2018학년도 2학기 언어와 컴퓨터

제15강 N-그램 언어 모형 (2)

박수지

서울대학교 인문대학 언어학과

2018년 11월 14일 수요일

오늘의 목표

1. N-그램 모형의 성능을 복잡도라는 척도로 평가할 수 있다.
2. N-그램 모형을 사용하여 문장을 생성할 수 있다.
3. N-그램 모형에서 발생하는 0 문제를 평탄화로 해결할 수 있다.
4. N-그램 모형의 한계를 설명할 수 있다.

단어 연쇄의 확률 계산

(1) 조건부확률의 연쇄법칙으로 단어 연쇄의 확률 표현하기

$$
\begin{aligned}
&P(\text{“하늘은 파랗고 단풍잎은 빨갛고 은행잎은 노랗고”}) \\
&= P(\text{‘‘하늘은}'') \times P(\text{“파랗고”}|\text{“하늘은”}) \\
&\quad \times P(\text{“단풍잎은”}|\text{“하늘은 파랗고”}) \\
&\quad \times P(\text{“빨갛고”}|\text{“하늘은 파랗고 단풍잎은”}) \\
&\quad \times P(\text{“은행잎은”}|\text{“하늘은 파랗고 단풍잎은 빨갛고”}) \\
&\quad \times P(\text{“노랗고”}|\text{“하늘은 파랗고 단풍잎은 빨갛고 은행잎은”})
\end{aligned}
$$

단어 연쇄의 확률 계산

(2) 단어 연쇄의 코퍼스 출현 빈도로 조건부확률 추정하기

$$P(\text{“은행잎은”}|\text{“하늘은 파랗고 단풍잎은 빨갛고”})$$
$$= \frac{Count(\text{“하늘은 파랗고 단풍잎은 빨갛고 은행잎은”})}{Count(\text{“하늘은 파랗고 단풍잎은 빨갛고”})}$$

문제

코퍼스에 한 번도 출현하지 않은 부분이 있으면 전체 확률이 0이 된다.

단어 연쇄의 확률 계산

(2) 단어 연쇄의 코퍼스 출현 빈도로 조건부확률 추정하기

문제

"하늘은 파랗고 단풍잎은 빨갛고 은행잎은 노랗고"에 0의 확률을 할당하는 언어모형을 쓰고 싶은가?

단어 연쇄의 확률 계산 — N-그램 모형

(3) 조건부확률의 근삿값을 사용하기

예시

트라이그램 ($N = 3$) 근사

$$
\begin{aligned}
&P(\text{“하늘은 파랗고 단풍잎은 빨갛고 은행잎은 노랗고”}) \\
&\approx P(\text{“‘하늘은”}) \times P(\text{“파랗고”} | \text{“하늘은”}) \\
&\quad \times P(\text{“단풍잎은”} | \text{“하늘은 파랗고”}) \\
&\quad \times P(\text{“빨갛고”} | \text{“파랗고 단풍잎은”}) \\
&\quad \times P(\text{“은행잎은”} | \text{“단풍잎은 빨갛고”}) \\
&\quad \times P(\text{“노랗고”} | \text{“빨갛고 은행잎은”})
\end{aligned}
$$

단어의 직전 $(N - 1)$ 개만 보자!

단어 연쇄의 확률 계산 — N-그램 모형

(3) 조건부확률의 근삿값을 사용하기

예시

바이그램 ($N = 2$) 근사

$$
\begin{aligned}
&P(\text{“하늘은 파랗고 단풍잎은 빨갛고 은행잎은 노랗고”}) \\
&\approx P(\text{“‘하늘은”}) \times P(\text{“파랗고”} | \text{“하늘은”}) \\
&\quad \times P(\text{“단풍잎은”} | \text{“파랗고”}) \\
&\quad \times P(\text{“빨갛고”} | \text{“단풍잎은”}) \\
&\quad \times P(\text{“은행잎은”} | \text{“빨갛고”}) \\
&\quad \times P(\text{“노랗고”} | \text{“은행잎은”})
\end{aligned}
$$

단어 연쇄의 확률 계산 — N-그램 모형

(3) 조건부확률의 근삿값을 사용하기

예시

바이그램 ($N = 2$) 근사

$$
\begin{aligned}
&P(\text{“하늘은 파랗고 단풍잎은 빨갛고 은행잎은 노랗고”}) \\
&\approx P(\text{“‘하늘은”}) \times \frac{C(\text{“하늘은 파랗고”})}{C(\text{“하늘은”})} \\
&\quad \times \frac{C(\text{“파랗고 단풍잎은”})}{C(\text{“파랗고”})} \times \frac{C(\text{“단풍잎은 빨갛고”})}{C(\text{“단풍잎은”})} \\
&\quad \times \frac{C(\text{“빨갛고 은행잎은”})}{C(\text{“빨갛고”})} \times \frac{C(\text{“은행잎은 노랗고”})}{C(\text{“은행잎은”})}
\end{aligned}
$$

단어 연쇄의 확률 계산 — N-그램 모형

(3) 조건부확률의 근삿값을 사용하기

유니그램	구글 검색결과
”하늘은”	3520000
”파랗고”	392000
”단풍잎은”	34600
”빨갛고”	339000
”은행잎은”	24300

바이그램	구글 검색결과
”하늘은 파랗고”	56100
”파랗고 단풍잎은”	23
”단풍잎은 빨갛고”	160
”빨갛고 은행잎은”	85
”은행잎은 노랗고”	198

문제

첫 단어의 확률 $P(\text{“하늘은”})$은 어떻게 계산하는가?

단어 연쇄의 확률 계산 — N-그램 모형

(3) 조건부확률의 근삿값을 사용하기

문제

"하늘은"의 확률이 실제로 의미하는 것

- 문장이나 구가 "하늘은"으로 시작할 확률

→ (임의의 위치가 아니라) 첫 단어로 "하늘은"이 나올 확률

→ 조건부확률 $P(\text{"하늘은"}|\text{<s>})$

문장이 경계지어져 있어야 한다.

언어 모형 평가 방법

외재적 모형을 다른 과제에 응용했을 때 성능이 얼마나 향상되는가?

- 과제 수행을 위한 시간과 비용이 필요하다.

내재적 외부 과제와 무관하게 모형의 품질이 얼마나 좋은가?

- 모형의 품질을 평가할 척도가 필요하다.

내재적 평가의 원칙 혹은 전제

실제로 존재하는 문장에 높은 확률을 부여해야 한다.

"실제로 존재하는 문장"을 어디에서 찾는가?

기계학습을 위한 코퍼스 분할

훈련 집합 …에서 확률을 학습하고

- 기계에게 "정답"을 알려주고 훈련시키는 단계

개발 집합 …에서 조율하기도 하면서

- 훈련 중간에 평가해 보고 훈련 방향을 조정하는 단계

실험 집합 …에서 학습이 잘 되었는지 평가한다.

- 기계에게 "정답"을 알려주지 않고 맞추게 하는 단계

일반적인 분할 방법

- train:dev:test=80:10:10

주의

훈련 집합에서 학습한 모형을 훈련 집합의 문장으로 평가하면 안 된다.

복잡도(perplexity)

정의

실험 집합 $W = w_1 w_2 \dots w_K$ 의 확률의 역수의 K 제곱근

$$PP(W) = P(w_1 w_2 \dots w_K)^{-\frac{1}{K}} = \sqrt[K]{\frac{1}{P(w_1 w_2 \dots w_K)}}$$

역수를 취했으므로, 확률이 커지면 복잡도가 낮아진다.

의문

왜 K 제곱근을 취하는가?

- 길이 K 인 연쇄의 확률: 조건부확률 K 개의 곱
- 효과: 문장이 길어질수록 확률이 낮아지는 현상을 보완해 준다.

복잡도(perplexity)

숙제 (일부)

$W =$ “하늘은 파랗고 단풍잎은 빨갛고 은행잎은 노랗고”의 복잡도를

- 바이그램 모형에서 계산하라.
- 트라이그램 모형에서 계산하라.

바이그램 모형으로 문장 자동완성 & 생성하기

A toy corpus

```
i like to eat pasta .
i want to eat food from malaysia .
show me the list again .
i 'd like the previous list please .
i 'd like to go to a japanese restaurant .
```

- Berkeley Restaurant Corpus에서 발췌
- 텍스트 정규화 및 문장·단어 분리 완료

바이그램 모형으로 문장 자동완성 & 생성하기

코퍼스 크기

- 문장 수 5
- 단어(token) 수 $N = 39$
- 어휘(type) 수 $|V| = 23$

```
V = {i, like, to, eat,
pasta, ., want, food,
from, malaysia, show, me,
the, list, again, 'd,
previous, please, go, a,
japanese, restaurant}
```

유니그램의 확률 추정

$$P(w) = \frac{Count(w)}{N}$$

generate.py

```
8  from pprint import pprint
9  from collections import defaultdict, Counter
10 from random import choice
11
12 text = '''i like to eat pasta .
13 i want to eat food from malaysia .
14 show me the list again .
15 i 'd like the previous list please .
16 i 'd like to go to a japanese restaurant .'''
```

generate.py

```
19 sents = ... # edit this line
```

할 일

text를 가지고 문장(단어의 리스트)의 리스트 sents를 만들기

목표

```
>>> pprint(sents)
[['i', 'like', 'to', 'eat', 'pasta', '.'],
 ['i', 'want', 'to', 'eat', 'food', 'from', 'malaysia',
'.'],
 ['show', 'me', 'the', 'list', 'again', '.'],
 ['i', "'d", 'like', 'the', 'previous', 'list', 'please
', '.'],
 ['i', "'d", 'like', 'to', 'go', 'to', 'a', 'japanese',
'restaurant', '.']]
```

generate.py

```
22 bigrams_dict = defaultdict(list)
23 for sent in sents:
24     filled_sent = ['<s>'] + sent
25     for wd1, wd2 in ...
```

목표

```
>>> pprint(bigrams_dict)
defaultdict(<class 'list'>,
            {"'d": ['like', 'like'],
             '<s>': ['i', 'i', 'show', 'i', 'i'],
             'a': ['japanese'],
             'again': ['.'],
             'eat': ['pasta', 'food'],
             ...
             'the': ['list', 'previous'],
             'to': ['eat', 'eat', 'go', 'a'],
             'want': ['to']})
```

문장 자동 완성

generate.py

```
31 def modes(seq):
32     counts = Counter(seq)
33     M = max(counts.values())
34     return [item for item, count in counts.
           items() if count == M]
```

용도

bigrams_dict['<s>']의 값
['i', 'i', 'show', 'i', 'i']에서 최빈값을 찾아야 한다.

문장 자동 완성

generate.py

```
36 def complete():
37     # first word
38     wd = choice(modes(bigrams_dict['<s>']))
39     # initialize sentence to be completed
40     completed = [wd]
41     # and update sentence
42     ...
43     ...
44     return '␣'.join(wd for wd in completed)
```

문장 자동 완성

목표

```
>>> for _ in range(10):
...     print(complete())
...
i 'd like to eat food from malaysia .
i 'd like to eat food from malaysia .
i 'd like to eat pasta .
i 'd like to eat pasta .
i 'd like to eat food from malaysia .
i 'd like to eat pasta .
i 'd like to eat food from malaysia .
i 'd like to eat food from malaysia .
i 'd like to eat pasta .
i 'd like to eat food from malaysia .
```

문장 생성

generate.py

```
49 def generate():
50     # edit this
51     return # edit this
```

문장 생성

목표

```
show me the previous list again .
i 'd like to eat food from malaysia .
i want to a japanese restaurant .
show me the list please .
i 'd like the list please .
i want to eat food from malaysia .
i 'd like to eat food from malaysia .
i 'd like to go to eat food from malaysia .
show me the previous list please .
show me the previous list please .
```

“영어스러운” 단어 생성

실습 코드 `ngrams.py`

다른 예

문자의 N-그램으로 “영어스러운” 단어 생성하기

- 음운론 직관 테스트
- 소설, 게임 등 창작

“영어스러운” 단어 생성

결과

```
From bigrams:
['mperent', 'mointh', 'douscelar', 'mbiticeub',
'chte', 'iningidentin', 'rfitosssyme', 'pacun',
'sele', 'thoulatinurditid', 'wribenthecealiernt
', 'renteash', 'curbere', 'rvidd', 'towertet',
'hoicll', 'bombloorulephrenerquccurpl', 'brater
e', 'pirdiramane', 'plaplarcanter', 'lires', 'f
ule', 'lond', 'cicach', 'vente', 'daymerites',
'armat', 'macre', 'yroce', 'ghrulon']
```

“영어스러운” 단어 생성

결과

```
From trigrams:
['fulippity', 'farkliginebratilay', 'cle', 'adive', 'phippilatic', 'crore', 'swon', 'visladive', 'adinly', 'rernalmin', 'prorume', 'moreed', 'aurew', 'sle', 'divercument', 'iss', 'dine', 'whilimbrion', 'latid', 'fidiveous', 'won', 'stichan', 'beseencent', 'eloameratidion', 'emin', 'vescocuous', 'tryman', 'stalay', 'fulent', 'shor']
```

“영어스러운” 단어 생성

결과

```
From 4-grams:
['repinence', 'shor', 'tenebrity', 'empyreal',
'sempyrean', 'lamber', 'core', 'sea-gird', 'lac
hrymalefic', 'susurrand', 'trounder', 'ful', 'd
olous', 'crescient', 'circumvallation', 'sempit
ed', 'puissans', 'ethe', 'nightsome', 'impuissa
nthine', 'ful', 'illy', 'greenswain', 'atrand',
'afarer', 'well', 'lacustranthine', 'tarrant',
'rutilation', 'arge']
```

Toy data: 훈련 코퍼스

바이그램	빈도
귀여운 고양이	25
귀여운 강아지	22
귀여운 다람쥐	2
귀여운 햄스터	1
귀여운 병아리	0
귀여운 망아지	0
귀여운	50

$$P(\text{“고양이”}|\text{“귀여운”}) = \frac{C(\text{“귀여운 고양이”})}{C(\text{“귀여운”})} = \frac{25}{50} = 0.5$$

$$P(\text{“병아리”}|\text{“귀여운”}) = \frac{C(\text{“귀여운 병아리”})}{C(\text{“귀여운”})} = \frac{0}{50} = 0.0$$

문제

“귀여운 병아리”가 실험 코퍼스에 없으리라고 확신할 수 있는가?

평탄화 (Smoothing)

문제 훈련 코퍼스에 **없는** N-그램이 실험 코퍼스에 나타날 수 있다.

목표 훈련 코퍼스에 **없는** N-그램에도 양의 확률을 부여한다.

사실 확률의 합은 1이다.

종합 훈련 코퍼스에 **있는** N-그램의 확률을 깎아야 한다.

라플라스 평탄화 (Laplace smoothing, Add-1 smoothing)

(0을 포함한) 모든 빈도에 1을 더해서 확률을 추정한다.

Toy data: 훈련 코퍼스

바이그램	빈도	평탄화
귀여운 고양이	25	25+1
귀여운 강아지	22	22+1
귀여운 다람쥐	2	2+1
귀여운 햄스터	1	1+1
귀여운 병아리	0	0+1
귀여운 망아지	0	0+1
귀여운	50	50 + 6

$$P(\text{“고양이”}|\text{“귀여운”}) = \frac{26}{56} = 0.4643 < 0.5$$

$$P(\text{“병아리”}|\text{“귀여운”}) = \frac{1}{56} = 0.0179 > 0$$

Add-k smoothing

1 대신 $k(< 1)$을 더한다.

0에 대처하는 또다른 방법

N-그램이 없으면 $(N-1)$-그램을 동원하자!

back-off N-그램이 없을 때만 $(N-1)$-그램을 사용한다.

보간법 항상 N-그램과 $(N-1)$-그램을 함께 사용한다.

보간법 (interpolation)

$$\hat{P}(w_n|w_{n-2}w_{n-1}) = \lambda_1 P(w_n|w_{n-2}w_{n-1}) + \lambda_2 P(w_n|w_{n-1}) + \lambda_3 P(w_n)$$

$\lambda_1, \lambda_2, \lambda_3$ 모형 매개변수 (model parameters)

λ_1 트라이그램의 가중치

λ_2 바이그램의 가중치

λ_3 유니그램의 가중치

N-그램의 한계

- 장거리 의존을 반영하지 못한다.
  예: The **computers** which I had just put into the machine room on the fifth floor **are** crashing
  - 확률 추정치: $P(\text{“floor is”}) \gg P(\text{“floor are”})$
- 자유어순언어에 대해서는 잘 작동하지 않는다.
  예: 돈을 그에게 주었다. 그에게 돈을 주었다.
  - 확률 추정치: $P(\text{“돈을 그에게”}) \simeq P(\text{“그에게 돈을”})$
    $\Rightarrow$ “돈을 그에게 돈을 그에게 돈을 …” 생성 가능

오늘 배운 것

- 언어 모형을 내재적으로 평가하는 척도
  - 복잡도
- N-그램 확률이 0이 되는 문제 및 해결 방법
  - 평탄화

다음 주에 배울 것

- 기계학습 — 단순 베이즈 분류
- 감정분석
  - SLP3e Ch. 4
  - 밑바닥부터… 6–7장