Word2Vec

명지대학교
MYONGJI UNIVERSITY

희소 표현과 원 핫 인코딩

희소 표현(Sparse Representation)이란?

표현하고자 하는 단어의 인덱스의 값만 1이고,
나머지 인덱스에는 전부 0으로 표현되는 벡터 표현 방법(One Hot Encoding)

Word2Vec 개요

희소 표현과 원 핫 인코딩

원 핫 인코딩(One Hot Encoding)이란?

전체 레이블(Label) 수에 대해 표현하고자 하는
특정 단어(Word)만 '1'로 활성화한 벡터로 변환하는 방법

사과	1	0	0	0	0
개	0	1	0	0	0
배	0	0	1	0	0
고양이	0	0	0	1	0
토마토	0	0	0	0	1

단어의 의미와 관계를 전혀 고려하지 않음

벡터의 크기는 총 단어 수만큼의 Sparse Vector

정답(Label)을 인식하게 하는 용도로 주로 사용

Word2Vec 개요

분산 표현과 Word2Vec

분산 표현(Distributed Representation)이란?

분포 가설(Distributional Hypothesis) 가정 하에 문자열을 표현하는 방법

비슷한 위치에서 등장하는 단어들은
비슷한 의미를 가질 확률이 높다는 가설

» ‘강아지’는 주로 ‘귀엽다’, ‘예쁘다’, ‘멍멍’ 등의 단어와 함께 등장

분산 표현과 Word2Vec

Word2Vec이란?

대표적인 분산 표현 방법으로 분포 가설을 이용

코퍼스(말뭉치)에서 각 단어를 단어의 의미에 따라
여러 차원에 분산하여 벡터로 표현

단어 간 유사도 계산 등의 연산 가능

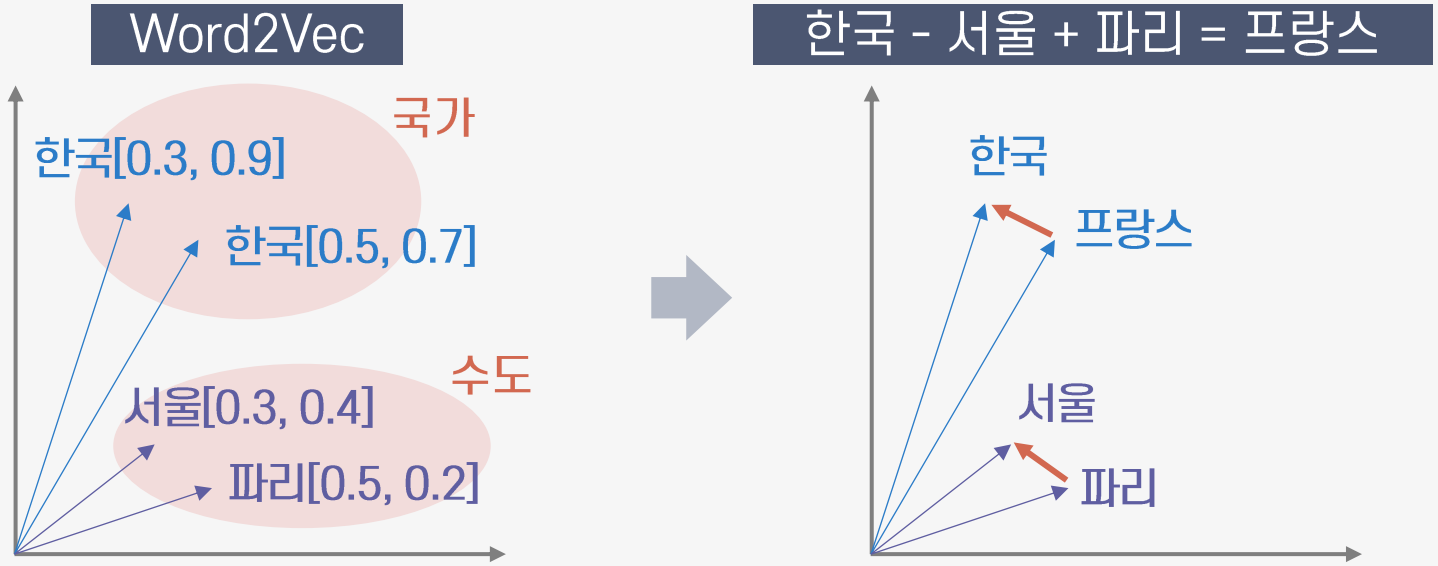
Word2Vec 개요

분산 표현과 Word2Vec

Word2Vec을 이용한 워드 임베딩

Word2Vec은 유사한 단어가 유사한 위치에 임베딩

단어 벡터 간 연산을 통해 의미 파악 가능



Word2Vec은 CBOW와 Skip-gram 등 두 가지 방법이 존재

Word2Vec 개요

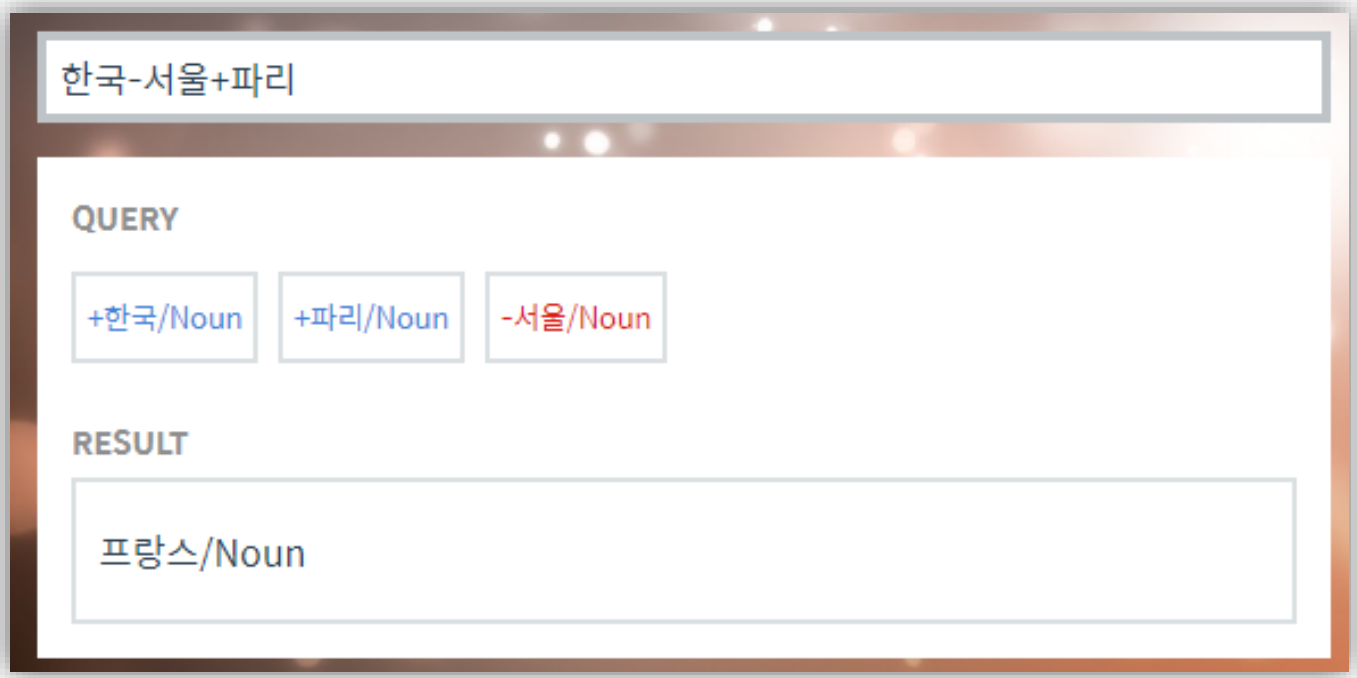
분산 표현과 Word2Vec

- Word2Vec을 이용한 단어 벡터 간 연산

https://word2vec.kr에 접속하여
“한국 - 서울 + 파리”를 입력

»

연산의 결과로
“프랑스”를 출력

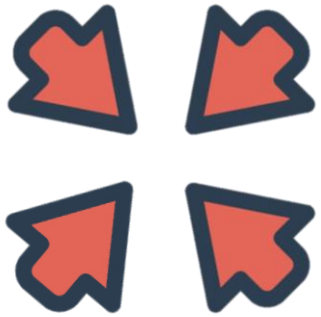


<출처: Word2Vec. https://word2vec.kr>

CBOW와 Skip-gram 개요

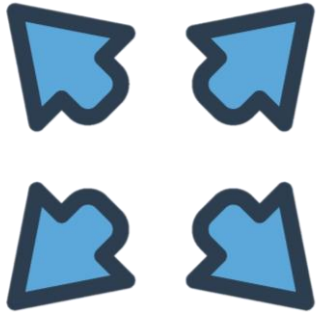
CBOW란? (Continuous Bag-of-Words)

주변 단어(Context Word)를
임베딩하여
중심 단어(Target Word)를
예측하기 위한 Word2Vec 방법



Skip-gram이란?

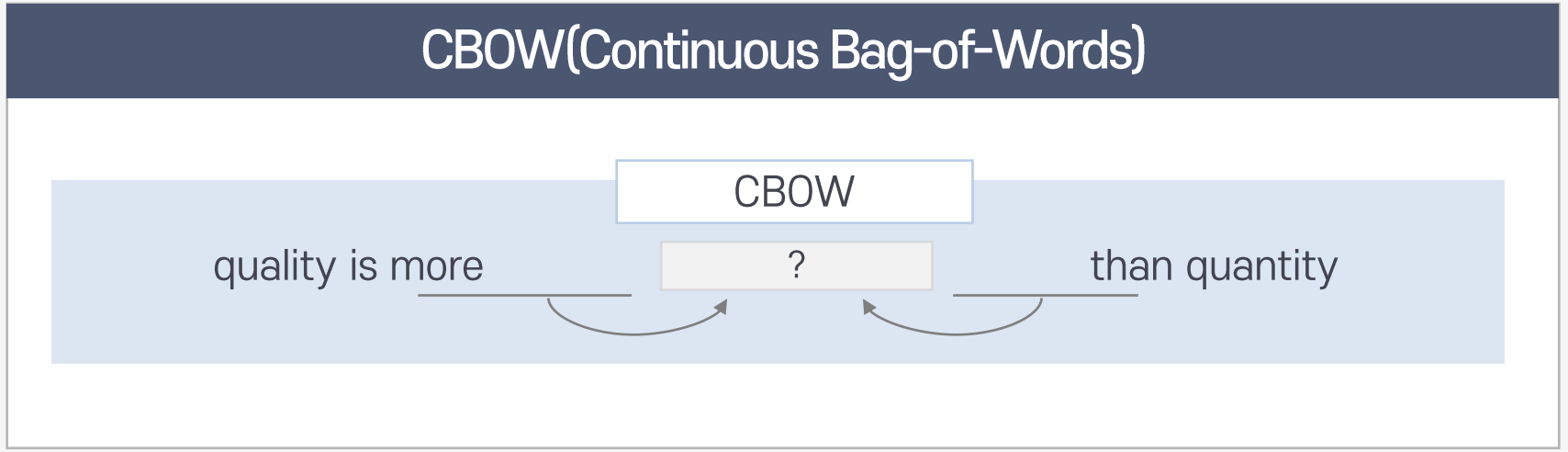
중심 단어(Target Word)를
임베딩하여
주변 단어(Context Word)를
예측하기 위한 Word2Vec 방법



CBOW과 Skip-gram의 예측 단어

문장에 대해 CBOW와 Skip-gram을 이용해 단어 예측

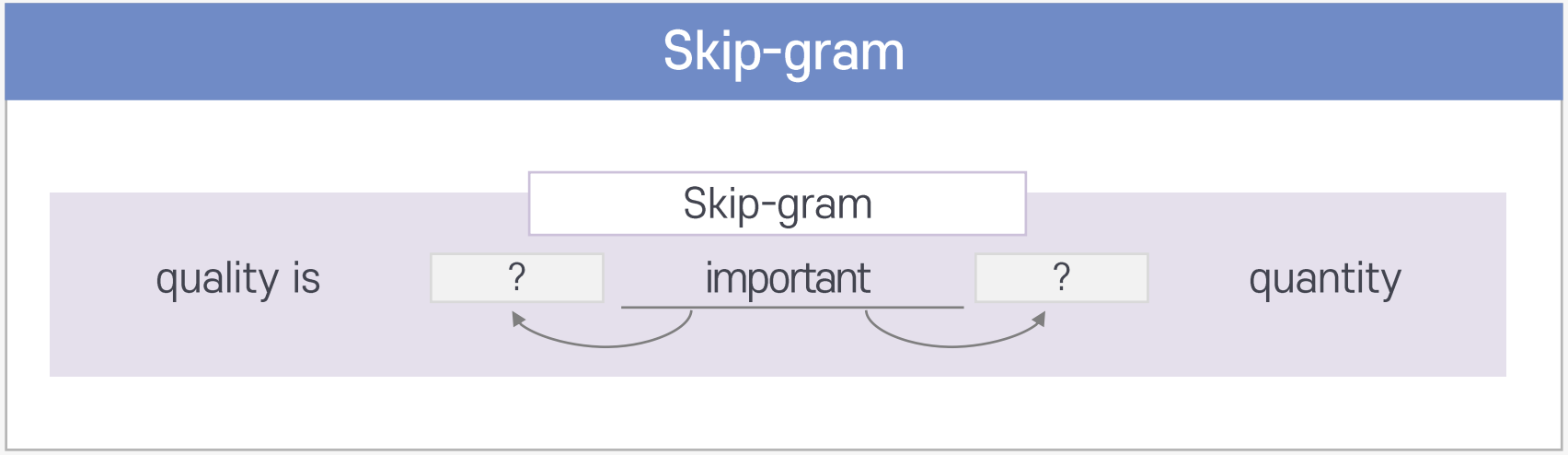
CBOW는 주변 단어(context word)를 임베딩(embedding)하여
중심 단어(target word)를 예측



CBOW과 Skip-gram의 예측 단어

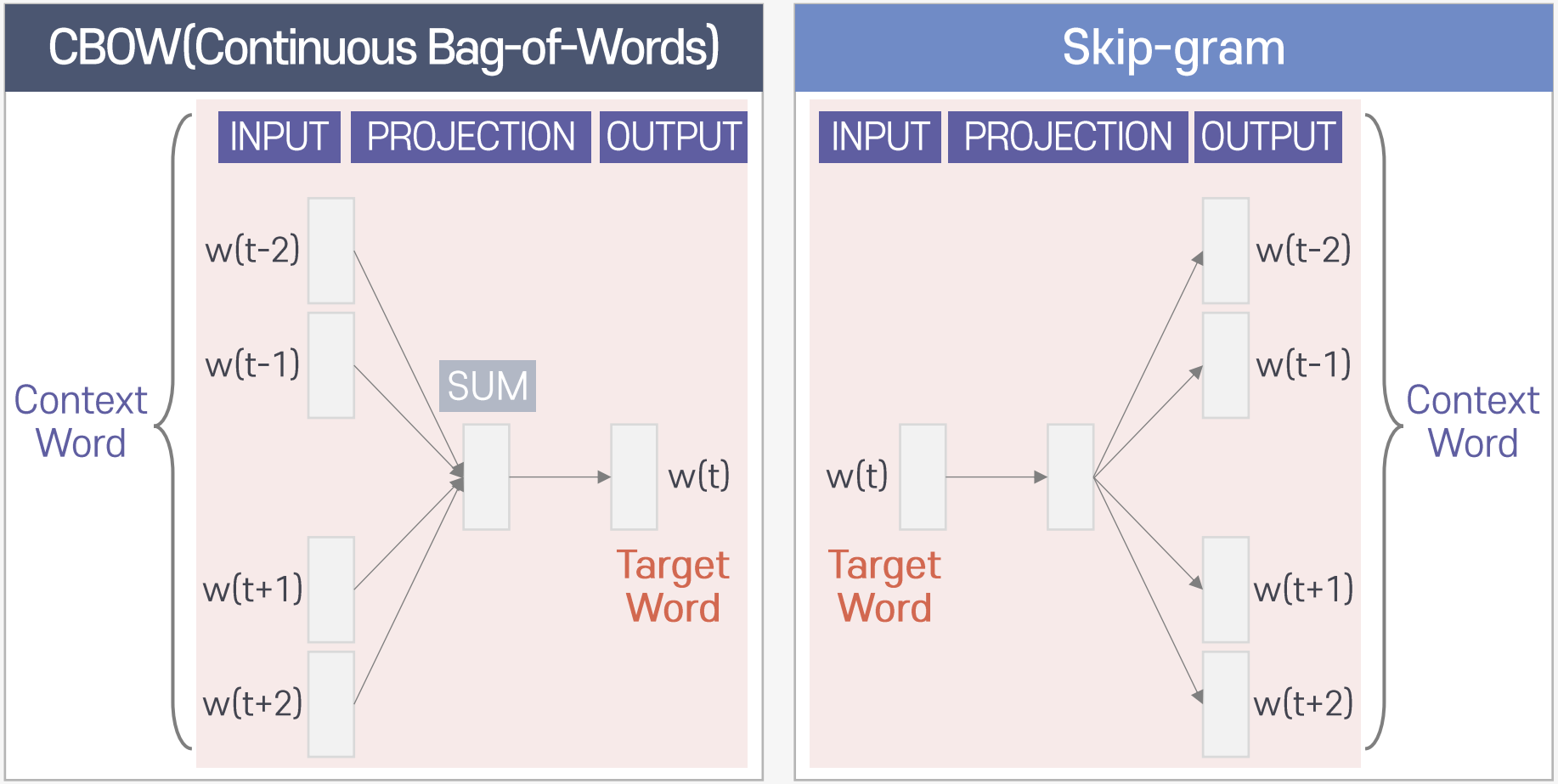
문장에 대해 CBOW와 Skip-gram을 이용해 단어 예측

skip-gram은 중심 단어(target word) 를
임베딩(embedding)하여 주변 단어(context word)를 예측



CBOW와 Skip-gram의 구조

CBOW와 Skip-gram은 서로 대칭적 구조로 이루어 짐



CBOW와 Skip-gram을 이용한 단어 예측

Word2Vec 학습 문장 “quality is more important than quantity”의
원 핫 인코딩(One Hot Encoding) 표현

quality	1	0	0	0	0	0
is	0	1	0	0	0	0
more	0	0	1	0	0	0
important	0	0	0	1	0	0
than	0	0	0	0	1	0
quantity	0	0	0	0	0	1

CBOW와 Skip-gram을 이용한 단어 예측

윈도우(Window)

윈도우(Window)란?

중심 단어(Target Word)를 기준으로
참조하고자 하는 주변 단어(Context Word)의 범위

Window size가 2인 경우 예시

quality	is	more

quality	is	more	important	than

CBOW와 Skip-gram을 이용한 단어 예측

슬라이딩 윈도우

슬라이딩 윈도우(Sliding Window)란?

전체 학습 문장에 대해
Window를 이동하며 학습하기 위해

중심 단어(Target Word)와
주변 단어(Context Word) 데이터 셋을 추출하는 과정

CBOW와 Skip-gram을 이용한 단어 예측

- 슬라이딩 윈도우

Window size가 2인 경우 예시

1	quality	is	more	important	than	quantity
2	quality	is	more	important	than	quantity
3	quality	is	more	important	than	quantity
4	quality	is	more	important	than	quantity
5	quality	is	more	important	than	quantity
6	quality	is	more	important	than	quantity

CBOW와 Skip-gram을 이용한 단어 예측

슬라이딩 윈도우

> 슬라이딩 윈도우를 통해 추출된
> 중심 단어(Target Word)와 주변 단어(Context Word) 데이터 셋

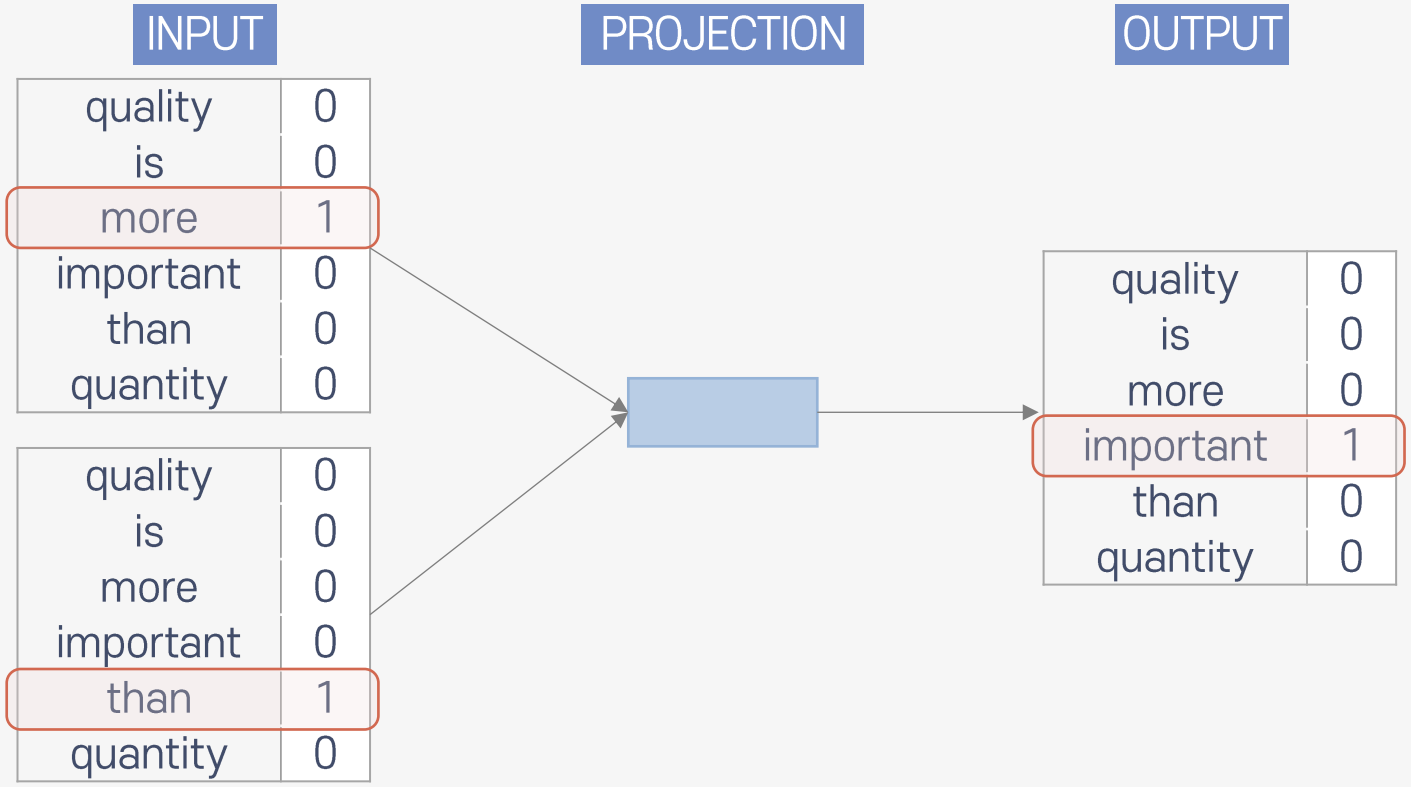
Sliding	Target Word	Context Word
1	[1, 0, 0, 0, 0, 0]	[0, 1, 0, 0, 0, 0], [0, 0, 1, 0, 0, 0]
2	[0, 1, 0, 0, 0, 0]	[1, 0, 0, 0, 0, 0], [0, 0, 1, 0, 0, 0], [0, 0, 0, 1, 0, 0]
3	[0, 0, 1, 0, 0, 0]	[1, 0, 0, 0, 0, 0], [0, 1, 0, 0, 0, 0], [0, 0, 0, 1, 0, 0], [0, 0, 0, 0, 1, 0]
4	[0, 0, 0, 1, 0, 0]	[0, 1, 0, 0, 0, 0], [0, 0, 1, 0, 0, 0], [0, 0, 0, 0, 1, 0], [0, 0, 0, 0, 0, 1]
5	[0, 0, 0, 0, 1, 0]	[0, 0, 1, 0, 0, 0], [0, 0, 0, 1, 0, 0], [0, 0, 0, 0, 0, 1]
6	[1, 0, 0, 0, 0, 1]	[0, 0, 0, 0, 1, 0], [0, 0, 0, 1, 0, 0]

CBOW와 Skip-gram을 이용한 단어 예측

CBOW의 중심 단어(Target Word) 예측

Window size가 1인 경우

- 입력 more [001000], than [000010]
- 출력 important [000100]

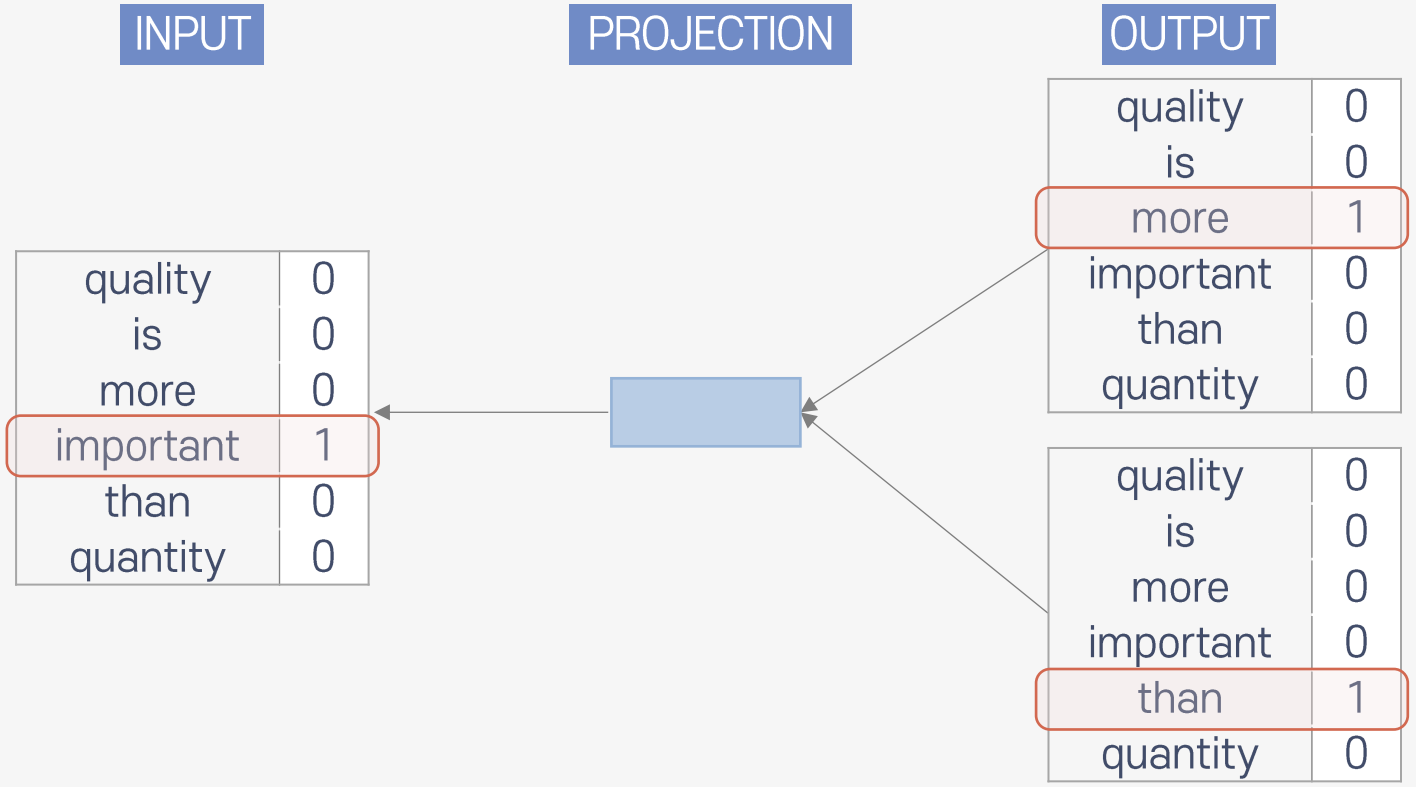


CBOW와 Skip-gram을 이용한 단어 예측

Skip-gram의 중심 단어(Target Word) 예측

Window size가 1인 경우

- 입력 important [000100]
- 출력 more [001000], than [000010]



Word2Vec 활용

파이썬에서 Word2Vec 적용을 위한 gensim 설치

gensim 패키지

파이썬에서 CBOW, Skip-gram등의
Word2Vec 적용을 위한 라이브러리

gensim 패키지 설치 방법

command 창 실행
(window키 + R)

»

Word2Vec을 위한
gensim 설치

- pip install gensim

Word2Vec 활용

파이썬에서 genism 패키지를 이용한 Word2Vec 적용

```
from gensim.models import Word2Vec
model = Word2Vec([data],        # 리스트 형태의 데이터
                 sg=1,          # 0: CBOW, 1: Skip-gram
                 size=100,      # 벡터 크기
                 window=3,      # 고려할 앞뒤 폭(앞뒤 3단어)
                 min_count=3)  # 사용할 단어의 최소 빈도(3회 이하 단어 무시)
```

Word2Vec 활용

파이썬에서 genism 패키지를 이용한 유사도 분석

유사한 단어 및 단어별 유사도 분석

코퍼스에서 '게임'과 유사한 단어 및 단어별 유사도 분석

```
result = model.most_similar('게임')
print(result)

[('이용', 0.17749272286891937),
 ('규정', 0.1271159648895263 7),
 ('적용', 0.1180865317583084 1),
 ('국내', 0.1066054329276084 9),
 ('질병', 0.1021090000867843 6),
 ('코드', 0.1015351489186286 9),
 ('지난', 0.0919529199600219 7),
 ('콘텐츠', 0.0783862918615341 2),
 ('게임중독', 0.0583686903119087 2),
 ('분류', 0.0482855737209320 1)]
```

Word2Vec 활용

- 파이썬에서 genism 패키지를 이용한 유사도 분석
  - 단어 간 유사도 분석

```
cos = model.wv.similarity('게임', '질병')
print(cos)

0.102109
```

Word2Vec 활용

Pre-trained Word2Vec 모델을 활용한 유사도 분석

Pre-trained 모델이란?

대용량 코퍼스 데이터를 이용하여 사전에 학습된 모델

Word2Vec 활용

To. 교수님

①번 내용 관련해,
해당 경로가 없어지거나 변경될 것을 고려해
경로 대신 다르게 제시 가능할까요?
-> 출처를 밝힐 필요가 있어 서 불가피 합니다.

Pre-trained Word2Vec 모델을 활용한 유사도 분석

- 한국어 Pre-trained Word2Vec 모델을 활용한 유사도 분석 방

① Pre-trained Word2Vec 모델 다운로드

» https://github.com/Kyubyong/wordvectors에
공개된 경로를 통해 한국어 버전 다운로드 가능

② 압축해제 및 ko.bin 파일을 data 디렉토리로 이동

Pre-trained Word2Vec 모델을 활용한 유사도 분석

한국어 Pre-trained Word2Vec 모델을 활용한 유사도 분석 방법

③ Pre-trained 모델 로드 및 활용

```
import gensim
model = gensim.models.Word2Vec.load('./data/ko.bin')

result = model.most_similar("게임")
print(result)

[('액션', 0.6752270460128784),
 ('게임기', 0.6604887843132019),
 ('콘솔', 0.6558898687362671),
 ('슈팅', 0.6405965089797974),
 ('아케이드', 0.6360284090042114),
 ('퍼즐', 0.6304599046707153),
 ('어드벤처', 0.6224750876426697),
 ('닌텐도', 0.6071302890777588),
 ('애니메이션', 0.6063960790634155),
 ('온라인', 0.6031370162963867)]
```

Pre-trained Word2Vec 모델을 활용한 유사도 분석

한국어 Pre-trained Word2Vec 모델을 활용한 유사도 분석 방법

③ Pre-trained 모델 로드 및 활용

```
result = model.most_similar("인공지능")
print(result)

[('컴퓨팅', 0.6520194411277771),
 ('가상현실', 0.6393702030181885),
 ('심리학', 0.63037109375),
 ('모델링', 0.625065267086029),
 ('신경망', 0.6200423836708069),
 ('로봇', 0.6109743118286133),
 ('시뮬레이션', 0.610107064247313),
 ('지능', 0.6092982888221741),
 ('기술', 0.6087721586227417),
 ('기술인', 0.595707654953002 9)]
```