

# 최종 보고서



아래 Emoji들은 emogenius를 사용한 결과입니다.



## 주제 변경 사유

### 교수님 강의 요약기 -> 한국어 문장 인식 후 emoji 추천

- 초기 목적은 **Google Bert** 모델을 **fine-tuning** 해보고 이를 이용해 application을 배포하는 것이었다. 따라서 다양한 자연어 처리 task들 중 고민했는데 교수님 강의 내용을 text로 변경한 후 이를 **skt-kobert** 모델로 요약해주는 프로젝트를 처음에 선택했는데 교수님 강의 audio를 google에서 제공하는 **STT**(speech-to-text)로 변경했더니 결과가 좋지 않았다. 결과는 아래와 같다.
- 그래서 다른 주제를 찾던 중 **kobert**로 감정을 분류해서 이에 맞는 **emoji**를 추천해 주는 기능을 생각했다.

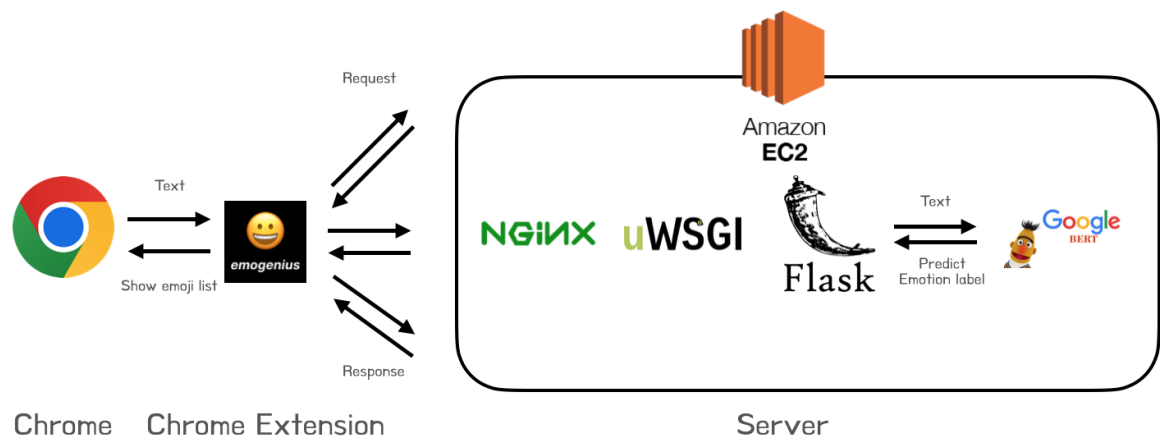


## 프로젝트 소개



문장에 맞는 **Emoji**를 사용할 수 있을까?

- **window나 mac**에서 emoji를 사용하면 전체 리스트에서 내가 자주 사용한 emoji들만 쓰게 된다. 이에 특정 상황(**문장**)에 맞는 **emoji**를 추천해주는 **app**을 만들면 어떨까라는 생각에서 시작했다.



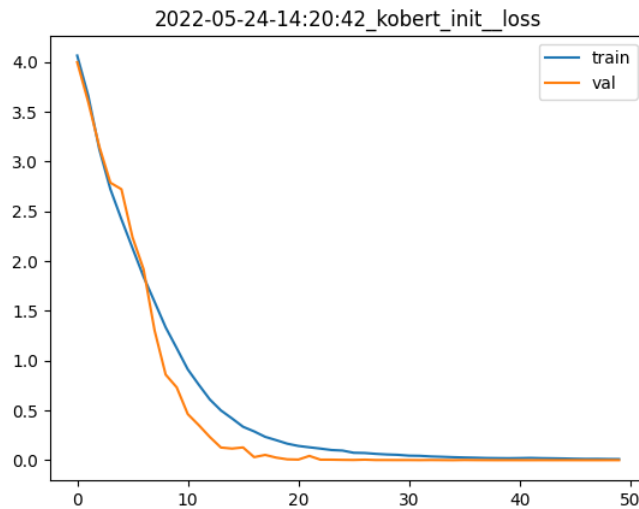
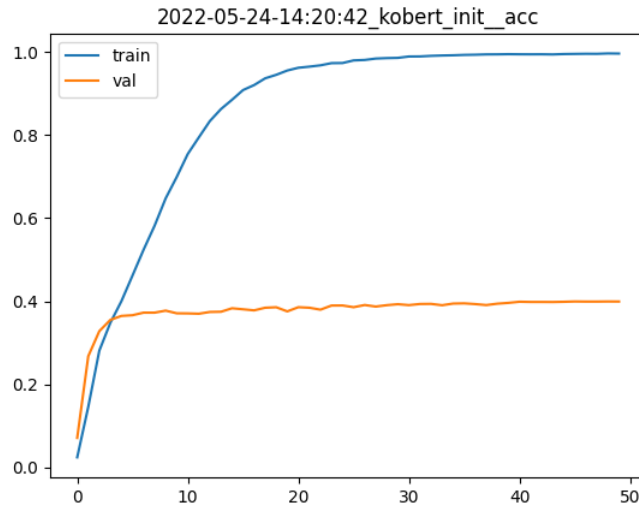
- 현재 결과물은 **chrome extension**으로 배포 준비중인 상태이다. 사용법은 README.md에 작성했다.
- dataset  
감성 대화 말뭉치
- model  
SKTBrain/KoBERT
- emoji  
emojddb

## 😞 프로젝트 수행 중 부딪힌 문제들과 해결 방법

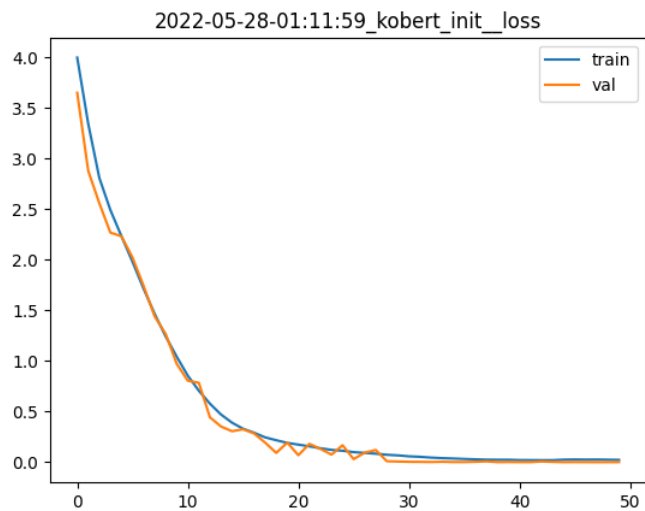
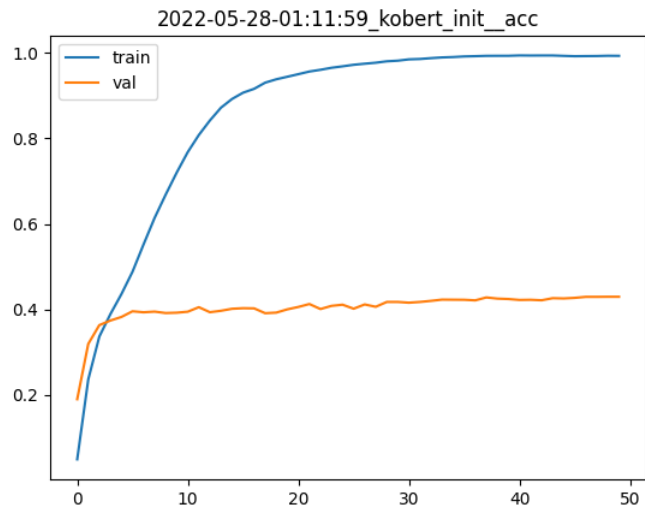
- 데이터 수집
  - 교수님이 알려주신 사이트나 AI-Hub 같이 다양한 데이터를 확인했지만 대부분이 Label 개수가 적었다. (Label 개수가 많을수록 좀 더 다양한 emoji를 추천할 수 있다고 생각했다) 그래서 blog 나 예제로 주어진 데이터 말고 새로운 데이터로 fine-tuning하고 싶었기에 데이터 선택에 있어 고민이 많았다. 그 중 감성 데이터셋을 찾을 수 있었고 다소 문장이 딱딱하지만 감정을 소분류로 분류할 경우 Label 개수가 58개나 됐기에 이를 선택했다.
- 모델 정확도
 

**skt-kobert**로 학습을 완료 했을 때 모델의 정확도가 **0.40** 정도로 높지 않은 결과가 나왔다. 따라서 이를 분석해봐야할 필요가 있었고 결과는 아래와 같다.

  - 초기 모델 돌린 결과



- 위 그래프를 확인했을 때 **3\_epoch** 이후 **validation**의 **accuracy**는 거의 변하지 않지만 **training accuracy**는 올라갔다. 하지만 **loss** 값을 봤을 때는 두 경우 모두 떨어지고 **validation**이 **역전**하는 모습은 보이지 않았다. 정확히 어떤 경우인지 모르지만 **overfitting**이라고 생각하고 다양한 학습 방식을 사용했지만 아래와 같은 방법에서 조금 효과를 보여 첨부했다.
- 학습 방법(K-fold), classifier\_layer 부분이 한층으로 되어있어서 2층으로 수정했다.

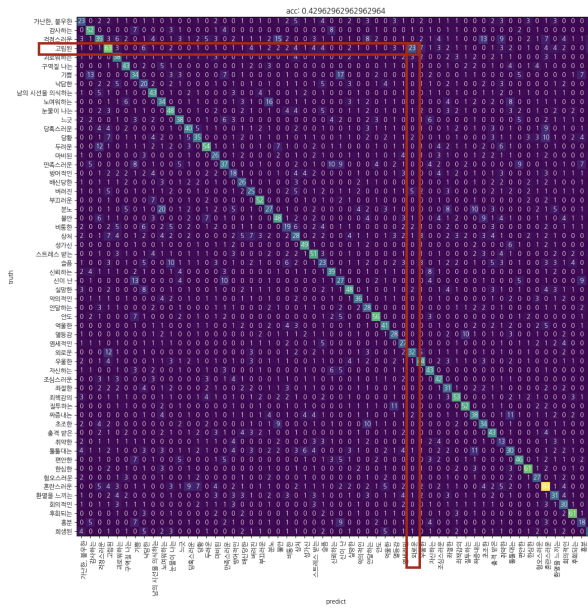
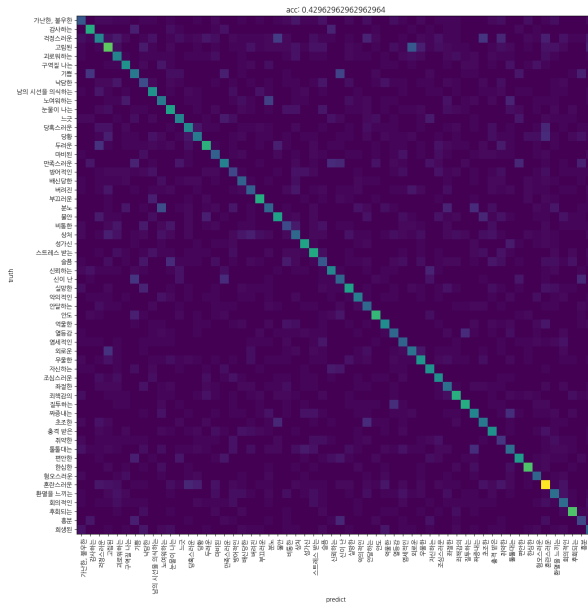


```
epoch 50 train acc 0.9926123437461778
100%|████████████████████████████████████████████████████████████████████████████████| 81/81 [0
epoch 50 test acc 0.42962962962962964
Training complete in 69m 6s
Best val Acc: 0.429630
```

- 위 그래프를 보서는 크게 증가한 지 확인이 불가능 하지만 실제 0.43정도로 0.02정도 올라갔다.

그래도 그리 높지 않은 결과이기에 **Confusion\_Matrix**를 찍어봤다. 결과는 아래와 같다.

- **Confusion\_Matrix**



가장 오차가 컸던 값을 확인해 보면 정확히는 분류하지 못했지만 상당히 분위기가 비슷한 것을 알 수 있었다. 따라서 이 모델을 사용하기로 결정했다.

	truth	predict	오차
#1	고립된	외로운	23
#2	분노	노여워하는	20
#3	기쁨	신이 난	17

- 서버 배포

- chrome\_extension에서 Javascript로 모델을 돌리는 방법(Tensorflow.js)를 사용 할까 했는데 조금 모험이라고 생각했다. 그래서 서버를 두고 chrome\_extension에서 text를 http 요청하는 방식을 선택했다.
- 하지만 Local에서 **ssl** 인증 없이 배포할 경우 **https** → **http**로 요청을 보내는 경우 Mixed content error가 발생했다. 그래서 **amazon ec2**를 통해 서버를 올리고 도메인을 구입해 **ssl 인증**을 적용해서 배포했다.



- chrome extension 배포

보고서 제출 전에 chrome extension 결과물을 배포하고 싶었지만 시간을 맞추지 못했다. 현재 검토 대기 중이고 꼭 배포해서 반응을 보고 싶다.

항목

+ 새 항목

보관처리됨

항목	유형	만든 날짜	최종 업데이트	등급	사용자	상태
 <b>emogenius</b> 버전 1.0.0	확장 프로그램	2022. 5. 27.	2022. 5. 27.	-	-	검토 대기 중
 <b>emogenius</b> 버전 1.0.0	확장 프로그램	2022. 5. 27.	2022. 5. 27.	-	-	검토 대기 중

<

항목 2개 중 1~2번째

>

## 프로젝트 수행 중 새롭게 배운 점 및 느낌점

- 프로젝트 수행 중 자연어 처리의 범주를 알 수 있었다. 이전에 한국어 임베딩이라는 책을 스터디한 적 있었는데 그때도 어렵다고 생각했지만 그 임베딩한 결과를 갖고 다시 **transformer**를 사용해 model을 만들고 학습한다는 과정이 매우 task가 많다고 생각했다. 프로젝트를 진행하고 수업을 들으면서 왜 요즘 **transformer**에 관심이 높고 **Computer\_vision** 분야에서도 사용할려고 하는지 깨닫게 됐다.
- Computer Vision 중 Object Detection에 관한 과제에 참여하고 있었는데 기존 YOLO 나 Mask-RCNN과 같이 이전 방식이 아닌 transformer를 적용한 모델에 대해서도 공부해 볼 동기가 생겼다.
- 서버 Model을 바꾸면 사용자들이 Model을 바로 사용할 수 있기 때문에 조금 더 BERT에 대해 공부하고 BART나 SOTA 모델을 다시 학습해 적용해 보고 싶다.

## 예상되는 자신의 프로젝트 점수

- 기본 점수: 20

- 완성도 점수: 20
- 아이디어 점수: 10
- 난이도 점수: 10

## 🔧 개선 사항

- KoBERT 기존 모델이 핵심이기 때문에 크게 변경하지 못했다. K-fold를 이용한 학습 방법과 마지막 classifier\_layer를 조금 수정했다.
- 기존 KoBERT 모델에 이용해 flask + javascript를 사용해 application을 배포

## 😭 참고 자료

### 감성 대화 말뭉치

데이터셋명 감성 대화 말뭉치 데이터 분야 음성/자연어 데이터 유형 텍스트, 오디오 구축기관 미디어젠 데이터 관련 문의처 담당자명 송민규 (미디어젠) 가공기관 미디어젠 전화번호 02-6429-7104 검수기관 미


 <https://aihub.or.kr/aidata/7978>



<https://github.com/SKTBrian/KoBERT>

### EmojiDB

Find emojis!

 <https://emojidb.org/>