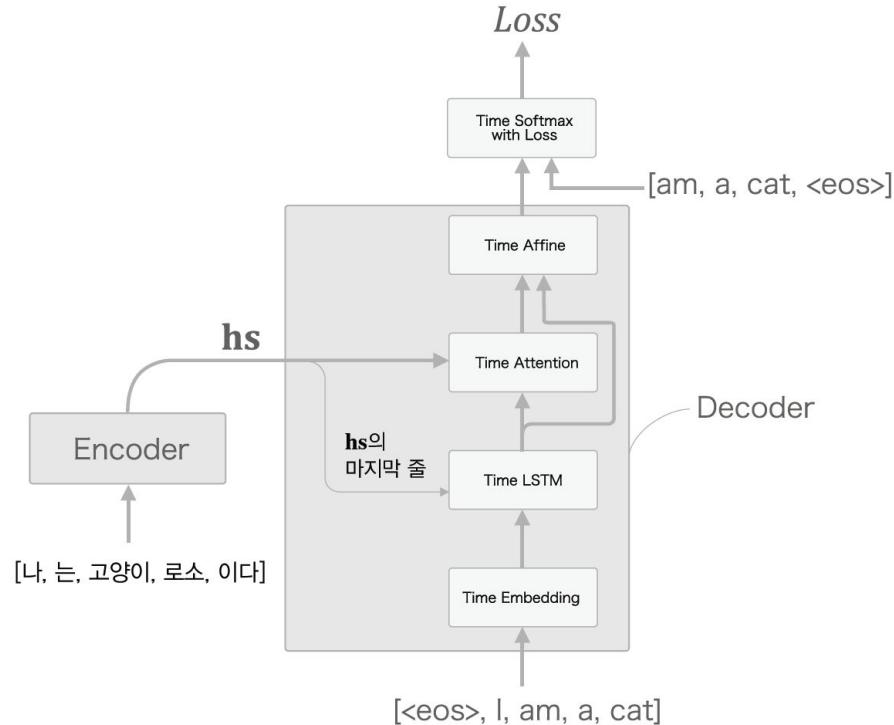
Decoder 개선



어텐션 (Attention)

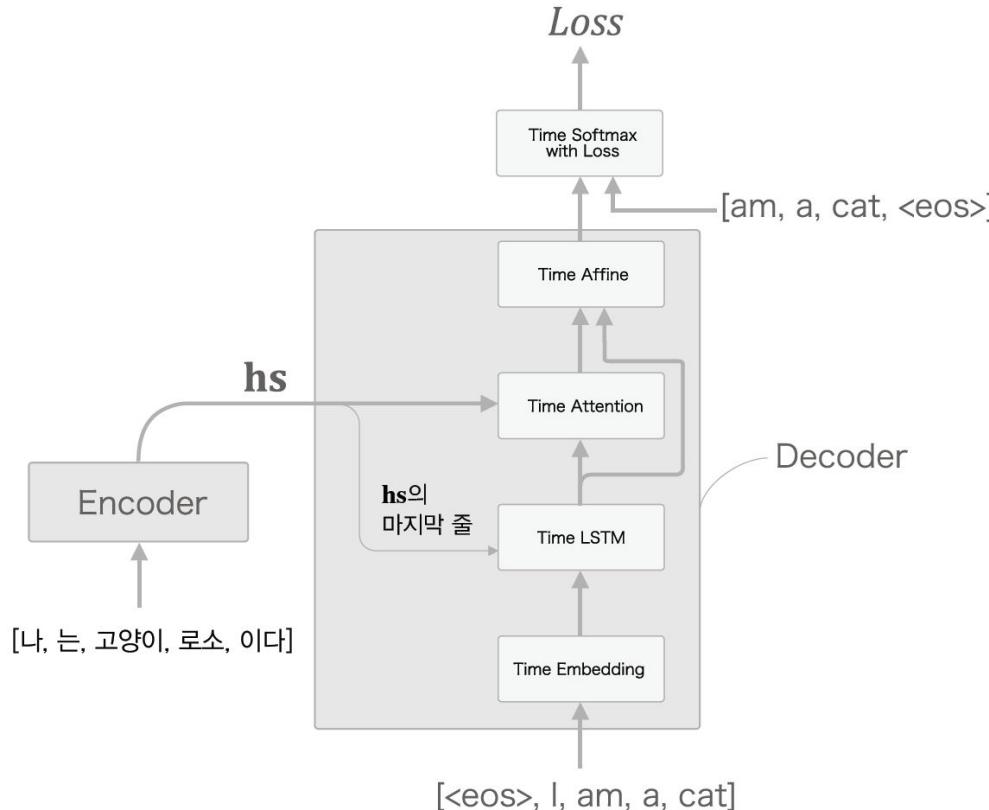
Attention을 갖춘 seq2seq 구현

그림 8-21 Decoder의 계층 구성



어텐션 (Attention)

Attention을 갖춘 seq2seq 구현



자연어 처리 실습

응용하기 좋은 데이터셋

- <https://github.com/niderhoff/nlp-datasets>
- <https://machinelearningmastery.com/datasets-natural-language-processing/>
- <https://www.conll.org/2019-shared-task>
- <http://ai.stanford.edu/~amaas/data/sentiment/>
- <https://rajpurkar.github.io/SQuAD-explorer/>
- <https://korquad.github.io/>
- <https://github.com/e9t/nsmc>

자연어 처리 실습

정량 평가 지표 - 기계 번역기의 성능 평가 지표

오늘은 날씨가 너무 좋고 바람도 선선하네요.



The weather is so nice and the wind is cool today.



The weather is so good today and the wind is cool.

정답 : The weather is fine and the wind is cool today.

자연어 처리 실습

정량 평가 지표 - 기계 번역기의 성능 평가 지표

Unigram precision

번역에 나온 단어 중 정답에 있는 단어의 수

번역에 있는 단어의 수

정답 : The weather is fine and the wind is cool today.



Google 번역

The weather is so nice and the wind is cool today. The weather is so good today and the wind is cool.

자연어 처리 실습

정량 평가 지표 - 기계 번역기의 성능 평가 지표

Unigram precision

번역에 나온 단어 중 정답에 있는 단어의 수

번역에 있는 단어의 수

정답 : The weather is fine and the wind is cool today.

The weather is fine fine fine cool cool.

자연어 처리 실습

정량 평가 지표 - 기계 번역기의 성능 평가 지표

Modified Unigram precision

번역에 나온 단어 중 정답에 있는 단어의 수 - 정답에서 등장 횟수를 넘는
수

번역에 있는 단어의 수

정답 : The weather is fine and the wind is cool today.

The weather is fine **fine** **fine** cool **cool**.

$$(8 - 3) / 8 = 5 / 8$$

자연어 처리 실습

정량 평가 지표 - 기계 번역기의 성능 평가 지표

N-Gram

정답 : The weather is fine and the wind is cool today.

bi-gram

The weather, weather is, is fine, fine and , ...

Tri-gram

The weather is, weather is fine, is fine and, fine and the, ...

4-gram

The weather is fine, weather is fine and, is fine and the, fine and the wind, ...

자연어 처리 실습

정량 평가 지표 - 기계 번역기의 성능 평가 지표

N-Gram

번역에 나온 단어 중 정답에 있는 **n-gram** 의 수

번역에 있는 n-gram의 수

정답 : The weather is fine and the wind is cool today.

The **weather**, **weather is**, **is so**, **so nice**, **nice and**, **and the**, **the wind**, **wind is**, **is cool**, **cool today**.

자연어 처리 실습

정량 평가 지표 - 기계 번역기의 성능 평가 지표

BLEU (Bilingual Evaluation Understudy)

$$BLEU = BP \times \exp \left(\sum_{n=1}^N w_n \log p_n \right)$$

$$BP = \begin{cases} 1, & \text{if } c > r \\ \exp(1 - r/c), & \text{if } c \leq r \end{cases}$$

Brevity Penalty: 길이가 짧을 경우 페널티

여러개의 N-Gram precision을 구하여 합산하는 지표