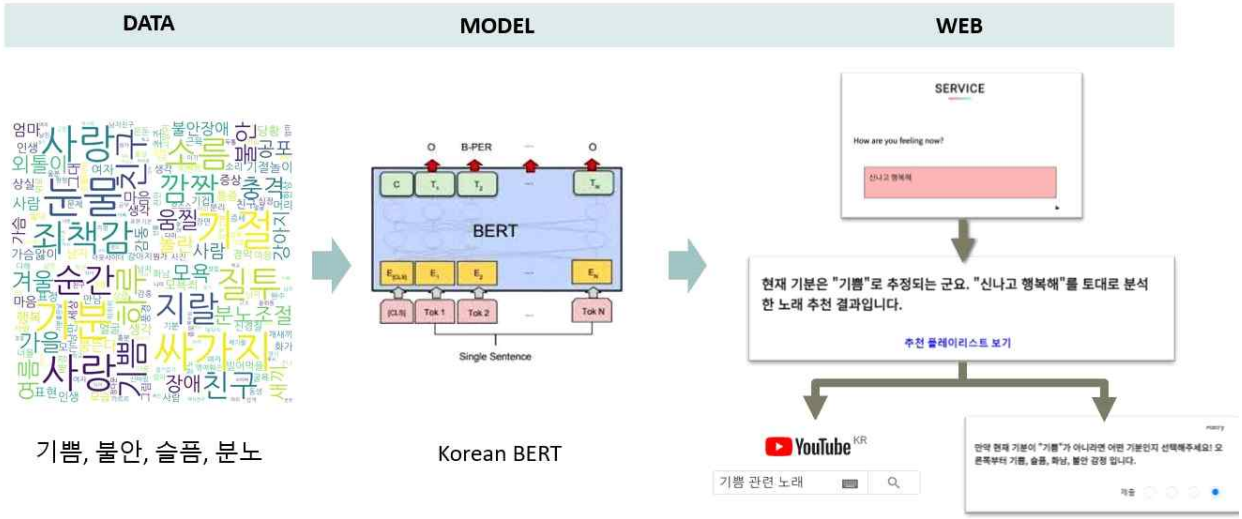
감정 분석을 통한 음악 추천 시스템

김해린, 이세미
세종대학교 지능기전공학부
https://github.com/lini1634/AdvancedML_project

요 약



웹앱에서 사용자가 자신의 감정을 문장으로 입력하면 그 문장을 기반으로 사용자의 감정을 분석하여 알려주고 관련된 음악을 유튜브 플랫폼에서 추천하여준다. 웹앱은 python Flask 프레임워크를 기반으로 하였으며 모바일에서도 사용이 용이하도록 반응형 웹으로 제작하였다. 감정 분석의 모델은 lstm과 kobert의 성능을 비교하여 최종적으로 kobert 모델을 채택하였다. 분석 결과가 잘못되었을 경우 사용자로부터 라벨을 입력받아 성능향상을 꾀하였다.

1. 개발 동기와 목적

1.1 개발 동기

최근 멜론, 플로, 유튜브 뮤직, 올레, 지니, 네이버, 벅스 등 다양한 뮤직플랫폼이 존재하여 사람들이 손쉽게 음악을 감상할 수 있다. 그에 걸맞게 수많은 음악이 존재하는데 선택의 폭이 증가하면서 어떤 음악을 들어야 할 지에 대한 고민 역시 증가하였다. 따라서 자신의 감정, 상태, 상황 등을 입력하면 그와 어울리는 노래를 추천해주는 서비스가 있으면 어떨까하는 생각을 가지게 되었고, 그를 기반으로 프로젝트를 진행하였다.

1.2 개발 목적

사용자가 웹에 임의의 문장을 입력하면 그 문장을 기쁨, 불안, 슬픔, 화남의 클래스로 분류하고, 추측한 감정을 기반으로 유튜브 플랫폼에서 노래를 검색하여 사용자에게 제공한다. 또한 추측 결과가 틀렸을 경우, 사용자에게 라벨을 입력받아 자체 데이터 수집과 추후 성능 개선이 가능하도록 하였다.

2. 개발 과정

데이터 클래스에 따라 3가지 버전을 거쳐 개발을 완료하였다. 첫 번째 버전은 기쁨, 슬픔, 우울, 화남, 설렘 다섯 가지의 클래스를, 두 번째 버전은 기쁨, 슬픔, 화남, 놀람 네 가지의 클래스를, 세 번째 버전은 기쁨, 슬픔, 화남, 불안 네 가지의 클래스를 분류하는 것을 목표로 개발하였다.

전체적으로는 우선 감정 인식을 하기에 적절한 텍스트 데이터를 각 클래스에 따라 네이버 지식인에서 크롤링하였다. 그리고 모은 데이터를 PCA를 이용하여 시각화한 결과와 wordCloud를 통하여 분석하였다. 분류 모델은 kobert와 lstm의 결과를 비교한 후, 최종적으로 kobert를 선택하였다. 이후 Flask를 기반으로 웹 애플리케이션의 back-end를 개발하고 html과 java script로 front-end를 개발하였다.

2.1 데이터 수집

네이버 지식인에서 각 클래스에 대한 관련 키워드를 사용하여 크롤링하였다. 관련 키워드는 한국어 감정표현단어의 추출과 범주화라는 논문을 참고하여 결정하였다. 데이터를 수집한 후, 중복데이터와 관련 없는 데이터를 정리 후에 분석 및 학습에 사용하였다.

2.2 데이터 분석

첫 번째 버전에서는 기쁨, 슬픔, 우울, 화남, 설렘에 대하여 데이터 수집을 하였다. 그리고 중복데이터를 제거하고 정리한 후 클래스당 데이터 개수는 기쁨 4300개, 설렘 3865개, 슬픔 3934개, 우울 3968개, 화남 2780개다. 그 결과를 wordCloud를 사용하여 분석한 결과는 아래와 같다.

[기쁨]

[설렘]

[슬픔]

[우울]

[화남]

위의 결과를 토대로 데이터를 분석해봤을 때, 기쁨과 설렘, 슬픔과 우울의 데이터가 비슷한 양상을 띠는 것을 확인할 수 있었다. 그러한 이유로 classification 결과에서 기쁨과 설렘, 슬픔과 우울 클래스 사이의 혼동이 많았다. 따라서 설렘과 우울 클래스를 삭제하고 놀람 클래스를 추가하기로 하고 두 번째 버전을 개발하였다.

두 번째 버전에서는 기쁨, 슬픔, 화남, 놀람에 대하여 데이터를 추가로 수집하여 보완하였다. 클래스당 데이터 개수는 기쁨 5739개, 슬픔 8639개, 화남 6252개, 놀람 4385개이다. 수집한 데이터를 wordCloud를 사용하여 분석한 결과는 아래와 같다.

[기쁨]

[슬픔]

[화남]

[놀람]

두 번째 버전의 경우 wordCloud상으로는 첫 번째 버전보다 데이터간의 유사성이 적지만 슬픔의 데이터 양이 다른 클래스보다 많아서 비교적 슬픔과 비슷한 클래스인 화남과 많이 혼동되는 것을 확인하였다. 또한 놀람 클래스의 경우 데이터를 수집하기가 쉽지않았다. 따라서 데이터 양의 편향을 줄이기 위하여 놀람 대신에 놀람과 공포를 포함하는 감정인 불안을 클래스로 사용하기로 결정하고 세 번째 버전을 개발하였다.

세 번째 버전에서는 기쁨, 불안, 슬픔, 화남 네 가지 클래스를 분류하는 것을 목표로 하였다. 그를 위하여 다시 데이터를 수집, 보완 후, 중복데이터를 제거한 결과 기쁨은 9144개, 불안 8086개, 슬픔 8589개, 화남 8468개이다. 그를 wordCloud로 시각화한 결과는 아래와 같다.

[행복]

[불안]

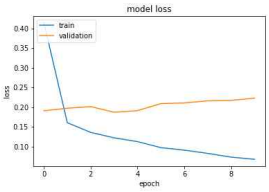
[슬픔]

[화남]

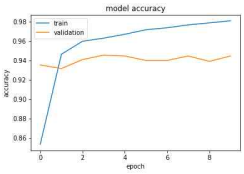
세 번째 버전의 경우 wordCloud상으로 봤을 때, 데이터간의 유사성도 적으면서 데이터가 클래스의 특징을 잘 표현하고 있는 것으로 확인이 되었다. 학습 후 분류 결과도 이전의 버전들보다 개선되었음을 확인하고 세 번째 버전을 최종 데이터로 채택하였다.

2.3 모델 성능 분석

첫 번째 버전의 경우 lstm 모델을 사용하여 학습을 돌린 결과는 다음과 같다.

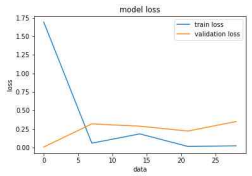


[lstm - loss]

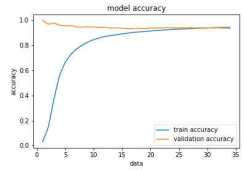


[lstm - acc]

위의 그래프를 보면 validation 결과에 비하여 train 정확도만 상승하는 것을 보아 과적합이 되고 있음을 확인할 수 있다.
아래는 kobert 모델을 사용하여 학습을 돌린 결과이다.



[kobert - loss]

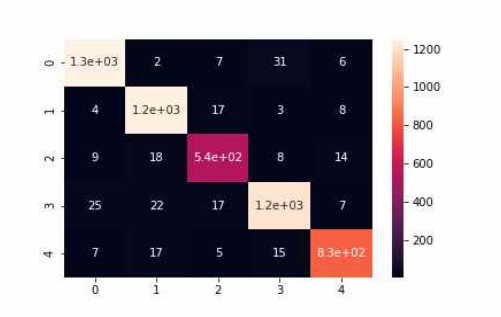


[kobert - acc]

위의 그래프를 보면 lstm보다는 덜하지만 과적합이 일어나고 있음을 확인할 수 있다.
아래는 kobert모델의 classification report와 confusion matrix이다.

	precision	recall	f1-score	support
happy	0.97	0.96	0.96	1298
sad	0.95	0.97	0.96	1277
depressed	0.92	0.92	0.92	591
love	0.95	0.94	0.95	1280
angry	0.96	0.95	0.95	875
accuracy			0.95	5321
macro avg	0.95	0.95	0.95	5321
weighted avg	0.95	0.95	0.95	5321

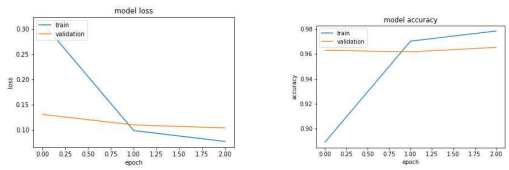
[classification-report]



[confusion-matrix]

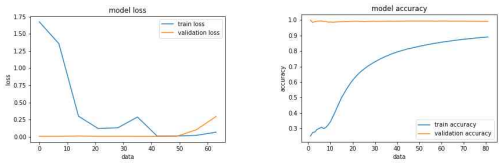
우울 클래스의 score들이 현저히 낮은 것을 확인할 수 있다.

두 번째 버전의 경우 lstm 모델을 사용하여 학습을 돌린 결과는 다음과 같다.



[lstm - loss] [lstm - acc]

이 역시도 과적합이 일어남을 확인할 수 있다.



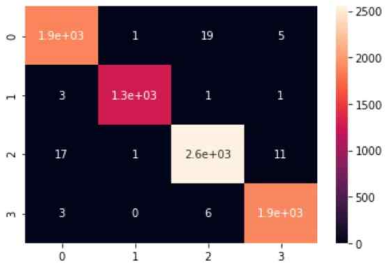
[kobert - loss] [kobert - acc]

위 그래프는 kobert 모델을 사용하여 학습을 돌린 결과로 학습이 잘 된 것을 확인할 수 있다.

다음은 kobert 모델에서의 classification report와 confusion matrix이다.

	precision	recall	f1-score	support
happy	0.99	0.99	0.99	1903
surprised	1.00	1.00	1.00	1280
sad	0.99	0.99	0.99	2579
angry	0.99	1.00	0.99	1906
accuracy			0.99	7668
macro avg	0.99	0.99	0.99	7668
weighted avg	0.99	0.99	0.99	7668

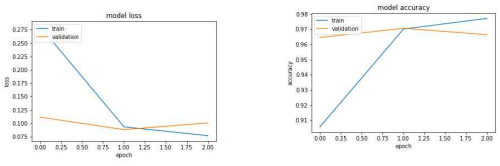
[classification-report]



[confusion-matrix]

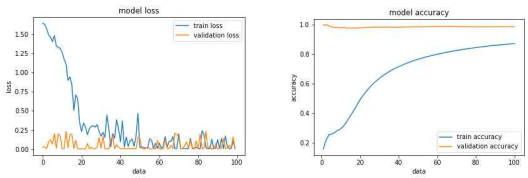
confusion matrix를 보면 2번째 클래스인 surprised가 다른 클래스들보다 추측이 빗나감을 확인할 수 있다.

세 번째 버전에서 lstm 모델을 사용하여 학습을 돌린 결과는 다음과 같다.



[lstm - loss] [lstm - acc]

이전 버전들보다는 괜찮지만 아직도 과적합이 있음을 확인할 수 있다.



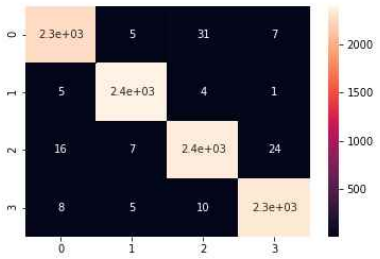
[kobert - loss] [kobert - acc]

위 그래프는 kobert 모델을 사용하여 학습한 결과로 학습이 잘 되었음을 확인할 수 있다.

아래는 kobert 모델에서의 classification report와 confusion matrix이다.

	precision	recall	f1-score	support
happy	0.99	0.98	0.98	2311
unstable	0.99	1.00	0.99	2415
sad	0.98	0.98	0.98	2410
angry	0.99	0.99	0.99	2368
accuracy			0.99	9504
macro avg	0.99	0.99	0.99	9504
weighted avg	0.99	0.99	0.99	9504

[classification-report]



[confusion-matrix]

confusion-matrix로 미루어보았을 때, 대체적으로 다 밝은 빛을 띠어 학습이 잘 되었음을 확인할 수 있다.

위 결과들을 토대로 lstm 모델보다 kobert 모델이 더 성능이 좋아 최종 모델로 kobert를 선택하였다.

2.4 웹 개발

웹은 기본적으로 front-end는 html과 java-script를 이용하였고, back-end는 python Flask를 이용하여 개발하였다.
사용자가 웹에 자신의 감정, 상태, 상황에 대하여 문장으로 입력을 주면, back-end에서 입력에 대한 추론을 진행한다. 그리고 추론 결과와 추론 결과를 기반으로 유튜브에서 검색을 진행한 웹페이지를 사용자에게 제공한다. 만약 사용자의 감정과 추론 결과가 다르다면 사용자에게 알맞은 라벨 입력을 요구한다. 그리하여 얻은 데이터를 엑셀 파일로 수집하여 추후 성능 개선 데이터로 이용한다.

또한 웹을 반응형으로 개발하여 사용자가 모바일에서도 사용이 용이하도록 하였다.

참고 자료

[1]https://github.com/SKTBrain/KoBERT
[2]충남대학교 사회과학대학 심리학과/ 뇌과학 연구소(2012). 한국어 감정표현단어의 추출과 범주화