빅데이터 최신기술

Word Count

목차

소프트웨어학부

20163162

차윤성

1. 과제 수행 방법

Python 내장 함수인 split과 dictionary 자료구조를 사용하면 간단히 구현할 수 있을 것 같아 과제 수행에 사용할 언어를 Python으로 선택하였다.

- 프로그램 개요

```
36  def main(argv):
37      if len(argv) != 2:
38          print("========== Invalid Command ==========\n")
39          print("\t C > wordcount.py wikisent.txt\n")
40          exit()
41
42      f = open(argv[1], 'r', encoding='utf8')
43
44      line = f.readlines()
45
46      wordSet = []
47
48      wordSet = getWordSet(line)
49
50      storeCountedWord(wordSet)
51
52      f.close()
53
54
55  if __name__ == "__main__":
56      main(sys.argv)
```

- main: 먼저 시스템의 인자를 확인해서 입력파일을 정상적으로 넘겨받았는지 확인한다. 이후 readlines를 수행한 다음 문장 단위로 단어를 찾고, 단어와 단어의 빈도수로 이루어진 리스트를 도출한다. 반환된 리스트를 통해 빈도수 상위 1,000개의 단어를 파일로 저장한 후 프로그램이 종료된다.

```
25  def getWordSet(l):
26      wSet = {}
27      for i in tqdm(range(len(l))):
28          temp = l[i].split()
29          for j in range(len(temp)):
30              word = ''.join(temp[j])
31              wSet[word] = wSet.get(word, 0) + 1
32
33      wSet = sorted(wSet.items(), reverse=True, key=lambda item:item[1])
34      return wSet
```

- getWordSet: main에서 수행된 readlines의 결과값을 인자로 넘겨받아 단어와 단어의 빈도수를 반환해주는 함수이다. 각 line별로 split을 이용해 word를 도출하고, dictionary의 킷값을 word로 추가하며 빈도수는 기존 key값의 value에 1을 더해 단어를 추가해준다. 이후 sorted를 통해 빈도수를 기준으로 내림차순 정렬 후 리스트를 main으로 반환한다.

```python
def storeCountedWord(wSet):
    w = open("out.txt", 'w', encoding='utf8')

    for i in range(1000):
        if(i > len(wSet) - 1):
            break
        word = wSet[i][0]
        freq = (int)(wSet[i][1])

        data = "{w}\t{f}\n".format(w=word, f=freq)
        w.write(data)

    w.close()
```

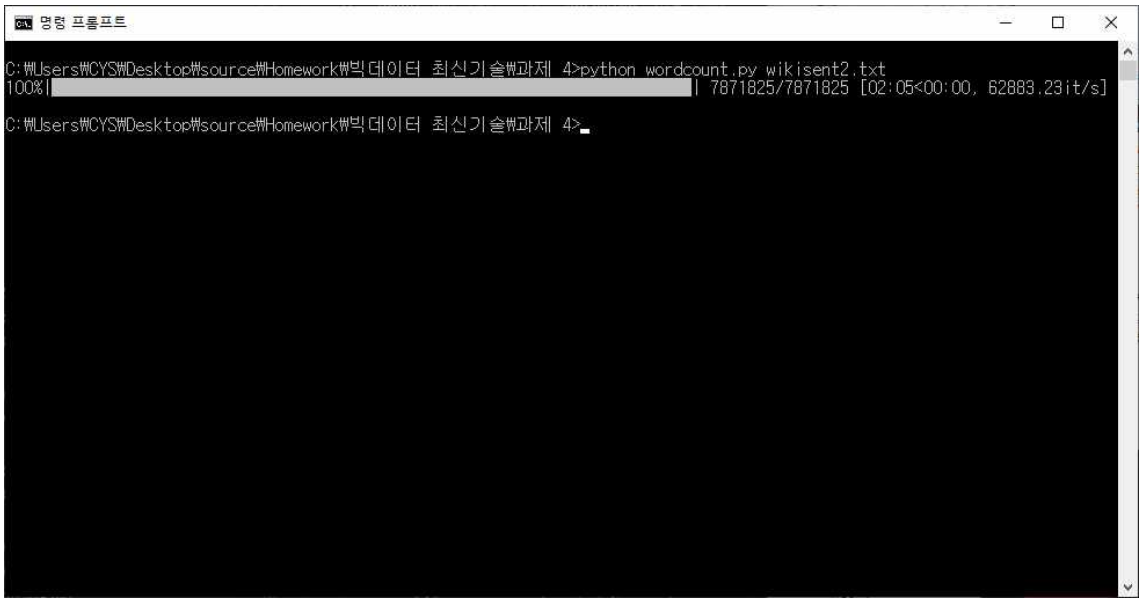
- storeCountedWord: getWordSet으로 반환된 빈도수를 기준으로 정렬되어 있는 wordSet을 인자로 받아와 빈도수 상위 1,000개의 단어를 out.txt에 저장한다. 이 때, out.txt에 저장되는 data의 format은 "word freq"이다.

- 사용한 라이브러리

```python
import sys
from tqdm import tqdm
```

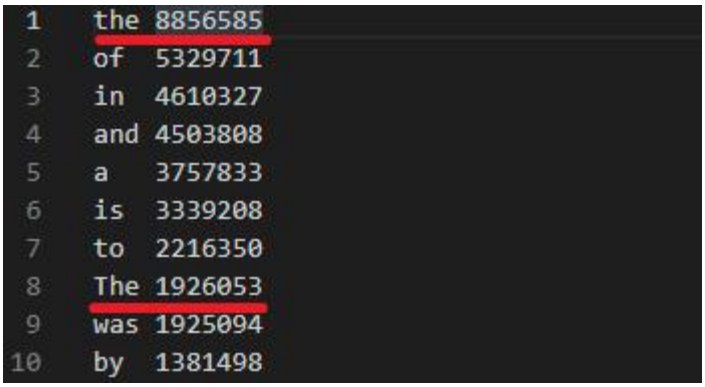
  - sys: command line으로 넘겨받은 인자를 관리하기 위해 사용하였다.
  - tqdm: pip install tqdm을 통해 설치한 외부 라이브러리로, progress bar를 표시하기 위해 사용하였다.

2. 실행 화면



- tqdm을 이용해 progress bar를 표시해본 결과, 저장을 제외한 각 line 별로 단어를 추출해 set을 만드는 것에 약 2분 5초정도 소요되었다.

3. 결과



out.txt

- 결과로 각 단어 중 the가 8,856,585번 등장해 가장 많이 등장한 단어였고, 그 다음으로 빈도수가 높은 단어는 5,329,711번 등장한 of였다.

- 첫 번재로 많이 등장한 단어는 the, 여덟 번째로 많이 등장한 단어는 The로, the와 The를 다른 단어로 구별하고 있음을 확인할 수 있다.

- 결과 파일로 저장된 out.txt의 용량은 약 14KB 였다.