# Word2Vec

곽지호

# 텍스트 단어를 분석하려면?

단어들을 벡터로 만들어야 한다.


# 가장 쉽고 고전적인 방법!

One-Hot Encoding

저는 데이터 분석을 좋아합니다

['저', '는', '데이터', '분석', '을', '좋아합니다']

| - | 0 | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|---|
| 저 | 1 | 0 | 0 | 0 | 0 | 0 |
| 는 | 0 | 1 | 0 | 0 | 0 | 0 |
| 데이터 | 0 | 0 | 1 | 0 | 0 | 0 |
| 분석 | 0 | 0 | 0 | 1 | 0 | 0 |
| 을 | 0 | 0 | 0 | 0 | 1 | 0 |
| 좋아합니다 | 0 | 0 | 0 | 0 | 0 | 1 |

# One-Hot Encoding의 한계는 무엇이죠?

1. Sparce Matrix (희소 행렬) 이 만들어진다.
   a. 주어진 Corpus 즉, 문서의 단어 개수가 N이라면 N차원 공간이 필요하다.
   b. But, 1개 차원의 벡터만 1로 기록되고, 나머지 차원은 모두 0으로 채워진다.

**차원의 저주 : 큰공간에 적은량의 정보는 과적합을 일으킵니다.**

1. 단어 간의 관계를 표현하지 못하고 모두 독립적이라고 가정한다.
   b. '데이터'와 '분석'의 관계나, '데이터'와 '좋아합니다'의 관계가 동일하게 간주된다.
   c. 수학적으로 각 단어는 차원공간상에서 서로 직교한다.

**단어가 가지는 의미와 맥락을 알아내지 못합니다.**

# 그렇다면 인코딩이 아닌 임베딩이다!

1. Embedding 이란?
   a. Dense Representation : 특정 차원수, 즉 특징수에 각 단어를 분산해서 표현.
   b. 각 차원은 임의의 특징벡터를 표현하여 각 차원에 분산해서 표현
   c. 차원의 의미는 알수 없음.
   d. Sparce Representation을 하는 one-hot encoding보다 저차원으로 표현가능

| - | 0 | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|---|
| 데이터 | 0 | 0 | 1 | 0 | 0 | 0 |
| 분석 | 0 | 0 | 0 | 1 | 0 | 0 |

차원축소

| - | f1 | f2 | f3 | f4 |
|---|---|---|---|---|
| 데이터 | 0.14 | 0.73 | 0.13 | -1.2 |
| 분석 | 0.13 | -0.5 | 0.4 | 1.1 |

특성요소 f1에 대해 두 단어가 유사하다고 해석 가능

# 임베딩 벡터는 어떻게 얻을수 있나요?

## 출현빈도 기반 임베딩

## (Frequency based Embedding)

**BoW → TDM → LSA or LDA**

문서별 단어 출현 빈도 카운트

**TF-IDF → TDM → LSA or LDA**

해당문서 출현빈도와 문서전체 출현빈도 비율

**출력 = 축소된 N차원 X 단어종류수**

## 뉴럴네트워크 기반 임베딩

## (NN based Embedding)

**Word2Vec**

N차원의 1개 은닉층의 두고 연관단어 예측

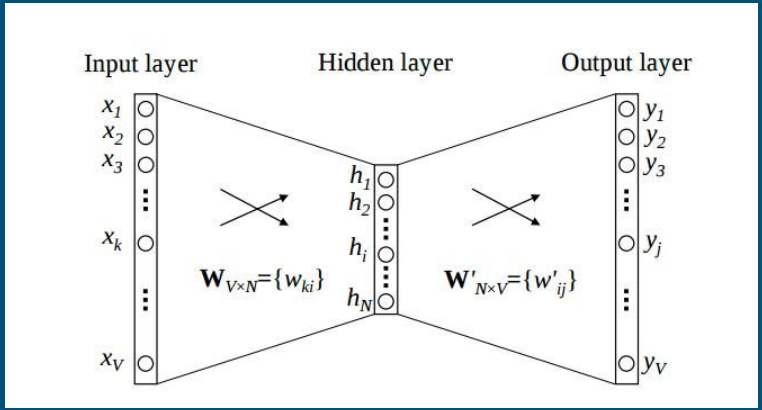
**FastText, GloVe, ELMo**

글자단위, 통계하이브리드, 다의어처리

**출력 = 단어종류수 X N차원**

# Word2Vec 아이디어

1. 맥락으로 단어를 예측할 수 있다는 predictive method 에 기반
   a. ‘나는 사과를 마트에서 자주 OO’ → “구매한다”
2. 신경망 역전파 알고리즘에 의해 지도학습 방식으로 가중치벡터 W를 획득
   a. 지도학습을 통해 비지도학습 문제해결을 달성



# Word2Vec 알고리즘

## CBOW

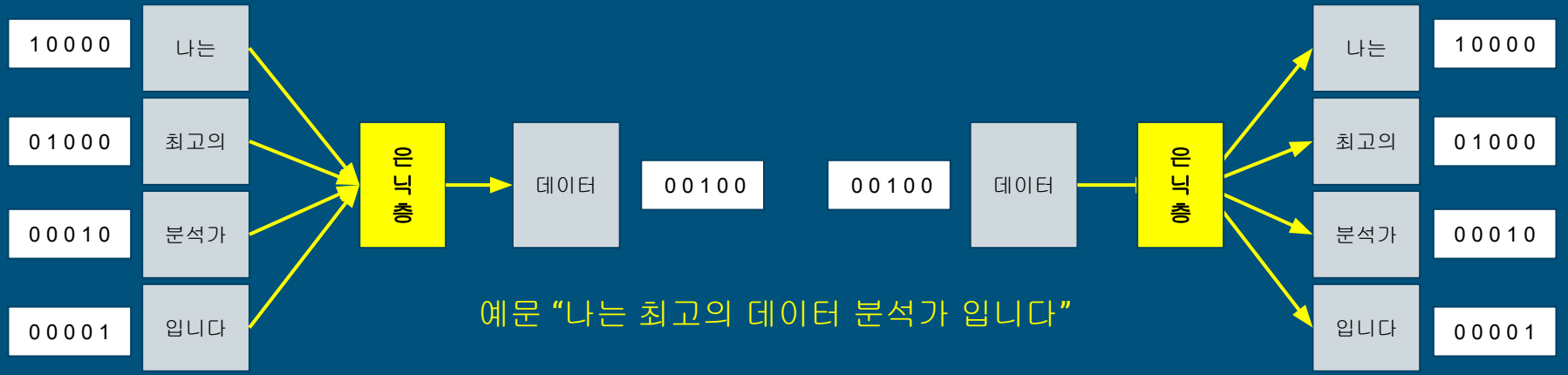
**주변단어로 중심단어 예측**

Input : 중심단어  Label : 주변단어

## Skip-gram

**중심단어로 주변단어 예측**

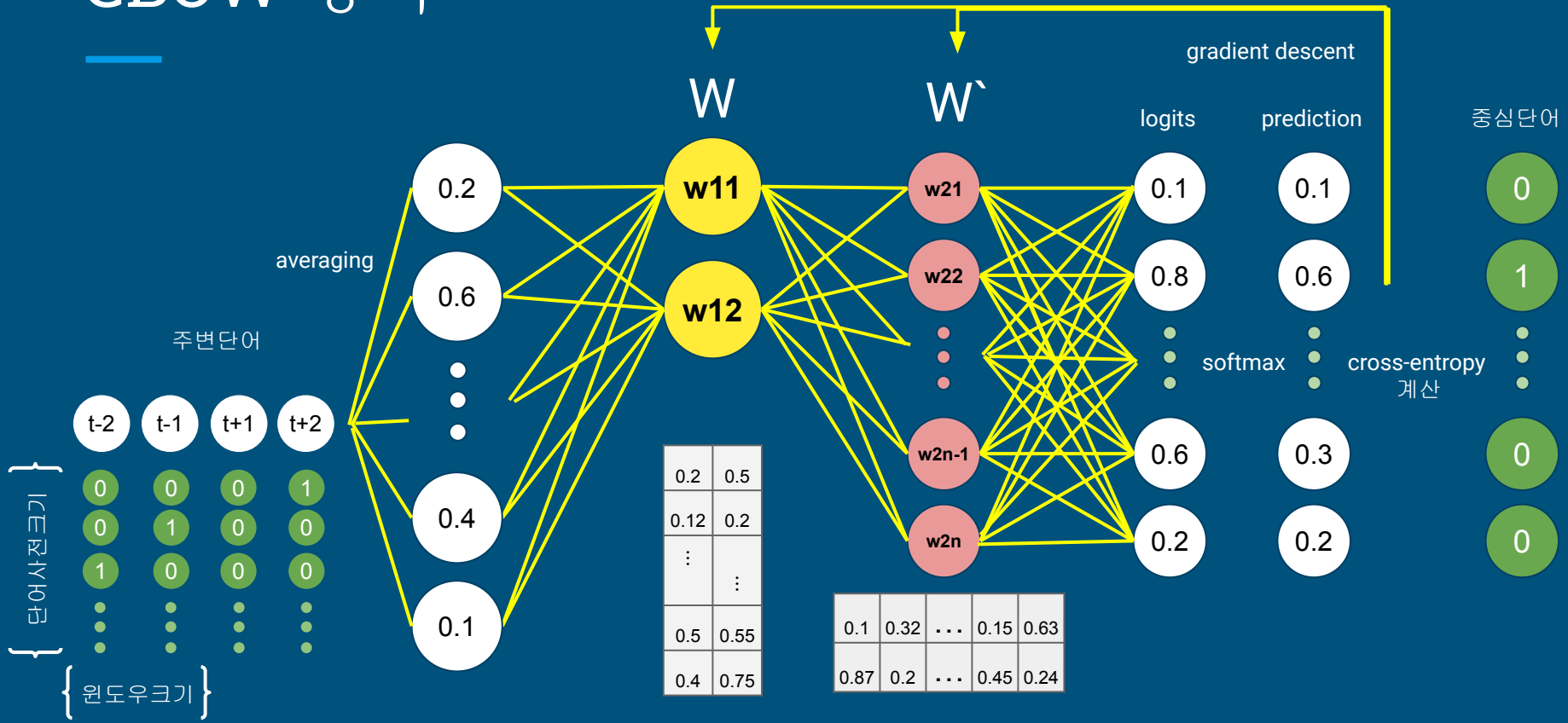
Input : 주변단어 Set  Label : 중심단어




예문 “나는 최고의 데이터 분석가 입니다”

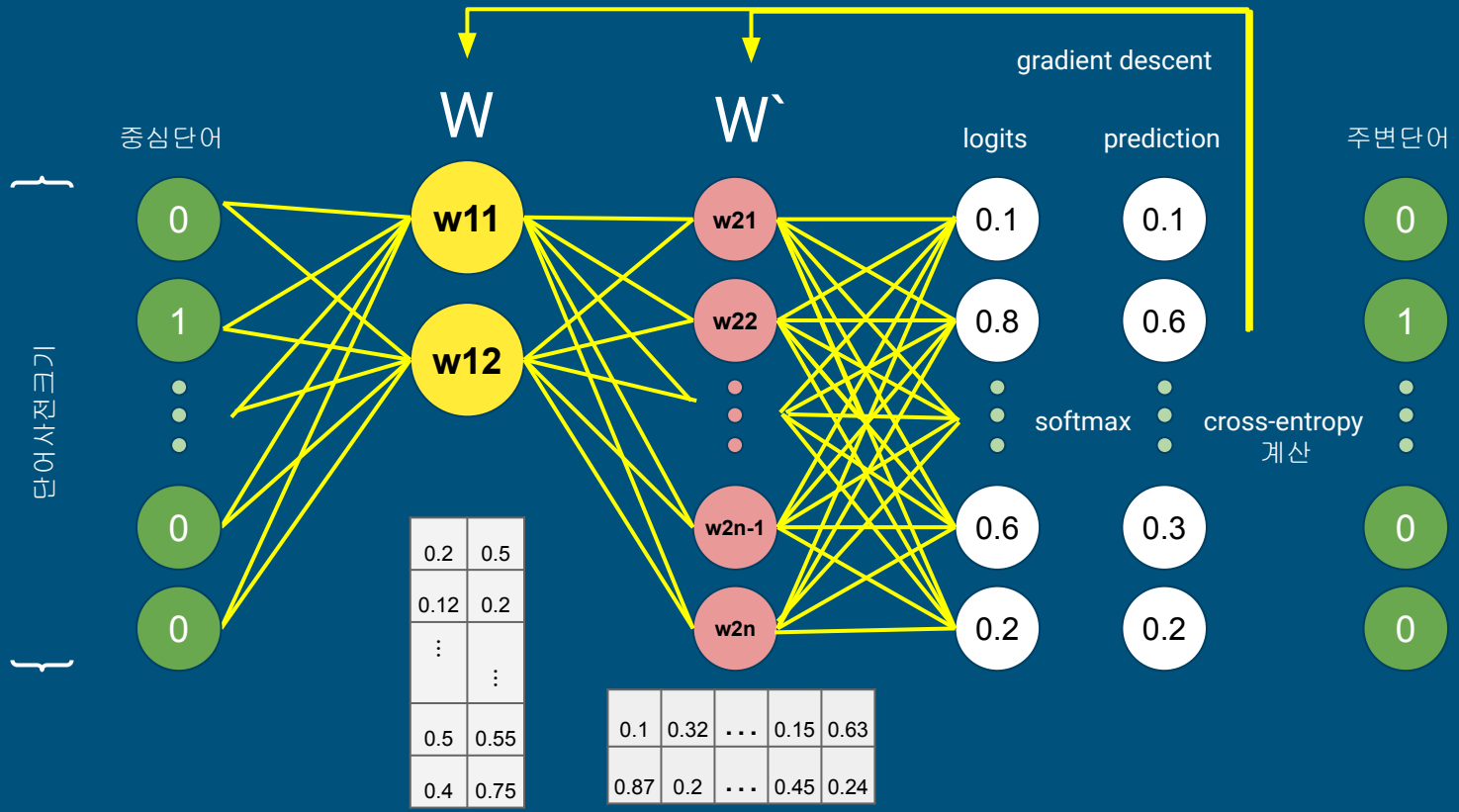
# CBOW 방식

- 윈도우사이즈 2의 CBOW 2차원 임베딩 개념도 예시



# Skip-gram 방식

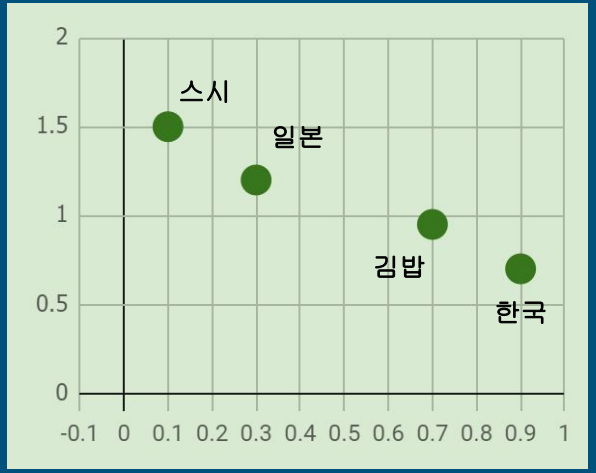
- 윈도우사이즈 2의 skim-gram 2차원 임베딩 개념도 예시



# 결국 단어 유사도를 파악할 수 있게된다

| one-hot encoding | | | | |
|---|---|---|---|---|
| 스시 | 1 | 0 | 0 | 0 | 0 |
| 일본 | 0 | 1 | 0 | 0 | 0 |
| 김밥 | 0 | 0 | 0 | 0 | 1 |
| 한국 | 0 | 0 | 0 | 1 | 0 |

| word2vec embedding | | |
|---|---|---|
| - | 1차원 | 2차원 |
| 스시 | 0.1 | 1.5 |
| 일본 | 0.3 | 1.2 |
| 김밥 | 0.7 | 0.95 |
| 한국 | 0.9 | 0.7 |




| 코사인 유사도 행렬 | | | | |
|---|---|---|---|---|
| - | 스시 | 일본 | 김밥 | 한국 |
| 스시 | 1.00 | 0.98 | 0.84 | 0.67 |
| 일본 | 0.98 | 1.00 | 0.92 | 0.79 |
| 김밥 | 0.84 | 0.92 | 1.00 | 0.96 |
| 한국 | 0.67 | 0.79 | 0.96 | 1.00 |

스시 - 일본 + 한국 = 김밥

# Word2Vec 시각화 시뮬레이션

https://ronxin.github.io/wevi/