문서의 표현 (Document Representation)

자연어처리 텍스트마이닝

문서의 표현

문서의 표현(Document representation / Sentence representation)이란?

문서를 자연어처리를 위해 연산할 수 있도록 숫자로 표현하는 방법

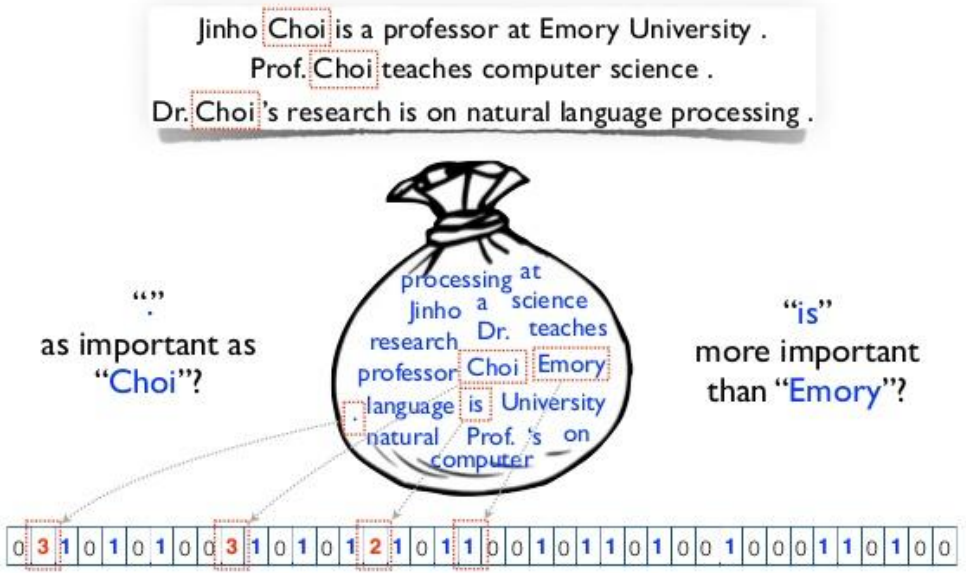
문서를 벡터로 표현하는 방법

BoW
(Bag of Words)

문서의 표현(Document Representation)

BoW (Bag of Words)

BoW(Bag of Words) : 문서 내 단어 출현 순서는 무시. 빈도수만 기반으로 문서를 표현하는 방법



BoW 생성 방법

문서1: 오늘 동물원에서 코끼리를 봤어
문서2:오늘 동물원에서 원숭이에게 사과를 줬어

Step1. 각 토큰에 고유 인덱스 부여

오늘	0
동물원에서	1
코끼리를	2
봤어	3
원숭이에게	4
사과를	5
줬어	6

Step2. 각 인덱스 위치에 토큰 등장 횟수를 기록

	오늘	동물원에서	코끼리를	봤어	원숭이에게	사과를	줬어
문서1	1	1	1	1	0	0	0

	오늘	동물원에서	코끼리를	봤어	원숭이에게	사과를	줬어
문서2	1	1	0	0	1	1	1

한계

- 단어의 **순서를 고려 하지 않음**

- **BoW 는 Spare** 함. 벡터 공간의 낭비, 연산 비효율성 초래

- **단어 빈도수가 중요도를 바로 의미 하지 않음.** 단어가 자주 등장한다고 중요한 단어는 아님.

- 전처리가 매우 중요함. 같은 의미의 다른 단어 표현이 있을 경우 다른것으로 인식될 수 있음.
  (뉴스와 같이 정제된 어휘를 사용하는 매체는 좋으나, 소셜에서는 활용하기 어려움)

TDM
(Term-Document Matrix)

문서의 표현(Document Representation)

TDM (Term-Document Matrix)

- BoW(Bag of Words) 중 하나
- 문서에 등장하는 각 단어 빈도를 행렬로 표현한 것

문서1: 동물원 코끼리
문서2: 동물원 원숭이 바나나
문서3: 엄마 코끼리 아기 코끼리
문서4: 원숭이 바나나 코끼리 바나나

	동물원	코끼리	원숭이	바나나	엄마	아기
문서1	1	1	0	0	0	0
문서2	1	0	1	1	0	0
문서3	0	2	0	0	1	1
문서4	0	1	1	2	0	0

TDM, DTM

TDM(Term-Document Matrix)
단어-문서 행렬

	문서1	문서2	문서3
단어1			
단어2			
단어3			
단어4			

DTM(Document-Term Matrix)
문서-단어 행렬

	단어1	단어2	단어3	단어4
문서1				
문서2				
문서3				

TDM (Term-Document Matrix) 의 한계

- 단어의 **순서를 고려 하지 않음**

- **TDM 는 Spare** 함. 벡터 공간의 낭비, 연산 비효율성 초래

- **단어 빈도수가 중요도를 바로 의미 하지 않음.** the와 같은 단어는 빈번하게 등장하고 TDM에서 중요한 단어로 판단 될 수 있음
  => 이를 보완 하기 위하여 TF - IDF 를 사용

TF-IDF
(Term Frequency-Inverse Document Frequency)

단어의 표현(Word Representation)

TF-IDF (Term Frequency-Inverse Document Frequency)

- 단어 빈도 - 역문서 빈도
- TDM 내 각 단어의 중요성을 가중치로 표현
- TDM을 사용하는 것보다 더 정확하게 문서비교가 가능

$$\mathrm{tfidf}(t, d, D) = \mathrm{tf}(t, d) \cdot \mathrm{idf}(t, D)$$

tf(d,t)	특정 문서 d에서의 특정 단어 t의 등장 횟수
df(t)	특정 단어 t가 등장한 문서의 수
idf(d, t)	df(t)의 역수

TF-IDF

$$\mathrm{tfidf}(t, d, D) = \mathrm{tf}(t, d) \cdot \mathrm{idf}(t, D)$$

tf(d,t)	특정 문서 d에서의 특정 단어 t의 등장 횟수
df(t)	특정 단어 t가 등장한 문서의 수
idf(d, t)	df(t)의 역수

TF	IDF	TF-IDF	설명
높	높	높	특정 문서에 많이 등장하고 타 문서에 많이 등장하지 않는 단어 (중요 키워드)
높	낮	-	특정 문서에도 많이 등장하고 타 문서에도 많이 등장하는 단어
낮	높	-	특정 문서에는 많이 등장하지 않고 타 문서에만 많이 등장하는 단어
낮	낮	낮	특정 문서에 많이 등장하지 않고 타 문서에만 많이 등장하는 단어

TF-IDF 활용

- 단어와 문서사이의 연관성

- 정보 검색 (Information retrieval)
  - 검색어와 가장 관련이 있는 문서를 찾아 결과 제공

    ex) 검색엔진에서 '인공지능'

- 키워드 추출(Keyword Extraction)
  - TF-IDF 점수가 가장 높은 점수를 가지고 있는 단어가 그 문서를 대표하는 키워드

TF-IDF 적용 사례

인공지능

'4차 산업혁명, 다가오는 변화의 물결'
'[AI리포트] AI 로봇기자가 이제는 시도 쓴다고?'
'포브스가 선정한 10년후 미래 유망 직업 Top 20'

TF-IDF

'4차 산업혁명, 다가오는 변화의 물결' -> 4차 산업혁명
'[AI리포트] AI 로봇기자가 이제는 시도 쓴다고?' -> 로봇기자
'포브스가 선정한 10년후 미래 유망 직업 Top 20' -> 직업

TF-IDF 가중치 계산

Variants of term frequency (tf) weight

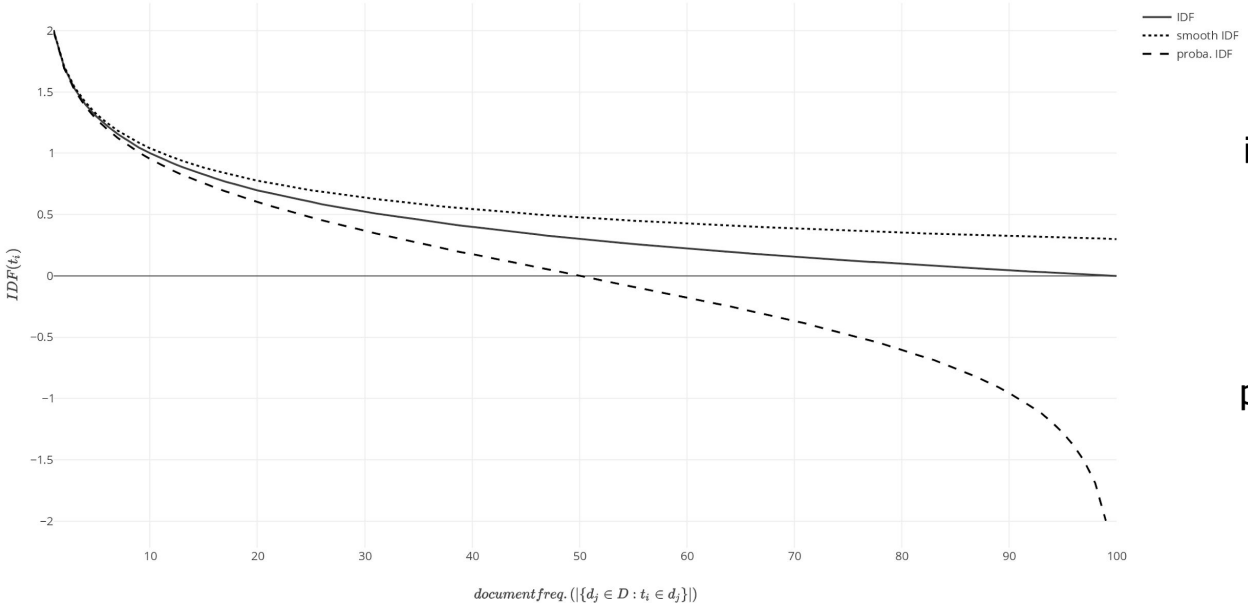
weighting scheme	tf weight
binary	$0, 1$
raw count	$f_{t,d}$
term frequency	$f_{t,d} \Big/ \sum_{t' \in d} f_{t',d}$
log normalization	$\log(1 + f_{t,d})$
double normalization 0.5	$0.5 + 0.5 \cdot \frac{f_{t,d}}{\max_{\{t' \in d\}} f_{t',d}}$
double normalization K	$K + (1 - K) \frac{f_{t,d}}{\max_{\{t' \in d\}} f_{t',d}}$

Variants of inverse document frequency (idf) weight

weighting scheme	idf weight ($n_t = \left\vert \{d \in D : t \in d\} \right\vert$)
unary	1
inverse document frequency	$\log \frac{N}{n_t} = -\log \frac{n_t}{N}$
inverse document frequency smooth	$\log \left(\frac{N}{1 + n_t} \right)$
inverse document frequency max	$\log \left(\frac{\max_{\{t' \in d\}} n_{t'}}{1 + n_t} \right)$
probabilistic inverse document frequency	$\log \frac{N - n_t}{n_t}$

출처 : https://en.wikipedia.org/wiki/Tf%E2%80%93idf

IDF에 로그를 사용하는 이유



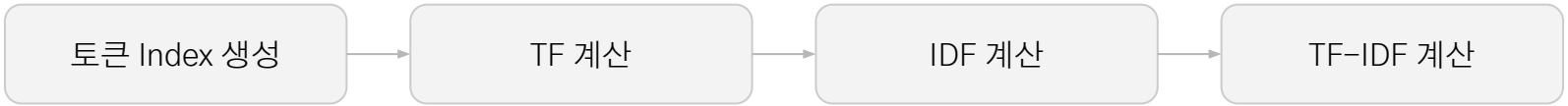
$$\text{idf} = \log\left(\frac{N}{n_t}\right) = -\log\left(\frac{n_t}{N}\right)$$

$$\text{smooth idf} = \log\left(\frac{N}{1+n_t}\right) + 1$$

$$\text{probabilistic idf} = \log\left(\frac{N-n_t}{n_t}\right)$$

출처 : https://en.wikipedia.org/wiki/Tf%E2%80%93idf

TF-IDF 계산절차



각 토큰(단어)을 그대로 사용할 수 없기 때문에 토큰에 Index를 부여

각 토큰(단어)의 등장 빈도를 계산

각 토큰(단어)의 문서 등장 빈도를 계산하여 역수를 취함

계산한 TF와 IDF를 곱하여 TF-IDF를 계산

예제 1 : 토큰 Index 생성

문서1 : d1 = "The cat sat on my face I hate a cat"
문서2 : d2 = "The dog sat on my bed I love a dog"

	Index
The	0
cat	1
sat	2
on	3
my	4
face	5
I	6
hate	7
a	8
dog	9
bed	10
love	11

예제 : TF 계산

문서1 : d1 = "The cat sat on my face I hate a cat"
문서2 : d2 = "The dog sat on my bed I love a dog"

$$f_{t,d} \Big/ \sum_{t' \in d} f_{t',d}$$

$f_{t,d}$ = 문서내 토큰 빈도
SUM($f_{t,d}$)= 문서내 전체 토큰빈도

문서1

	문서내 토큰 빈도	문서내 전체 토큰빈도	TF
The	1	10	0.1
cat	2	10	0.2
sat	1	10	0.1
on	1	10	0.1
my	1	10	0.1
face	1	10	0.1
I	1	10	0.1
hate	1	10	0.1
a	1	10	0.1
dog	0	10	0
bed	0	10	0
lov	0	10	0

문서2

	문서내 토큰 빈도	문서내 전체 토큰빈도	TF
The	1	10	0.1
cat	0	10	0
sat	1	10	0.1
on	1	10	0.1
my	1	10	0.1
face	0	10	0
I	1	10	0.1
hate	0	10	0
a	1	10	0.1
dog	2	10	0.2
bed	1	10	0.1
love	1	10	0.1

예제 : IDF 계산

문서1 : d1 = "The cat sat on my face I hate a cat"
문서2 : d2 = "The dog sat on my bed I love a dog"

$$\log \frac{N}{n_t} = -\log \frac{n_t}{N}$$

N = 문서수

n_t = 토큰이 등장한 문서수

	문서수	토큰이 등장한 문서수	IDF
The	2	2	0
cat	2	1	0.301
sat	2	2	0
on	2	2	0
my	2	2	0
face	2	1	0.301
I	2	2	0
hate	2	1	0.301
a	2	2	0
dog	2	1	0.301
bed	2	1	0.301
love	2	1	0.301

예제 : TF-IDF 계산

문서1 : d1 = "The cat sat on my face I hate a cat"
문서2 : d2 = "The dog sat on my bed I love a dog"

문서1

	TF	IDF	TF-IDF
The	0.1	0	0
cat	0.2	0.301	0.060
sat	0.1	0	0
on	0.1	0	0
my	0.1	0	0
face	0.1	0.301	0.301
I	0.1	0	0
hate	0.1	0.301	0.301
a	0.1	0	0
dog	0	0.301	0.301
bed	0	0.301	0.301
lov	0	0.301	0.301

문서2

	TF	IDF	TF-IDF
The	0.1	0	0
cat	0	0.301	0
sat	0.1	0	0
on	0.1	0	0
my	0.1	0	0
face	0	0.301	0.301
I	0.1	0	0
hate	0	0.301	0.301
a	0.1	0	0
dog	0.2	0.301	0.601
bed	0.1	0.301	0.301
love	0.1	0.301	0.301

TF-IDF

$$\mathrm{tfidf}(t, d, D) = \mathrm{tf}(t, d) \cdot \mathrm{idf}(t, D)$$

tf(d,t)	특정 문서 d에서의 특정 단어 t의 등장 횟수
df(t)	특정 단어 t가 등장한 문서의 수
idf(d, t)	df(t)의 역수

TF	IDF	TF-IDF	설명
높	높	높	특정 문서에 많이 등장하고 타 문서에 많이 등장하지 않는 단어 (중요 키워드)
높	낮	-	특정 문서에도 많이 등장하고 타 문서에도 많이 등장하는 단어
낮	높	-	특정 문서에는 많이 등장하지 않고 타 문서에만 많이 등장하는 단어
낮	낮	낮	특정 문서에 많이 등장하지 않고 타 문서에만 많이 등장하는 단어

LSA
(Latent Semantic Analysis)

문서의 표현(Document Representation)

그 외 문서의 표현

문서의 표현(Document Representation)

단어-동시빈도 행렬 (Term-Cooccurrence Matrix)

- 단어간의 동시등장(co-occurrence) 행렬

1. I enjoy flying.
2. I like NLP.
3. I like deep learning.

$$X = \begin{array}{c} \\ I \\ like \\ enjoy \\ deep \\ learning \\ NLP \\ flying \\ . \end{array} \begin{array}{c} \begin{array}{cccccccc} I & like & enjoy & deep & learning & NLP & flying & . \end{array} \\ \begin{bmatrix} 0 & 2 & 1 & 0 & 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \end{bmatrix} \end{array}$$

단어-문맥 행렬 (Term-Context Matrix)

- 단어-문맥 간의 동시등장(co-occurrence) 행렬
- 문맥은 사용자가 설정한 window의 크기로 결정
- 문맥 내 등장하는 단어의 빈도를 표기