

A close-up photograph of a yellow pencil and a metal sharpener on a lined notebook. The sharpener is a small, silver-colored metal device with a textured surface. The pencil is yellow with a black eraser and a sharpened lead tip. Several wood shavings are scattered around the sharpener and pencil. The notebook has horizontal blue lines and a vertical red margin line. The background is slightly blurred, showing the pages of the notebook.

NLP – Word2Vec

18기 심준교

목차

1

Word Embedding

2

NNLM

3

Word2Vec

4

Negative Sampling

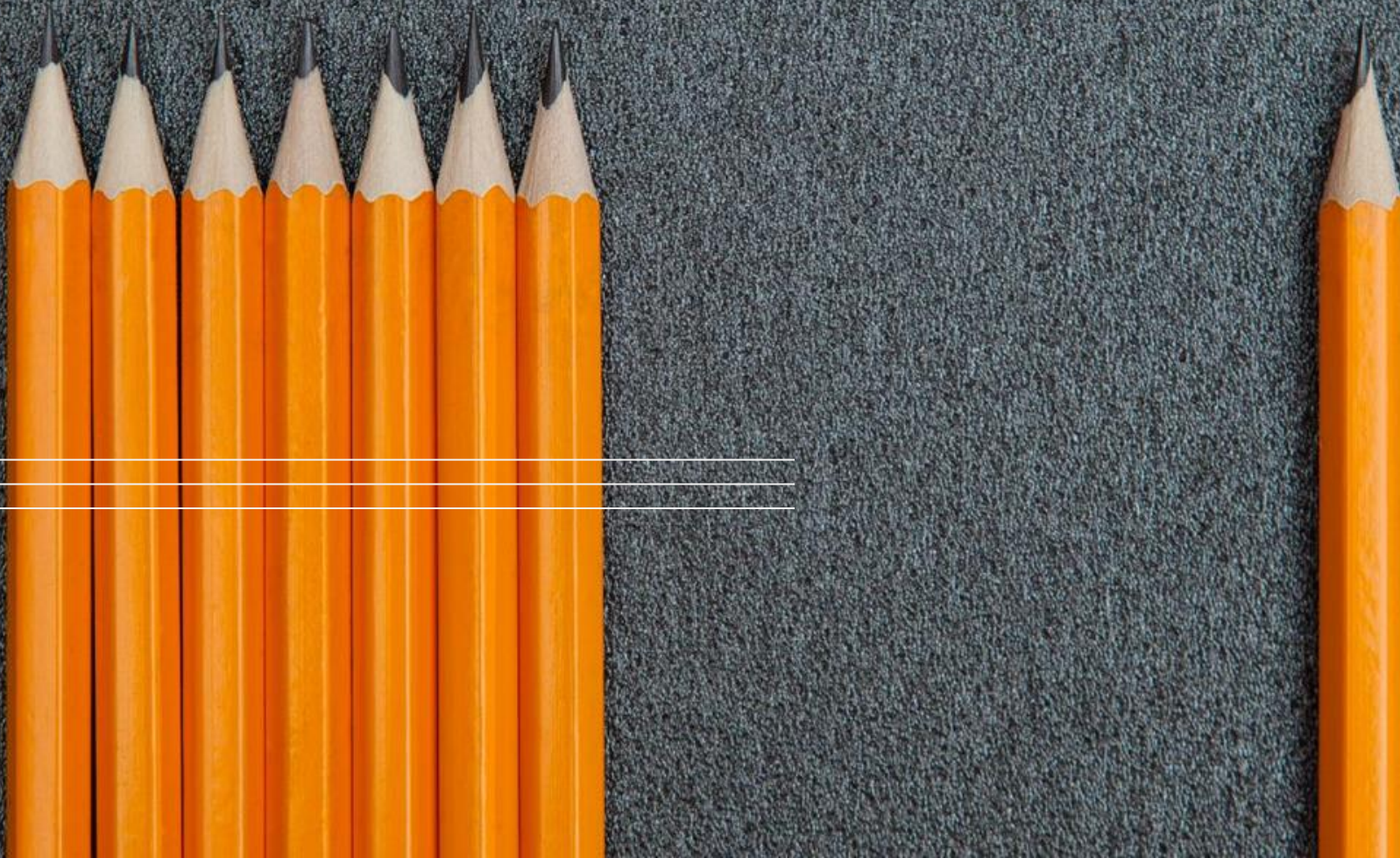
5

Glove , FastText



Part 1

Word Embedding



Word Embedding

Ex) Corpus 의 단어의 개수 : 10000
'강아지'라는 단어 표현

기존의 희소 표현(Sparse representation)

강아지 = [0,0,0,0,1,0,0,0, ... ,0]

- 공간적 낭비(10000차원의 벡터)
- 단어의 의미를 담지 못함



밀집 표현(Dense representation)

강아지 = [0.2, 1.8, 1.1, -2.1, 1.1, 2.8 ...]

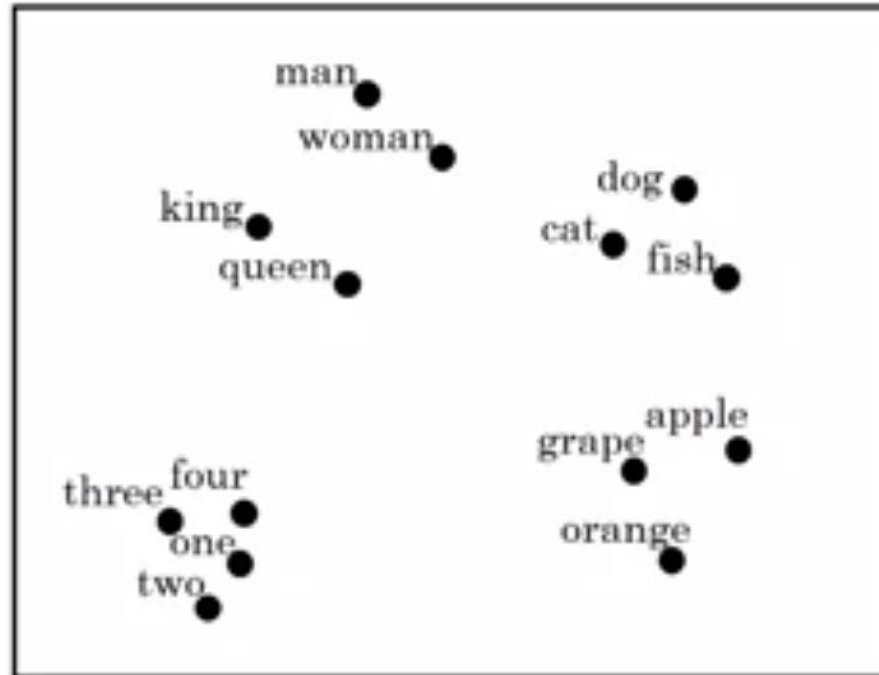
- 사용자가 지정한 차원으로 표현
- 학습을 통해 단어의 의미 표현 가능

Word Embedding : 단어를 dense vector 로 표현하는 방법
dense vector 를 embedding vector 라고도 한다

Word Embedding

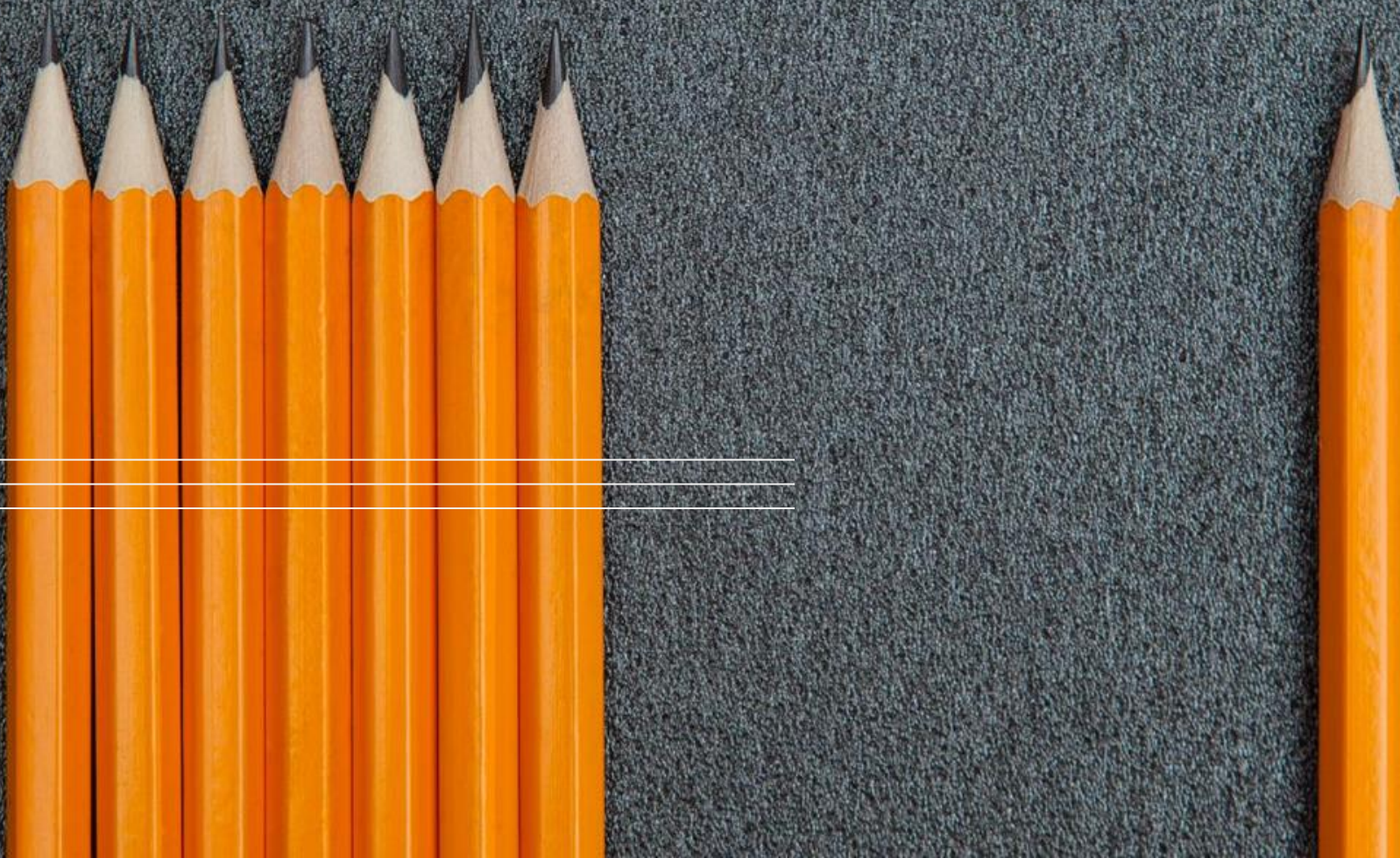
단어의 '의미' ?

Man : Woman = King : ?



Part 2

NNLM (Neural Network Language Model)



NNLM(Neural Network Language Model)

기존 언어 모델의 한계

1. 학습 데이터에 존재하지 않는 n-gram 이 포함된 문장이 나타날 확률 값을 0으로 부여한다.

Ex) An adorable little boy is spreading ____ . → 4-gram 언어 모델로 예측

훈련 corpus 에 'boy is spreading smiles' 라는 단어 시퀀스가 존재하지 않으면

$P(\text{smiles} \mid \text{boy is spreading}) = 0$ 이 되어버린다.

현실에서는 자주 사용되는 표현이기 때문에 제대로 된 모델링이라 할 수 없다.

2. n이 커질수록 등장 확률이 0인 단어 시퀀스가 기하급수적으로 늘어나 n 을 5 이상으로 설정할 수 없다. 따라서 문장의 장기 의존성을 포착하기 힘들다.

3. 단어/ 문장 간 유사도를 계산할 수 없다.

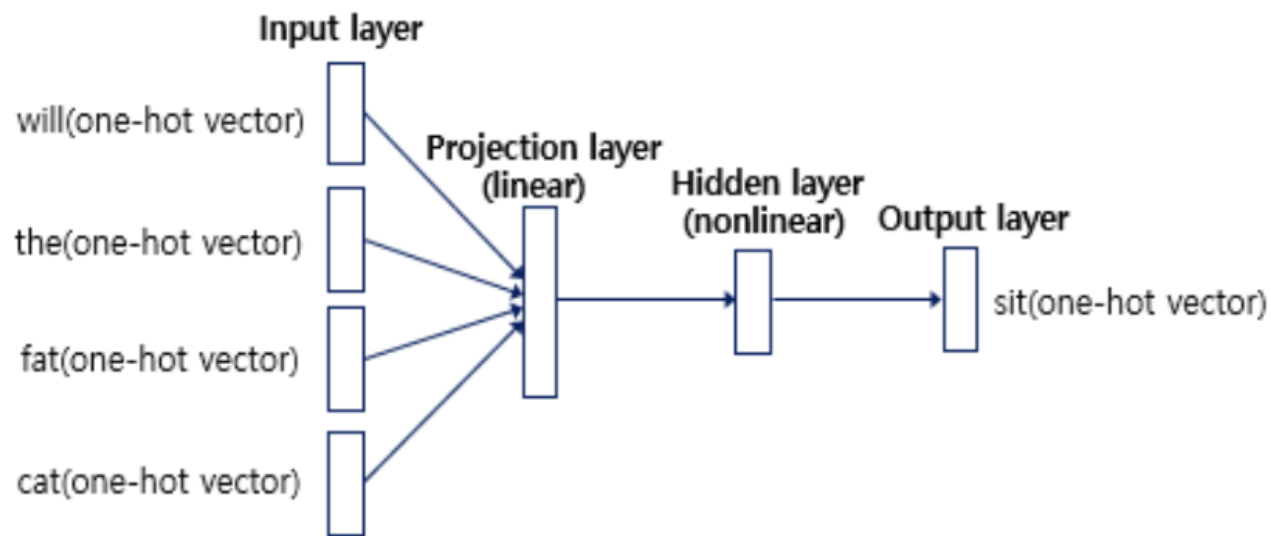
NNLM(Neural Network Language Model)

Ex) "what will the fat cat sit on"



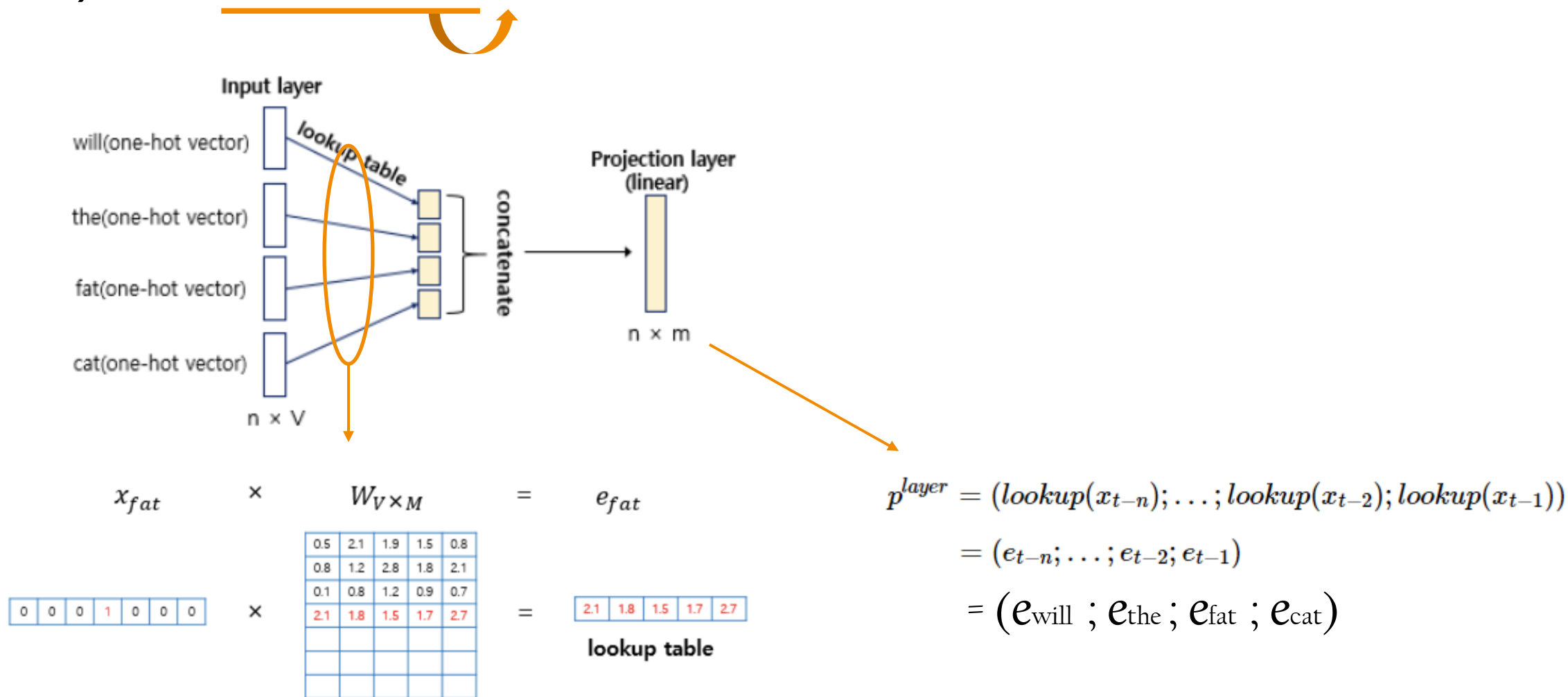
One-hot encoding (7개의 단어만 있다고 가정)

```
what = [1, 0, 0, 0, 0, 0, 0]
will = [0, 1, 0, 0, 0, 0, 0]
the  = [0, 0, 1, 0, 0, 0, 0]
fat  = [0, 0, 0, 1, 0, 0, 0]
cat  = [0, 0, 0, 0, 1, 0, 0]
sit  = [0, 0, 0, 0, 0, 1, 0]
on   = [0, 0, 0, 0, 0, 0, 1]
```



NNLM(Neural Network Language Model)

Ex) "what will the fat cat sit on"



V : 단어 집합의 크기
M : 투사층의 크기

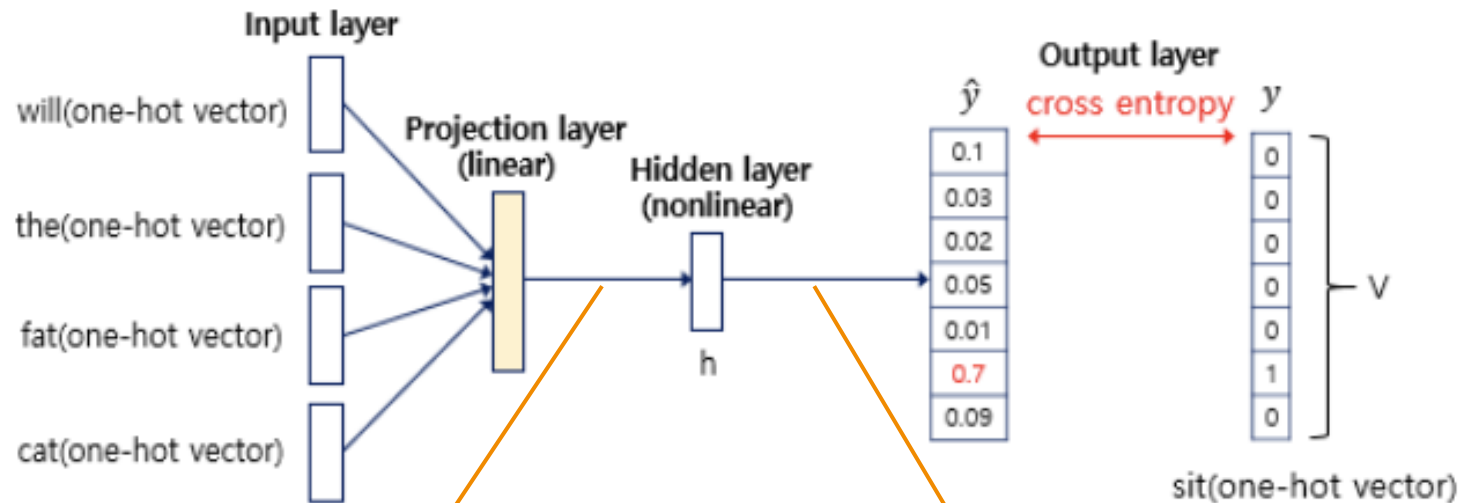
$$p^{layer} = (lookup(x_{t-n}); \dots; lookup(x_{t-2}); lookup(x_{t-1}))$$

$$= (e_{t-n}; \dots; e_{t-2}; e_{t-1})$$

$$= (e_{will} ; e_{the} ; e_{fat} ; e_{cat})$$

NNLM(Neural Network Language Model)

Ex) "what will the fat cat sit on"



$$h^{layer} = \tanh(W_h p^{layer} + b_h)$$

$$\hat{y} = \text{softmax}(W_y h^{layer} + b_y)$$

NNLM(Neural Network Language Model)

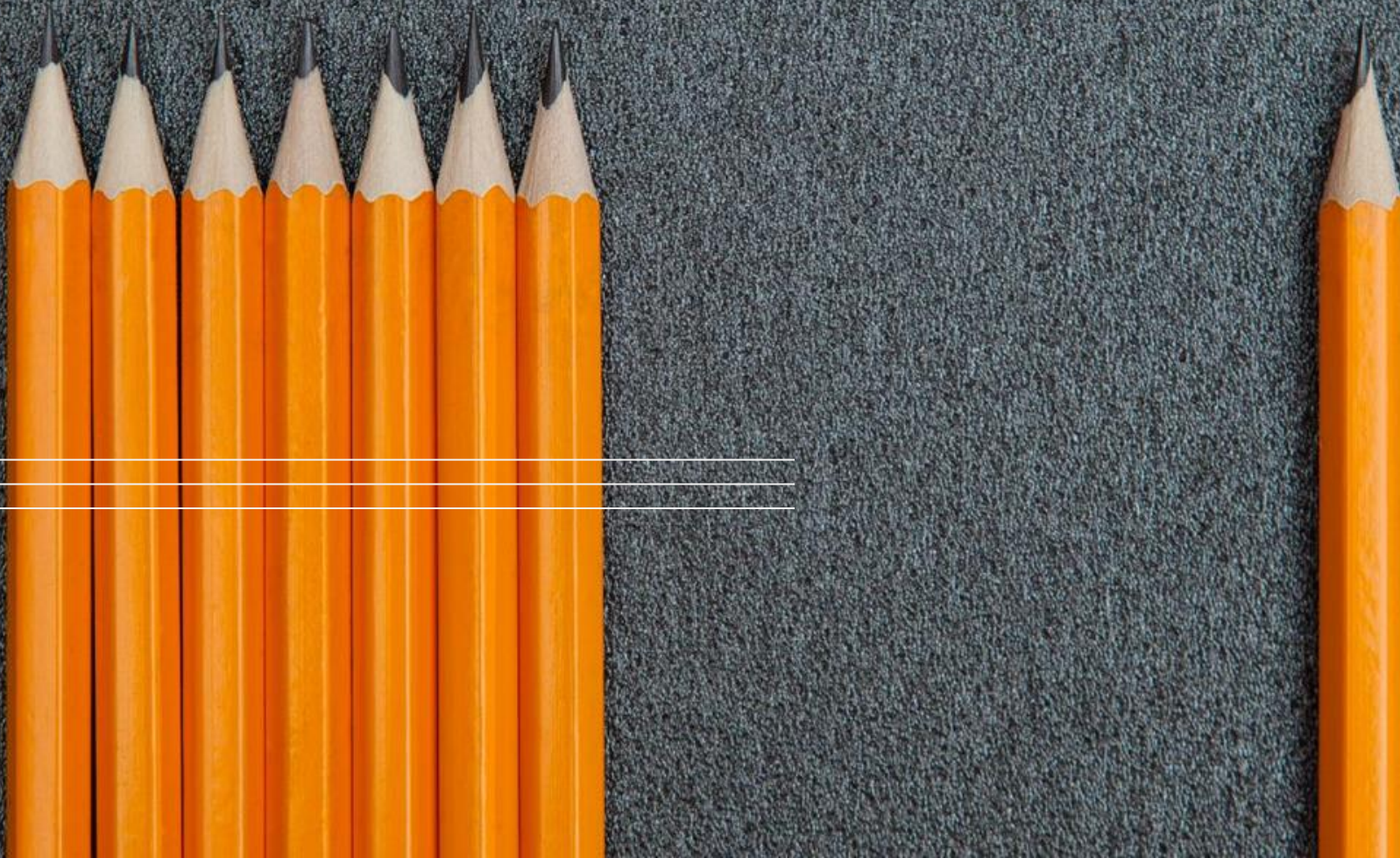
$$W_{V \times M}$$

what	0.5	2.1	1.9	1.5	0.8
will	0.8	1.2	2.8	1.8	2.1
the	0.1	0.8	1.2	0.9	0.7
fat	2.1	1.8	1.5	1.7	2.7
cat					
sit					
on					

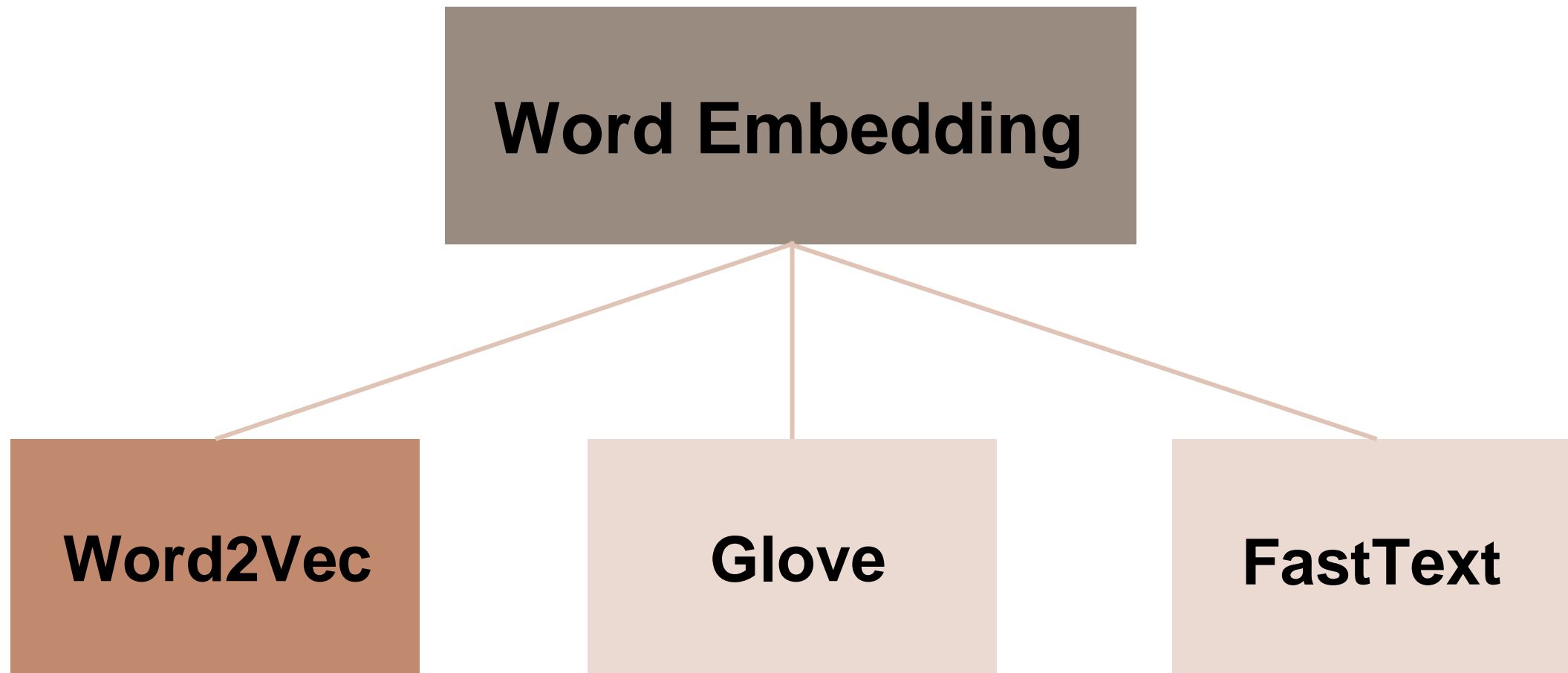
W의 각 행이 곧 단어의 M 차원 임베딩 벡터가 된다

Part 3

Word2Vec



Word2Vec




Word2Vec

Ex) "The fat cat sat on the mat"
 주변 중심 주변

CBOW

"The fat cat ? on the mat"



주변에 있는 단어로 **중심**에 있는 단어를 예측

Skip-gram

"The fat ? sat ? the mat"



중심에 있는 단어로 주변에 있는 단어를 예측

Word2Vec - CBOW

Ex) "The fat cat sat on the mat"

중심 단어
↓
주변 단어

The fat cat sat on the mat

The fat cat sat on the mat

The fat cat sat on the mat

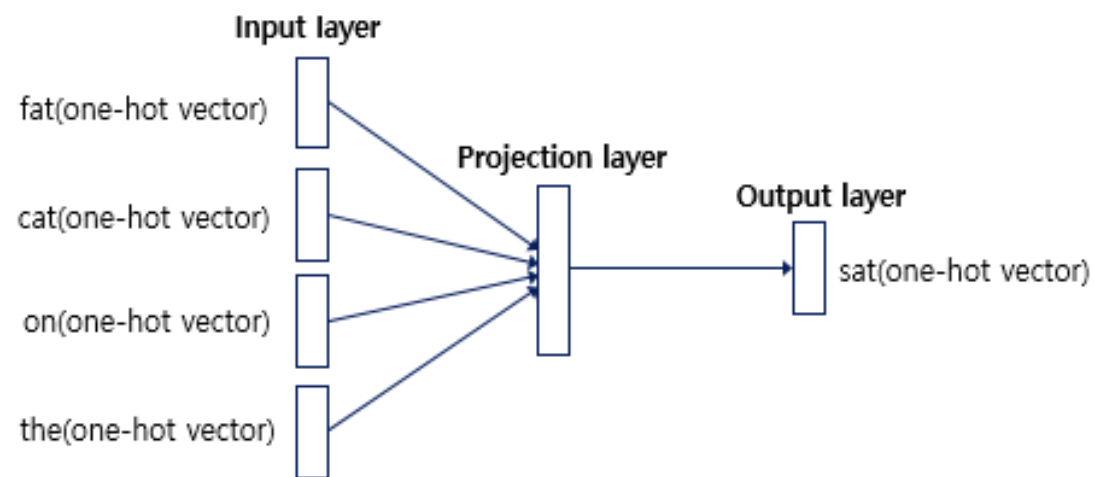
The fat cat sat on the mat

The fat cat sat on the mat

The fat cat sat on the mat

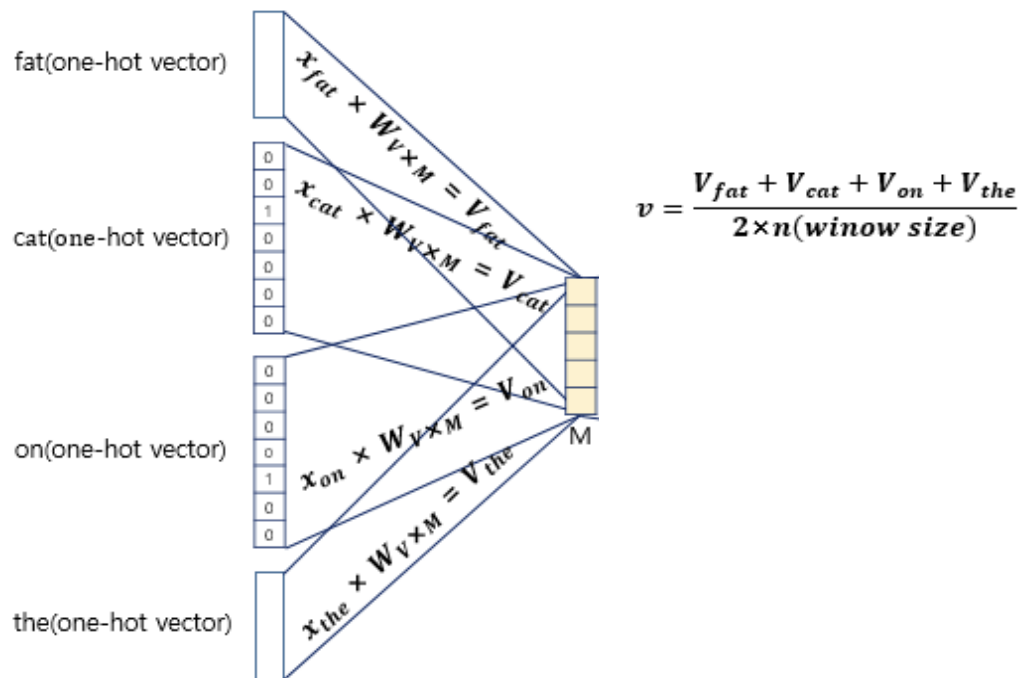
The fat cat sat on the mat

중심 단어	주변 단어
[1, 0, 0, 0, 0, 0, 0]	[0, 1, 0, 0, 0, 0, 0], [0, 0, 1, 0, 0, 0, 0]
[0, 1, 0, 0, 0, 0, 0]	[1, 0, 0, 0, 0, 0, 0], [0, 0, 1, 0, 0, 0, 0], [0, 0, 0, 1, 0, 0, 0]
[0, 0, 1, 0, 0, 0, 0]	[1, 0, 0, 0, 0, 0, 0], [0, 1, 0, 0, 0, 0, 0], [0, 0, 0, 1, 0, 0, 0], [0, 0, 0, 0, 1, 0, 0]
[0, 0, 0, 1, 0, 0, 0]	[0, 1, 0, 0, 0, 0, 0], [0, 0, 1, 0, 0, 0, 0], [0, 0, 0, 0, 1, 0, 0], [0, 0, 0, 0, 0, 1, 0]
[0, 0, 0, 0, 1, 0, 0]	[0, 0, 1, 0, 0, 0, 0], [0, 0, 0, 1, 0, 0, 0], [0, 0, 0, 0, 0, 1, 0], [0, 0, 0, 0, 0, 0, 1]
[0, 0, 0, 0, 0, 1, 0]	[0, 0, 0, 1, 0, 0, 0], [0, 0, 0, 0, 1, 0, 0], [0, 0, 0, 0, 0, 0, 1]
[0, 0, 0, 0, 0, 0, 1]	[0, 0, 0, 0, 1, 0, 0], [0, 0, 0, 0, 0, 1, 0]

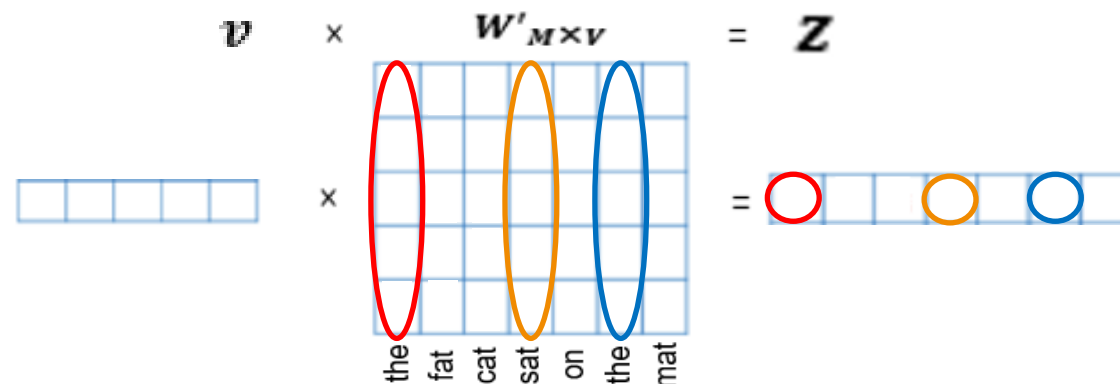
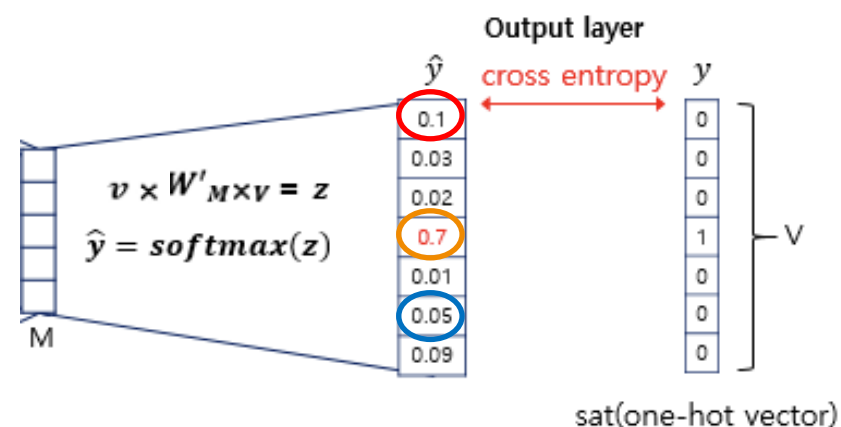


Word2Vec - CBOW

Ex) "The fat cat sat on the mat"

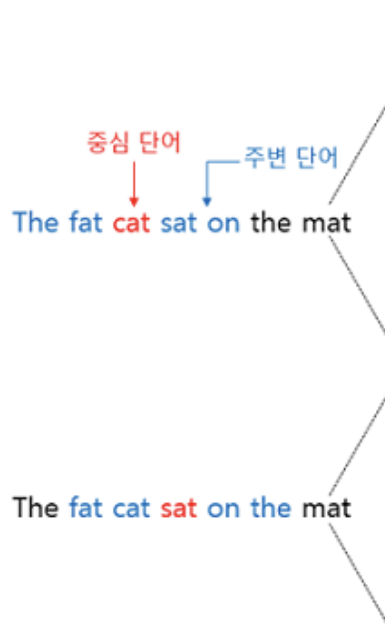


Loss function $H(\hat{y}, y) = - \sum_{j=1}^{|V|} y_j \log(\hat{y}_j)$

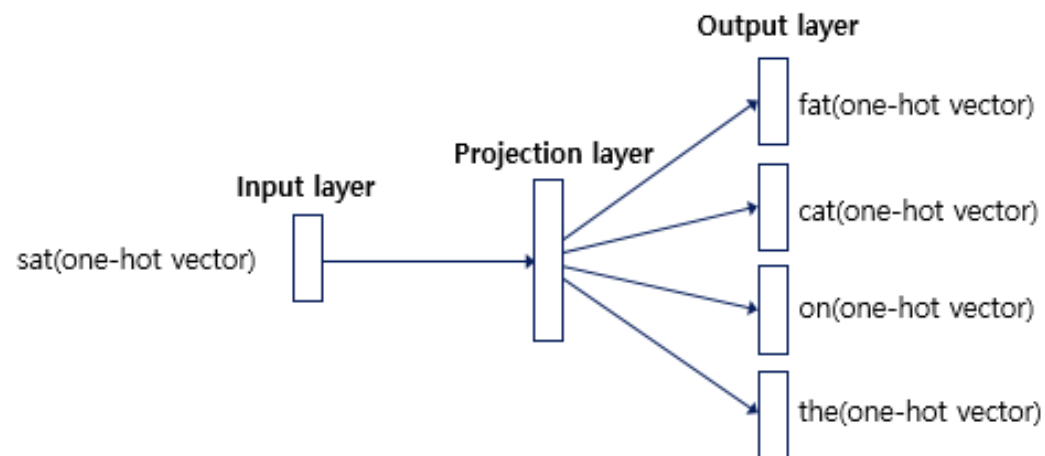


Word2Vec - Skip-gram

Ex) "The fat cat sat on the mat"

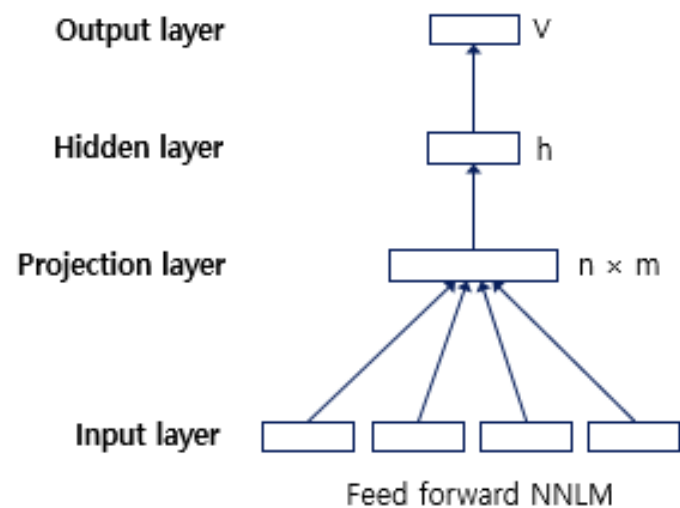


중심 단어	주변 단어
cat	The
cat	Fat
cat	sat
cat	on
sat	fat
sat	cat
sat	on
sat	the

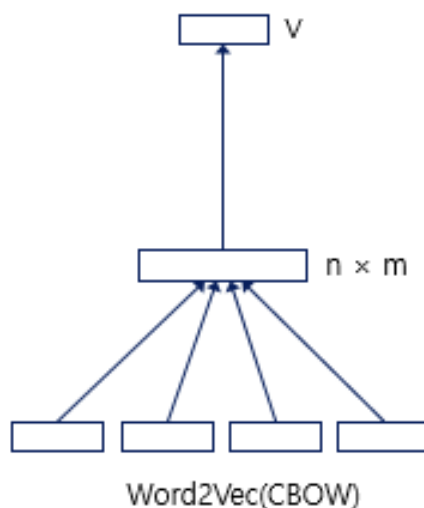


4번의 backpropagation → 가중치 업데이트 횟수가 많아 성능이 더 좋다

NNLM vs Word2Vec



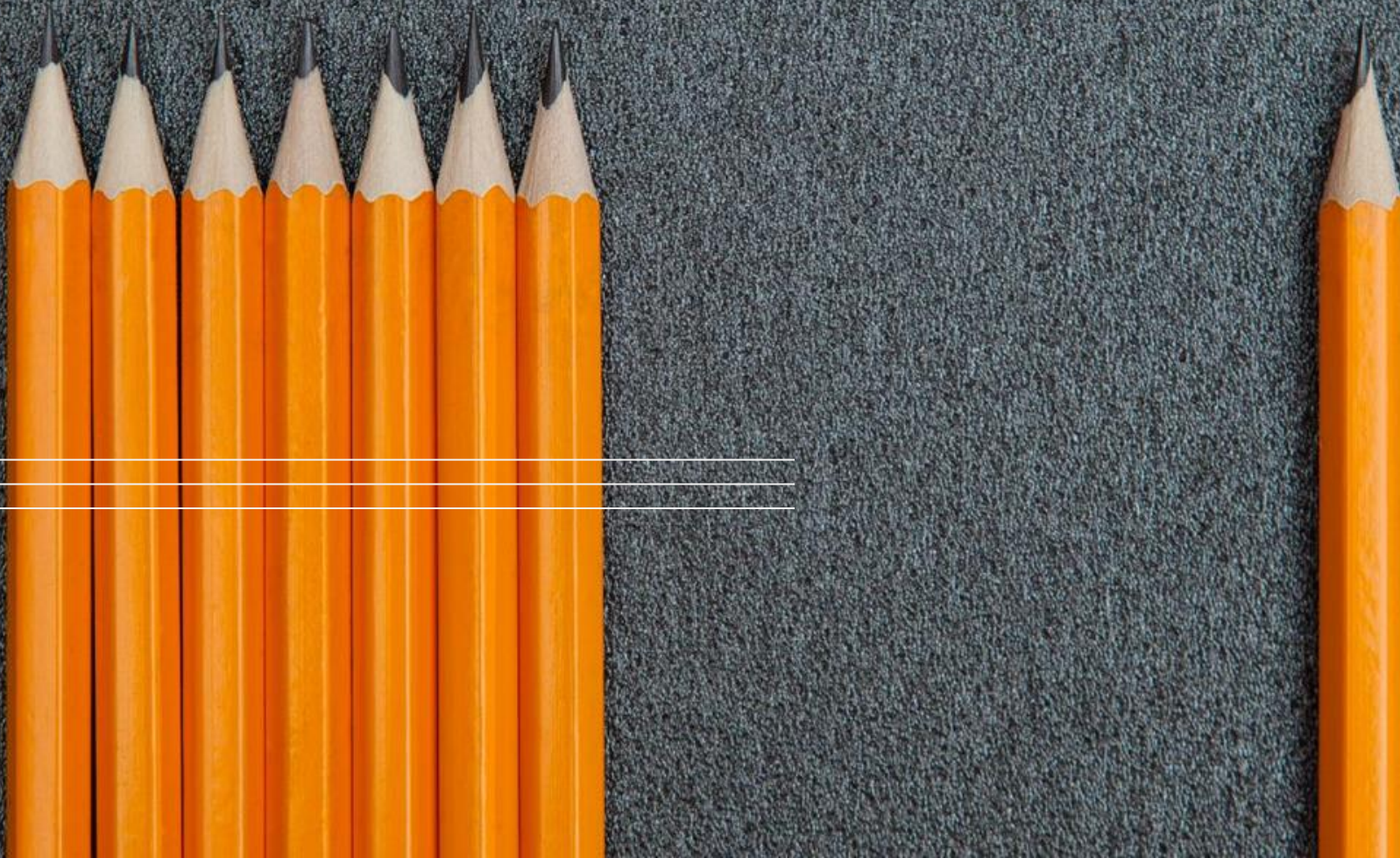
- 다음 단어를 예측하는 것이 목적인 '언어 모델'
- 예측 단어의 이전 단어만을 참고
- 연산량 : $(n \times m) + (n \times m \times h) + (h \times V)$



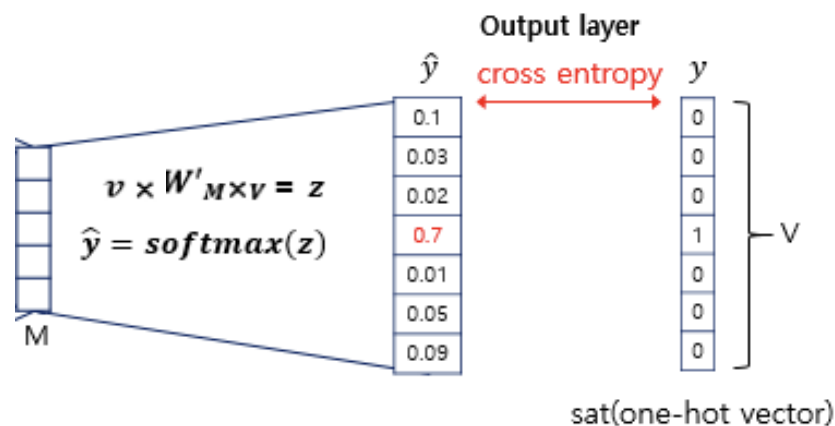
- 워드 임베딩 자체가 목적
- 예측 단어의 주변 단어를 모두 참고
- 연산량 : $(n \times m) + (m \times \log(V))$

Part 4

Negative Sampling



Negative Sampling



기본적인 Word2Vec : 출력층에서 모든 단어 집합에 대한 임베딩 벡터 값 업데이트

→ 학습하고 있는 중심, 주변 단어 : ‘강아지’, ‘고양이’, ‘귀여운’ 일 때, ‘돈가스’, ‘컴퓨터’ 와 같은 연관관계가 적은 수많은 단어의 벡터값도 업데이트

→ 비효율적, softmax 적용하기 때문에 연산량 ↑

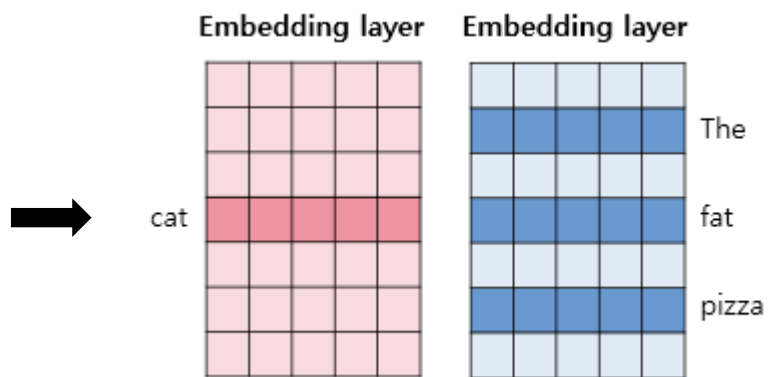


추가하는 단어 수 : 2 ~ 5 in large dataset
5 ~ 20 in small dataset

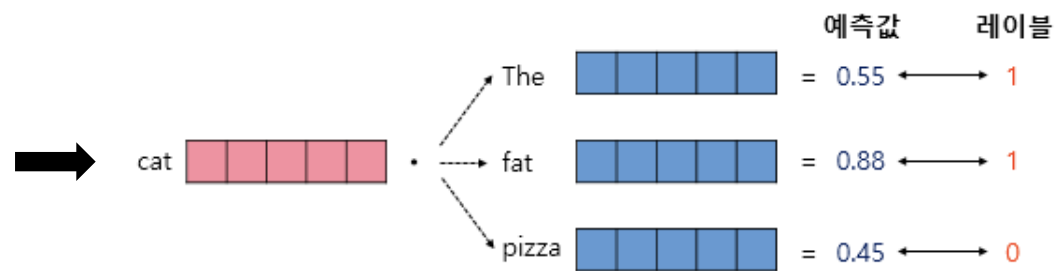
Negative Sampling

X		Y
입력1	입력2	레이블
cat	The	1
cat	fat	1
cat	pizza	0
cat	computer	0
cat	sat	1
cat	on	1
sat	fat	1
sat	cat	1
...

임베딩 벡터로 변환

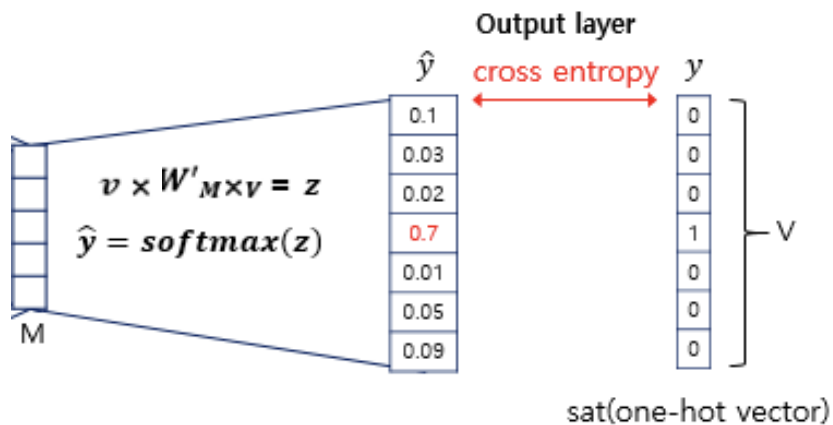


임베딩 벡터들의 내적값에 sigmoid 적용



Negative Sampling

전체 단어 집합 크기만큼의
Softmax classification



VS.

입력1과 입력2가 서로 중심단어와 주변단어의
관계면 1, 아니면 0 인 binary classification



Negative Sampling

단어 w_i 를 negative sample 로 선택할 확률

$$P_{negative}(w_i) = \frac{U(w_i)^{3/4}}{\sum_{j=0}^n U(w_j)^{3/4}}$$

$U(w_j)$: 전체 corpus 에서 w_i 의 비율

n : 전체 corpus 의 단어 개수

Ex) 전체 단어가 ‘강아지’, ‘고양이’ 2개이고
그 비율이 99 : 1인 경우

$$P(\text{강아지}) = \frac{0.99^{0.75}}{0.99^{0.75} + 0.01^{0.75}} = 0.97$$

$$P(\text{고양이}) = \frac{0.01^{0.75}}{0.99^{0.75} + 0.01^{0.75}} = 0.03$$

Subsampling

Negative Sampling 과는 별개의 방법

입력1	입력2	레이블
cat	The	1
cat	fat	1
cat	pizza	0
cat	computer	0
cat	sat	1
cat	on	1
...

학습되는 데이터 쌍이 매우 많다

→ 자주 등장하는 단어는 학습에서 제외시키자

$$P_{\text{subsampling}}(w_i) = 1 - \sqrt{\frac{t}{f(w_i)}} \quad , t = 10^{-5}$$

$f(w_i) = 0.01$ 인 경우 $P_{\text{subsampling}}(w_i) = 0.9684$

→ 해당 단어가 가질 수 있는 100번의 학습 기회 중
96번 정도를 제외

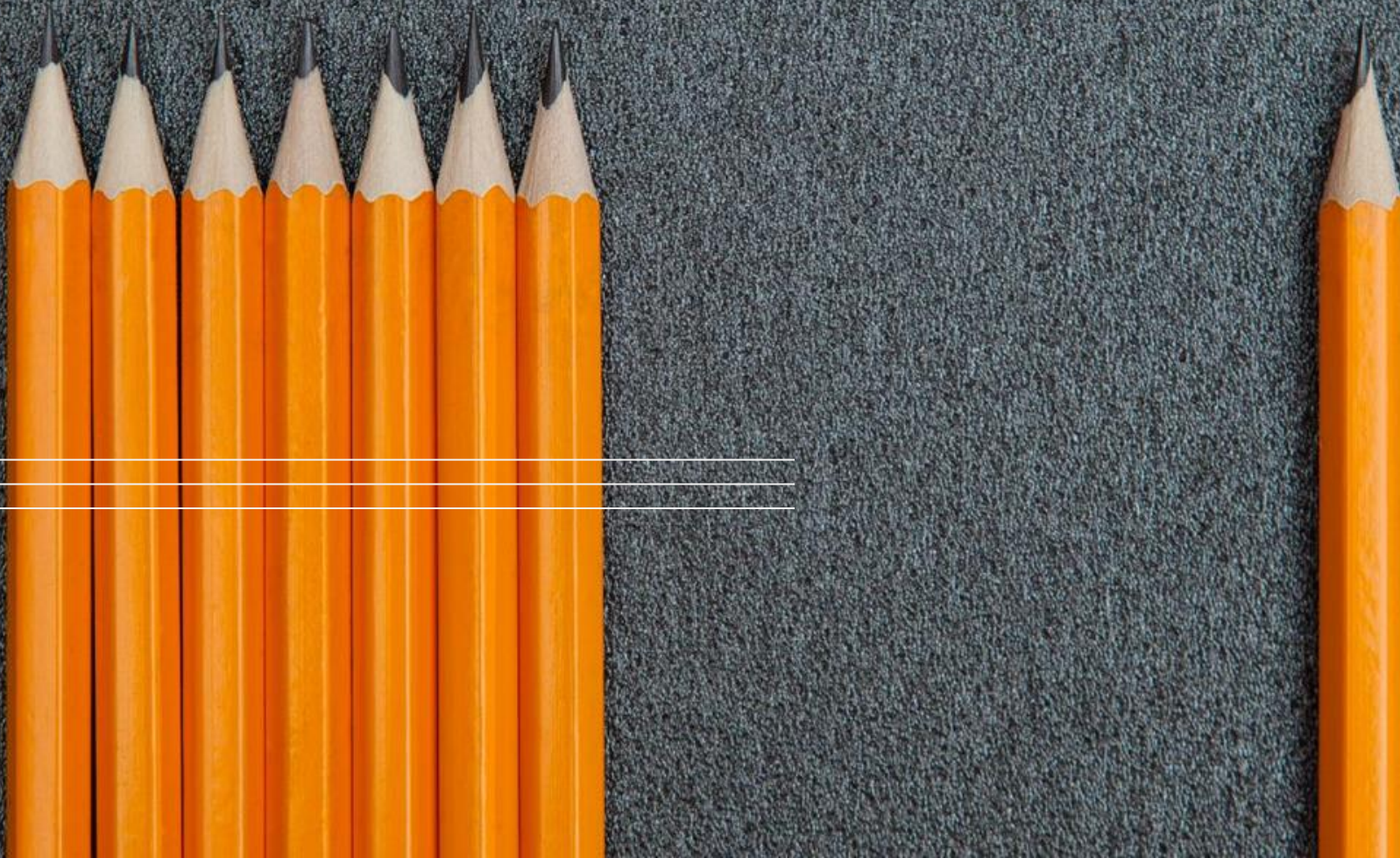
$f(w_i) = 10^{-5}$ 인 경우 $P_{\text{subsampling}}(w_i) = 0$

→ 등장하면 무조건 학습

「
실습!
」

Part 5

Glove, FastText



Glove, FastText

Word2Vec

Glove

FastText

2013년, 구글

예측 기반(실제값과 예측값의
오차 최소화)

<https://drive.google.com/file/d/0B7XkCwpI5KDYNINUTTISS21pQmM>

2014년, 스탠포드

Word2Vec(예측 기반)
+
LSA(카운트 기반)

<https://nlp.stanford.edu/projects/glove/>

2016년, 페이스북

하나의 단어 안에 여러
단어(Subword)가 존재한다고 간주

Ex) 'birthplace' , 'apple'
학습 가능

<https://fasttext.cc/>



감사합니다 !!