

## <빅데이터 최신기술 2차과제>

### 한글 문장의 유사도 계산(2)

20153167 김현중

음절 bigram 방식에 의한 유사도 계산.

구현 코드

```
# -*- coding: utf-8 -*-
doc1 = raw_input("첫번째 문장을 입력해주세요 : ")
doc2 = raw_input("두번째 문장을 입력해주세요 : ")
text1 = tuple(doc1) #첫번째 입력 문장을 음절 단위로 쪼갬
text2 = tuple(doc2) #두번째 입력 문장을 음절 단위로 쪼갬
bigram1 = [text1[x:x+2] for x in range(0,len(text1))] # 음절 단위로 쪼개진 문장1을 bigram(음절2개)씩 묶어준다
bigram2 = [text2[x:x+2] for x in range(0,len(text2))] # 음절 단위로 쪼개진 문장2를 bigram(음절2개)씩 묶어준다
bigram1_size = len(bigram1)
bigram2_size = len(bigram2)
bigram_intersection = [value for value in bigram1 if value in bigram2] ## 문장 1과 문장 2에서의 음절 bigram 교집합
bigram_intersection_size = len(bigram_intersection) ## ngrams_intersection_size : bigram 교집합 사이즈
if len(doc1) <= len(doc2):
    short_bigram = bigram1_size
else:
    short_bigram = bigram2_size ## 문장 1과 문장 2의 길이를 비교하여 짧은 문장의 음절 bigram을 short_bigram에 저장
similarity = float(bigram_intersection_size) / float(short_bigram) ## 공통 bigram 개수 / 짧은 문장 bigram 개수 = 유사도
print(similarity*100) ##음절 bigram 유사도 출력
```

구현 Step

1. 두개의 문장을 입력받는다
2. 두개의 문장을 각각 bigram(음절2개)씩 묶어준다  
(bigram1과 bigram2의 list에 저장)
3. bigram1과 bigram2 list를 비교하여 공통되는 요소를  
bigram\_intersection(교집합 리스트)에 저장해준다.
4. 두 개의 문장 중 짧은 문장을 판별한다
5. 공통된 bigram수(bigram\_intersection의 length)를 짧은 문장의

- bigram수(short\_bigram)으로 나누어주어 유사도를 계산한다
6. 유사도를 출력해준다

## 출력결과

```
Run: bigram x
/Users/gimhyeonjung/PycharmProjects/bigdata/venv/bin/python /Users/gimhyeonjung/PycharmProjects/bigdata/bigram.py
첫번째 문장을 입력해주세요 : 나는 오늘 카페에서 아이스 아메리카노를 마셨다
두번째 문장을 입력해주세요 : 나는 오늘 집에서 아이스커피를 마셨다
91.8367346939
Process finished with exit code 0
```

## 음절 trigram 방식에 의한 유사도 계산.

### 구현 코드

```
# -*- coding: utf-8 -*-
doc1 = raw_input("첫번째 문장을 입력해주세요 : ")
doc2 = raw_input("두번째 문장을 입력해주세요 : ")
text1 = tuple(doc1) #첫번째 입력 문장을 음절 단위로 쪼갬다
text2 = tuple(doc2) #두번째 입력 문장을 음절 단위로 쪼갬다
trigram1 = [text1[x:x+3] for x in range(0,len(text1))] # 음절 단위로 쪼개진 문장1을 trigram(음절3개)씩 묶어준다
trigram2 = [text2[x:x+3] for x in range(0,len(text2))] # 음절 단위로 쪼개진 문장1을 trigram(음절3개)씩 묶어준다
trigram1_size = len(trigram1)
trigram2_size = len(trigram2)
trigram_intersection = [value for value in trigram1 if value in trigram2] ## 문장 1과 문장 2에서의 음절 trigram 교집합
trigram_intersection_size = len(trigram_intersection) ## trigram_intersection_size : trigram 교집합 사이즈
if len(doc1) <= len(doc2):
    short_trigram = trigram1_size
else:
    short_trigram = trigram2_size ## 문장 1과 문장 2의 길이를 비교하여 짧은 문장의 음절 trigram을 short_trigram에 저장
similarity = float(trigram_intersection_size) / float(short_trigram) ## 공통 trigram 개수 / 짧은 문장 trigram 개수 = 유사도
print(similarity*100) ##음절 trigram 유사도 출력
```

### 구현 Step

1. 두개의 문장을 입력받는다
2. 두개의 문장을 각각 trigram(음절3개)씩 묶어준다  
(trigram1과 trigram2의 list에 저장)
3. trigram1과 trigram2 list를 비교하여 공통되는 요소를  
trigram\_intersection(교집합 리스트)에 저장해준다

4. 두 개의 문장 중 짧은 문장을 판별한다
5. 공통된 trigram수(trigram\_intersection의 length)를 짧은 문장의 trigram수(short\_trigram)으로 나누어 주어 유사도를 계산한다
6. 유사도를 출력해준다

## 출력결과

```
Run: trigram x
/Users/gimhyeonjung/PycharmProjects/bigdata/venv/bin/python /Users/gimhyeonjung/PycharmProjects/bigdata/trigram.py
첫번째 문장을 입력해주세요 : 나는 오늘 카페에서 머리스 머무르려노를 마셨다
두번째 문장을 입력해주세요 : 나는 오늘 집에서 머리스를 마셨다
85.7142857143
Process finished with exit code 0
```

## 음절 bi + trigram 방식에 의한 유사도 계산.

### 구현 코드

```
# -*- coding: utf-8 -*-
doc1 = raw_input("첫번째 문장을 입력해주세요 : ")
doc2 = raw_input("두번째 문장을 입력해주세요 : ")
text1 = tuple(doc1) #첫번째 입력 문장을 음절 단위로 쪼갬다
text2 = tuple(doc2) #두번째 입력 문장을 음절 단위로 쪼갬다

bigram1 = [text1[x:x+2] for x in range(0,len(text1))] # 음절 단위로 쪼개진 문장1을 bigram(음절2개)씩 묶어준다
trigram1 = [text1[x:x+3] for x in range(0,len(text1))] # 음절 단위로 쪼개진 문장1을 trigram(음절3개)씩 묶어준다
bitri1 = bigram1+trigram1 ##bigram으로 쪼개진 리스트와 trigram으로 쪼개진 리스트를 합쳐준다
bitri1_size = len(bitri1)

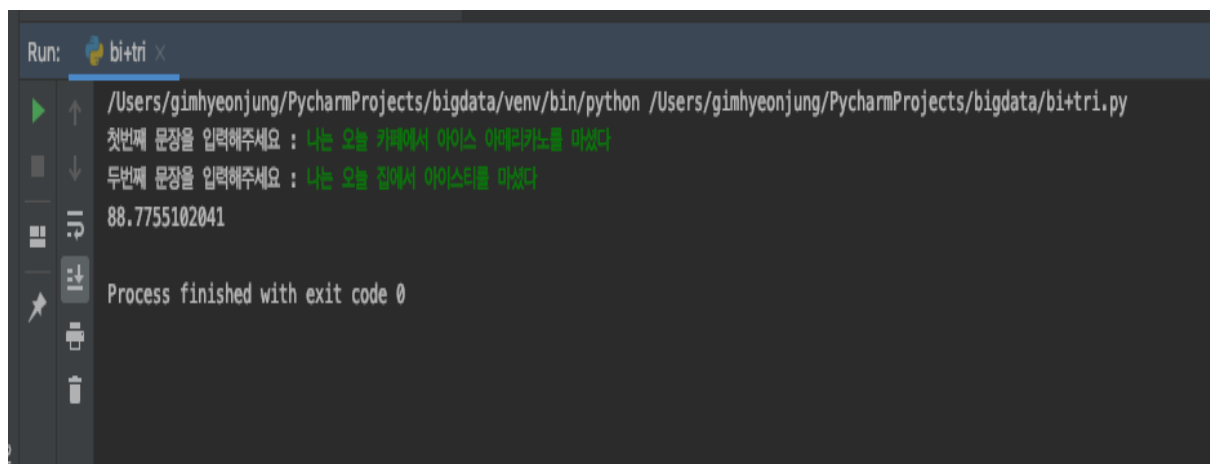
bigram2 = [text2[x:x+2] for x in range(0,len(text2))] # 음절 단위로 쪼개진 문장2를 bigram(음절2개)씩 묶어준다
trigram2 = [text2[x:x+3] for x in range(0,len(text2))] # 음절 단위로 쪼개진 문장2 trigram(음절3개)씩 묶어준다
bitri2 = bigram2+trigram2 ##bigram으로 쪼개진 리스트와 trigram으로 쪼개진 리스트를 합쳐준다
bitri2_size = len(bitri2)

bitri_intersection = [value for value in bitri1 if value in bitri2] ## 문장 1과 문장 2에서의 음절 bitrigram 교집합
bitri_intersection_size = len(bitri_intersection) ## bitri_intersection_size : bitrigram 교집합 사이즈
if len(doc1) <= len(doc2):
    short_bitri = bitri1_size
else:
    short_bitri = bitri2_size ## 문장 1과 문장 2의 음절 bitrigram 수를 비교하여 짧은 문장의 음절 bitrigram을 short_bitri에 저장
similarity = float(bitri_intersection_size) / float(short_bitri) ## 공통 bi + trigram 개수 / 짧은 문장 bi + trigram 개수 = 유사도
print(similarity*100) ##음절 bi+trigram 유사도 출력
```

## 구현 Step

1. 두개의 문장을 입력받는다
2. 두개의 문장을 각각 bigram(음절2개)씩 묶어준다  
(bigram1과 bigram2의 list에 저장)
3. 두개의 문장을 각각 trigram(음절3개)씩 묶어준다  
(trigram1과 trigram2의 list에 저장)
4. bigram1과 trigram1 list를 bitri1 리스트로 합쳐준다  
bigram2와 trigram2 list를 bitri2 리스트로 합쳐준다
5. bitri1과 bitri2 list를 비교하여 공통되는 요소를  
bitri\_intersection(교집합 리스트)에 저장해준다
6. 두 개의 문장 중 짧은 문장을 판별한다
7. 공통된 bi+trigram수(bitri\_intersection의 length)를 짧은 문장의  
bi+trigram수(short\_bitri)로 나누어주어 유사도를 계산한다
8. 유사도를 출력해준다

## 출력결과



```
Run: bi+tri x
/Users/gimhyeonjung/PycharmProjects/bigdata/venv/bin/python /Users/gimhyeonjung/PycharmProjects/bigdata/bi+tri.py
첫번째 문장을 입력해주세요 : 나는 오늘 카페에서 아이스 아메리카노를 마셨다
두번째 문장을 입력해주세요 : 나는 오늘 집에서 아이스티를 마셨다
88.7755102041
Process finished with exit code 0
```

## 형태소 분석기(KoNLPy)에 의한 유사도 계산.

### 구현 코드

```
# -*- coding: utf-8 -*-
from konlpy.tag import Hannanum
hannanum = Hannanum()
doc1 = input("첫번째 문장을 입력해주세요 : ")
doc2 = input("두번째 문장을 입력해주세요 : ")
hannanum1 = hannanum.morphs(doc1) ## 문장 1을 형태소 단위로 쪼개준다
hannanum2 = hannanum.morphs(doc2) ## 문장 2를 형태소 단위로 쪼개준다
hannanum1_size = len(hannanum1)
hannanum2_size = len(hannanum2)
hannanum_intersection = [value for value in hannanum1 if value in hannanum2] ## 문장 1과 문장 2에서의 형태소 교집합
hannanum_intersection_size = len(hannanum_intersection) ## hannanum_intersection_size : 겹치는 형태소의 개수(사이즈)
if len(doc1) <= len(doc2):
    short_hannanum = hannanum1_size
else:
    short_hannanum = hannanum2_size ## 문장 1과 문장 2의 길이를 비교하여 짧은 문장의 형태소 개수를 short_hannanum에 저장
similarity = float(hannanum_intersection_size) / float(short_hannanum) ## 공통 형태 개수 / 짧은 문장소 형태소 개수 = 유사도
print(similarity*100) ##형태소 유사도 출력
```

### 구현 Step

1. 두개의 문장을 입력받는다(Konlpy의 hannanum class 이용)
2. Hannanum class의 morphs method를 이용하여 입력받은 두개의 문장을 형태소 단위로 쪼개준다  
(hannanum1 list, hannanum2 list)
3. 두개의 리스트(hannanum1,hannanum2)를 비교하여 공통된 형태소를 hannanum\_intersection list(교집합 리스트)에 저장한다
4. 두개의 문장 중 짧은 문장을 판별한다
5. 공통된 형태소의 개수(hannanum\_intersection의 length)를 짧은 문장의 형태소 개수(short\_hannanum)로 나누어주어 유사도를 계산한다
6. 유사도를 출력해준다

## 출력결과

```
/Users/gimhyeonjung/PycharmProjects/untitled1/venv/bin/python /Users/gimhyeonjung/PycharmProjects/untitled1/venv/hw4.py
첫번째 문장을 입력해주세요 : 나는 오늘 카페에서 아이스 아메리카노를 마셨다
첫번째 문장을 입력해주세요 : 나는 오늘 집에서 아이스티를 마셨다
80.0

Process finished with exit code 0
```