단어의 표현 (Word Representation)

자연어처리 텍스트마이닝

단어의 표현이 필요한 이유

자연어 처리를
시작해보겠습니다.

문자열

- 확률 계산
- 더하고
- 빼고
- 등 ...

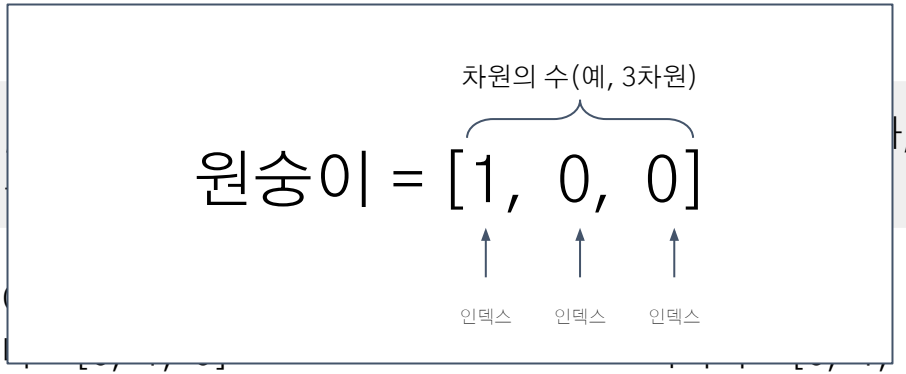
수학적 연산

원핫-인코딩
(One-Hot-Encoding)

단어의 표현(Word Representation)

원핫-인코딩(One-Hot-Encoding)

원핫-인코딩은 단어(word)를 숫자로 표현하고자 할 때 적용할 수 있는 간단한 방법론



원숭이
바나
사과 = [0, 0, 1]

0, 0]
0, 0]
사과 = [0, 0, 1, 0]
코끼리 = [0, 0, 0, 1]

원핫-인코딩(One-Hot-Encoding) 한계점

1) 차원 크기의 문제

원숭이, 바나나, 사과를
표현할 때

원숭이 = [1, 0, 0]
바나나 = [0, 1, 0]
사과 = [0, 0, 1]

단어의 수만큼 차원이 필요함

단어수가 많아진다면?

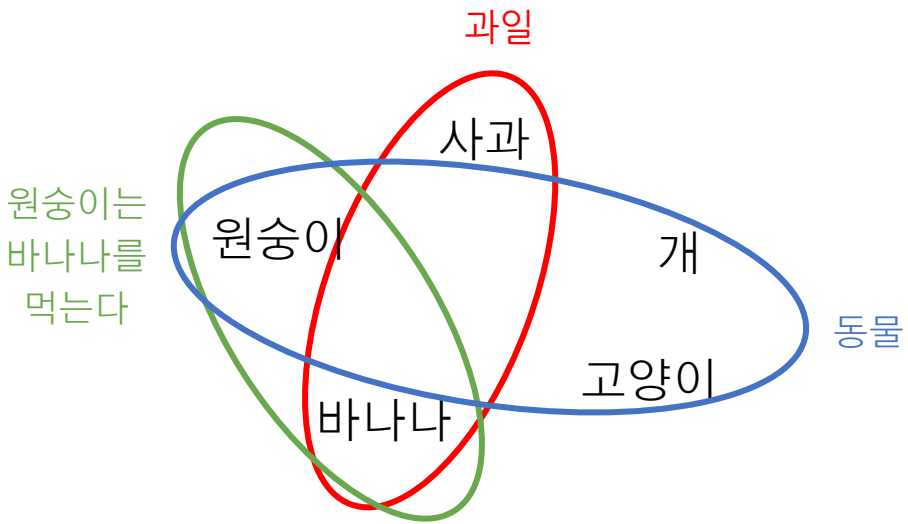
2017년 표준국어대사전에 등재된 단어 수 약 50만개
=> 50만개의 차원이 필요

원숭이 = [1, 0, 0, 0, 0, 0, 0, 0, 0, ... 0, 0, 0, 0, 0]

50만 차원 벡터

원핫-인코딩(One-Hot-Encoding) 한계점

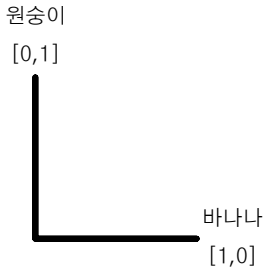
2) 의미를 담지 못하는 문제



원핫-인코딩(One-Hot-Encoding) 한계점

2) 의미를 담지 못하는 문제

$$\text{similarity} = \cos(\theta) = \frac{A \cdot B}{\|A\|\|B\|} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n} (A_i)^2} \times \sqrt{\sum_{i=1}^{n} (B_i)^2}}$$



$$\text{similarity} = \frac{(0 \times 1) + (1 \times 0)}{(1^2 + 0^2) \times (0^2 + 1^2)} = 0$$

원핫-인코딩(One-Hot-Encoding) 한계점

2) 의미를 담지 못하는 문제

원숭이, 바나나, 사과, 개, 고양이를 표현할 때

원숭이 = [1, 0, 0, 0, 0]
바나나 = [0, 1, 0, 0, 0]
사과 = [0, 0, 1, 0, 0]
개 = [0, 0, 0, 1, 0]
고양이 = [0, 0, 0, 0, 1]

- “원숭이, 사과” 코사인 유사도 : 0
- “원숭이, 바나나” 코사인 유사도 : 0
- “개, 고양이” 코사인 유사도 : 0

=> 원핫 벡터간 코사인 유사도는 모두 0

=> 따라서 의미를 분간 하기 어려움

BoW
(Bag of Words)

문서의 표현(Document Representation)

TF-IDF
(Term Frequency-Inverse Document Frequency)

단어의 표현(Word Representation)

단어 임베딩
(Word Embedding)

단어의 표현(Word Representation)

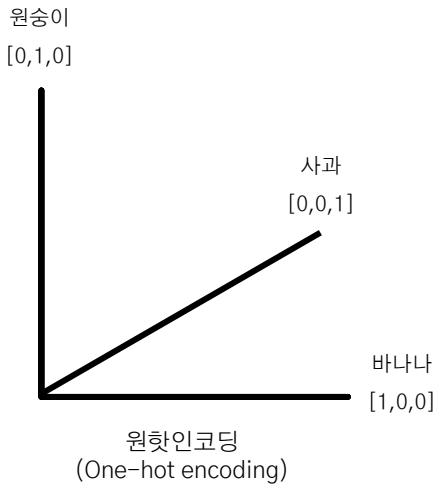
원핫-인코딩(One-Hot-Encoding) 한계점

벡터로 표현한 단어 차원이 너무 큼	연산이 낭비되어 모델 학습에 불리하게 적용
단어 의미를 담지 못함	분석을 효과적으로 수행할 수 없음

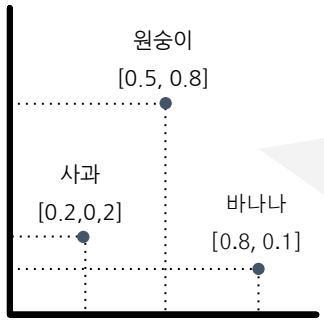
단어 임베딩(Word embedding)

단어 임베딩은 단어의 의미를 간직하는 밀집 벡터(Dense Vector)로 표현하는 방법

원숭이, 바나나, 사과를 표현할 때



원핫인코딩
(One-hot encoding)



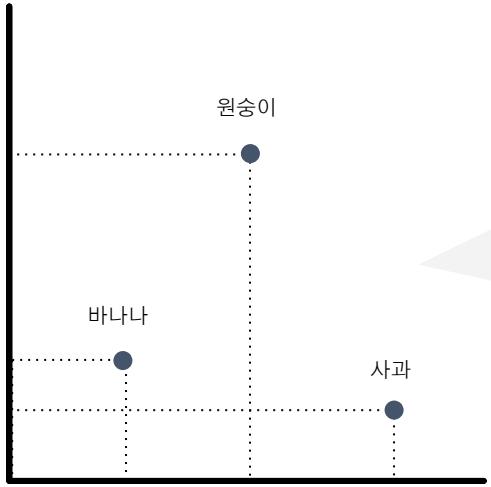
밀집 벡터
(Dense Vector)

- 벡터가 공간에 꽉차 있음
- 새로운 단어 추가시 차원을 추가할 필요가 없음
  => 차원을 줄일 수 있음
  => 추후 분류나 예측 모델을 학습할 때 연산을 줄일 수 있는 이점을 갖음

단어 임베딩 (Word Embedding)의 한계

벡터로 표현한 단어 차원이 너무 큼

밀집 벡터(Dense vector)로 해결



사과 벡터는 어디에
표현되는 것이 맞을까요?

=> 단어를 벡터로
표현하는 명확한 방법이
존재하지 않음

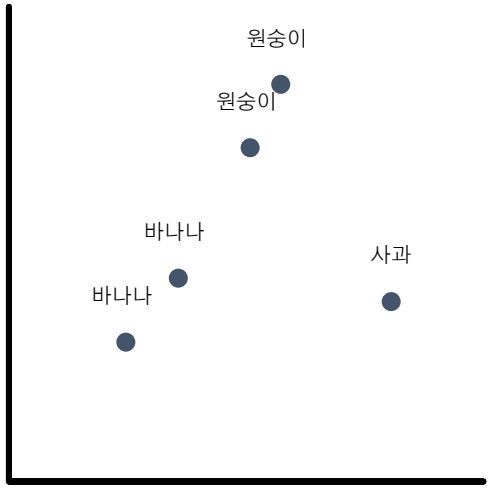
단어 의미를 담지 못함

?

밀집 벡터를 만드는 방법

분포 가설이란,

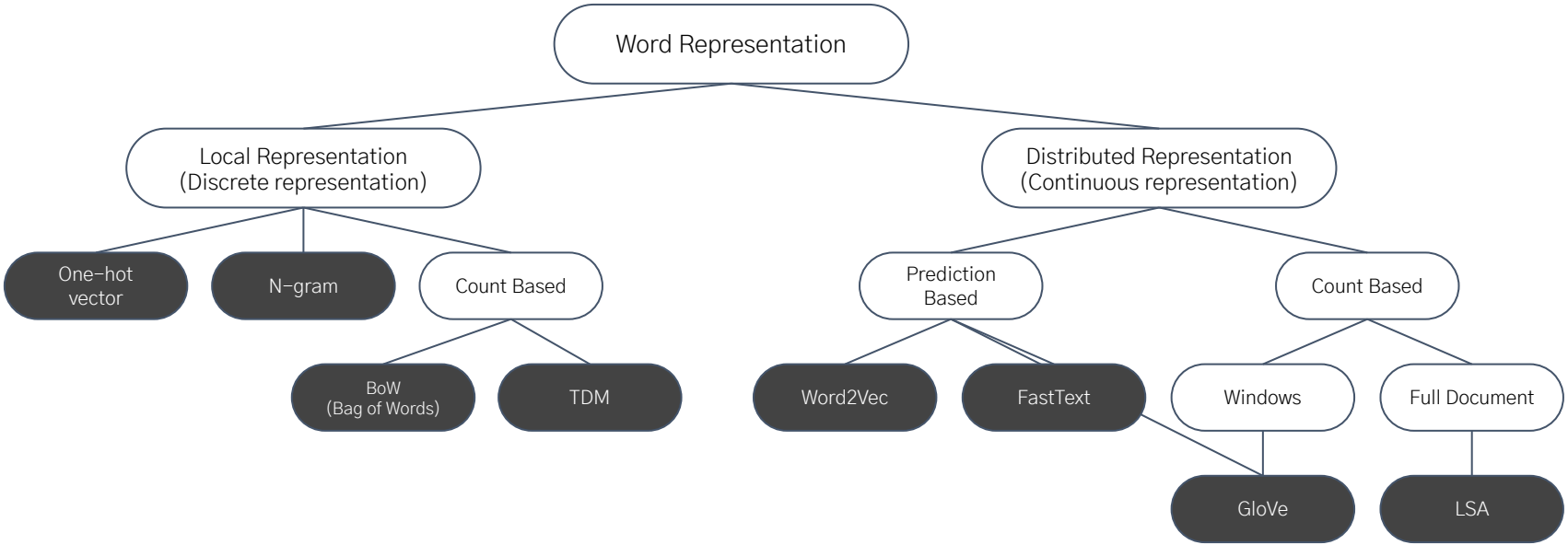
'같은 문맥에서 등장하는 단어는 유사한 의미를 지닌다'



1) 임의의 위치에 벡터 생성

2) 같은 문맥이 등장하는 단어를 더 가까이 표현

Word Representation

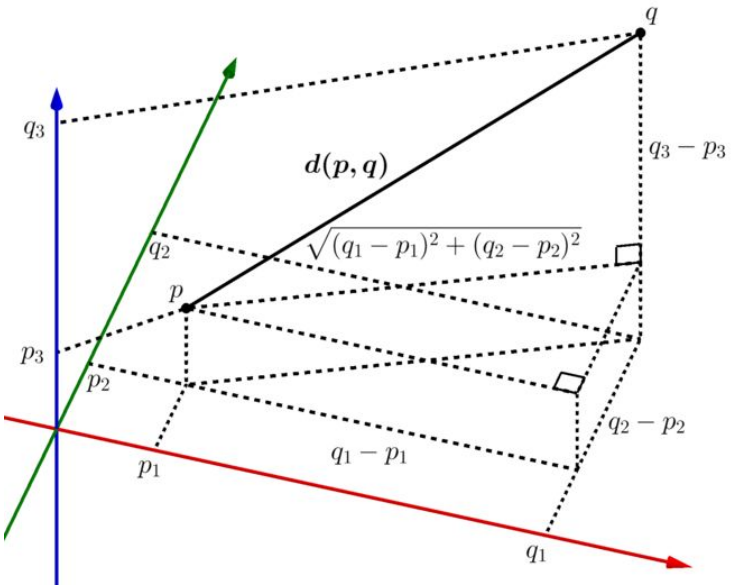


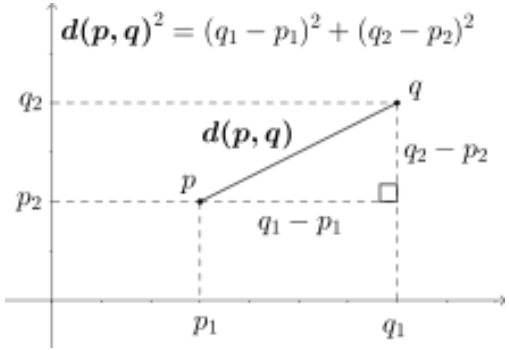
- Local representation (Discrete representation) : 해당 단어 그 자체만 보고 값을 매핑하여 표현
- Distributed representation (Continuous representation) : 단어를 표현하기 위해 주변을 참조

유사도 계산
(Text Similarity)

단어의 표현(Word Representation)

유클리디언 거리(Euclidean distance)





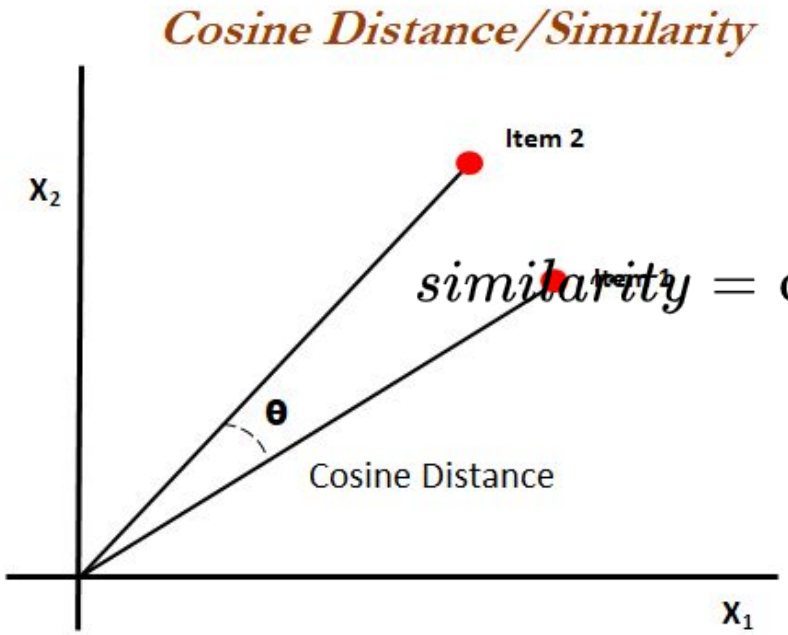
$$d(\mathbf{p}, \mathbf{q}) = d(\mathbf{q}, \mathbf{p}) = \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2 + \cdots + (q_n - p_n)^2}$$

$$= \sqrt{\sum_{i=1}^{n} (q_i - p_i)^2}.$$

https://en.wikipedia.org/wiki/Euclidean_distance

벡터간 직진거리 계산

코사인 유사도(Cosine Similarity)



$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}},$$

$$similarity = \cos(\theta) = \frac{A \cdot B}{\|A\| \cdot \|B\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}}$$

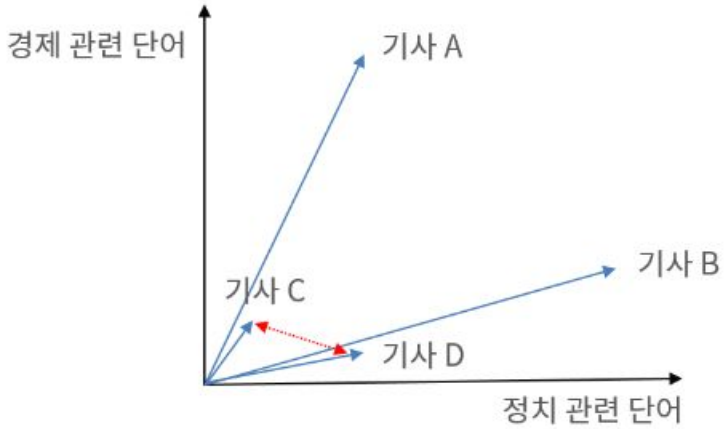
- 두 벡터간의 유사도를 측정하는 방법 중 하나
- 두 벡터 사이의 코사인을 측정
- 0도 = 1, 90도 = 0, 180도 = -1
  => 1에 가까울수록 유사도가 높음
  => 유사도가 높다는 것은 유사한 의미를 가짐을 의미

https://en.wikipedia.org/wiki/Cosine_similarity

두 벡터간 각(코사인 유사도)을 이용한 유사도 측정

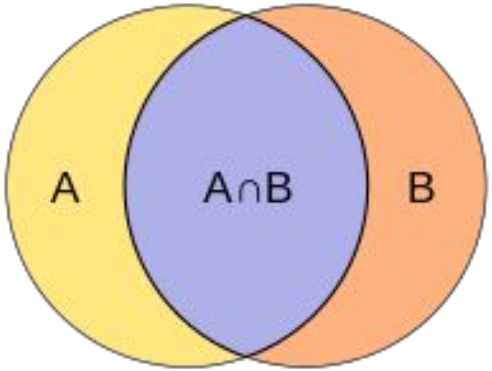
유클리디안 vs 코사인

상대적인 크기를 측정할 때



https://brunch.co.kr/@gimmesilver/39

자카드 유사도(Jaccard index)



$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}$$

https://en.wikipedia.org/wiki/Jaccard_index

문서 혹은 문장간 유사도 측정 (겹치는 토큰의 비율)

레벤슈타인 유사도(Levenshtein distance)

$$\operatorname{lev}_{a,b}(i,j) = \begin{cases} \max(i,j) & \text{if } \min(i,j) = 0, \\ \min \begin{cases} \operatorname{lev}_{a,b}(i-1,j) + 1 \\ \operatorname{lev}_{a,b}(i,j-1) + 1 \\ \operatorname{lev}_{a,b}(i-1,j-1) + 1_{(a_i \neq b_j)} \end{cases} & \text{otherwise.} \end{cases}$$

H		O	N	D	A	
H	Y	U	N	D	A	I

H	O		N	D	A	
H	Y	U	N	D	A	I

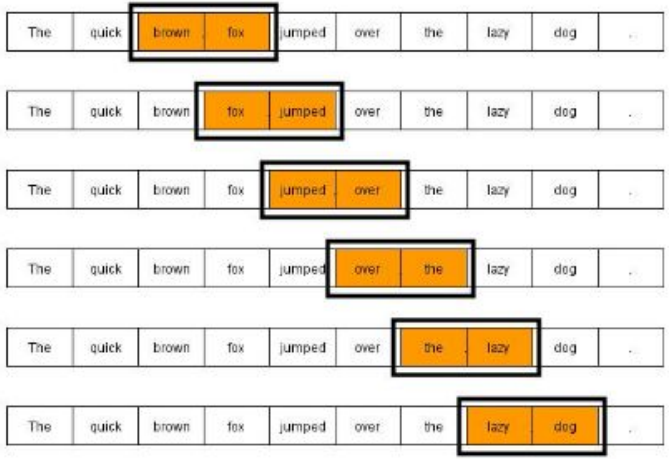
두 문자열이 얼마나 다른지를 나타내는 거리 중 하나

n-Gram

단어의 표현(Word Representation)

n-Gram 이란?

- 복수개(n개) 단어를 보는냐에 따라 unigram, bigram, trigram 등 으로 구분
- 제한적으로 문맥을 표현할 수 있음



n-Gram 이란?

an adorable little boy is spreading smile

- unigrams : an, adorable, little, boy, is, spreading, smiles
- bigrams : an adorable, adorable little, little boy, boy is, is spreading, spreading smiles
- trigrams : an adorable little, adorable little boy, little boy is, boy is spreading, is spreading smiles
- 4-grams : an adorable little boy, adorable little boy is, little boy is spreading, boy is spreading smiles

n-Gram 활용

- 연어 처리

  ex) 금융 통화 위원회, 국회 의원, 고객 서비스

- Language Modeling에 사용
  - 분야(Domain)에 따라 단어들의 확률 분포는 다름
    (금융 분야는 금융 관련 용어가 많이 등장하고, 마케팅은 관련 용어가 많이 등장할 것임)
  - 분야에 적합한 코퍼스를 사용하면 언어 모델의 성능이 높아질 수 있음
    (훈련에 사용되는 코퍼스에 따라 언어 모델의 성능이 달라짐 이는 언어 모델읜 약점으로 분류되기도함)