언어모델&카운트 기반의 단어 표현

안유진

1. 언어모델

1.1 언어모델 & 통계적 언어 모델

1.2 N-gram

1.3 한국어와 언어 모델

1.4 Perplexity

1. 카운트 기반의 단어 표현

2.1 Bag of Words

2.2 Document-Term Matrix

2.3 TF-IDF

Chapter 1

언어모델

1-1 언어모델 & 통계적 언어 모델

- **언어모델 : 단어 시퀀스에 확률을 할당하는 모델**
  -> ex) P(나는 버스를 탔다) > P(나는 버스를 태운다)

- **통계적 언어 모델(Statistic Language Model, SLM)의 접근 방법**
  - 조건부 확률 : P(B|A) = P(A,B)/P(A) -> P(A,B,C,D) = P(A)P(B|A)P(C|A,B)P(D|A,B,C)
  - chain rule : $P(x_1, x_2, x_3 \dots x_n) = P(x_1)P(x_2|x_1)P(x_3|x_1, x_2) \dots P(x_n|x_1 \dots x_{n-1})$
  - 문장에 대한 확률 : 이전 단어가 주어졌을 때 다음 단어로 등장할 확률의 곱

$$P(w_1, w_2, w_3, w_4, w_5, \dots w_n) = \prod_{n=1}^{n} P(w_n|w_1, \dots, w_{n-1})$$

  - 카운트기반의 접근

$$P(\text{is}|\text{An adorable little boy}) = \frac{\text{count}(\text{An adorable little boy is})}{\text{count}(\text{An adorable little boy })}$$

  - 희소문제 : 코퍼스에 현재 단어 시퀀스가 없어 확률이 0이 되는, 충분한 데이터를 관측하지 못하여 언어를 정확히 모델링하지 못하는 문제

1.2 N-gram

- **마르코프 가정 : 특정 시점의 상태 확률은 단지 그 직전 상태에만 의존한다는 논리**

- **N-gram : 마르코프 가정을 기반으로 확률을 추정하는 방식** (n은 k+1, k의 단위 뭉치로 끊음)
  unigrams : an, adorable, little, boy, is, spreading, smiles(2-gram)_
  bigrams : an adorable, adorable little, little boy, boy is, is spreading, spreading smiles(3-gram)
  trigrams : an adorable little, adorable little boy, little boy is, boy is spreading, is spreading smiles(3-gram)
  4-grams : an adorable little boy, adorable little boy is, little boy is spreading, boy is spreading smiles}

  -> 4-gram ex) $$P(w|\text{boy is spreading}) = \frac{\text{count}(\text{boy is spreading } w)}{\text{count}(\text{boy is spreading})}$$

- **N-gram model의 한계**
  - 희소 문제
  - n의 trade off
    -> n이 커질수록 희소문제 심각
    -> n이 작을수록 근사의 정확도가 현실의 확률분포와 멀어짐

1.3 한국어와 언어 모델

종류	대표적 언어	특징
교착어	한국어, 일본어, 몽골어	어간에 접사가 붙어 단어를 이루고 의미와 문법적 기능이 더해짐
굴절어	라틴어, 독일어, 러시아어	단어의 형태가 변함으로써 문법적 기능이 정해짐
고립어	영어, 중국어	어순에 따라 단어의 문법적 기능이 정해짐

- 한국어 특징 1 : 한국어는 교착어
- 한국어 특징 2 : 띄어쓰기가 제대로 지켜지지 않는다
- 한국어 특징 3 : 한국어는 어순이 중요하지 않다

1.4 Perplexity(PPL)

$$PPL(W) = P(w_1, w_2, w_3, \dots, w_N)^{-\frac{1}{N}} = \sqrt[N]{\frac{1}{P(w_1, w_2, w_3, \dots, w_N)}} = \sqrt[N]{\frac{1}{\prod_{i=1}^{N} P(w_i|w_1, w_2, \dots, w_{i-1})}}$$

- PPL : 단어의 수로 정규화된 테스트 데이터에 대한 확률의 역수
  -> PPL을 최소화 : 문장의 확률을 최대화!
- 분기 계수(Branching factor) -> 선택할 수 있는 가능한 경우의 수
- ex)

$$PPL(W) = P(w_1, w_2, w_3, \dots, w_N)^{-\frac{1}{N}} = (\frac{1}{10}^{N})^{-\frac{1}{N}} = \frac{1}{10}^{-1} = 10$$

-> 다음 단어를 예측할 때 평균 10개의 단어를 가지고 정답을 고민한다!

Chapter 2

카운트 기반의 단어 표현

2.1 Bag of Words

- **Bag of Words**
  : 단어들의 순서는 전혀 고려하지 않고, 단어들의 출현 빈도(frequency)에만 집중하는 텍스트 데이터의 수치화 표현 방법

- **BoW의 과정**
  - 각 단어에 고유한 정수 인덱스 부여
  - 각 인덱스의 위치에 단어 토큰의 등장 횟수를 기록한 벡터 사용

- **BoW 예시**
  -> 정부가 발표하는 물가상승률과 소비자가 느끼는 물가상승률은 다르다.
  -> ('정부': 0, '가': 1, '발표': 2, '하는': 3, '물가상승률': 4, '과': 5, '소비자': 6, '느끼는': 7, '은': 8, '다르다': 9)
  -> [1, 2, 1, 1, 2, 1, 1, 1, 1, 1]

2.1 Bag of Words

- BoW에서 중요한 것은 단어의 등장 빈도!
  -> ('발표': 0, '가': 1, '정부': 2, '하는': 3, '소비자': 4, '과': 5, '물가상승률': 6, '느끼는': 7, '은': 8, '다르다': 9)
  -> [1, 2, 1, 1, 1, 1, 2, 1, 1, 1]
  ==
  -> ('정부': 0, '가': 1, '발표': 2, '하는': 3, '물가상승률': 4, '과': 5, '소비자': 6, '느끼는': 7, '은': 8, '다르다': 9)
  -> [1, 2, 1, 1, 2, 1, 1, 1, 1, 1]
  -> 위 둘은 동일한 BoW

- BoW는 각 단어가 등장한 횟수를 수치화하는 방법
  -> 단어 등장 횟수에따라 문서의 성격 판단 가능

2.2 Document-Term Matrix

문서1 : 먹고 싶은 사과
문서2 : 먹고 싶은 바나나
문서3 : 길고 노란 바나나 바나나
문서4 : 저는 과일이 좋아요

BoW

-	과일이	길고	노란	먹고	바나나	사과	싶은	저는	좋아요
문서1	0	0	0	1	0	1	1	0	0
문서2	0	0	0	1	1	0	1	0	0
문서3	0	1	1	0	2	0	0	0	0
문서4	1	0	0	0	0	0	0	1	1

2.2 Document-Term Matrix

- 문서 단어 행렬(DTM)의 **한계**
  - **희소 표현(Sparse representation)**
    -> 전체 단어 집합의 크기를 가지기 때문에, 차원이 커진다
    -> 대부분의 값이 0일수도 있다
    -> 많은 양의 저장공간과 계산을 위한 리소스 필요
  - **단순 빈도 수 기반 접근**
    -> 불용어인 the는 빈도수가 높다(그렇지만 유사한 문서는 X)
    -> TF-IDF를 사용하자

2.3 TF-IDF(Term Frequency-Inverse Document Frequency)

- **tf(d.t)** : 특정 문서 d에서의 특정 단어 t의 등장 횟수
-
- $idf(d,t) = log(\frac{n}{1+df(t)})$ 의 수

2.3 TF-IDF(Term Frequency-Inverse Document Frequency)

$$idf(d,t) = log(\frac{n}{1+df(t)})$$

- 왜 idf에 log를 취할까? -> 기하급수적으로 값이 커지는걸 막기 위함
- idf(d, t) : df(t)에 반비례하는 수

$idf(d,t) = log(n/df(t))$

$n = 1,000,000$

단어 t	$df(t)$	$idf(d,t)$
word1	1	6
word2	100	4
word3	1,000	3
word4	10,000	2
word5	100,000	1
word6	1,000,000	0

$idf(d,t) = n/df(t)$

$n = 1,000,000$

단어 t	$df(t)$	$idf(d,t)$
word1	1	1,000,000
word2	100	10,000
word3	1,000	1,000
word4	10,000	100
word5	100,000	10
word6	1,000,000	1

2.3 TF-IDF(Term Frequency-Inverse Document Frequency)

$$idf(d,t) = log(\frac{n}{1+df(t)})$$

- 왜 idf의 Log안의 식에서 분모에 1을 더해줄까?
- -> 특정 단어가 전체 문서에서 등장하지 않을 경우 분모가 1이 되는 상황
- idf(d, t) : df(t)에 반비례하는 수
-> 하지만 df(t)가 n-1일 경우 log(1)이 되어 log(~) + 1을 해주기도함

- idf의 계산

-	과일이	길고	노란	먹고	바나나	사과	싶은	저는	좋아요
문서 1	0	0	0	1	0	1	1	0	0
문서 2	0	0	0	1	1	0	1	0	0
문서 3	0	1	1	0	2	0	0	0	0
문서 4	1	0	0	0	0	0	0	1	1

단어	IDF(역 문서 빈도)
과일이	ln(4/(1+1)) = 0.693147
길고	ln(4/(1+1)) = 0.693147
노란	ln(4/(1+1)) = 0.693147
먹고	ln(4/(2+1)) = 0.287682
바나나	ln(4/(2+1)) = 0.287682
사과	ln(4/(1+1)) = 0.693147
싶은	ln(4/(2+1)) = 0.287682
저는	ln(4/(1+1)) = 0.693147
좋아요	ln(4/(1+1)) = 0.693147

2.3 TF-IDF(Term Frequency-Inverse Document Frequency)

-

-	과일이	길고	노란	먹고	바나나	사과	싶은
문서 1	0	0	0	0.287682	0	0.693147	0.287682
문서 2	0	0	0	0.287682	0.287682	0	0.287682
문서 3	0	0.693147	0.693147	0	0.575364	0	0
문서 4	0.693147	0	0	0	0	0	0

TF-IDF : 모든 문서에서 자주 등장하는 단어는 중요도가 낮다고 판단하고, 특정 문서에서만 자주 등장하는 단어는 중요도가 높다고 판단
-> 값이 낮으면 중요도가 낮다!
-> 값이 크면 중요도가 크다!

End of Document

Thank You.