<빅데이터 최신기술 3차과제>

대규모 말뭉치(KCC 원시말뭉치)에서 가장 유사한 문장 상위 n 개 추출

20153167 김현중

구현 코드

```python
import time
start = time.time()
from konlpy.tag import Hannanum
doc = input("문장을 입력해주세요 : ") ## 문장을 입력받는다
n = input("입력문장과 유사한 몇개의 문장을 출력할까요? : ") ## 유사한 문장을 몇개 출력할지 입력 받는다
print(" ")
hannanum = Hannanum() ##형태소 분석을 위하여 konlpy의 Hannanum class를 이용하여 준다
doc_tokenized = hannanum.morphs(doc) ##입력문장을 형태소 단위로 쪼개준다
doc_tokenized_size = len(doc_tokenized) ##입력문장의 형태소 개수

list=[]

with open('KCCq28.txt', 'r', encoding='utf-8') as input:  # 유사도 검사할 말뭉치를 가져옴
    for line in input: ##말뭉치에서 라인 단위로 읽어준다
        file_tokenized = hannanum.morphs(line)  ##읽어들인 라인을 형태소 단위로 쪼개준다
        file_tokenized_size = len(file_tokenized)  ##읽어들인 라인의 형태소 개수
        intersection_size = 0  ##교집합 형태소 개수를 초기화
        for x in doc_tokenized:
            if x in file_tokenized:
                intersection_size+=1  ##입력문장과 라인문장을 비교하여 겹치는 형태소가 있다면 교집합 형태소 개수 증가시키기
        if len(doc) <= len(line):
            short = doc_tokenized_size
        else:
            short = file_tokenized_size  ## 입력문장과 파일 각 문장의(라인단위) 길이를 비교하여 짧은 문장의 형태소 개수를 short에 저장
        similarity = float(intersection_size) / float(short)  ## 공통 형태소 개수 / 짧은 문장소 형태소 개수 = 유사도
        list.append([line, similarity * 100])  ##list 배열에 문장과, 그문장의 입력문장에 대한 유사도를 집어넣어준다

    sorted_list = sorted(list, key=lambda x: -x[1])  ##list 배열을 유사도 순으로 정렬하여 준다(유사도가 높은순으로, 내림차순)
    for i in range(int(n)):
        print(sorted_list[i][0])  ##입력받은 n개의 유사도가 높은 문장을 출력한다.
        print("유사도는 " + str(sorted_list[i][1]) + "% 입니다.")  ##입력받은 n개의 유사도가 높은 문장의 유사도를 출력한다.
        print(" ")
        print(" ")

    print("소요시간 :", time.time() - start, "초")  # 소요시간 출력
```

구현 Step

1. Konlpy의 Hannanum class를 import한다.

2. 문장을 입력받는다.

3. 유사한 문장을 몇개를 출력할지 입력받는다(n개).

4. Hannanum class로부터 hannanum 객체를 생성해준다.

5. Hannanum class의 morphs 메소드를 이용하여 입력받은 문장을 형태소 단위로 쪼개어준다(tokenizing).

6. 형태소 단위로 쪼개진 입력문장에서 형태소의 개수를 세준다.

7. KCCq28 대용량 파일을 입력으로 가져온다.
   (KCCq28-01 파일을 사용하였습니다)

8. 대용량 파일을 라인단위로 가져온다

9. 라인 단위로 가져온 문장을 Hannanum class의 morphs 메소드를 이용하여 형태소 단위로 쪼개어준다(tokenizing).

10. 형태소 단위로 쪼개진 문장의 형태소 개수를 세준다.

11. 교집합 형태소의 개수를 0으로 초기화 해준다.

12. 입력문장의 tokenized list와 라인의 tokenized list를 비교해주어 겹치는 형태소가 존재할 경우 카운트를 증가시켜준다.

13. 입력문장과 라인의 길이를 비교하여 짧은 문장의 형태소 개수를 short 변수에 넣어준다.

14. Interserction_size(교집합 형태소 개수) / short(짧은 문장 형태소 개수) 식을 이용하여 두 문장간의 유사도를 similarity 변수에 넣어준다.

15. 각 라인(문장)과 그 문장의 입력문장에 대한 유사도 (similarity*100)를 2차원 배열 list에 넣어준다.

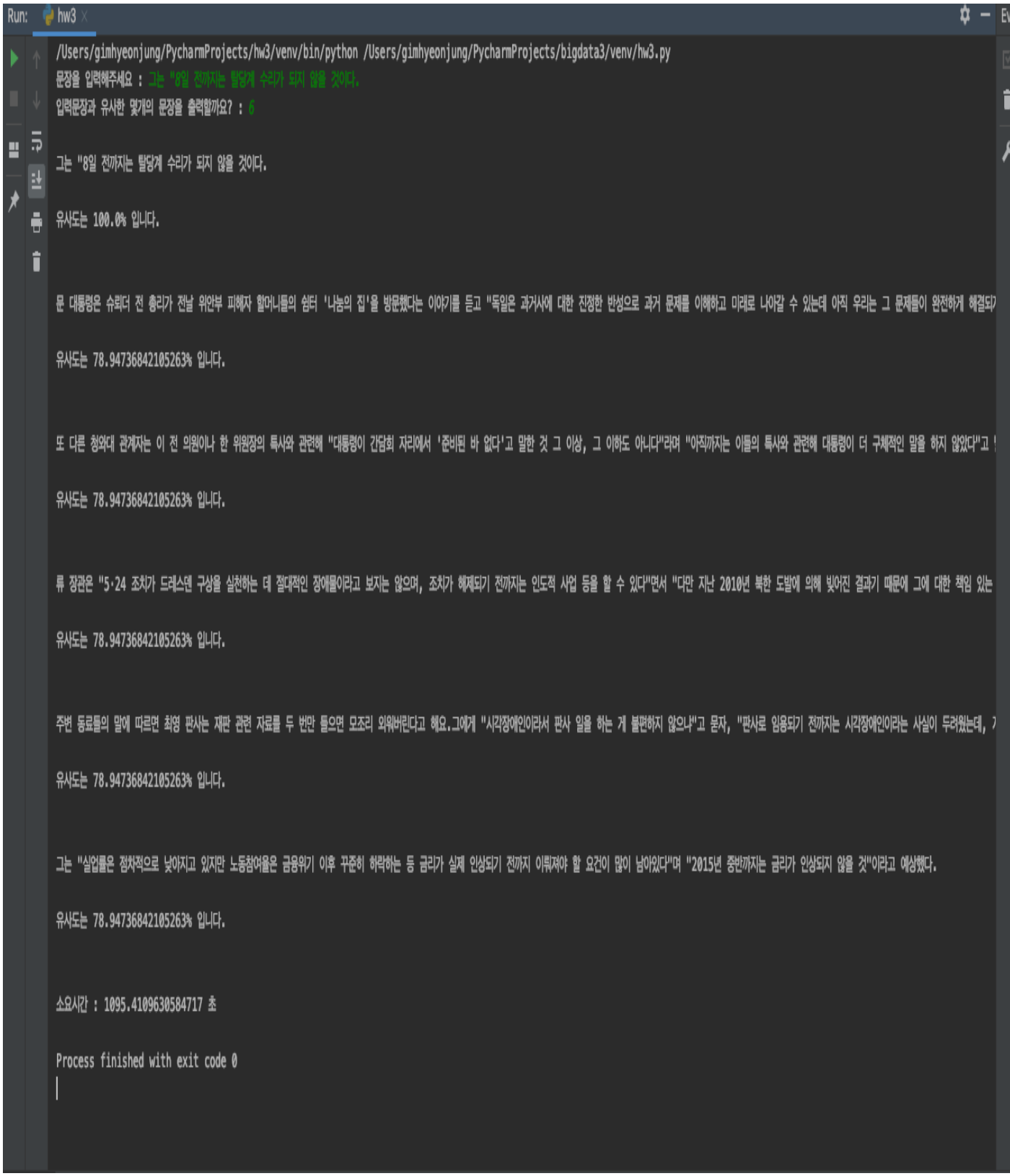
16. 대용량 한국어 텍스트 파일을 모두 읽었을 때 list 배열에는 파일의 라인 순서대로 [문장, 유사도]가 저장이 된다.

17. list 배열을 similarity가 높은 순서대로 정렬한다

18. 출력할 문장과 유사도는 n개이므로 정렬된 배열에서 처음부터
For in range(int(n))
n개의 문장(sorted_list[i][0]),
N개 문장의 유사도(sorted_list[i][1])를 출력하면 된다.

19. 모든 작업을 마쳤을 시 소요시간을 출력한다.

출력 결과


Run: hw3
/Users/gimhyeonjung/PycharmProjects/hw3/venv/bin/python /Users/gimhyeonjung/PycharmProjects/bigdata3/venv/hw3.py
문장을 입력해주세요 : 그는 "8일 전까지는 탈당계 수리가 되지 않을 것이다.
입력문장과 유사한 몇개의 문장을 출력할까요? : 6
그는 "8일 전까지는 탈당계 수리가 되지 않을 것이다.
유사도는 100.0% 입니다.
문 대통령은 슈뢰더 전 총리가 전날 위안부 피해자 할머니들의 쉼터 '나눔의 집'을 방문했다는 이야기를 듣고 "독일은 과거사에 대한 진정한 반성으로 과거 문제를 이해하고 미래로 나아갈 수 있는데 아직 우리는 그 문제들이 완전하게 해결되
유사도는 78.94736842105263% 입니다.
또 다른 청와대 관계자는 이 전 의원이나 한 위원장의 특사와 관련해 "대통령이 간담회 자리에서 '준비된 바 없다'고 말한 것 그 이상, 그 이하도 아니다"라며 "아직까지는 이들의 특사와 관련해 대통령이 더 구체적인 말을 하지 않았다"고
유사도는 78.94736842105263% 입니다.
류 장관은 "5·24 조치가 드레스덴 구상을 실천하는 데 절대적인 장애물이라고 보지는 않으며, 조치가 해제되기 전까지는 인도적 사업 등을 할 수 있다"면서 "다만 지난 2010년 북한 도발에 의해 빚어진 결과기 때문에 그에 대한 책임 있는
유사도는 78.94736842105263% 입니다.
주변 동료들의 말에 따르면 최영 판사는 재판 관련 자료를 두 번만 들으면 모조리 외워버린다고 해요.그에게 "시각장애인이라서 판사 일을 하는 게 불편하지 않으냐"고 묻자, "판사로 임용되기 전까지는 시각장애인이라는 사실이 두려웠는데,
유사도는 78.94736842105263% 입니다.
그는 "실업률은 점차적으로 낮아지고 있지만 노동참여율은 금융위기 이후 꾸준히 하락하는 등 금리가 실제 인상되기 전까지 이뤄져야 할 요건이 많이 남아있다"며 "2015년 중반까지는 금리가 인상되지 않을 것"이라고 예상했다.
유사도는 78.94736842105263% 입니다.
소요시간 : 1095.4109630584717 초
Process finished with exit code 0