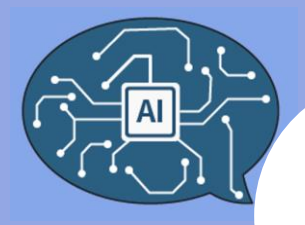
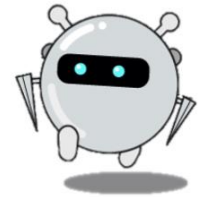


Word Embedding/ Autoencoder



멘토 황주훈

워드 임베딩 1

희소 표현

- 대표적인 희소 표현: 원-핫 벡터
- 표현하고 싶은 단어의 인덱스에만 1을 부여하고 다른 인덱스에는 0을 부여하는 벡터의 표현 방식
- 벡터의 차원 = 단어 집합의 크기

```
dog      =  [1,0,0]
cat      =  [0,1,0]
animal   =  [0,0,1]
pet      =  [0,0,1]
```

희소 표현의 한계

- BOW 가 커지면 많아질수록 사용하는 메모리가 계속 증가한다.

(공간적 낭비)

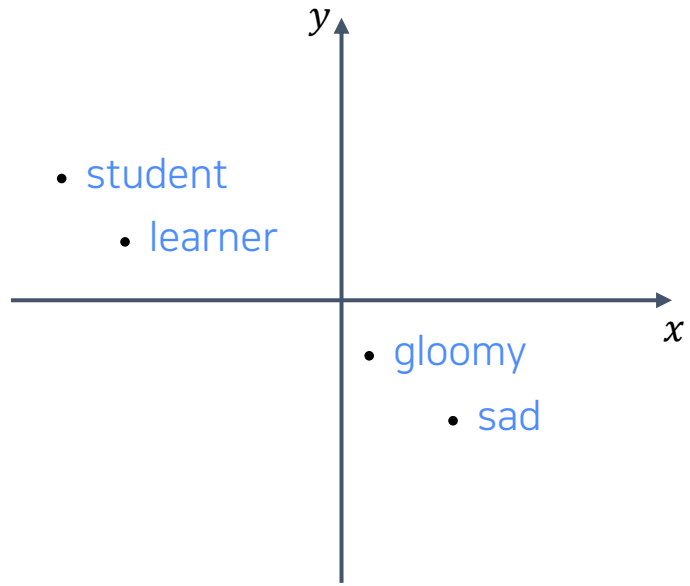
- 유사어 판별 불가능

사과	=	[1,0,0]
로봇	=	[0,1,0]
참외	=	[0,0,1]

워드 임베딩 정의

- Nlp task에서 단어를 하나의 벡터로 표현하는 기술
  (컴퓨터는 자연어를 그대로 처리할 수 없기 때문)
- 단어를 밀집 표현(dense vector)으로 변환
- 워드 임베딩을 통해 나온 결과 : 임베딩 벡터
- 유사어 판별 가능

예시)'sad': [0.02, -0.02]
'gloomy':[0.01, -0.01]



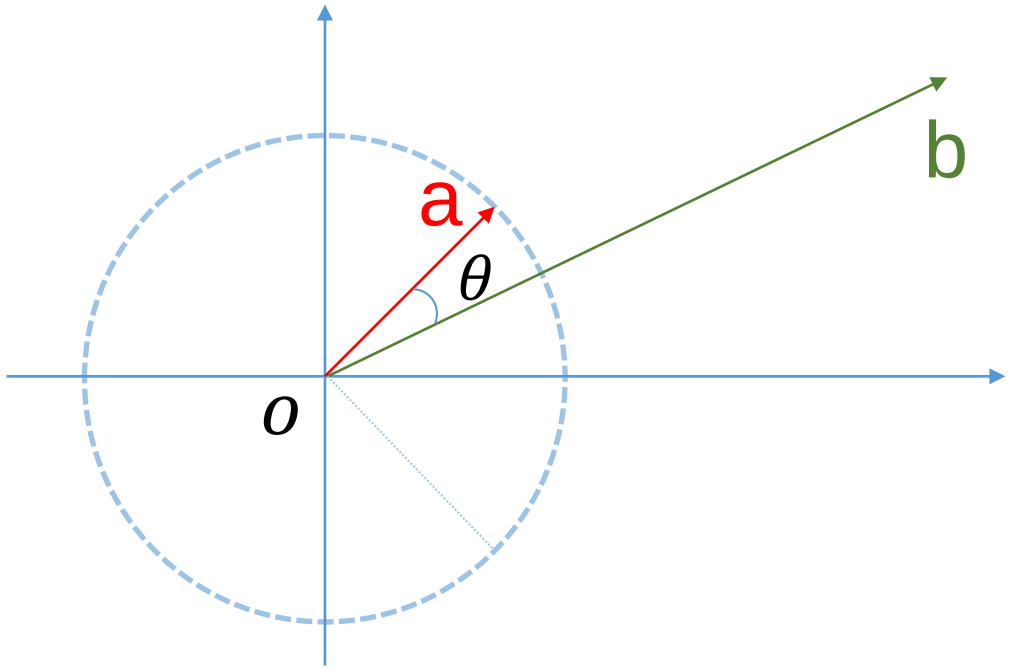
밀집 표현

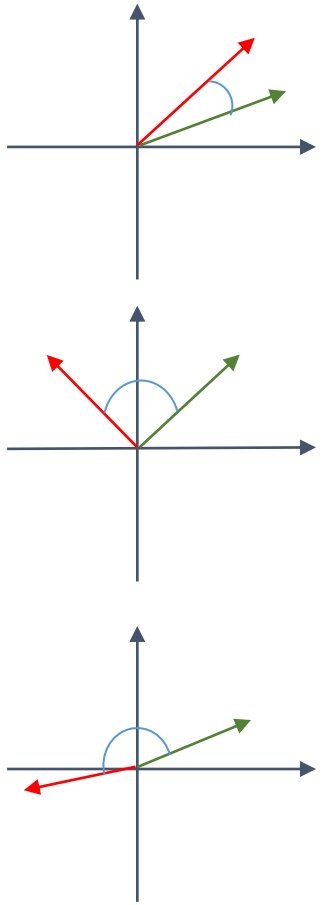
- 1과 0이 아닌 실수값을 갖는다
- 훨씬 적은 차원으로 단어를 표현할 수 있음.
- 사용자가 설정한 값으로 벡터의 차원 설정
- 두 단어의 유사도를 구할 수 있음.

dog	=	[-0.2, 0.5, 0.1, ···, 0.3]
cat	=	[0.3, 1.4, 0.1, ···, -0.2]
animal	=	[0.1, 0.2, -2.8, ···, 0.4]
pet	=	[0.4, -0.3, 0.1, ···, 0.2]

코사인 유사도

$$\cos\theta = \frac{A \cdot B}{\|A\|\|B\|}$$





두 단어를 벡터로 나타냈을 때 두 벡터 사이의 각도를 측정해서 유사도를 측정하는 방법. (-1 ~ 1 사이의 값을 가진다)

코사인 유사도

```
cosine_similarity(u, v) = (u . v) / (||u|| * ||v||)
```

두 단어를 벡터로 나타냈을 때 두 벡터 사이의 각도를 측정해서 유사도를 측정하는 방법. (-1 ~ 1 사이의 값을 가진다)

원핫-인코딩에서는 불가능.

비슷한 단어끼리는
코사인 유사도가 크겠다!

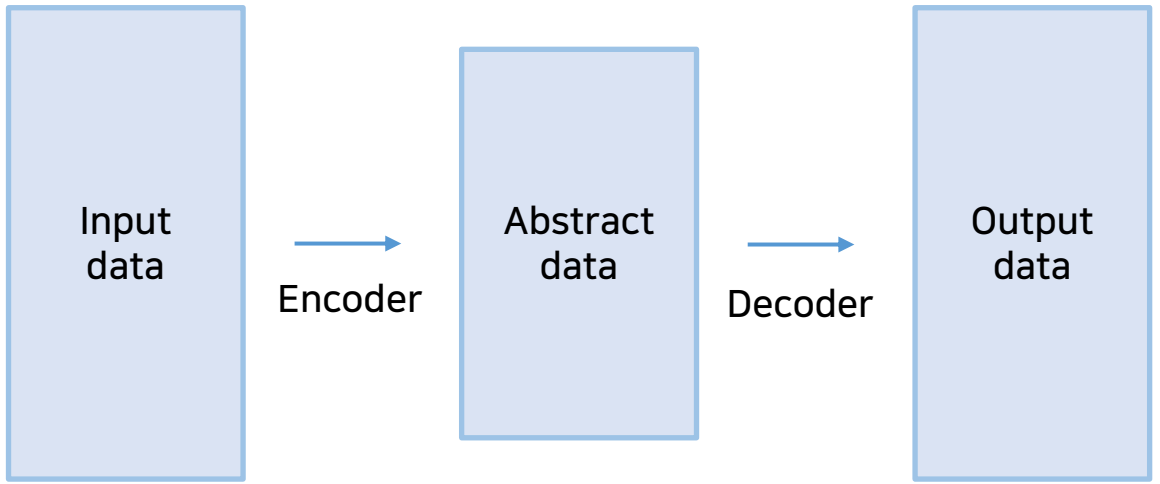
	원-핫 벡터	임베딩 벡터
차원	고차원(단어 집합의 크기)	저차원
표현 방법	수동	훈련 데이터로부터 학습함
값의 타입	1과 0	실수

오토 인코더

오토인코더

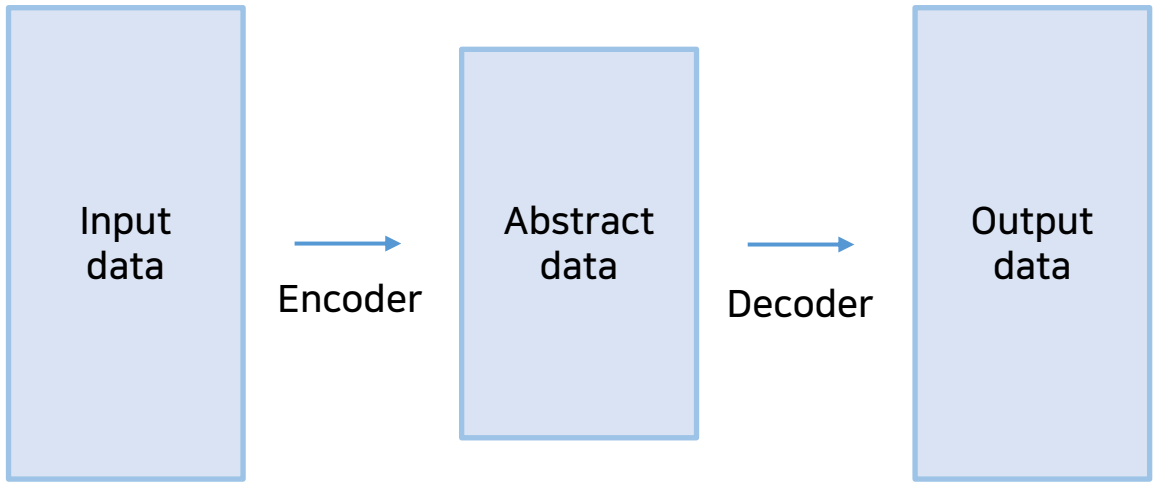
- 데이터를 압축된 형태로 학습시킬 수 있도록 하는 신경망의 하나의 형태
- 데이터 특징 추출로 인한 용량 감소

  예) 50차원의 데이터 1000개(50x1000) -> 2차원의 데이터 1000개(2x1000)



오토인코더 구조

- 인코더: 데이터의 차원 축소, 중요한 특징을 골라내고 그렇지 않은 부분 삭제
- Abstract data에는 데이터의 중요한 특징을 내포
- 디코더: 축소된 데이터로부터 원래 데이터 복원



오토인코더 학습방식

- 데이터를 압축할 때 발생할 수 있는 손실 최적화
- 예)MSE를 Loss로 두고 경사하강법(다양한 최적화, Loss 사용 가능)

$$L_{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_{Pred} - y_{true})^2$$

Q) 여기서 y_{Pred} 와 y_{true} 는 무엇일까?

오토인코더 학습방식

- 데이터를 압축할 때 발생할 수 있는 손실 최적화
- 예)MSE를 Loss로 두고 경사하강법(다양한 최적화, Loss 사용 가능)

$$L_{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_{Pred} - y_{true})^2$$

Q) 여기서 y_{Pred} 와 y_{true} 는 무엇일까?

y_{Pred} : Output data y_{true} : Input data

좋은 모델일 수록 데이터를 압축했다 복원해도 원래 데이터와 유사

워드 임베딩 2

지도학습 vs 비지도 학습

지도 학습(Supervised Learning)

- 정답(label)이 있는 상황에서 모델 학습
- 회귀, 분류 문제
- MNIST 데이터셋

비지도 학습(Unsupervised Learning)

- 정답(label)이 없는 상황에서 모델 학습
- 대부분의 워드 임베딩 기술, 클러스티링

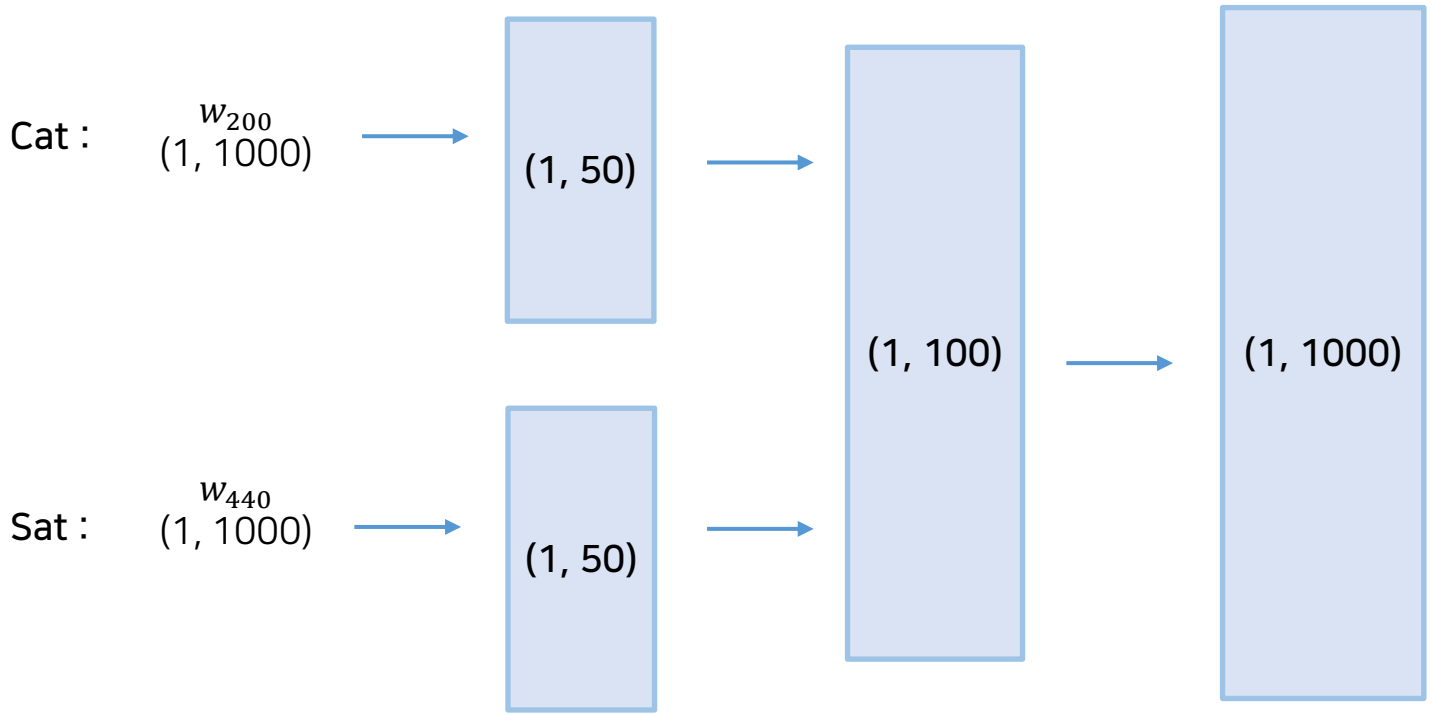
오토 인코더는 자기 지도 학습(self-supervised learning), Label = Input data

One-hot 인코딩을 이용한 단어 예측 모델

- 오토인코더를 통한 단어 예측
- Word2Vec

오토 인코더를 통한 단어 예측

예) ‘cat’, ‘sat’이 순서대로 주어졌고, 다음에 나올 단어를 예측하는 경우



오토 인코더를 통한 단어 예측

예) ‘cat’, ‘sat’이 순서대로 주어졌고, 다음에 나올 단어를 예측하는 경우

최종 (,1000)의 값을 Softmax함수를 통해 확률로 표현해줌

-> 학습된 모델에서 결과값이 [0.01, 0.02, ⋯, 0.9, ⋯, 0.01]이 된다면

0.9 값의 index에 대응 되는 단어: on 이 될 것

Word2Vec

- 워드 임베딩의 가장 대표적인 모델
- 학습 방식에 2가지 방식 CBOW와 Skip-Gram 이 있음.

Word2Vec - CBOW

중심단어 ↓ 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

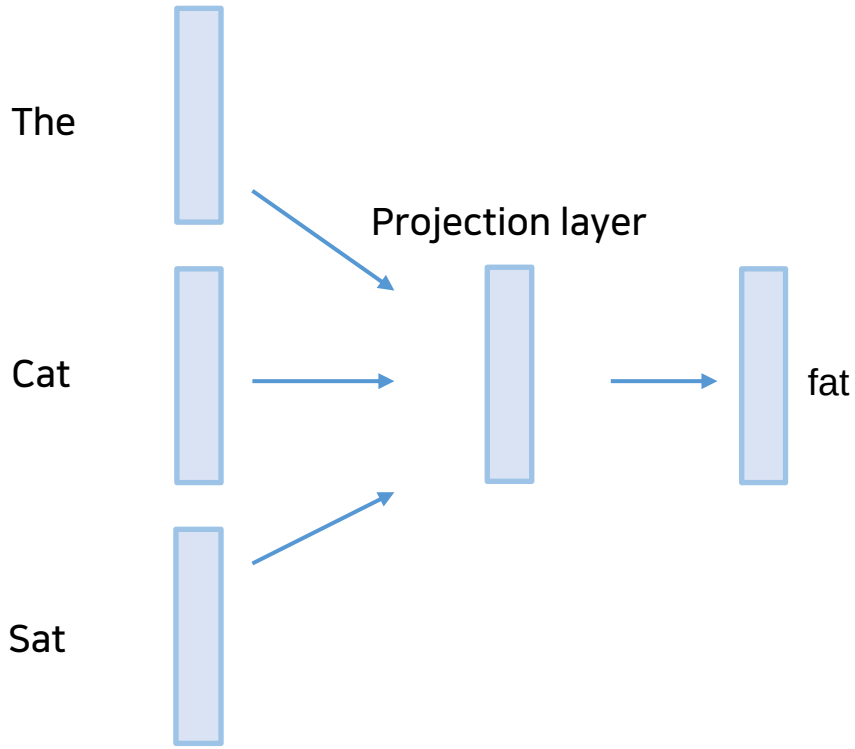
The fat cat sat on the mat

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]

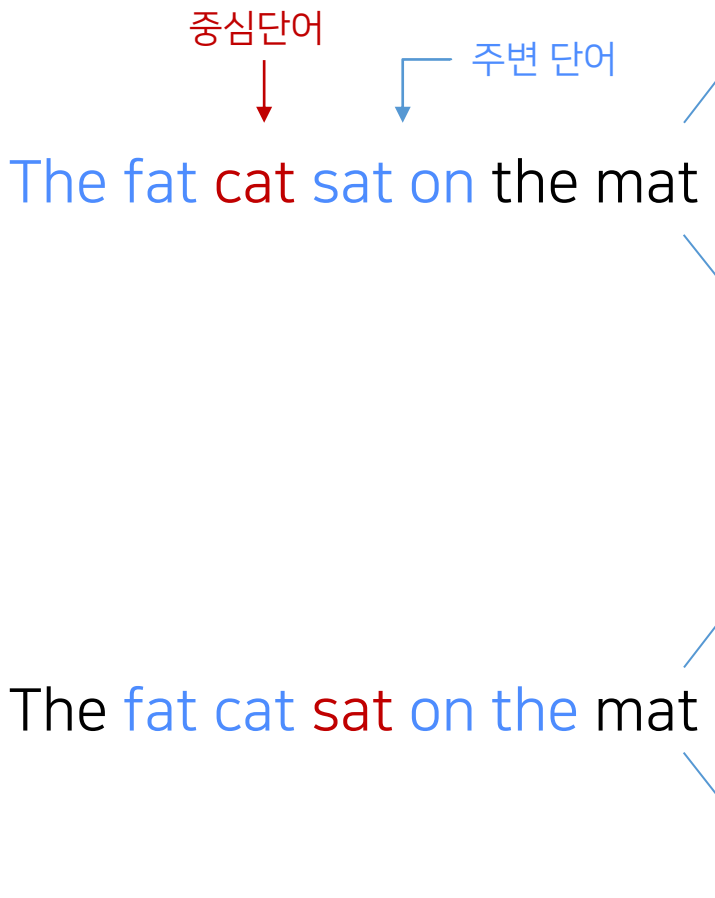
Word2Vec - CBOW

- 주변단어(입력)가 주어질 때 중심단어(출력)을 벡터 형태로 얻는 학습

예) 'The', 'Cat', 'Sat' -> Y_{pred} = 'fat'



Word2Vec - Skip-gram

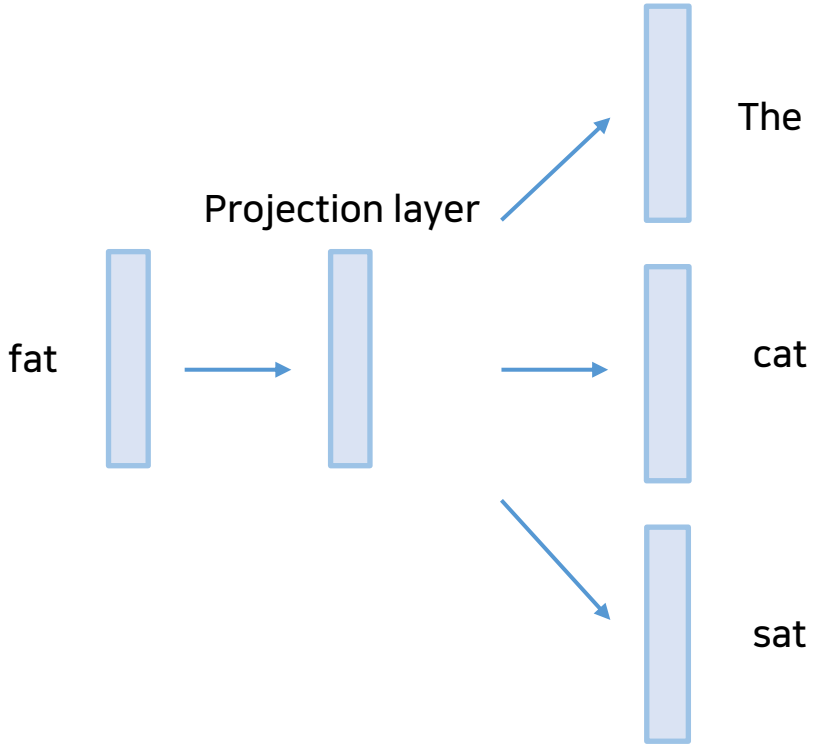


중심 단어	주변 단어
cat	The
cat	fat
cat	sat
cat	on
sat	fat
sat	cat
sat	on
sat	the

Word2Vec - Skip-gram

- 중심단어(입력)가 주어질 때 주변단어(출력)를 벡터 형태로 얻는 학습

예) 'fat' -> Y_{pred} = 'The', 'Cat', 'Sat'



의미론적 유사성 기반 인코딩

- 중심어 주위 문맥어(context word)의 등장 횟수를 기반으로 한 인코딩
- 문맥 행렬(Context Matrix)로 표현
- 행 : 중심어, 열 : 문맥어

문맥어 중심어	...cute	...litte	...scary
...			
dragon	0	1	8
kitten	28	42	4
puppy	30	39	0

kitten이 중심어일 때, 주위에 cute와 scary가 등장한 횟수는? 28, 4

의미론적 유사성 기반 인코딩

손실함수의 정의

$$L = \frac{1}{n}\sum_{i,j} f(x_{ij})\left(\overrightarrow{w_i} \cdot \overrightarrow{\tilde{w}_j} + b_i + \tilde{b}_j - \log(x_{ij})\right)^2$$

$$f(x) = \begin{cases} x, x_{\max} (x < x_{\max}) \\ 1 \qquad (x \geq x_{\max}) \end{cases}$$

$\overrightarrow{w_i}$: i 번째 중심어의 임베딩 벡터 $\overrightarrow{\tilde{w}_j}$: j번째 문맥어의 임베딩 벡터

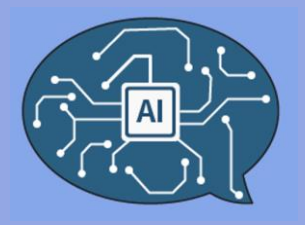
$b_i + \tilde{b}_j$: 중심어, 문맥어에 대응되는 편향

$x_{\max}$ 의 역할

- 가중치의 개념으로 사용자가 직접 설정한 값
- 횟수에 따른 가중치 부여, 또한 너무 높은 가중치를 방지(최대 1)

참고

https://wikidocs.net/22885



감사합니다