워드 임베딩

23.05.25 유하영

언어 모델

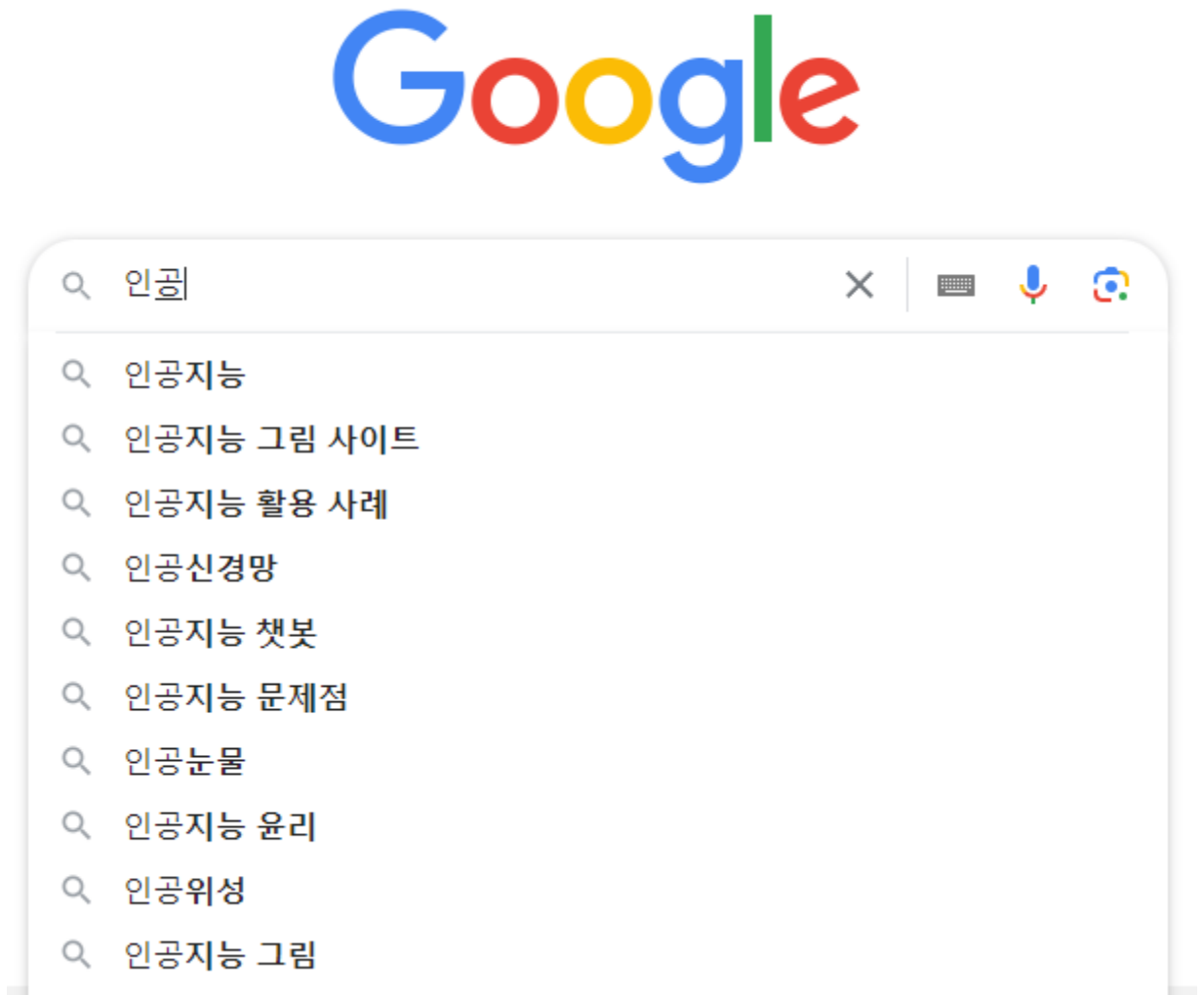
대규모의 텍스트 데이터를 학습하여
단어 간의 확률 관계를 파악

단어 간의 관계, 문장 구조, 문법 규칙 등을 학습

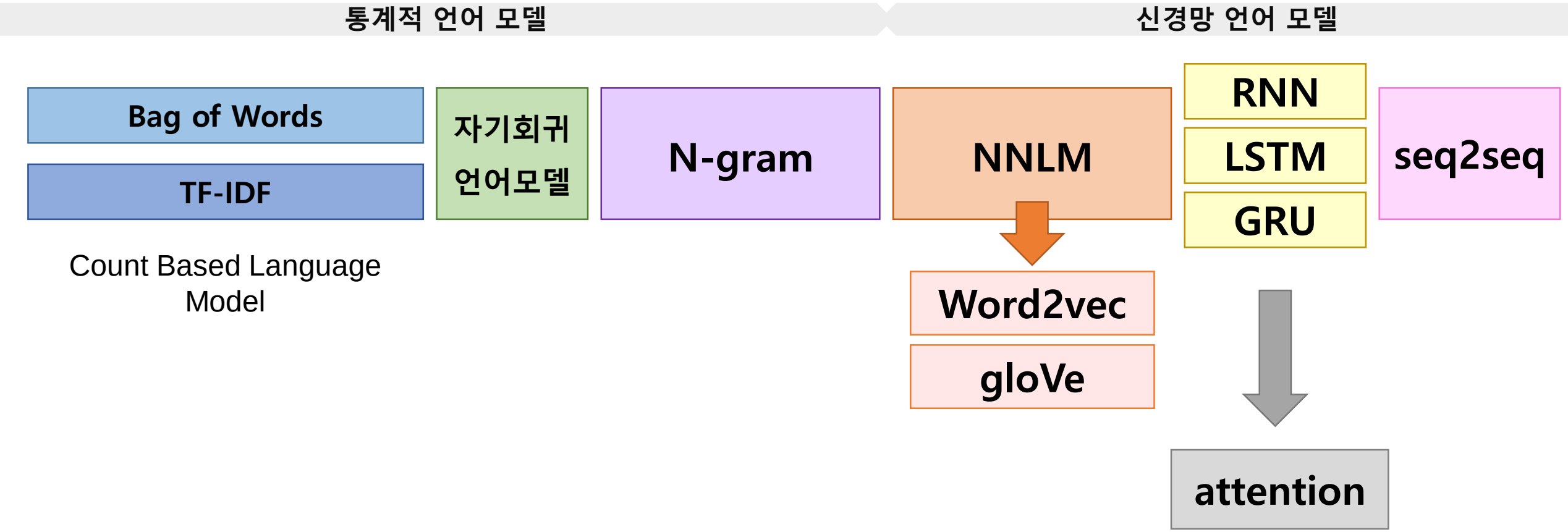
텍스트 데이터
"**이해**" + "**생성**"

1) 단어 생성
모델에 문장이나 단어 제공 시
다음 단어를 예측하여 새로운 텍스트 생성

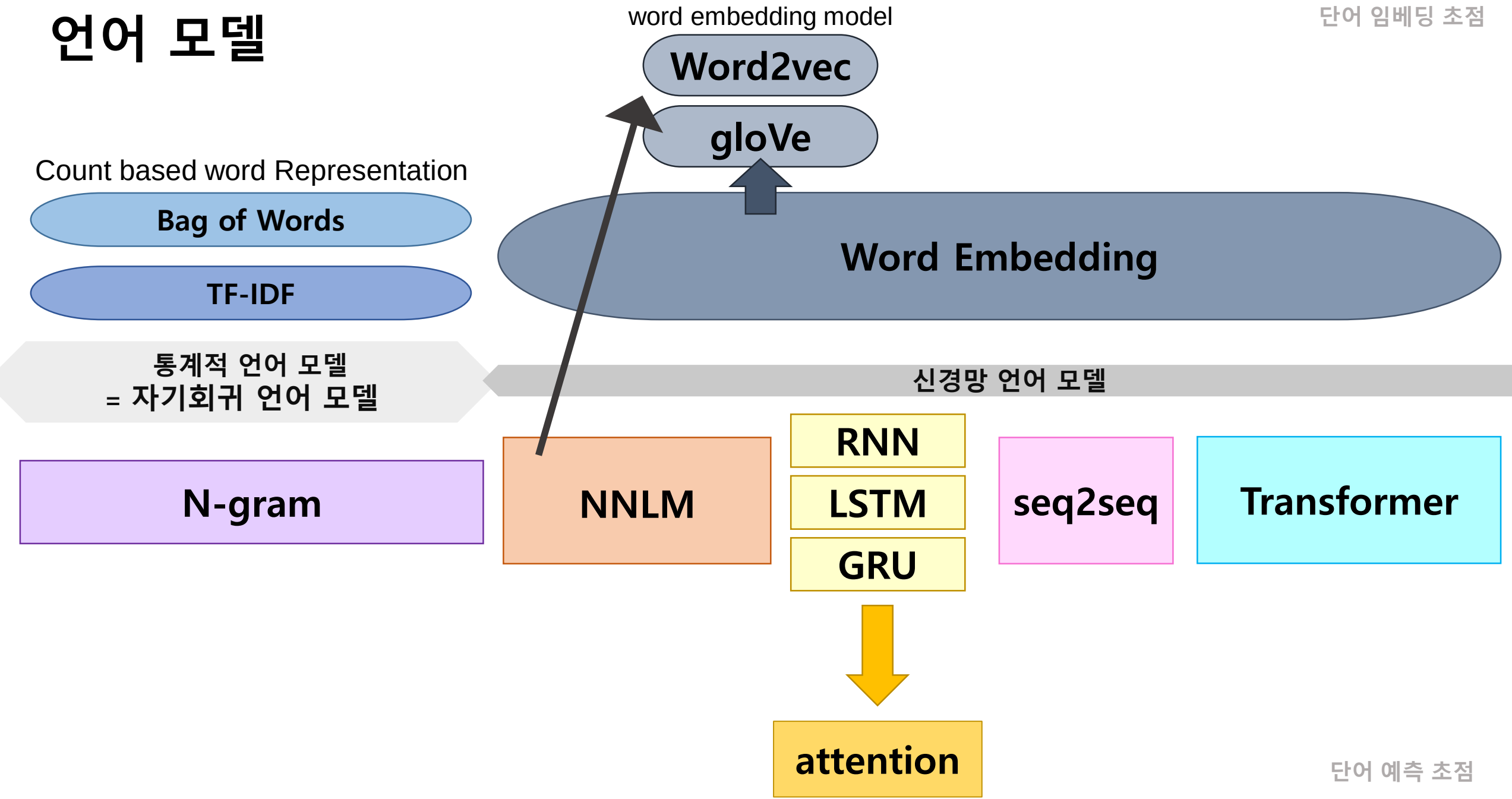
2) 문장 생성
문맥이나 조건에 기반하여 **텍스트 생성**
(질의응답, 대화형 챗봇)


Google
인공
인공지능
인공지능 그림 사이트
인공지능 활용 사례
인공신경망
인공지능 챗봇
인공지능 문제점
인공눈물
인공지능 윤리
인공위성
인공지능 그림

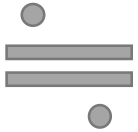
언어 모델


통계적 언어 모델
신경망 언어 모델
Bag of Words
TF-IDF
Count Based Language Model
자기회귀 언어모델
N-gram
NNLM
Word2vec
gloVe
RNN
LSTM
GRU
seq2seq
attention

언어 모델


word embedding model
Word2vec
gloVe
단어 임베딩 초점
Count based word Representation
Bag of Words
TF-IDF
Word Embedding
통계적 언어 모델
= 자기회귀 언어 모델
신경망 언어 모델
N-gram
NNLM
RNN
LSTM
GRU
seq2seq
Transformer
attention
단어 예측 초점

통계적 언어 모델



자기회귀 LM

문장이나 문서의 확률을 추정

이전에 생성된 단어들을 참고하여
다음 단어를 예측하는 언어 모델

= 통계적인 방식으로 문장의 확률을 모델링

but 모든 언어 모델이 자기회귀인건 아니다
(BERT)

자기회귀 LM

이전에 등장한 모든 단어 고려

$$P(w_1, w_2, w_3, w_4, w_5, \dots w_n) = \prod_{n=1}^{n} P(w_n | w_1, \dots, w_{n-1})$$

$$P(x_1, x_2, x_3 \dots x_n) = P(x_1)P(x_2|x_1)P(x_3|x_1, x_2) \dots P(x_n|x_1 \dots x_{n-1})$$

확률을 차례로 곱해나감

N-gram LM

연속된 일부 단어만 고려

$$P(\boldsymbol{x}^{(t+1)} | \boldsymbol{x}^{(t)}, \dots, \boldsymbol{x}^{(1)}) = \frac{\text{count}(\boldsymbol{x}^{(t+1)}, \boldsymbol{x}^{(t)}, \dots, \boldsymbol{x}^{(t-n+2)})}{\text{count}(\boldsymbol{x}^{(t)}, \dots, \boldsymbol{x}^{(t-n+2)})}$$

등장횟수 카운트

오늘 점심은 토마토 w

자기회귀 LM

P(w|오늘 점심은 토마토) =

P(오늘) X P(점심은|오늘)
X P(토마토|오늘 점심은) X P(w|오늘 점심은 토마토)

$$P(w_1, w_2, w_3, w_4, w_5, \dots w_n) = \prod_{n=1}^{n} P(w_n | w_1, \dots, w_{n-1})$$

$$P(x_1, x_2, x_3 \dots x_n) = P(x_1)P(x_2|x_1)P(x_3|x_1, x_2) \dots P(x_n|x_1 \dots x_{n-1})$$

확률을 차례로 곱해나감

N-gram LM

(n=3) P(w|점심은 토마토) =

$$\frac{count(\text{점심은 토마토 } w)}{count(\text{점심은 토마토})}$$

$$P(\boldsymbol{x}^{(t+1)} | \boldsymbol{x}^{(t)}, \dots, \boldsymbol{x}^{(1)}) = \frac{\text{count}(\boldsymbol{x}^{(t+1)}, \boldsymbol{x}^{(t)}, \dots, \boldsymbol{x}^{(t-n+2)})}{\text{count}(\boldsymbol{x}^{(t)}, \dots, \boldsymbol{x}^{(t-n+2)})}$$

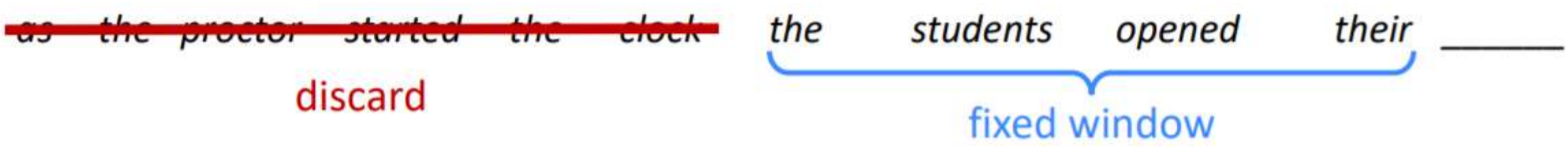
등장횟수 카운트

희소 문제, 단어 간 유사도 파악X

고차원 벡터의 사용 문제

NNLM(Neural Network LM)

정해진 n개의 단어만을 참고하여 다음 단어를 예측



- input : 이전 단어들의 시퀀스
- output : 다음 단어에 대한 확률 분포-> 단어 선택 및 생성

~~what~~ will the fat cat sit …

sit = [0, 0, 0, 0, 1, 0] (window size(N) = 4)

Input layer

Projection layer (피드 포워드 신경망)

will [1, 0, 0, 0, 0, 0]

the [0, 1, 0, 0, 0, 0]

fat [0, 0, 1, 0, 0, 0]

cat [0, 0, 0, 1, 0, 0]

단어 임베딩

x_{will}

[1, 0, 0, 0, 0, 0]

$\times$

$W_{V \times M}$

2.1	1.8	1.5	1.7	2.7
0.1	0.8	1.3	2.7	1.1
		.		
		.		
		.		

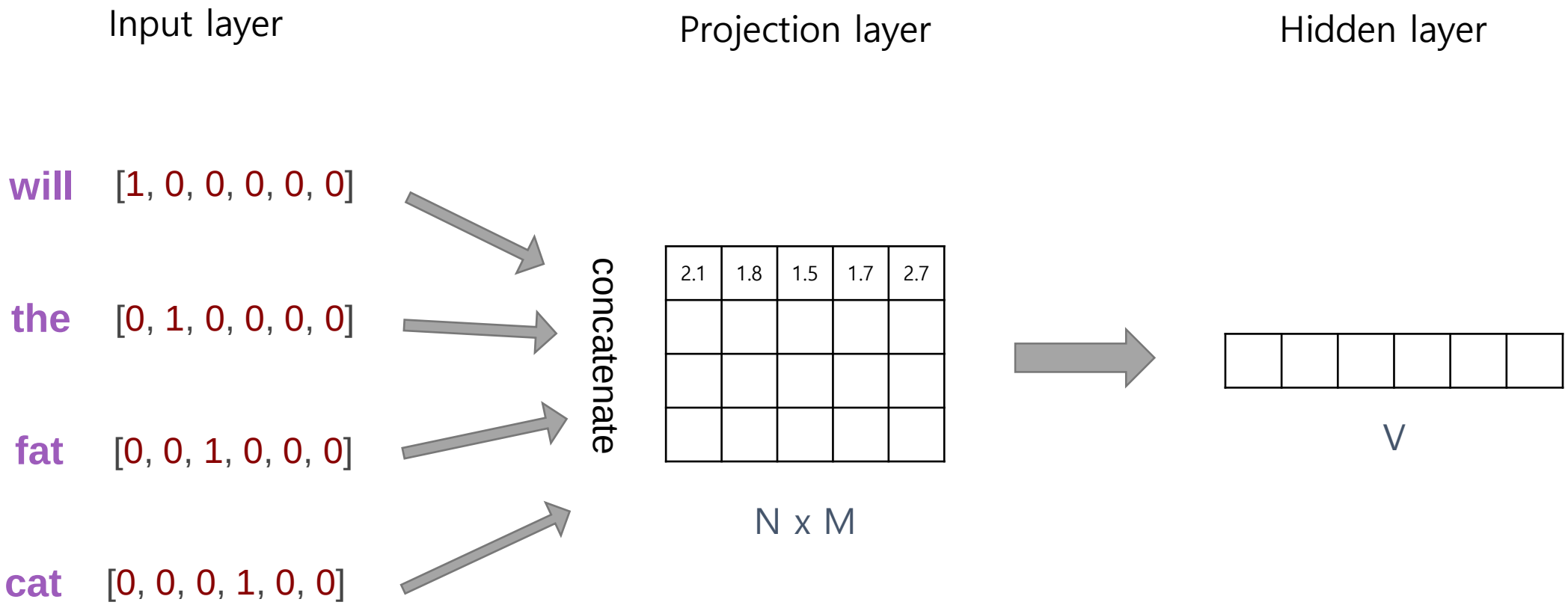
$=$

e_{will}

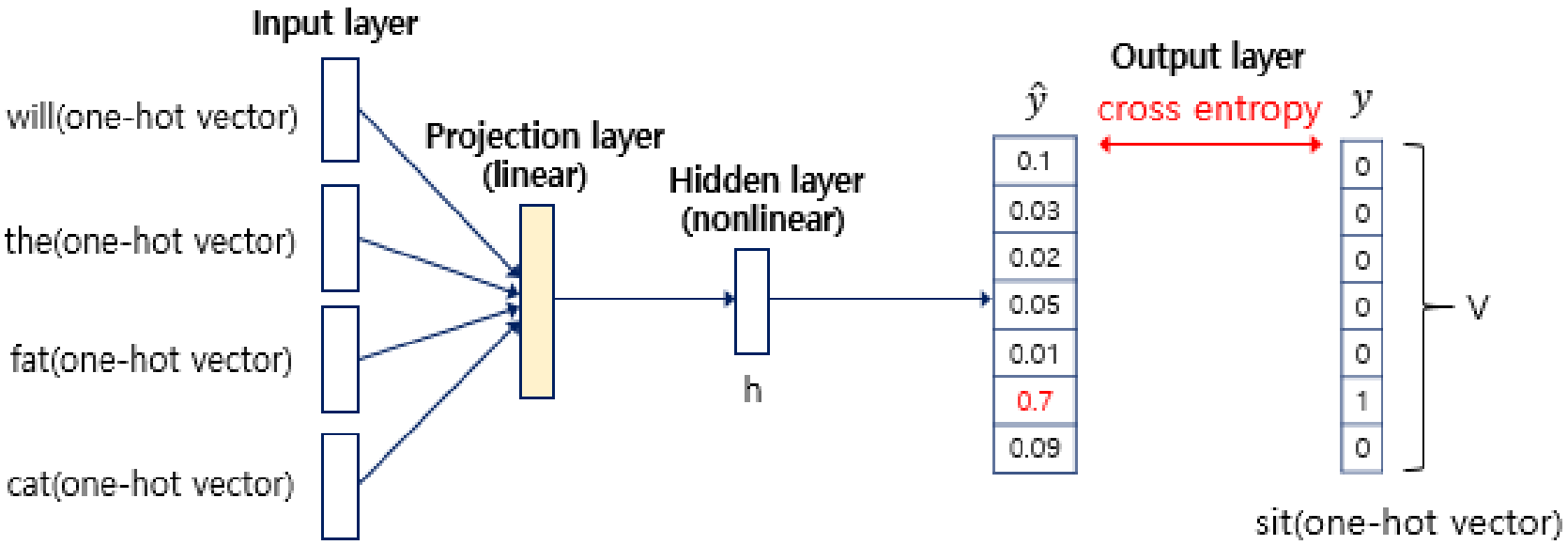
2.1	1.8	1.5	1.7	2.7

희소 표현 → 밀집 표현

신경망 언어 모델


Input layer
Projection layer
Hidden layer
will
[1, 0, 0, 0, 0, 0]
the
[0, 1, 0, 0, 0, 0]
fat
[0, 0, 1, 0, 0, 0]
cat
[0, 0, 0, 1, 0, 0]
concatenate
2.1
1.8
1.5
1.7
2.7
N x M
V

신경망 언어 모델


Input layer
will(one-hot vector)
the(one-hot vector)
fat(one-hot vector)
cat(one-hot vector)
Projection layer
(linear)
Hidden layer
(nonlinear)
h
Output layer
$\hat{y}$
cross entropy
y
0.1
0.03
0.02
0.05
0.01
0.7
0.09
0
0
0
0
0
1
0
V
sit(one-hot vector)

word embedding

: 단어 간의 관계를 학습해 vector에 저장하는 기법

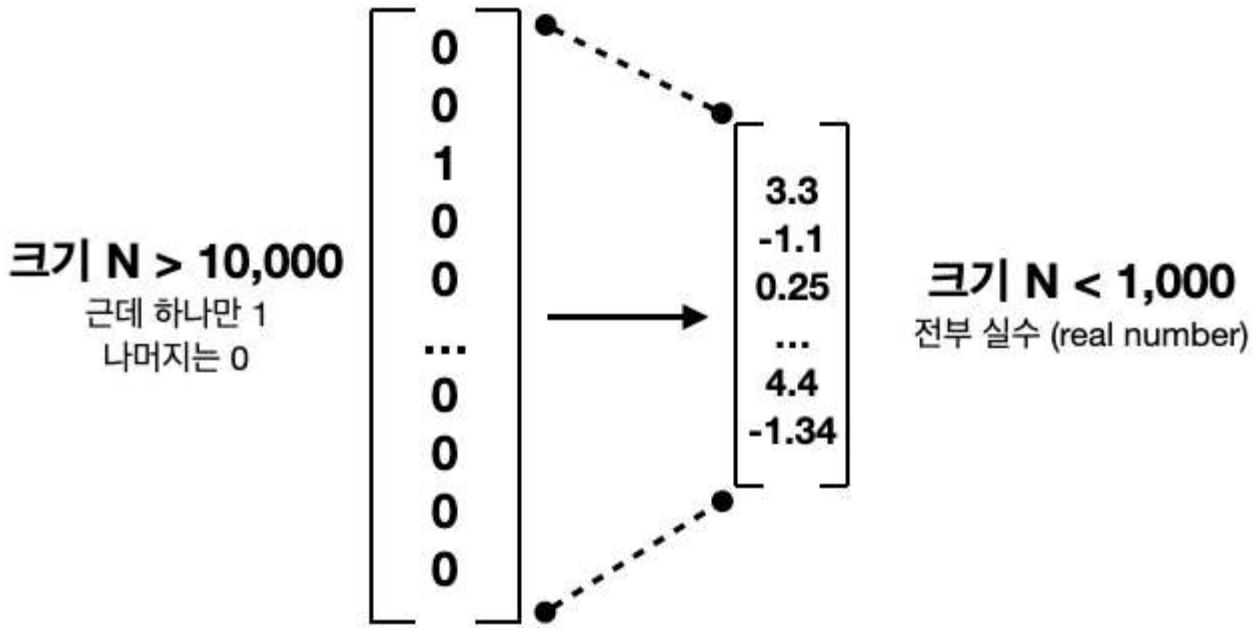
word embedding

강아지 = [0 0 0 0 1 0 0 0 0 0 0 0 ... 중략 … 0] **희소 표현**

차원은 10,000, 0은 9,999
(공간낭비, 유사도 반영 X)

강아지 = [0.2 1.8 1.1 -2.1 1.1 2.8 ... 중략 … 0.2] **밀집 표현**

사용자가 설정한 값이 벡터의 차원



word2vec

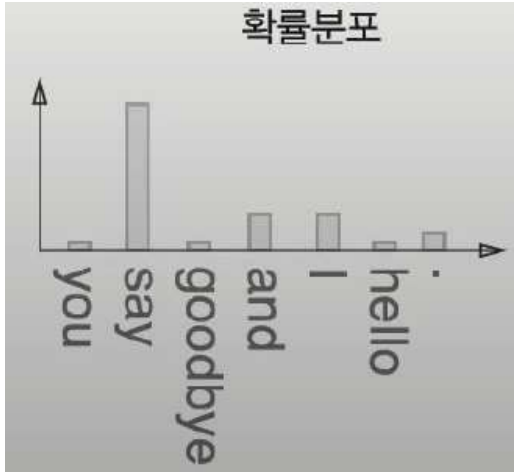
word2vec

CBOW

you ? goodbye and I say hello.

Skip-gram

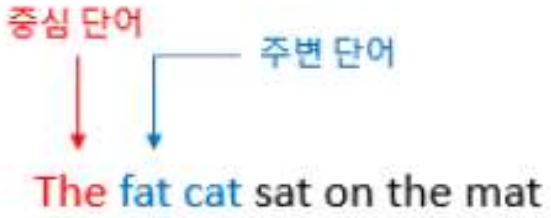
? say ? and I say hello.



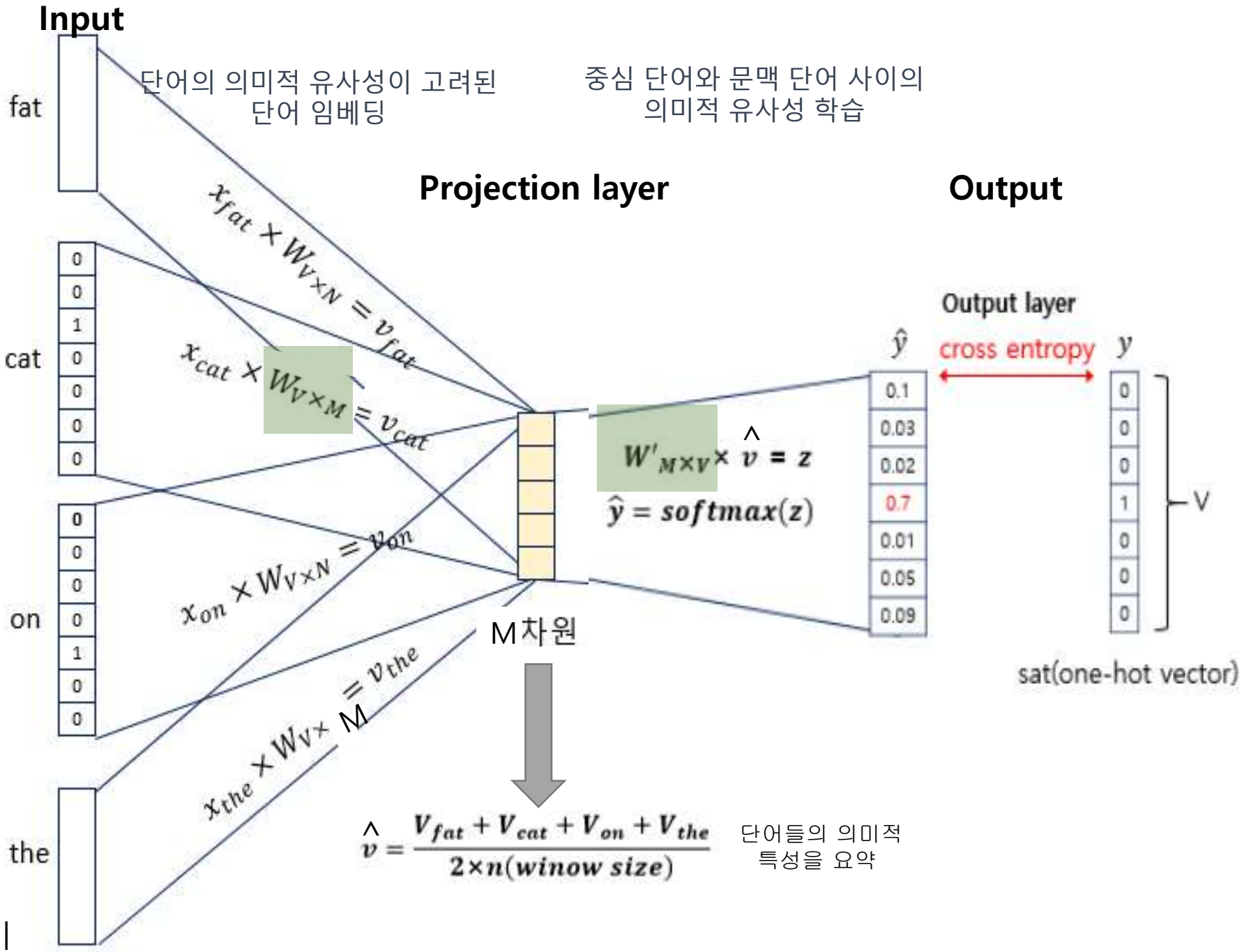
말뭉치 후보군 중에서 하나를 예측

CBOW

(window size = 2)

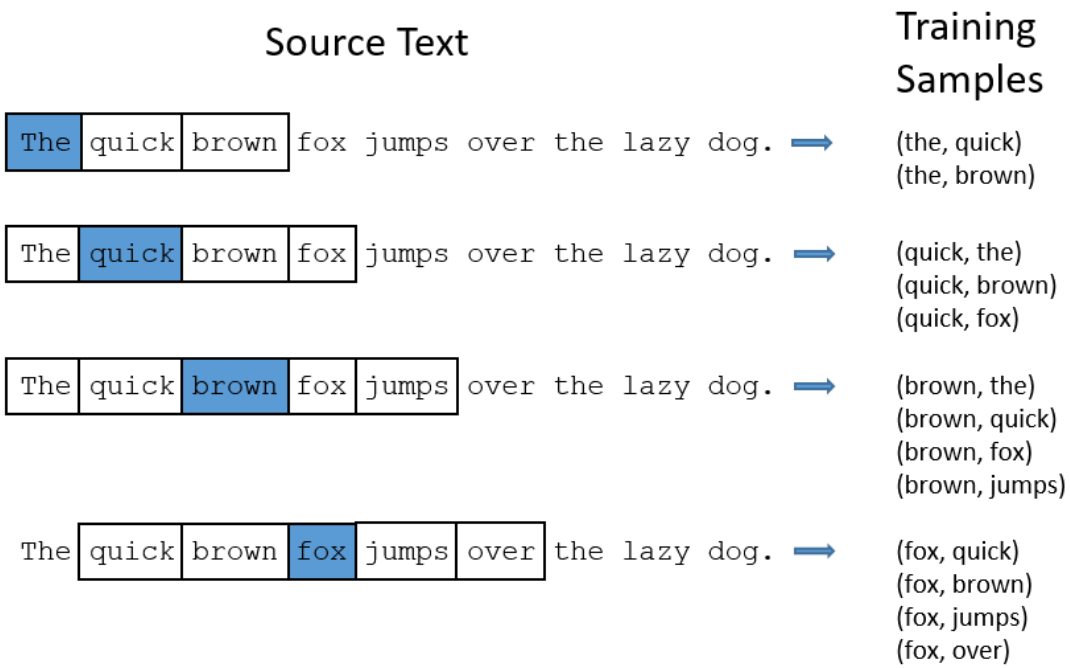


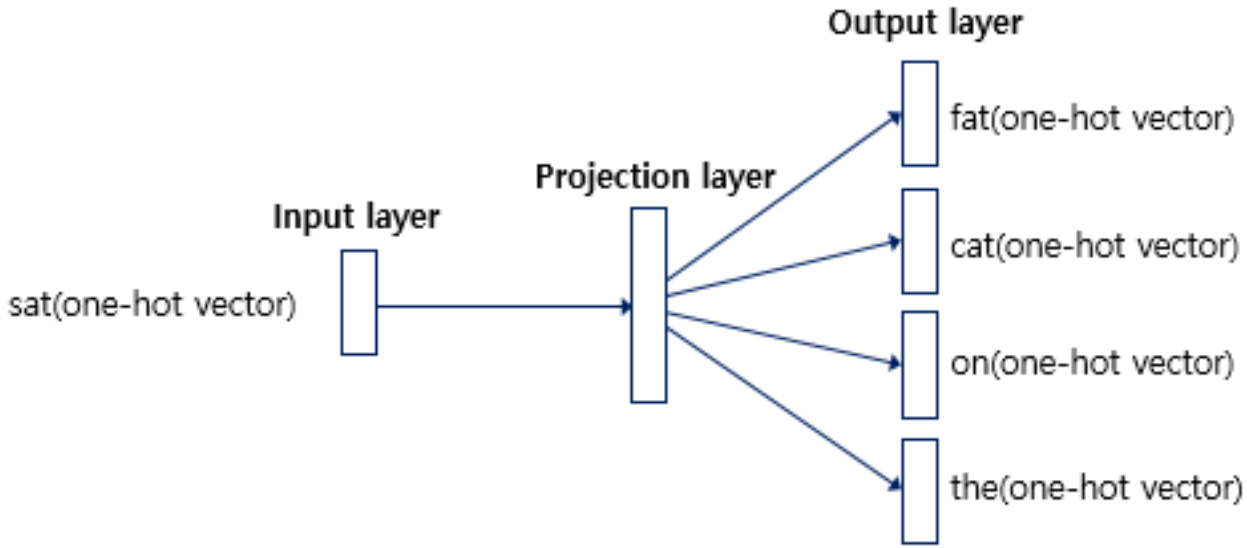
중심 단어	주변 단어
[1, 0, 0, 0, 0, 0, 0]	[0, 1, 0, 0, 0, 0, 0], [0, 0, 1, 0, 0, 0, 0]
[0, 1, 0, 0, 0, 0, 0]	[1, 0, 0, 0, 0, 0, 0], [0, 0, 1, 0, 0, 0, 0], [0, 0, 0, 1, 0, 0, 0]
[0, 0, 1, 0, 0, 0, 0]	[1, 0, 0, 0, 0, 0, 0], [0, 1, 0, 0, 0, 0, 0], [0, 0, 0, 1, 0, 0, 0], [0, 0, 0, 0, 1, 0, 0]
[0, 0, 0, 1, 0, 0,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

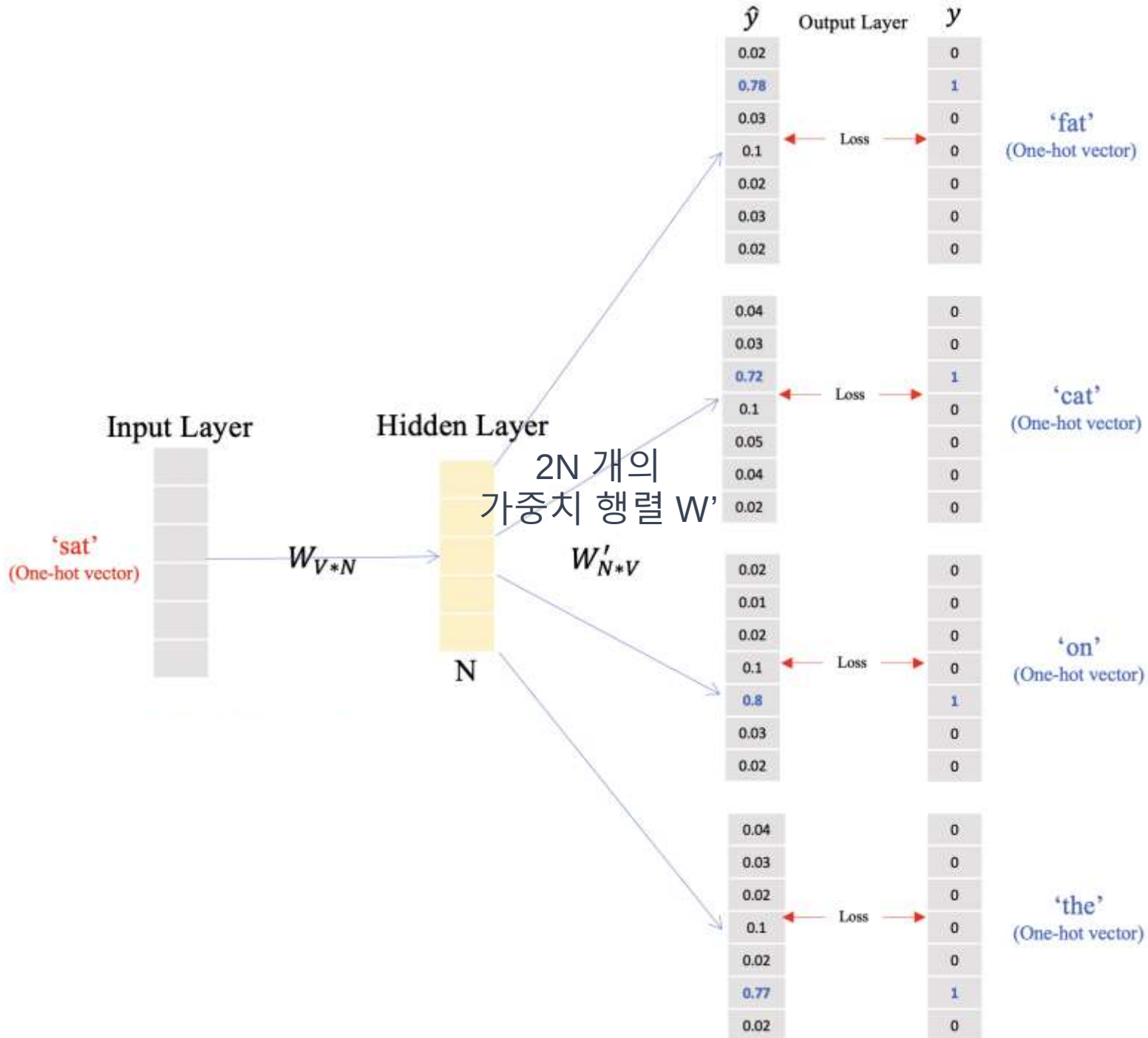


W : 가중치
V : 단어 집합의 크기
M : 투사층의 크기(임베딩 한 후의 벡터의 차원)

Skip-gram

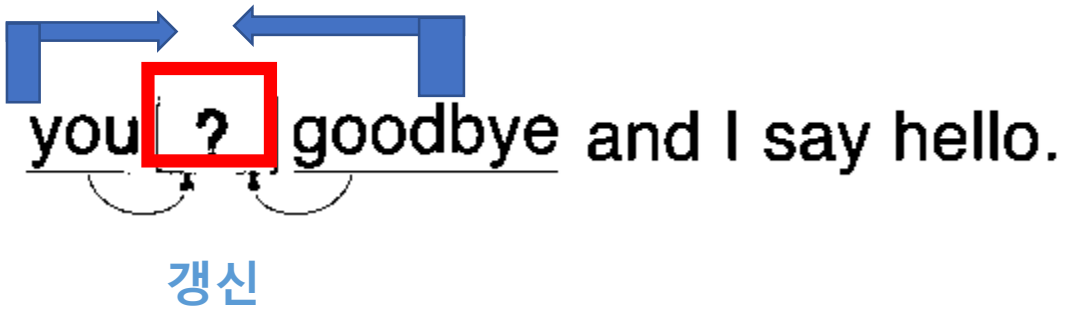

Source Text
Training Samples
The quick brown fox jumps over the lazy dog.
(the, quick)
(the, brown)
The quick brown fox jumps over the lazy dog.
(quick, the)
(quick, brown)
(quick, fox)
The quick brown fox jumps over the lazy dog.
(brown, the)
(brown, quick)
(brown, fox)
(brown, jumps)
The quick brown fox jumps over the lazy dog.
(fox, quick)
(fox, brown)
(fox, jumps)
(fox, over)


Output layer
Projection layer
Input layer
sat(one-hot vector)
fat(one-hot vector)
cat(one-hot vector)
on(one-hot vector)
the(one-hot vector)


$\hat{y}$
Output Layer
y
0.02
0.78
0.03
0.1
0.02
0.03
0.02
Loss
0
1
0
0
0
0
0
'fat'
(One-hot vector)
0.04
0.03
0.72
0.1
0.05
0.04
0.02
Loss
0
0
1
0
0
0
0
'cat'
(One-hot vector)
0.02
0.01
0.02
0.1
0.8
0.03
0.02
Loss
0
0
0
0
1
0
0
'on'
(One-hot vector)
0.04
0.03
0.02
0.1
0.02
0.77
0.02
Loss
0
0
0
0
0
1
0
'the'
(One-hot vector)
Input Layer
Hidden Layer
'sat'
(One-hot vector)
W_{V*N}
N
2N 개의
가중치 행렬 W'
W'_{N*V}

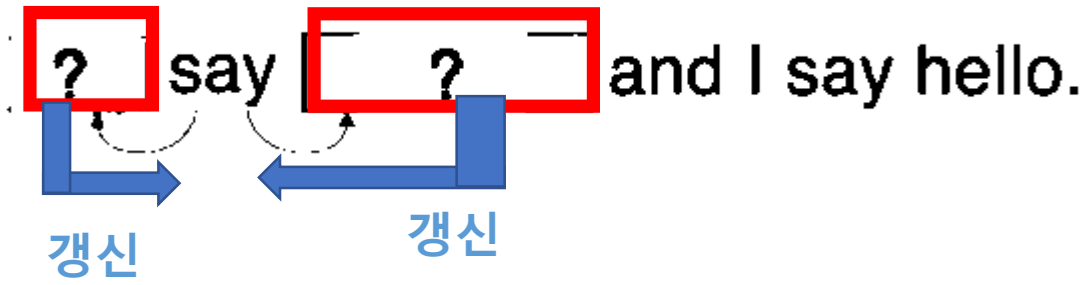
CBOW

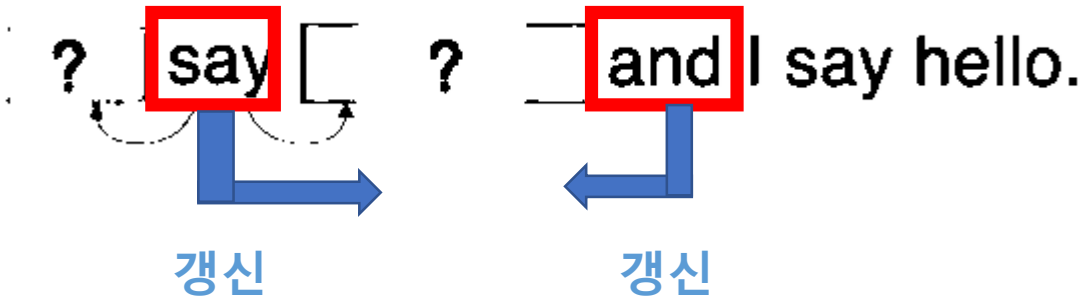
중심단어는 단 1번의 업데이트 기회를 가짐



Skip-gram

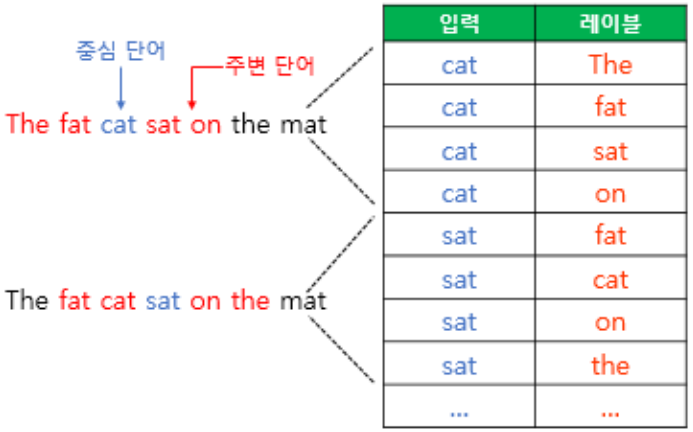
중심단어의 업데이트 기회를 여러 번 확보





SGNS

주변 단어들을 긍정(positive),
랜덤으로 샘플링 된 단어들을 부정(negative)으로 레이블링



입력	레이블
cat	The
cat	fat
cat	sat
cat	on
sat	fat
sat	cat
sat	on
sat	the
...	...

입력과 레이블의 변화

입력1	입력2	레이블
cat	The	1
cat	fat	1
cat	sat	1
cat	on	1

Negative Sampling

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

단어 집합에서 랜덤으로
선택된 단어들을
레이블 0의 샘플로 추가.

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