

Python

**Production of programs
that analyze Korean PDF
and extract keywords**

Final Report

Date : 2023.12.24

Name : Haegeon Lee

ID : 183014

1. Introduction

1) Background

With the advent of the information age, the era of reading and interpreting a large amount of documents has arrived. If the amount of documents is vast when a particular document is first encountered, fatigue accumulates before reading. Therefore, if the Python program provides keywords that readers should focus on first, the speed increases rapidly and more efficient reading is possible. Therefore, the existence of these programs is necessary.

2) Project goal

It aims to develop a program that analyzes pdf to analyze what key keywords are based on the frequency of words and provides them to users.

3) Differences from existing programs

Existing programs are produced based on English, so there is a big difference in reading Korean pdf and it is impossible to use. Therefore, there is a difference from existing programs because we focus on extracting keywords from Korean pdf file for Koreans.

2. Functional Requirement

1) Function 1 – Extract text from file

- Extract the text file from the pdf file and store it in the list.

2) Function 2 - Remove non-critical words

- Remove investigations such as '은','는','이','가' from list token. And import Morpheme analyzer to make up a list with only nouns. We will also add the ability to remove words that are frequently used but are not important.

3) Function 3 - ranking based on frequency to output the top 15

- Saves the frequency of nouns in dictionary form and outputs the most written value.

4) Function 4

- We plan to save the result value as a txt file so that it can be used in the future.

3. Implementation

(1) Extract text from file

- In

Pdf file

-out

Text(Words / variable)

- Explanation

After receiving pdf, import the txt file on a page-by-page basis from pdf and save it in text(variable).

- applied learning

For loop, package, file inout, function

- screen shot

```
def extract_text_from_pdf(pdf_input):  
    pdf_reader = PdfReader(pdf_input)  
    pages = pdf_reader.pages  
    text = ""  
    for page in pages:  
        text += page.extract_text()  
    return text
```

(2) Remove non-critical words

- Explanation

Other are excluded and only alphabetic or numeric words are added to the word list.

Gets the list of words that are not important.

Put the rest in the list called yes_words, except for the list of words mixed with numbers and non-important words.

- applied learning

for loop, list comprehension

- screen shot

```
# 안쓰는 단어 리스트/ 여기에 추가하여 업데이트 가능
no_words = [
    '은', '는', '이', '가', '을', '를', '에', '에서', '도', '만', '뿐', '만큼', '여러분', '지금', '그리고', '이렇게',
    '그렇게', '그래서', '그러나', '으로', '.', ',', '의', '이', '등', '및', '수', '있는', '사무관', '위한', '통해', '있도록',
]

text = H6pdf.extract_text_from_pdf(pdf_input)
no_words_removed_text = ' '.join([H6pdf.remove_suffix(word, no_words) for word in text.split()])
```

```
# 불필요한 접미사를 제거하는 함수
2 개의 사용 위치
def remove_suffix(word, suffix_list):
    for suffix in suffix_list:
        if word.endswith(suffix):
            return word[:-len(suffix)]
    return word
```

```
def get_word_frequency(text, no_words=None):
    if no_words is None:
        no_words = []
    # 숫자가 섞인 리스트 num_and_words랑 안쓰는 단어 리스트 no_words 제외하고
    num_and_words = [word for word in text.split() if word.isalnum()]
    # remove_suffix 함수를 사용하여 불필요한 접미사 제거 후 나머지를 yes_words 라는 리스트로 넣음
    yes_words = [remove_suffix(word, no_words) for word in num_and_words]
```

(3) ranking based on frequency to output the top 15

- In

Yes_words

-out

1~15 등 : (단어) n회

- Explanation

Find the frequency of words from the yes_words list and add them to the dictionary.

After sorting by frequency, it outputs up to the top 15 according to the format.

- applied learning

Dictionary, sorted, for loop

- screen shot

```
# 리스트 yes_words에서 각 단어 빈도수를 리스트를 통해서 딕셔너리에 추가
word_rank = {}
for word in yes_words:
    word_rank[word] = word_rank.get(word, 0) + 1

# 등장 횟수 순으로 정렬
sorted_word_rank = sorted(word_rank.items(), key=lambda x: x[1], reverse=True)
return sorted_word_rank

# 기능 3 단어의 빈도 수를 기준으로 랭킹을 선정하여 랭킹이 높은 순으로 정렬 후 상위 15위만 출력
1개의 사용 위치
def print_ranking(sorted_word_rank):
    print(f"상위 15개의 단어 랭킹:")
    for rank, (word, frequency) in enumerate(sorted_word_rank[:15], start=1):
        print(f"{rank} 등 : {word} {frequency} 회")
```

(4) save ranking to txt file

- In

Sorted_word_rank

-out

순위 빈도수 단어

N n n

N n n

“단어랭킹이 word_ranking.txt 파일로 저장되었습니다.”

- Explanation

The content of saving the results to a txt file for future use. The file to be saved is specified, and the sorted ranking list is taken and each content is sequentially stored in the txt file through the for loop.

- applied learning

File in out, dictionary, print, for loop

- screen shot

```
# 기능 4 저장된 랭킹을 txt 파일로 출력하는 기능
1개의 사용 위치
def save_ranking(sorted_word_rank, filename="word_ranking.txt"):
    with open(filename, 'w', encoding='utf-8') as textfile:
        textfile.write("순위\t\t\t빈도수\t\t\t단어\n")
        for rank, (word, frequency) in enumerate(sorted_word_rank, start=1):
            textfile.write(f"{rank}\t\t\t{frequency}\t\t\t{word}\n")
    print(f"단어 랭킹이 {filename} 파일로 저장되었습니다.")
```

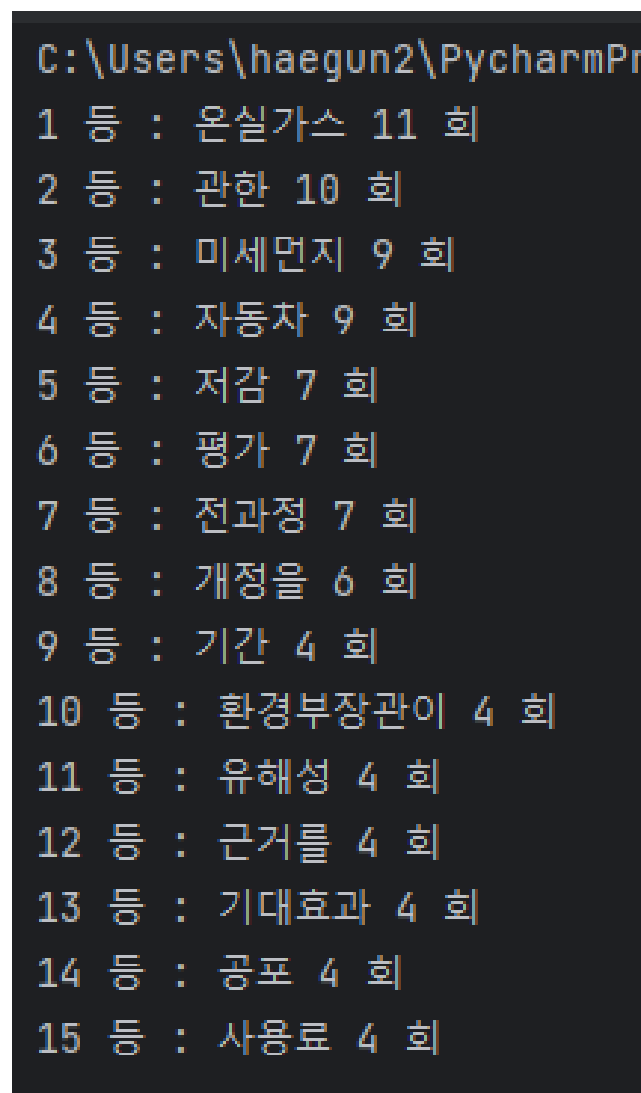
4. Test Result

(1)

- Explanation


Read the letter as txt from the input pdf, remove insignificant words, determine the frequency, and output to the top 15 depending on the format.

- Test Result screen shot



```
C:\Users\haegun2\PycharmPr
1 등 : 온실가스 11 회
2 등 : 관한 10 회
3 등 : 미세먼지 9 회
4 등 : 자동차 9 회
5 등 : 저감 7 회
6 등 : 평가 7 회
7 등 : 전과정 7 회
8 등 : 개정을 6 회
9 등 : 기간 4 회
10 등 : 환경부장관이 4 회
11 등 : 유해성 4 회
12 등 : 근거를 4 회
13 등 : 기대효과 4 회
14 등 : 공포 4 회
15 등 : 사용료 4 회
```


Example of pdf used for testing

 환경부	보도참고자료	다시 대한민국! 새로운 국민의 나라
보도시점	2023. 12. 8.(금) (배포 후 즉시)	배포 2023. 12. 8.(금)

미세먼지법 등 5개 환경법안 국회 통과

- 국민이 안심할 수 있는 깨끗하고 안전한 환경조성에 기여

환경부(장관 한화진)는 △‘미세먼지 저감 및 관리에 관한 특별법’, △‘대기환경보전법’, △‘화학물질의 등록 및 평가 등에 관한 법률’, △‘자원의 절약과 재활용촉진에 관한 법률’, △‘자연환경보전법’ 등 5개 환경법안이 12월 8일 국회 본회의를 통과했다고 밝혔다.

먼저, ‘미세먼지 저감 및 관리에 관한 특별법’은 미세먼지 배출저감 관리를 위해 초미세먼지(PM2.5) 월평균 농도가 심화되는 그해 12월 1일부터 이듬해 3월 31일까지의 기간 동안, 중앙행정기관과 지자체, 공공기관 등이 운영하는 공공배출시설에 대해 미세먼지 배출 저감조치를 시행하는 ‘미세먼지 계절관리제’를 적용해 왔다. 그러나 ‘미세먼지 계절관리제’ 기간 전후로도 지역별 초미세먼지 농도의 차이가 발생하고, 민간배출시설의 저감조치*는 의무적으로 적용되지 않는 등 실질적인 미세먼지 저감 효과를 기대하기에 일부 미흡한 측면이 있었다.

* 발전, 제철, 석유화학 등 대형사업장과 일부 중소규모 사업장이 자발적 협약을 통해 고농도 미세먼지 기간 저감대책에 참여 중

이에, 법률 개정을 통해 지역민의 건강 피해나 경제 영향 등을 고려하여, 시도지사가 필요 시 ‘미세먼지 계절관리제’ 기간을 연장할 수 있도록 하고, 환경부 장관의 미세먼지 저감조치 요청 대상을 공공배출시설에서 환경부령으로 정하는 민간배출시설까지 확대할 수 있도록 함에 따라 지역 특성에 보다 부합하고, 효과적인 미세먼지 배출 저감 효과를 기대할 수 있게 되었다.

‘대기환경보전법’은 개정을 통해 ‘자동차 온실가스 전과정 평가’의 정의 규정을 신설하고, 환경부장관이 관계부처와 함께 구체적인 평가 방법을 정하도록 했으며 자동차제작자에게 필요한 행정적·기술적 지원을 할 수 있는 근거가 마련됐다.

#Examples of saved text files

1	순위	빈도수	단어
2	1	12	온실가스
3	2	10	관한
4	3	9	평
5	4	9	미세먼지
6	5	9	전과정
7	6	9	근거
8	7	9	자동차
9	8	8	
10	9	7	저감
11	10	6	관리
12	11	6	개정
13	12	6	배출량
14	13	6	사용료
15	14	5	있
16	15	5	기간
17	16	5	마련
18	17	4	환경부
19	18	4	저감조치
20	19	4	일부
21	20	4	환경부장관
22	21	4	유해성
23	22	4	기대효과
24	23	4	공포
25	24	3	5개
26	25	3	국회
27	26	3	특별법
28	27	3	등록
29	28	3	절약과
30	29	3	배출
31	30	3	계절관리제
32	31	3	법률

5. Changes to plan

1) Change History Subject

- before

Remove investigations such as '은','는','이','가' from list token. And import Morpheme analyzer to make up a list with only nouns.

- after

Write a code that removes words and investigations that you don't use directly without importing morpheme analyzers.

- reason

I was thinking of importing morpheme analyzers. However, since the analyzer to be imported works using the java environment, we found that it is possible only if a specific environment is established. There was a problem that it was not easily and conveniently available on any computer. Therefore, we didn't import the content, but we directly configured it to be simple and maintained.

In that case, the removal of frequently used but insignificant words can also be maintained from time to time.

6. Lessons Learned & Feedback

파이썬으로는 키오스크를 만들어서 프로젝트를 진행해야겠다고 생각하고 기능을 구상하고 있었는데 중간고사 이후 파일 관련 내용이 꼭 들어가야하는 단서 조항이 생겨서 급선회 하면서 시행착오가 많이 있었습니다.

제가 생각하고 오 기발한데 했던 모든것들이 파이썬 라이브러리로 기본적으로 지원하는게 많아서 좌절감을 많이 느꼈습니다. 이번 프로젝트만 하더라도 원래 형태소 분석기를 통하여 형태소만 남기고 추출을 해주는 라이브러리가 있었는데 자바 환경도 깔려있어야 구동된다는 홈페이지 설명을 읽고 이참에 라이브러리 의존 없이 내가 만들어 보자! 생각을 하고 해당 형태소 분석기 내용은 제거한 후 수작업으로 조사나 불용어를 제거하는 내용을 작성하는 쪽으로 방향을 돌려 진행하였습니다. 다만 언어 분야 특성상 너무 방대한 조사나 불용어가 존재하여 그때 그때 랭킹을 확인하고 불용어 같은 경우를 no_word 리스트에 추가하면서 랭킹을 재 출력하는 번거로움이 붙을 수 있는 내용이지만 직접 구축하려고 시도했다는 점을 높게 사주시면 감사하겠습니다.

다만 라이브러리 의존 없이 진행하려고 하니 계속 여러군대에서 막히다 보니 결과적으로 대화형 인공지능에 의존을 하게 되는 아이러니함이 조금 생겼는데 그래도 제 실력으로 만드려고 최대한 노력하였고 인공지능이 만들어준 부분도 제가 읽고 수정하고 ppt를 확인하면서 복습하고 코드에 주석을 달며 여러 번을 고심한 끝에 만들면서 공부했습니다.

한학기동안 열정적인 모습으로 강의해 주셔서 진심으로 감동받았습니다. 추후 일반선택 학점이 남거나 제가 희망하고 있는 지능실감융합전공 복수전공의 전공 필수 학점이 인공지능학부의 교과목과 극소수 일부 겹치는 것으로 알고 있는데 교수님 수업과 겹친다면 꼭 다시 뵙고 싶습니다! 감사합니다.