Embedding

텍스트를 숫자로 표현하기 위한 방법

Embedding이란?

em·bed·ding

+ Add to Wordbook

Pronunciation

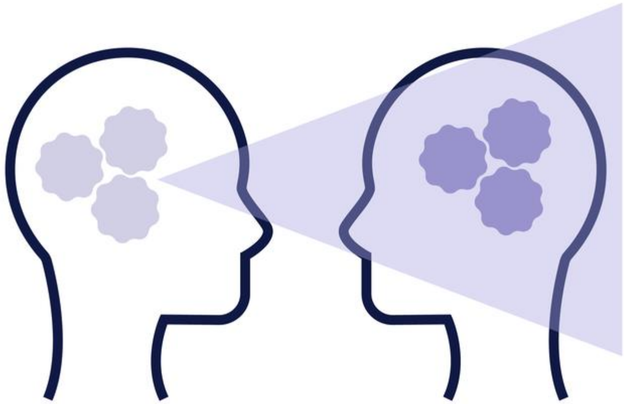
All US GB AU IN

Noun | EN-EN Dictionary

Noun

1. 수학
   어떤 위상(位相) 공간에서 다른 위상 공간으로의 동상 사상(同相寫像).

Source : YBM All in All English-Korean Dictionary

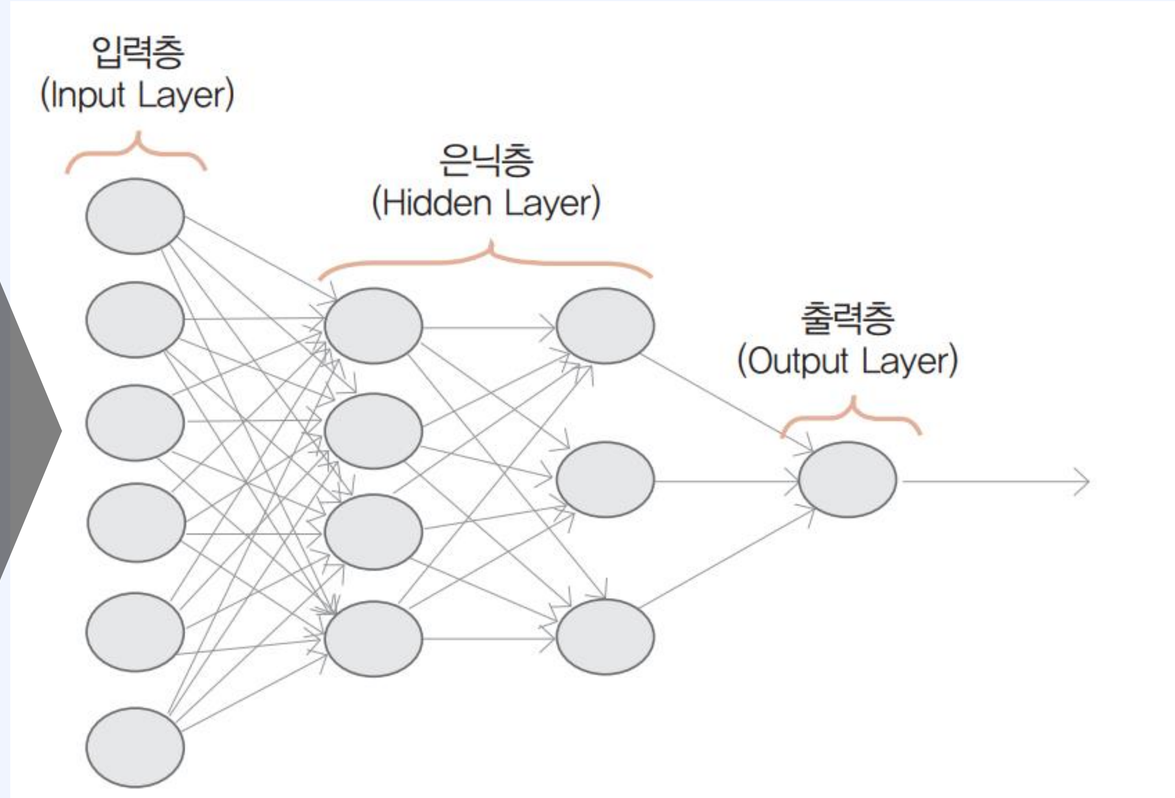


이미지출처: 네이버 사전, https://www.inkblotadvertising.com/blog/2021/1/11/behavioral-science-nudges-and-projection-bias

Embedding이 필요한 순간

➢ 인공신경망의 구조

Number,
Image,
Text,
Sound,
Video,
Category,
Graph …



이미지출처: 파이썬으로 시작하는 머신러닝+딥러닝(아이리포)

범주형 데이터의 벡터화

➢ One-hot vector

id	color
1	red
2	blue
3	green
4	blue

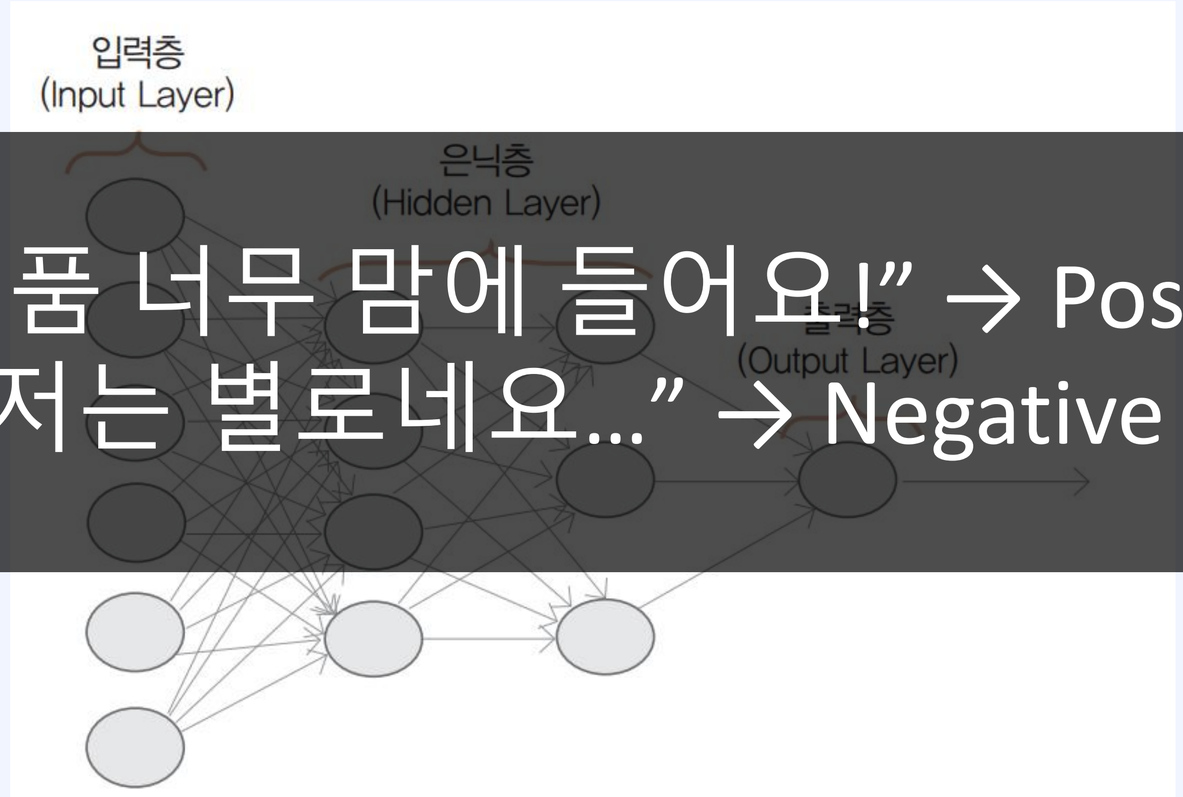
One Hot Encoding

id	color_red	color_blue	color_green
1	1	0	0
2	0	1	0
3	0	0	1
4	0	1	0

[이미지출처]: https://towardsdatascience.com/building-a-one-hot-encoding-layer-with-tensorflow-f907d686bf39

텍스트를 신경망에 넣는 방법?

➢ Sentimental Analysis



"이 제품 너무 맘에 들어요!" → Positive
"저는 별로네요..." → Negative

출처: 파이썬으로 시작하는 머신러닝+딥러닝(아이리포)

텍스트를 원핫 벡터로 표현할 경우의 한계

- Sparse한 구조
- 의미 반영 불가

id	color
1	red
2	blue
3	green
4	blue

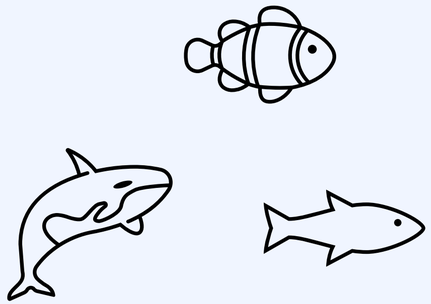
One Hot Encoding

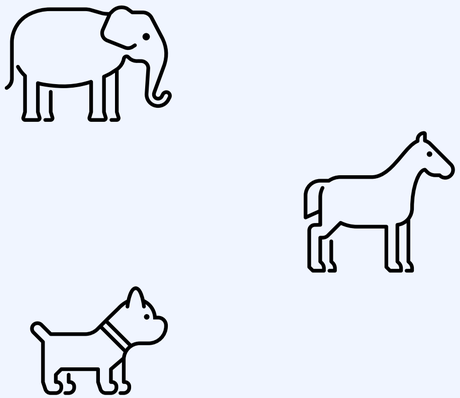
id	color_red	color_blue	color_green
1	1	0	0
2	0	1	0
3	0	0	1
4	0	1	0

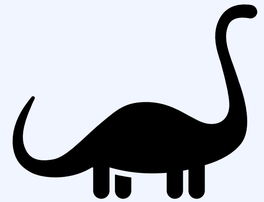
[이미지출처]: https://towardsdatascience.com/building-a-one-hot-encoding-layer-with-tensorflow-f907d686bf39

Embedding

- 원본 데이터의 '특징(feature)'을 담을 수 있는 표현방법이 필요
- 특징을 알고 있다면, 유사한 특징을 가진 데이터를 판단하고 수학적으로 다룰 수 있음



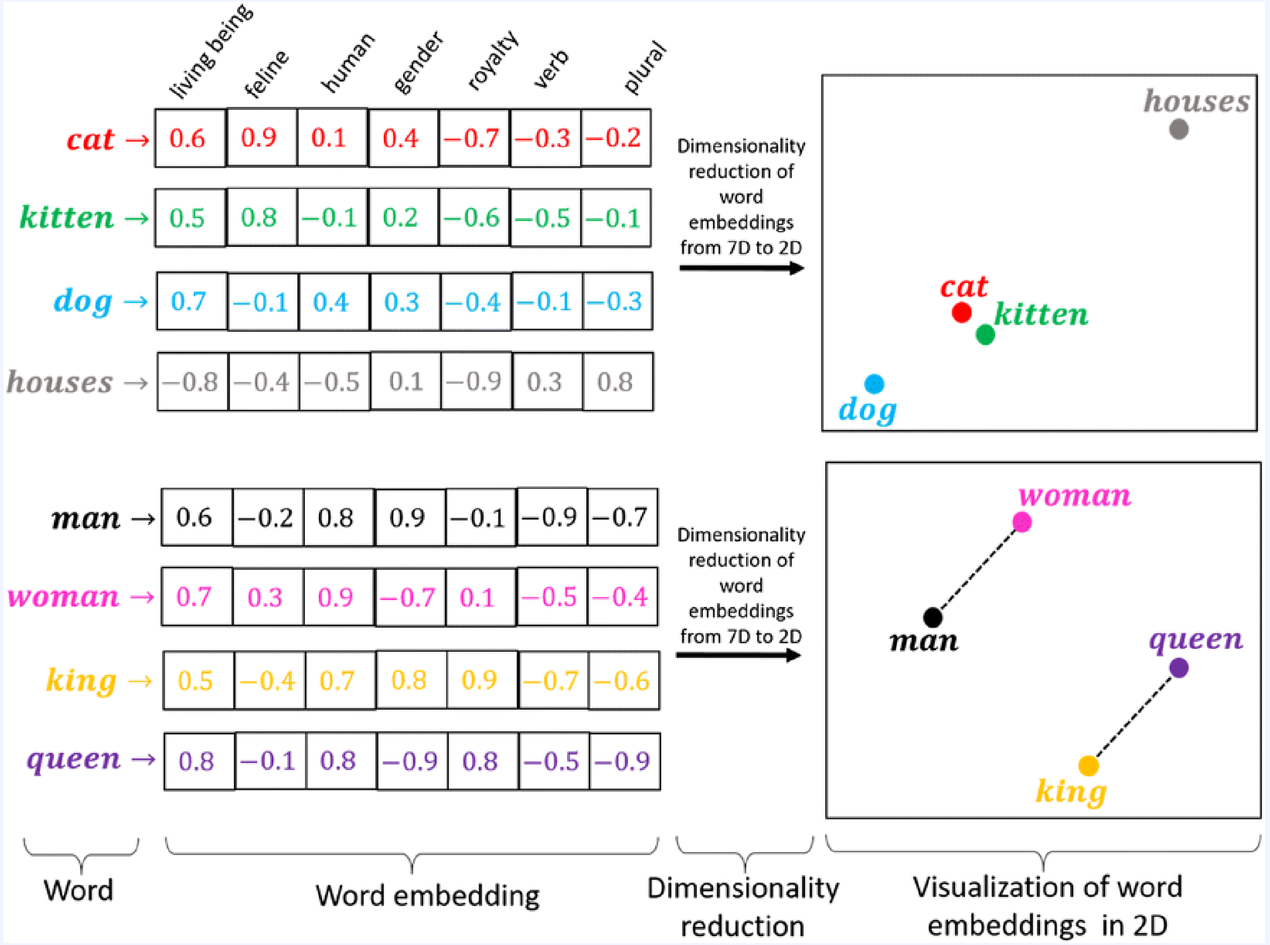




새로운 데이터?

Embedding

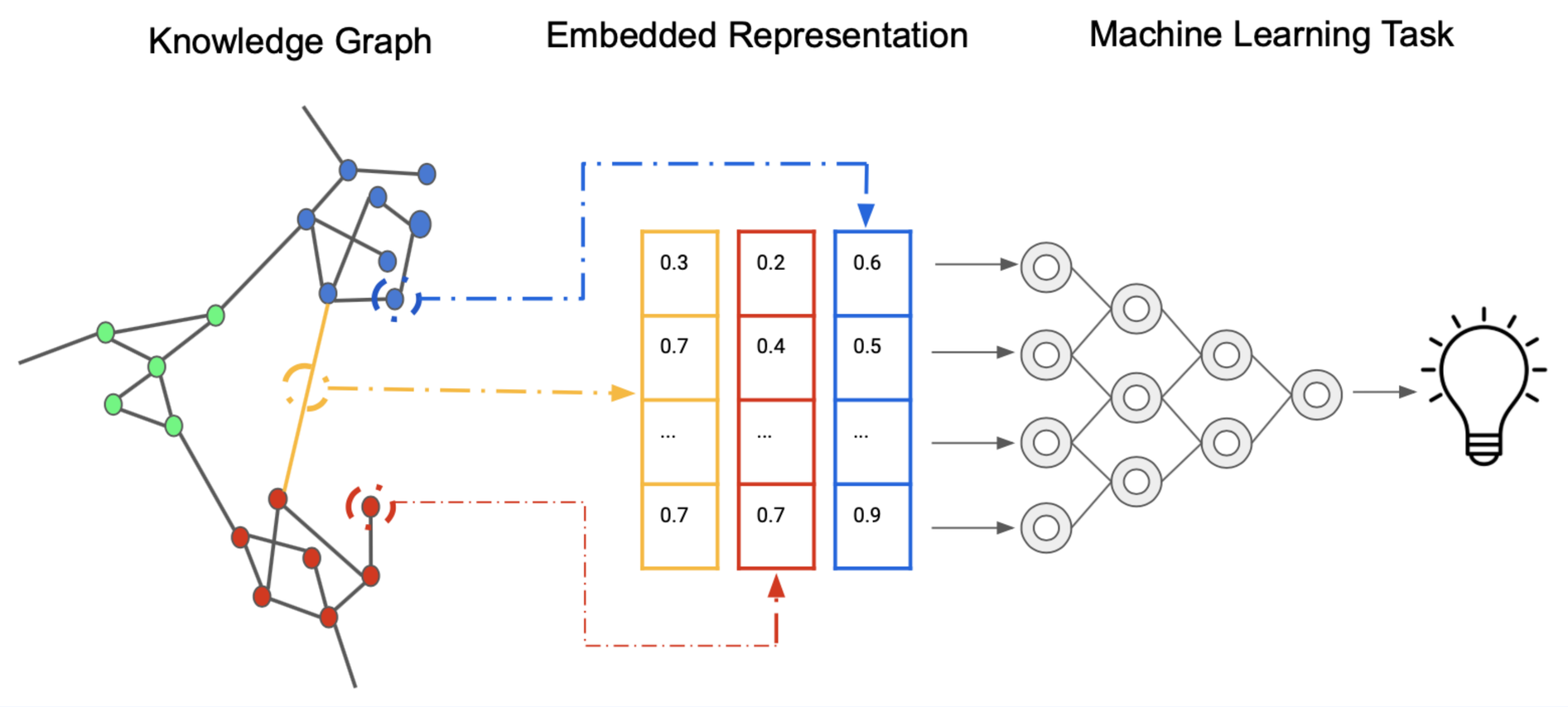
➢ Word Embedding



https://medium.com/@hari4om/word-embedding-d816f643140

Embedding

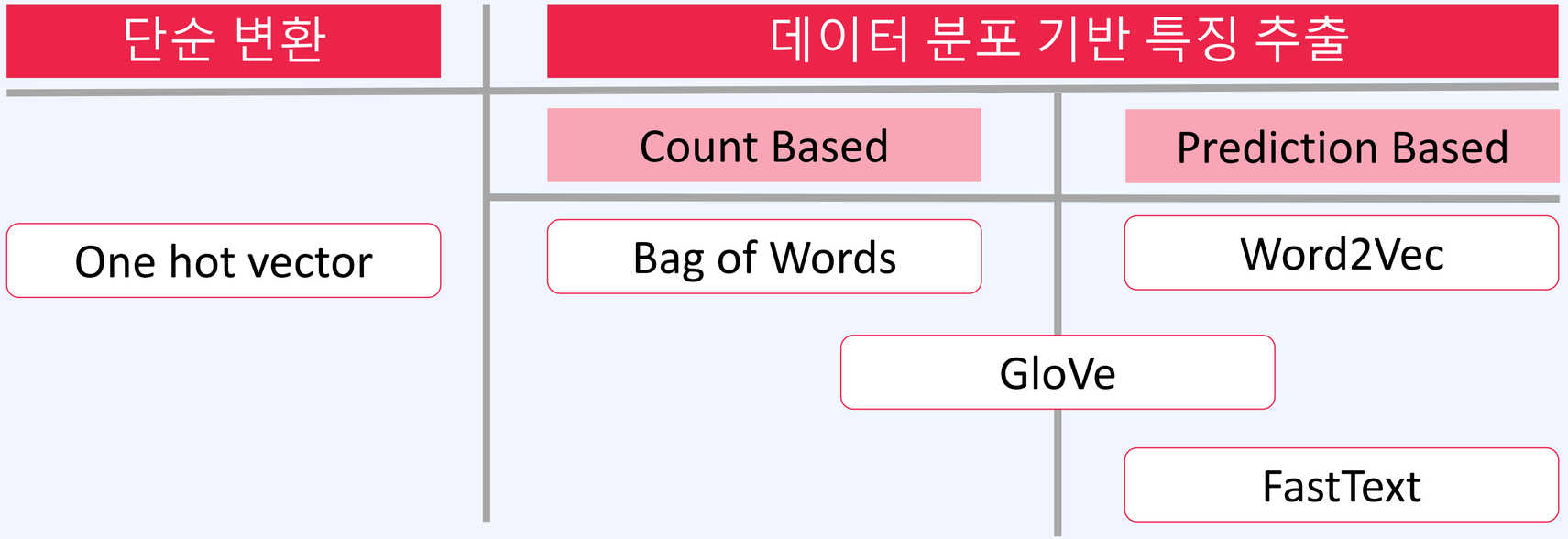
➢ Graph Embedding



https://en.wikipedia.org/wiki/Knowledge_graph_embedding#/media/File:KnowledgeGraphEmbedding.png

Word Embedding

➢ 단어가 가진 의미를 그대로 보존하면서 의미와 맥락을 고려하여 단어를 벡터로 표현
- 의미를 보존한다= 개별 단어가 가진 속성을 보존한다
- 벡터로 표현한다 = 실수값으로 표현한다

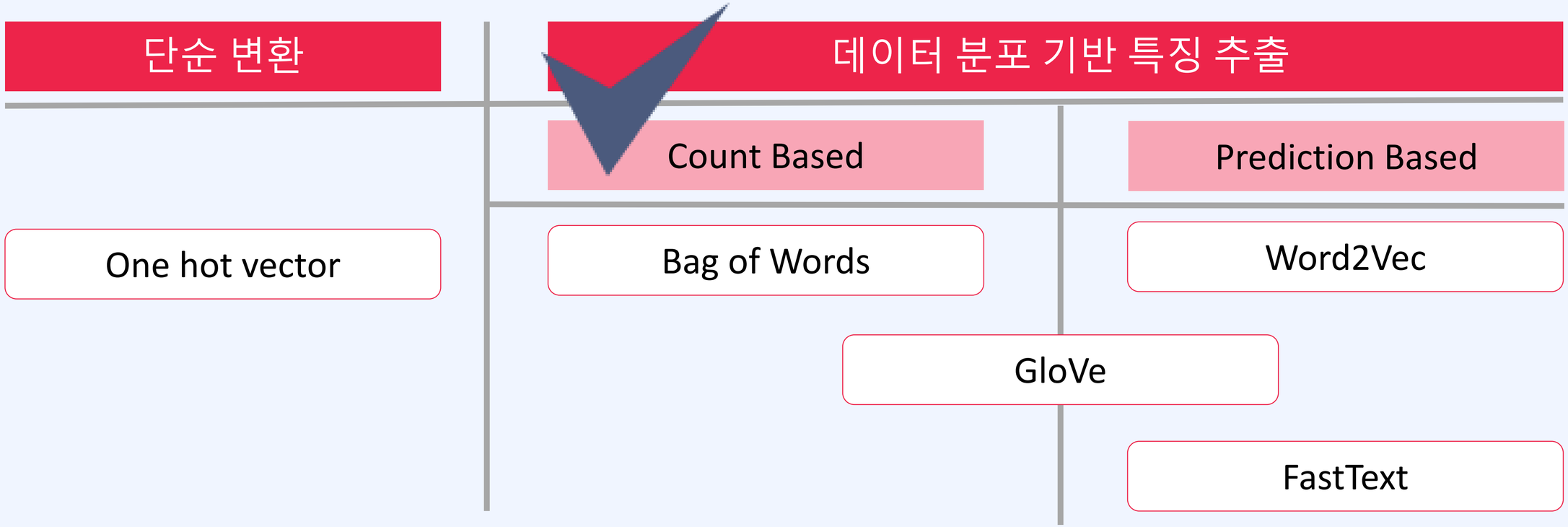


Embedding

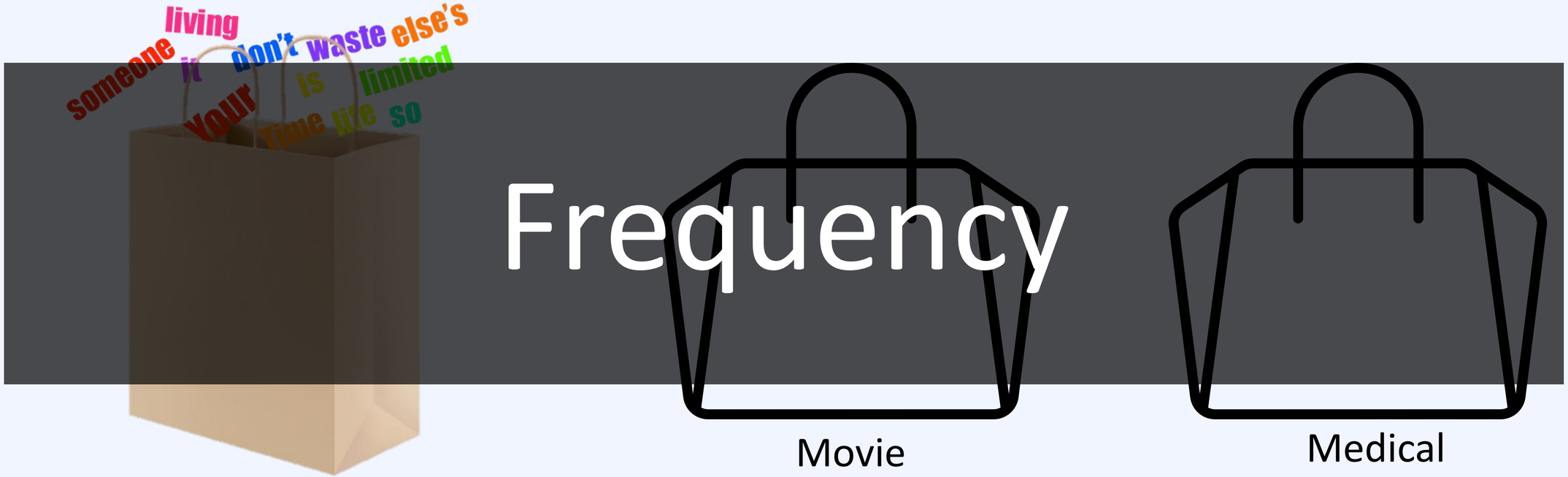
단어 빈도를 활용한 벡터 표현방법

Word Embedding

➢ Word Representation



Bag of Words



https://khuloodnasher.medium.com/bag-of-words-e2969110843

기본 빈도수

➢ CountVectorizer

```
from sklearn.feature_extraction.text import CountVectorizer
corpus = [
    'This is the first document.',
    'This document is the second document.',
    'And this is the third one.',
    'Is this the first document?',
]
vectorizer = CountVectorizer()
X = vectorizer.fit_transform(corpus)
vectorizer.get_feature_names_out()
```

```
array(['and', 'document', 'first', 'is', 'one', 'second', 'the', 'third',
       'this'], dtype=object)
```

```
print(X.toarray())
```

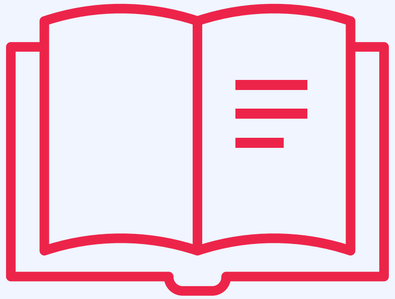
```
[[0 1 1 1 0 0 1 0 1]
 [0 2 0 1 0 1 1 0 1]
 [1 0 0 1 1 0 1 1 1]
 [0 1 1 1 0 0 1 0 1]]
```

https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.CountVectorizer.html#sklearn-feature-extraction-text-countvectorizer

TF-IDF

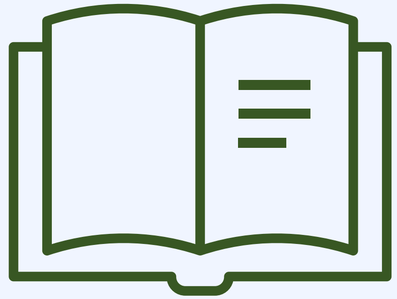
- TF(Term Frequency)
  - 단어가 문서에 나타난 횟수
- IDF(Inverse Document Frequency)
  - 단어가 포함된 문서의 수의 역수

$$\text{TF-IDF}(w,d) = \frac{\text{TF}(w,d)}{\text{DF}(w)}$$



옛날 옛적 한겨울에, 하늘에서 눈송이가 깃털처럼 내리고 있었습니다. 새로운 **왕비**는 신비한 거울을 가지고 있었는데, 그 앞에 서서 **거울**을 쳐다보면서 말했습니다.
"거울아, 벽에 있는 거울아 이 세상에서 누가 가장 아름답니?“

- 백설공주 中



옛날에 솜씨가 뛰어난 **악사**가 있었는데, 그의 **바이올린 연주**는 마음과 귀를 즐겁게 해주고 무척 감동적이었습니다. 한번은 그가 숲을 거닐고 있다가 너무 무료하여 중얼거렸습니다.
"나 혼자 있으니 너무 울적하군. 친구를 찾아봐야겠어."

- 떠돌이 악사 中

TF-IDF

- TF(Term Frequency)
  - 단어가 문서에 나타난 횟수
- IDF(Inverse Document Frequency)
  - 단어가 포함된 문서의 수의 역수

$$\text{TF-IDF}(w,d) = \frac{\text{TF}(w,d)}{\text{DF}(w)}$$

단어가 다른 문서에는 별로 등장하지 않고,
특정 문서에만 집중적으로 등장하는 단어야
말로 핵심어라 할 수 있다

옛날 옛적 한겨울에, 하늘에서 눈송이가 깃털처럼 내리고 있었습니다. 새로운 **왕비**는 신비한 거울을 가지고 있었는데, 그 앞에 서서 **거울**을 쳐다보면서 말했습니다.
"거울아, 벽에 있는 거울아 이 세상에서 누가 가장 아름답니?“

- 백설공주 中

옛날에 솜씨가 뛰어난 **악사**가 있었는데, 그의 **바이올린 연주**는 마음과 귀를 즐겁게 해주고 무척 감동적이었습니다. 한번은 그가 숲을 거닐고 있다가 너무 무료하여 중얼거렸습니다.
"나 혼자 있으니 너무 울적하군. 친구를 찾아봐야겠어."

- 떠돌이 악사 中

TF-IDF

➢ TfidfVectorizer

```
from sklearn.feature_extraction.text import TfidfVectorizer
corpus = [
    'This is the first document.',
    'This document is the second document.',
    'And this is the third one.',
    'Is this the first document?',
]
vectorizer = TfidfVectorizer()
X = vectorizer.fit_transform(corpus)
vectorizer.get_feature_names_out()
```

```
array(['and', 'document', 'first', 'is', 'one', 'second', 'the', 'third',
       'this'], dtype=object)
```

```
print(X.toarray())
```

```
[[0.         0.46979139 0.58028582 0.38408524 0.         0.
  0.38408524 0.         0.38408524]
 [0.         0.6876236  0.         0.28108867 0.         0.53864762
  0.28108867 0.         0.28108867]
 [0.51184851 0.         0.         0.26710379 0.51184851 0.
  0.26710379 0.51184851 0.26710379]
 [0.         0.46979139 0.58028582 0.38408524 0.         0.
  0.38408524 0.         0.38408524]]
```

https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html#sklearn.feature_extraction.text.TfidfVectorizer

Summary

➢ 단어 빈도를 활용한 벡터 표현방법

- CountVectorizer
- TfidfVectorizer

Quiz

Q **단어 빈도수를 기반으로 핵심어를 분석할 경우, 어디에나 자주 사용하는 단어는 오히려 중요성이 떨어지기 때문에 핵심어가 아닐 수 있다. 특정 문서에는 자주 나타나면서 말뭉치 전체에 걸쳐서는 자주 나타나지 않는 단어를 바탕으로 핵심어를 분석하는 것을 무엇이라고 하는가?**

① TF-IDF

② Topic Modeling

③ Tokenization

④ Stemming

해설

▶ 용어빈도-역문서빈도(TF-IDF)에서 용어빈도(TF)는 특정 단어가 특정 문서에 몇 번이나 출현했는지를 나타내는 지표이고, 문서빈도(DF)는 특정 단어가 전체 말뭉치에 몇 번이나 출현했는지를 나타내는 지표이다.

Quiz

Q 다음 중 빈도수 기반 단어 표현 방법은?

① SNA

② DTM

③ BoW

④ LSA

해설

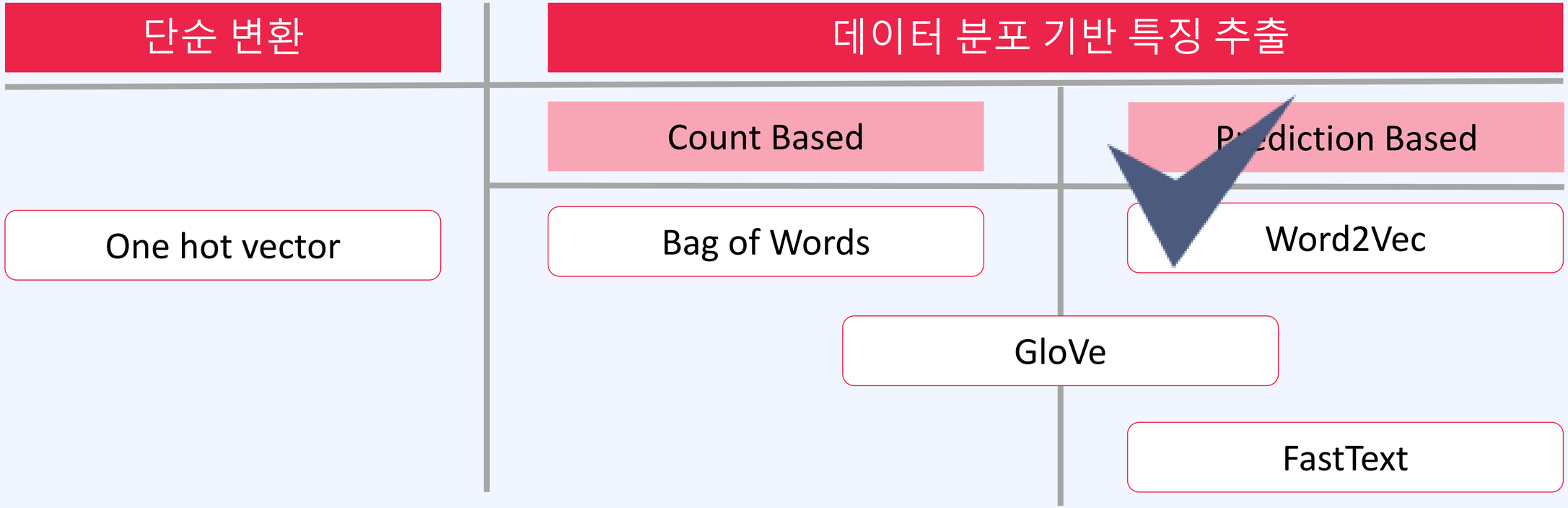
▶ BoW는 Bag of Words의 약자로 빈도수 기반 단어 표현 방법이다. DTM(Document Term Matrix)는 서로 다른 문서들의 단어를 하나의 매트릭스로 만들어서 분석하는 방법이다. LSA(Latent Semantic Analysis)는 잠재 의미 분석으로 주로 토픽모델링을 할 때 사용한다.

Embedding

Word2Vec

Word Embedding

➢ Word Representation



Word2Vec

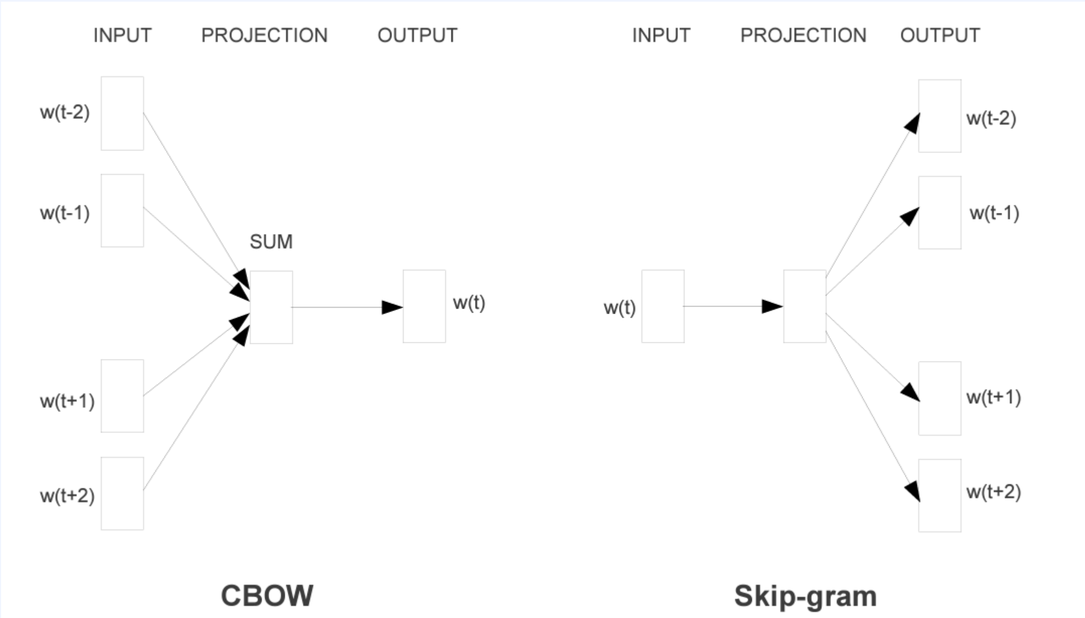
- Paper: Efficient Estimation of Word Representations in Vector Space(Tomas Mikolov et al., 2013)
- Goals of the paper

> We use recently proposed techniques for measuring the quality of the resulting vector representations, with the expectation that not only will similar words tend to be close to each other, but that words can have **multiple degrees of similarity** [20]. This has been observed earlier in the context of inflectional languages - for example, nouns can have multiple word endings, and if we search for similar words in a subspace of the original vector space, it is possible to find words that have similar endings [13, 14].

① 유사한 단어는 **근처에 위치**한다
② 단어는 **여러 개의 유사도**를 가질 수 있다

Word2Vec

- 신경망 기반의 벡터 생성방법
  - CBOW, Skip-gram
- **CBOW(Continuous Bag Of Words)**
  - 주변 단어를 통해 주어진 단어를 예측
- **Skip-gram**
  - 한 단어를 기준으로 주변에 올 수 있는 단어를 예측



Word2Vec

➢ CBOW(Continuous Bag Of Words)
- 주변 단어를 통해 주어진 단어를 예측

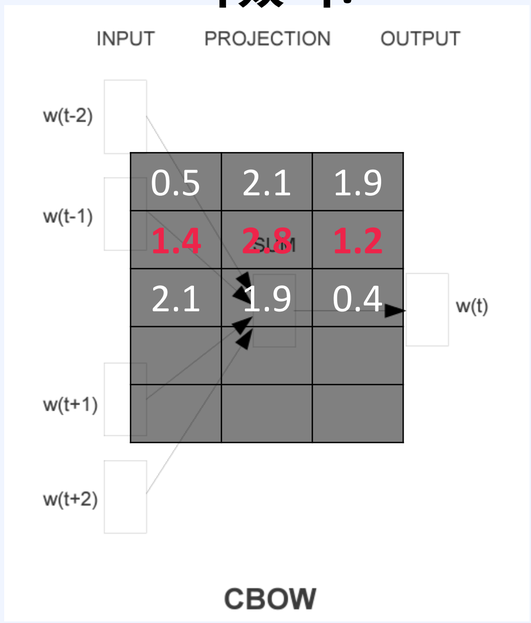
오늘 점심에 중국집에서 [자장면]을 먹었다.

[0, 0, 0, 0, 1] **오늘**

[0, 0, 0, 1, 0] **점심**

[0, 1, 0, 0, 0] **중국집**

[1, 0, 0, 0, 0] **먹었다**



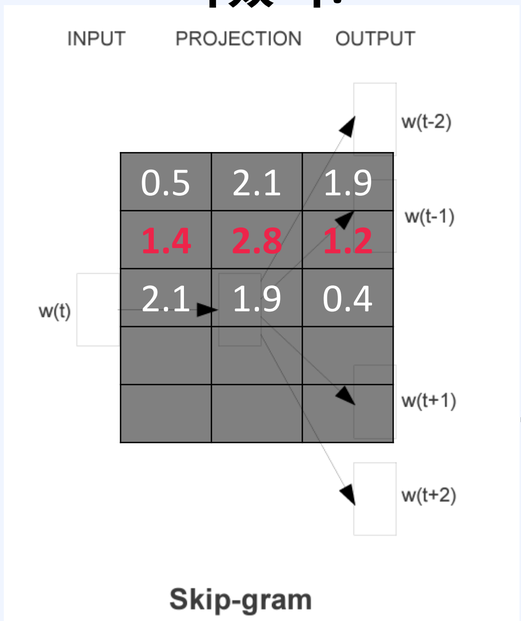
자장면 **[0, 0, 1, 0, 0]**

Word2Vec

➢ Skip-gram
- 한 단어를 기준으로 주변에 올 수 있는 단어를 예측

오늘 점심에 중국집에서 [자장면]을 먹었다.

[0, 0, 1, 0, 0] 자장면



오늘 [0, 0, 0, 0, 1]

점심 [0, 0, 0, 1, 0]

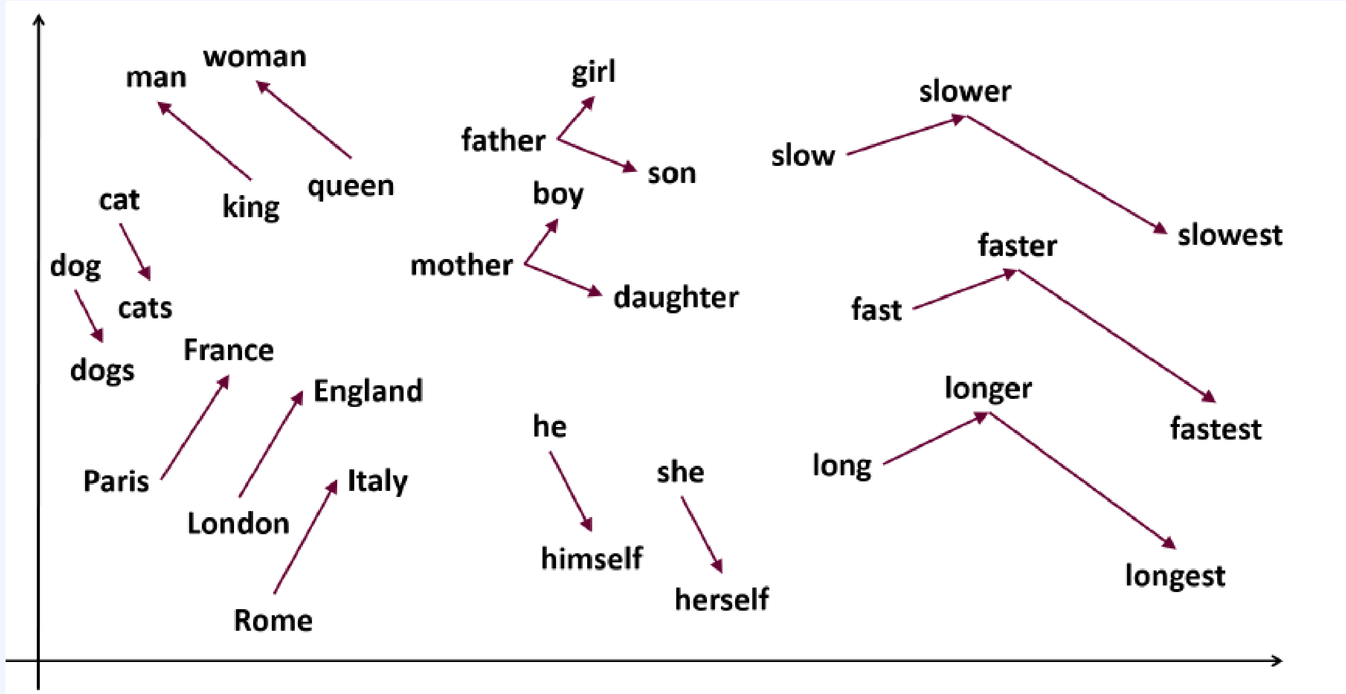
중국집 [0, 1, 0, 0, 0]

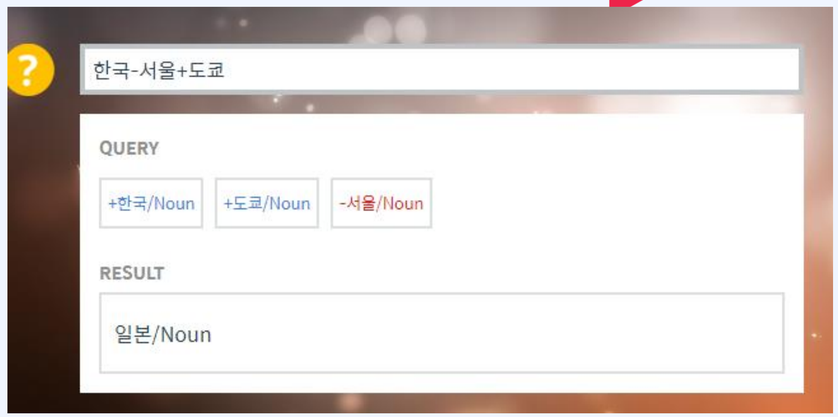
먹었다 [1, 0, 0, 0, 0]

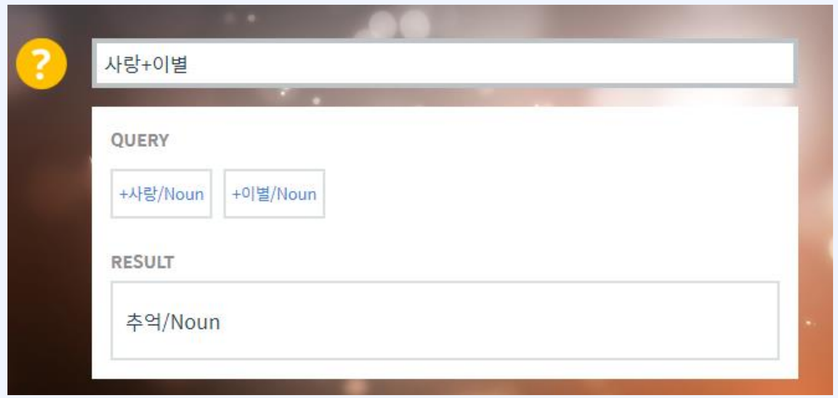
Word2Vec

➢ 단어의 의미가 벡터로 표현되므로 다음과 같은 벡터 연산이 가능

$$\vec{\omega}_{king} - \vec{\omega}_{man} + \vec{\omega}_{woman} \approx \vec{\omega}_{queen}$$



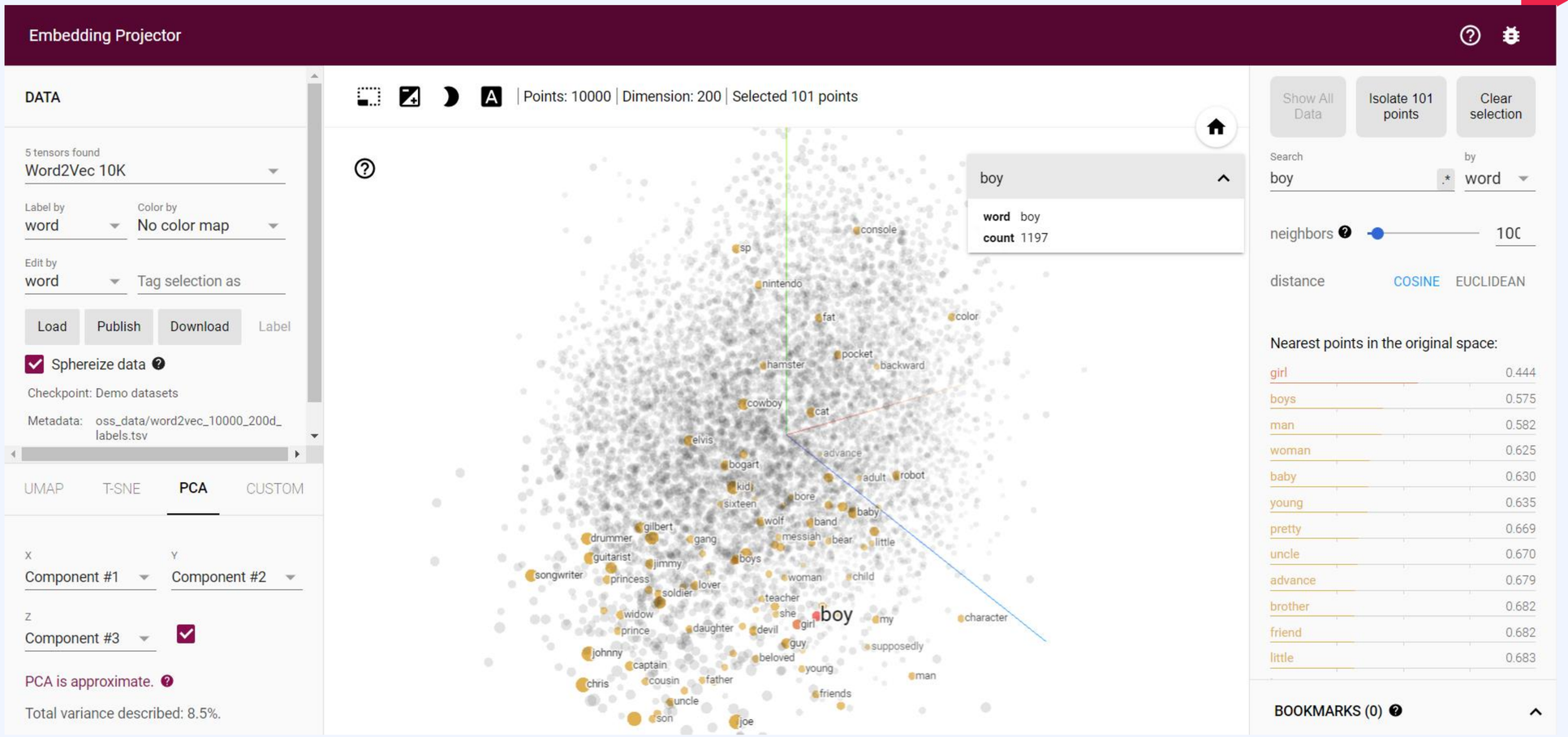




https://word2vec.kr/search/

Word2Vec

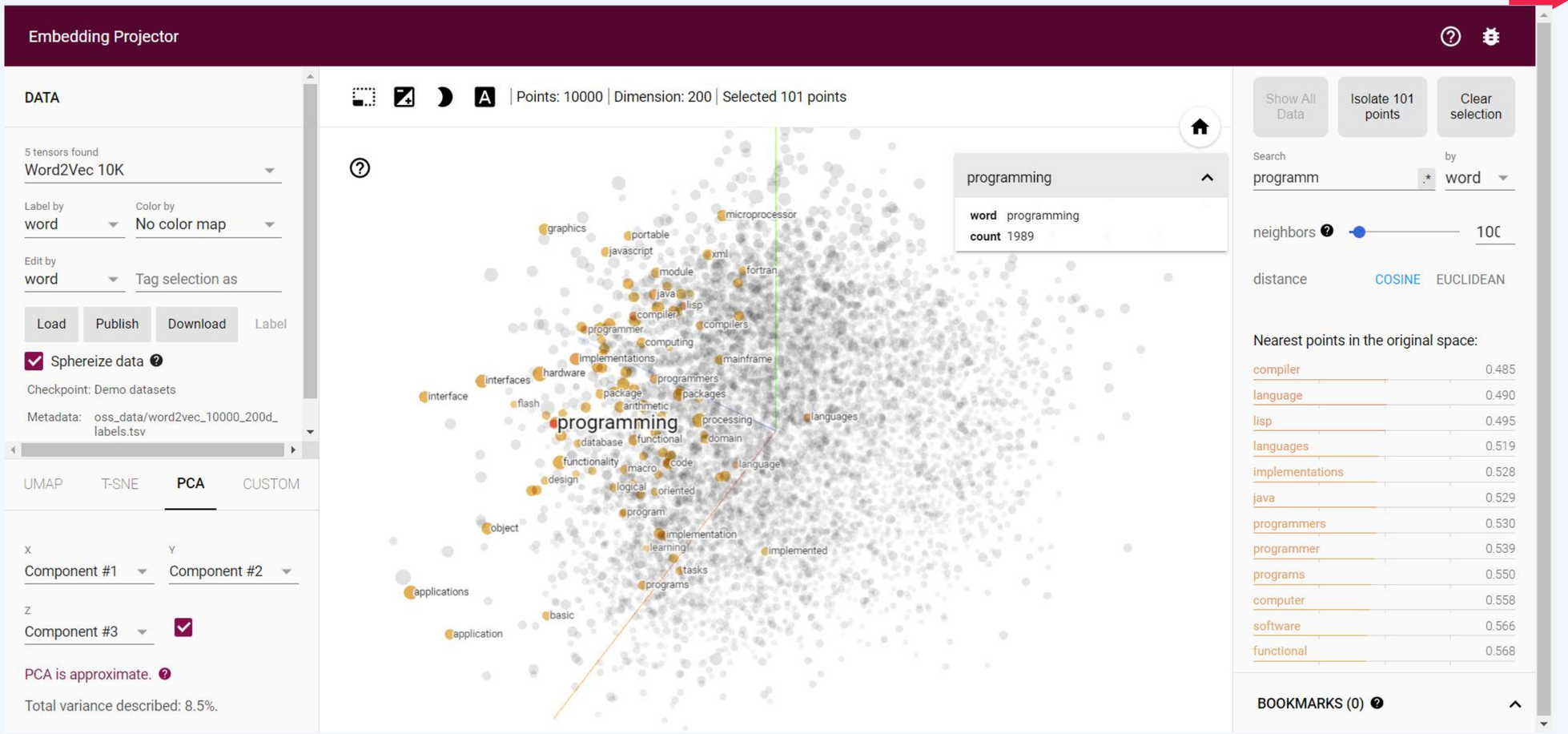
➢ boy



https://projector.tensorflow.org/

Word2Vec

➢ programming



https://projector.tensorflow.org/

Summary

➢ Word2Vec

- 중심 단어의 주변 단어들을 이용해 중심단어를 추론하는 방식으로 학습하여 단어가 가지는 의미 자체를 다차원 공간에 벡터화 하는 방법
- 단어 간의 유사도 측정이 가능
- 단어 간의 관계 파악을 할 수 있음
- 벡터 연산을 통한 추론이 가능 (예: $king - man + woman$ = ?)

Quiz

Q 다음 중 word2vec에 대한 설명으로 옳지 않은 것은?

① 신경망을 사용하여 단어 임베딩 벡터를 계산하는 방법이다.

② CBOW와 skip-gram 방식이 있다.

③ 비슷한 의미의 단어라고 하더라도, 단어 임베딩 결과는 차이가 많이 발생할 수 있다.

④ 특정 단어를 기준으로 윈도우 내의 주변 단어들을 사용하여 단어 임베딩을 학습한다.

해설

▶ word2vec을 사용하여 단어임베딩을 하면 비슷한 의미의 단어는 비슷한 벡터값을 가지게 된다.

Quiz

Q 다음 중 skip-gram 방식의 설명으로 옳지 않은 것은?

① word2vec 방식의 단어 임베딩 방식이다.

② 신경망을 사용하여 주변에 나타나는 단어들을 원핫 인코딩된 벡터로 입력받아 해당 단어를 예측한다

③ 대상 단어를 원핫 인코딩된 벡터로 입력받아 주변에 나타나는 단어를 예측한다.

④ 일반적으로 CBOW 방식보다 더 성능이 뛰어나다.

해설

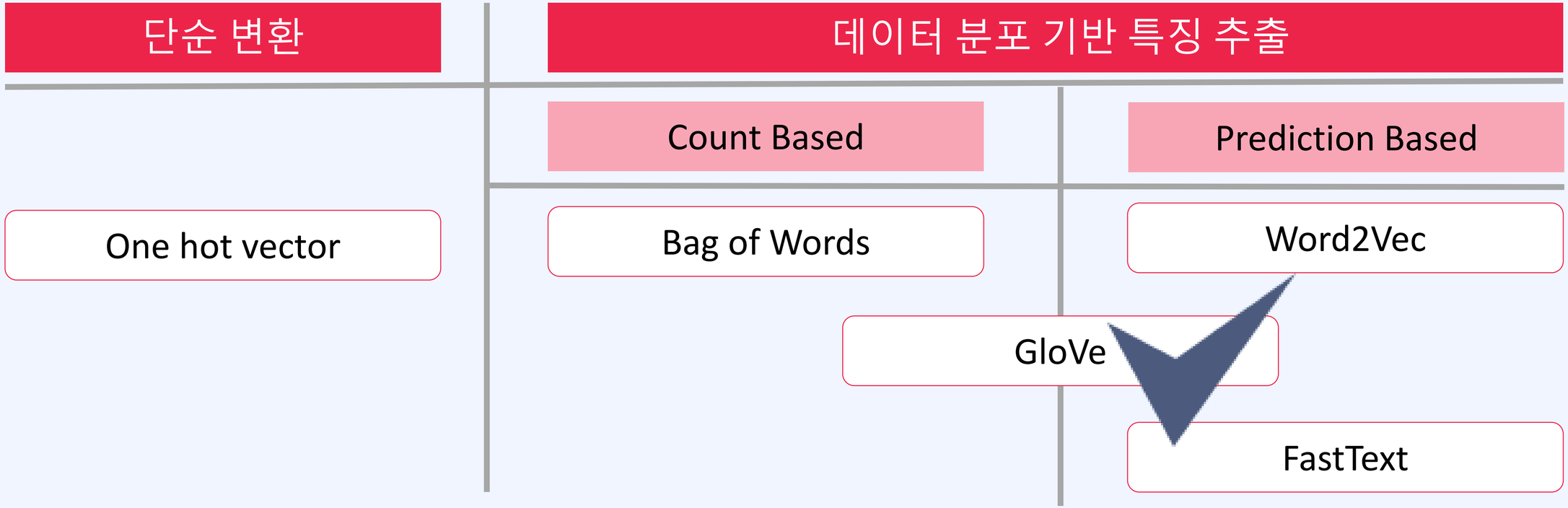
▶ CBOW(Continuous Bag Of Word)는 신경망을 사용하여 주변에 나타나는 단어들을 원핫 인코딩된 벡터로 입력받아 해당 단어를 예측한다

Embedding

GloVe, FastText

Word Embedding

➢ Word Representation



GloVe

- Paper: GloVe: Global Vectors for Word Representation (Jeffrey Pennington et al., 2014)
- 윈도우 내에 함께 출현한 단어들의 출현 빈도를 맞추도록 훈련

Table 1: Co-occurrence probabilities for target words *ice* and *steam* with selected context words from a 6 billion token corpus. Only in the ratio does noise from non-discriminative words like *water* and *fashion* cancel out, so that large values (much greater than 1) correlate well with properties specific to ice, and small values (much less than 1) correlate well with properties specific of steam.

Probability and Ratio	$k = solid$	$k = gas$	$k = water$	$k = fashion$
$P(k\|ice)$	1.9×10^{-4}	6.6×10^{-5}	3.0×10^{-3}	1.7×10^{-5}
$P(k\|steam)$	2.2×10^{-5}	7.8×10^{-4}	2.2×10^{-3}	1.8×10^{-5}
$P(k\|ice)/P(k\|steam)$	8.9	8.5×10^{-2}	1.36	0.96

ice와 solid가 함께 나타날 확률은
steam과 solid가 함께 나타날 확률의 8.9배이다.

GloVe

- Paper: GloVe: Global Vectors for Word Representation (Jeffrey Pennington et al., 2014)
- 윈도우 내에 함께 출현한 단어들의 출현 빈도를 맞추도록 훈련

In this work, we analyze the model properties necessary to produce linear directions of meaning and argue that global log-bilinear regression models are appropriate for doing so. We propose a specific weighted least squares model that trains on global word-word co-occurrence counts and thus makes efficient use of statistics. The model produces a word vector space with meaningful substructure, as evidenced by its state-of-the-art performance of 75% accuracy on the word analogy dataset. We also demonstrate that our methods outperform other current methods on several word similarity tasks, and also on a common named entity recognition (NER) benchmark.

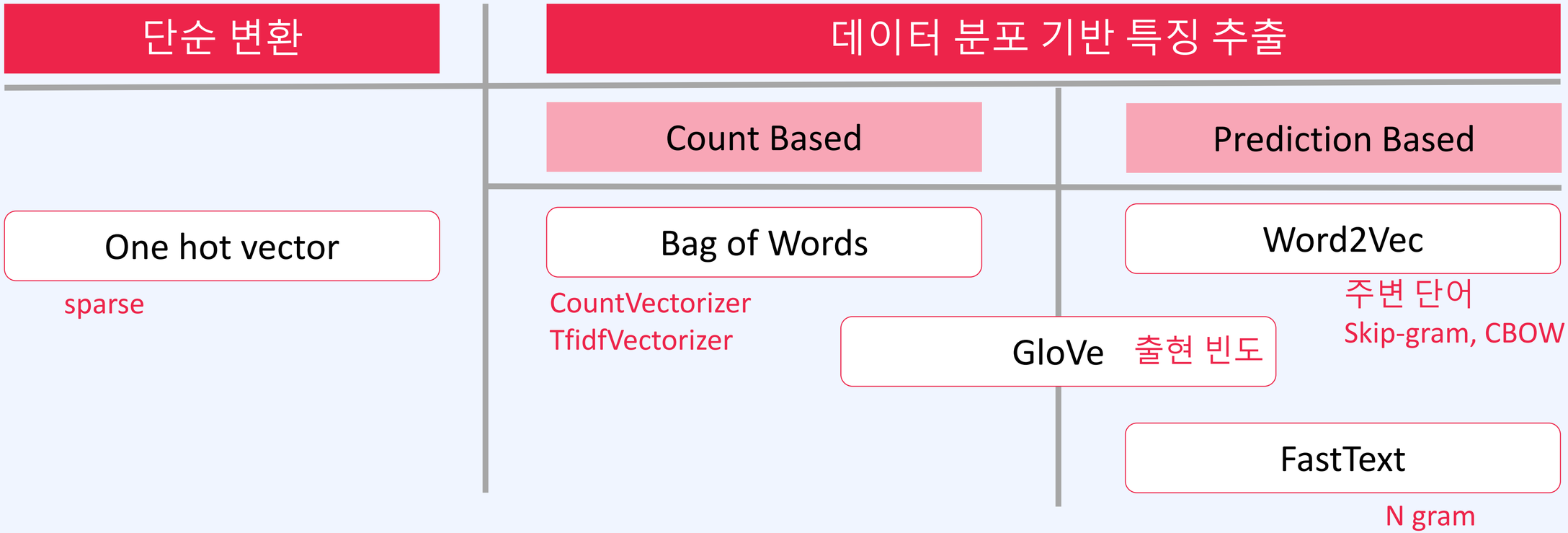
FastText

- Docs: https://fasttext.cc/
- Paper: Enriching Word Vectors with Subword Information(Bojanowski et al., 2016)
- Why
  - 기존의 word2vec은 문서에 자주 나타나지 않은 단어에 대한 학습과 OOV(Out of Vocabulary)에 대한 대처가 어려웠음
- What
  - 단어를 n-gram으로 분리한 후, 모든 n-gram vector를 합산한 후, 평균을 통해 단어 벡터를 획득

> Each word w is represented as a bag of character n-gram. We add special boundary symbols < and > at the beginning and end of words, allowing to distinguish prefixes and suffixes from other character sequences. We also include the word w itself in the set of its n-grams, to learn a representation for each word (in addition to character n-grams). Taking the word *where* and $n = 3$ as an example, it will be represented by the character n-grams:
>
> `<wh, whe, her, ere, re>`

Summary

➢ Word Embedding



Quiz

Q 다음 중 단어 임베딩(word embedding) 모형에 해당하지 않는 것은?

① Word2Vec

② FastText

③ GloVe

④ CNN

해설

▶ CNN은 Convolution Neural Network의 약자로 합성곱 연산을 수행하는 신경망을 의미한다.

Quiz

Q **다음 중 Glove에 대한 설명으로 옳지 않은 것은**

① 대상 단어에 대해서 코퍼스에 함께 나타난 단어별 출현 빈도를 예측하는 형태로 손실함수에 가중치를 부여한다.

② skip-gram에 비해서 학습 속도가 빠르다.

③ skip-gram에 비해서 출현 빈도가 적은 단어들은 부정확한 결과를 보인다.

④ 단어를 벡터로 변환하는 워드 임베딩 방식이다.

해설

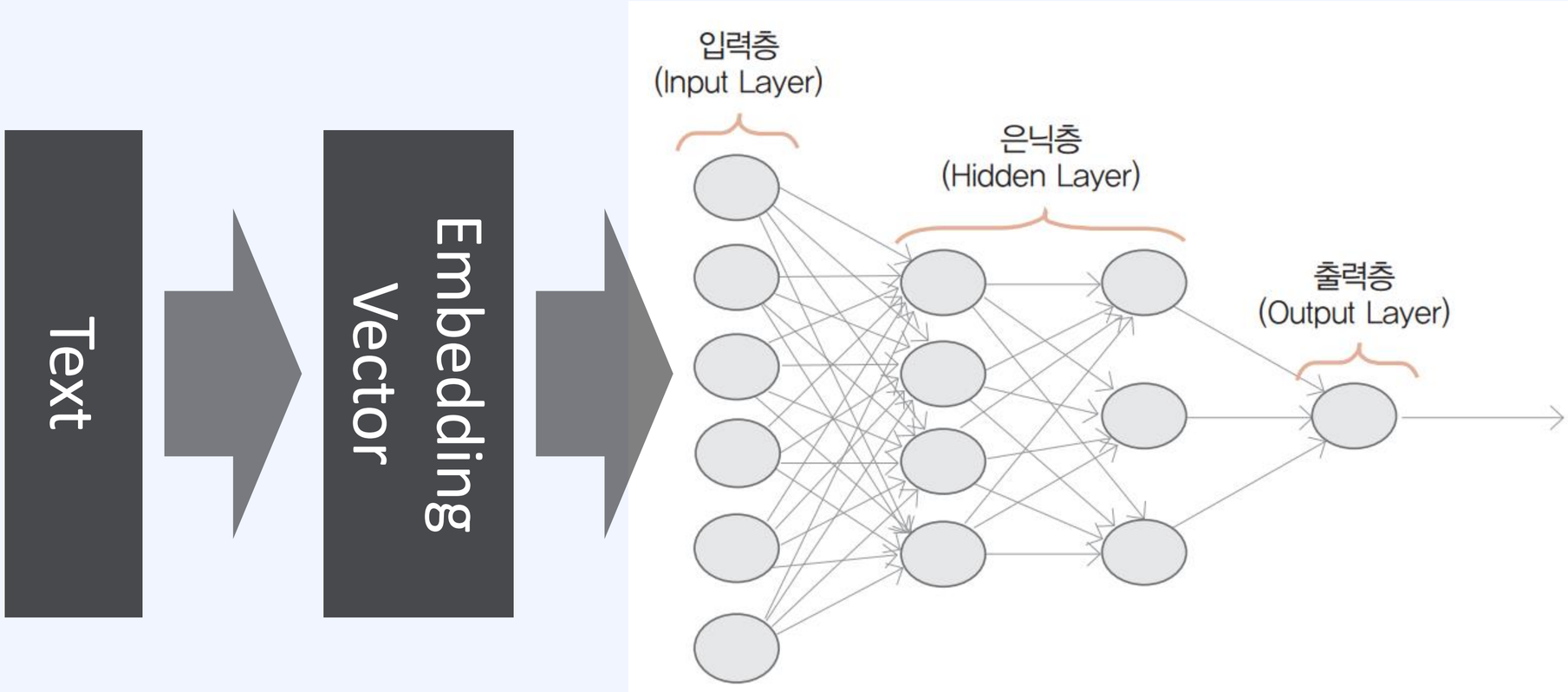
▶ Glove는 skip-gram에서 나타난 출현 빈도가 적은 단어들의 성능이 떨어진 부분을 개선한 워드 임베딩 방식이다.

Embedding

Embedding Layer

Embedding Layer

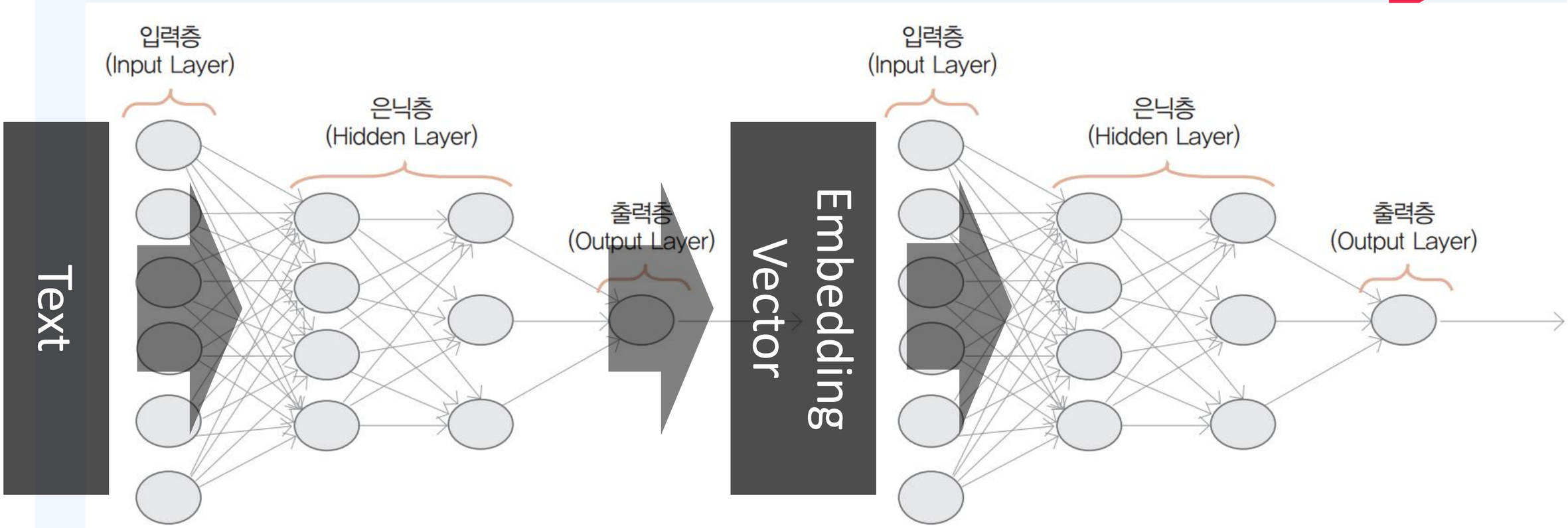
➢ Embedding Vector



이미지출처: 파이썬으로 시작하는 머신러닝+딥러닝(아이리포)

Embedding Layer

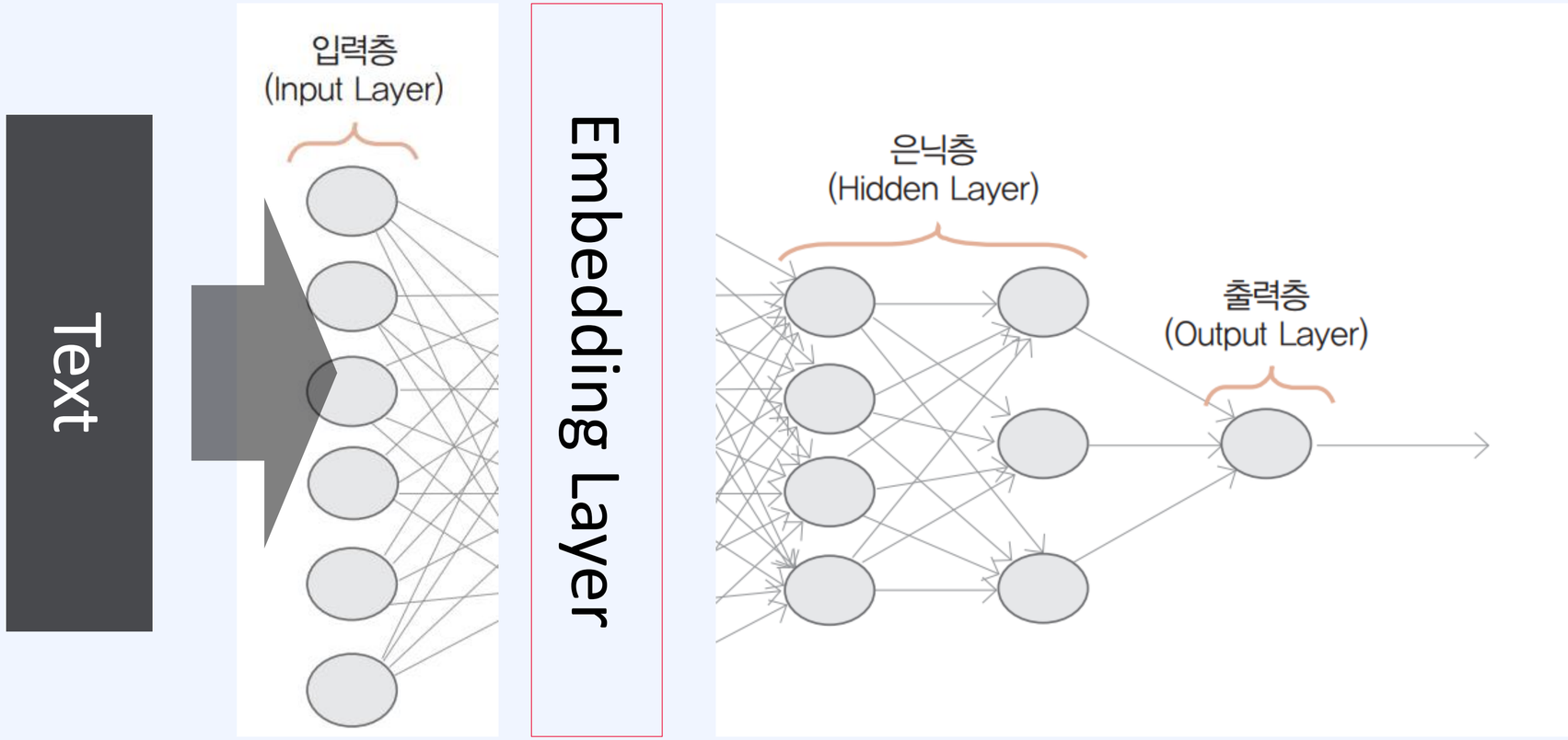
➢ Embedding Vector



이미지출처: 파이썬으로 시작하는 머신러닝+딥러닝(아이리포)

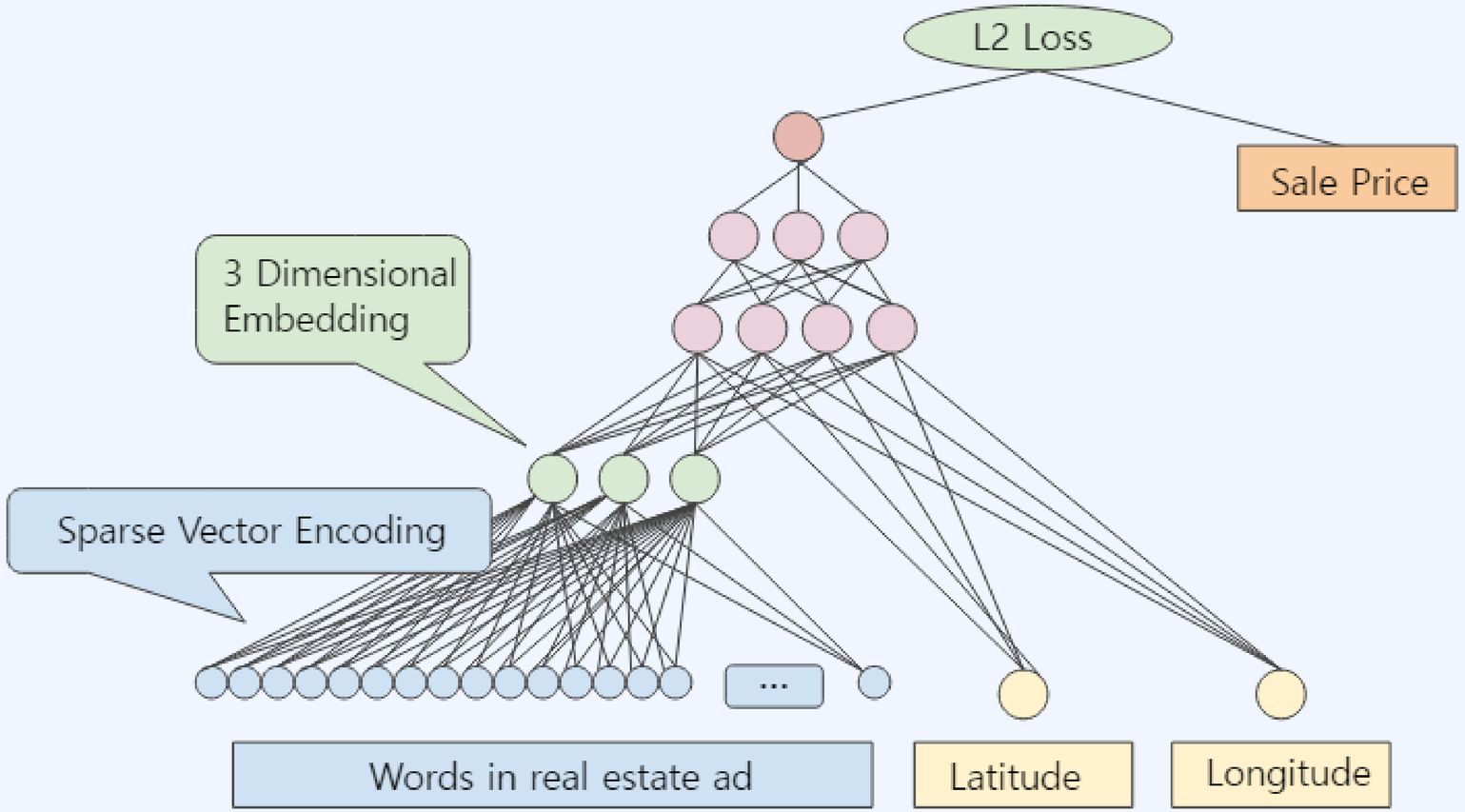
Embedding Layer

➢ Embedding Vector



이미지출처: 파이썬으로 시작하는 머신러닝+딥러닝(아이리포)

Embedding Layer



Code 맛보기

```
model = Sequential()
model.add(Embedding(2000, 128))
model.add(Dropout(0.5))
model.add(Conv1D(64, 5, padding='valid', activation='relu',strides=1))
model.add(MaxPooling1D(pool_size=4))
model.add(LSTM(55))
model.add(Dense(1, activation = 'sigmoid’))

model.compile(loss='binary_crossentropy’, optimizer='adam’, metrics=['accuracy'])
model.fit(x_train, y_train, batch_size=100, epochs=5, validation_data=(x_test, y_test))
```

Summary

➢ Embedding layer

- 미리 학습을 통해 얻어진 word embedding vector를 신경망에 넣어 주는 것이 아니라
- 학습을 통해서 embedding vector 를 획득하는 방법