이미지 분류 대회 Wrap up Report

Recsys 4조 이게딱이조
이채원, 김소미, 서현덕, 백승주, 유종문

1. 프로젝트 개요

A. 개요

이미지 분류 작업은 마스크 착용 여부 분류, 성별 분류, 연령대 분류 등 총 3가지 세부 작업으로 구성되어 있었다. 마스크 착용 여부는 착용, 미착용, 오착용, 성별은 남, 여, 연령대는 30대 이하, 30~60대, 60대 이상으로 각각 3개, 2개, 3개의 클래스를 갖고 있어, 분류해야 하는 최종 클래스는 18(3*2*3)개였다.

B. 활용 장비(도구)

- 개발환경: VSCode를 사용한 UpStage GPU와 SSH 연결
- 협업 tool: Notion, GitHub
- 의사소통: 카카오톡, Zoom, Slack, 필요시에 오프라인으로 프로젝트 진행

C. 기대 효과

사람들의 마스크 착용여부와 오착용 여부를 판별하고, 판별 대상의 연령대와 성별을 분류한다.

- 연령대 별로 마스크를 잘 착용할 수 있도록 만들 수 있다. (인식 확인)
- 마스크를 착용한 상태의 얼굴 인식 방법에 “성별” feature를 사용했다.

2. 프로젝트 팀 구성 및 역할

전체	문제 정의, 계획 수립, 모델 튜닝, 아이디어 제시
이채원	데이터 전처리, 데이터 Augmentation 전략 수립(Canny), 타 모델 조사, cross-validation, 모델/참고 자료 조사
유종문	EDA, 모델/참고 자료 조사 및 도입, 임시 제출 검증 시트 작성, ensemble (hard-voting) 구현, 실험 관리, train/test 데이터에 대한 RGB별 mean/std 값 재계산
김소미	EDA, 모델/참고 자료 조사, 데이터 Augmentation (FaceNet), 시각화, Custom Wandb 구축
서현덕	EDA, 데이터 Augmentation 전략 수립, 논문 모델 구현, 시각화, ensemble (soft-voting) 구현
백승주	데이터 전처리, 데이터 Augmentation 전략 수립(Cutmix), K-fold, 잘못 라벨링 된 데이터 처리, Github 관리

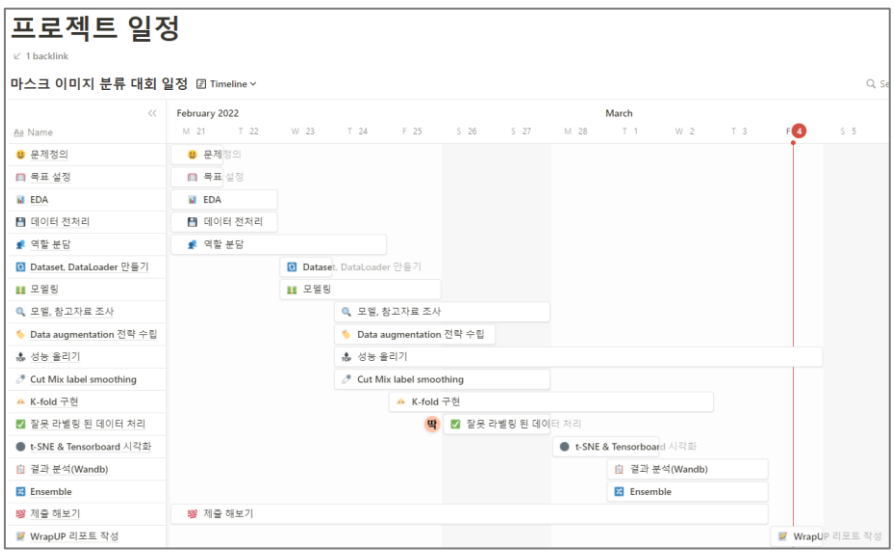
3. 프로젝트 수행 절차 및 방법

A. 목표 설정

- Validation accuracy 90% 이상 달성하기
- 모든 프로젝트 수행 과정 경험해 보기
- 리더보드 5등 이내 달성하기

B. 프로젝트 사전기획

i. 프로젝트 Task를 정의하고, 다음과 같은 일정을 수립한다.



ii. 역할 분배를 하고, P-Stage에서 진행되는 그라운드 룰을 생성한다.

① 기한 내에 못할 것 같으면 미리 말하기
② 언제든 도움 요청하기
③ 코딩 컨벤션 정하기
④ jupyter notebook파일 업로드 시, README와 markdown으로 설명 추가하기
⑤ 한 방법을 다 같이 고민해보고 안되면 다음 방법으로 넘어가기
⑥ 개선할 점과 단점을 이야기할 때 기분을 생각해서 이야기하기
⑦ 업무 현황을 잘 공유하고 소통하기

iii. GitHub 버전 관리 규칙

① Commit 할 때, 다음과 같은 Header 양식을 사용하기로 하였다. ([TYPE][Author] contents(#issue number))
TYPE 종류: FEAT, FIX, DOCS, STYLE, REFACTOR, TEST, CHORE

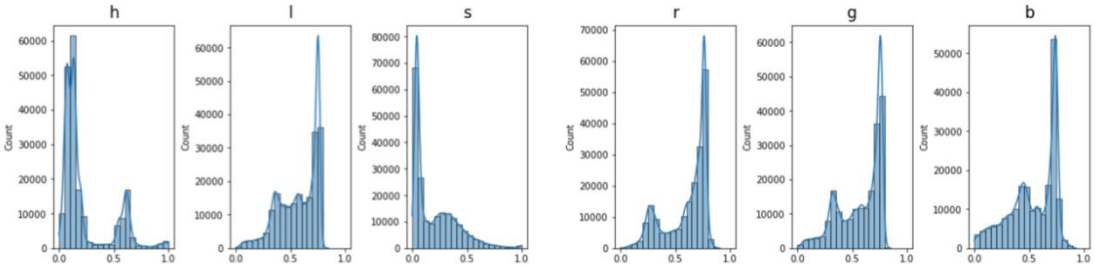
② Branch의 이름을 정할 때는 다음 규칙에 맞게 정한다. (feature/{기능}, hotfix/{#issue})
사용하는 branch: main, develop, feature, hotfix

C. 탐색적 분석 및 전처리 (학습 데이터 소개)

i. EDA

먼저 입력에 대한 csv 파일에 결측치가 존재하는지 확인하고, 입력 이미지의 크기가 어떻게 되는지 살펴보았다. 그 후, 실제 이미지가 어떻게 구성되어 있고 어떤 특징을 갖는지 확인해 보았다. 이제는 시각화를 사용해서 학습데이터의 분포가 어떻게 나타나는지 성별 분포, 나이대별 분포(단위: 1, 10, 30), 분류 클래스(18개)별 분포를 확인하였다.

이미지에 대한 RGB & HLS 분포를 확인하였는데, 항상 일정 수준 이상으로 비중을 차지하는 값이 있었다. 해당하는 값은 사람의 피부에 해당하는 색이었고, 이를 통해 얼굴 영역을 추출하여 변형할 수 있을 것으로 기대하였다.



ii. Data Preprocessing

폴더별로 구분되어 있는 학습 데이터를 프로그램에서 읽기 쉽게 구현하도록 하나의 폴더에 넣어서 정리하고, 그에 따른 CSV 파일도 다시 생성하였다. 이후, 분류하고자 하는 라벨로 재 라벨링을 진행하였다. 이미지를 다시 정리할 때는 이름을 라벨별로 다르게 지정하여 알아보기 쉽고, 프로그램상에서도 쉽게 이해하도록 구성하였다. 데이터에는 잘못 라벨링 된 데이터도 존재했는데, 이를 자동으로 수정하는 코드를 작성하였다.

D. 데이터

i. Dataset & DataLoader

① Split by Profile: 각 이미지의 파일을 사람별로 구분해서 train dataset과 validation dataset으로 구분하게 하였다. 이는 validation dataset에 학습 데이터 셋의 정보가 흘러 들어가는 것을 막기 위해서 사용하였다.

② Weighted Sampler: 학습에 사용되는 데이터의 라벨 별 분포를 고려하여 균등하게 이미지를 batch 크기로 반환하는 sampler를 직접 구현해서 사용하였다.

③ 나이 경곗값 제거(27~29, 57~59): 나이 클래스를 특정하기 힘든 경계에 있는 데이터를 제거하여 성능을 향상시키도록 노력하였다.

④ 60대 이상의 나이에 대해서만 50% 확률로 Horizontal Flip을 진행하였다. 이는 데이터가 부족한 연령대의 데이터를 더 많이 학습시킬 수 있게 만들어주었다.

⑤ CutMix Dataset

⑥ 나이, 성별, 마스크 착용 여부 별 Dataset

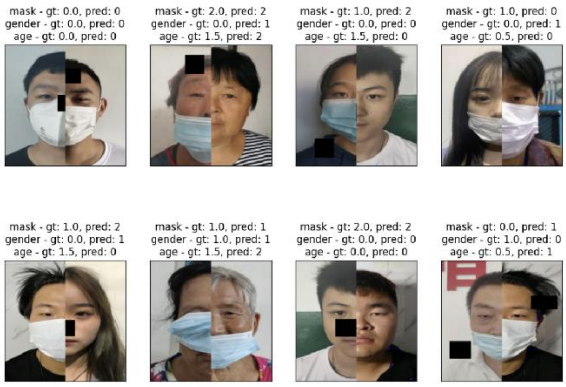
ii. Data Augmentation

① Normalize: 학습/테스트 데이터에 맞는 RGB 픽셀값의 평균과 표준편차 값을 구하여 이미지 정규화를 진행했다.

② FaceNet: 얼굴인식을 진행하도록 도와주는 FaceNet의 MTCNN을 사용하여 얼굴 이미지만 추출하도록 했다.

③ Canny Edge: 나잇값에 주름이 영향을 미칠 것으로 보아, 윤곽을 부각시켜주도록 Canny를 입히는 시도를 하였다.

④ CutMix: 동일한 라벨을 가진 이미지를 Horizontal 하게 잘라 이어 붙이고, 하나의 배치 내에 있는 이미지들을 섞는 CutMix 기법을 도입하였다.



⑤ Basic Transforms: HorizontalFlip, ColorJitter, CenterCrop, GaussianBlur, RGB to HLS & HLS manipulation

⑥ Random Erasing: 0.05 x 0.05 영역의 해당하는 부분을 지우는 RandomErasing 기법을 사용하였다.

⑦ Resize Image Size: EfficientNet의 기본 입력 이미지 사이즈에 맞추어서 입력 이미지의 크기를 조절하였다.

	B0	B1	B2	B3	B4	B5	B6	B7
Input	224	240	260	300	380	456	528	600
output	1280	1280	1408	1536	1792	2048	2304	2560

E. 시각화

i. Wandb 결과분석

① validation 과정에서 잘못 추론된 데이터의 이미지와 각각의 라벨을 출력했다.

② 첫번째 배치의 학습 데이터의 이미지와 라벨을 출력하여 확인할 수 있도록 했다.

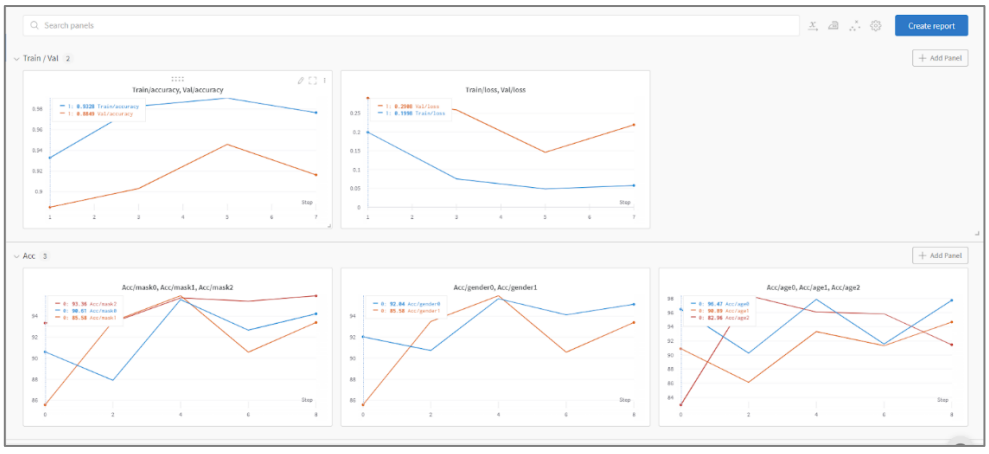
③ 학습 진행과정에서 Epoch당 loss와 accuracy를 기록했다.

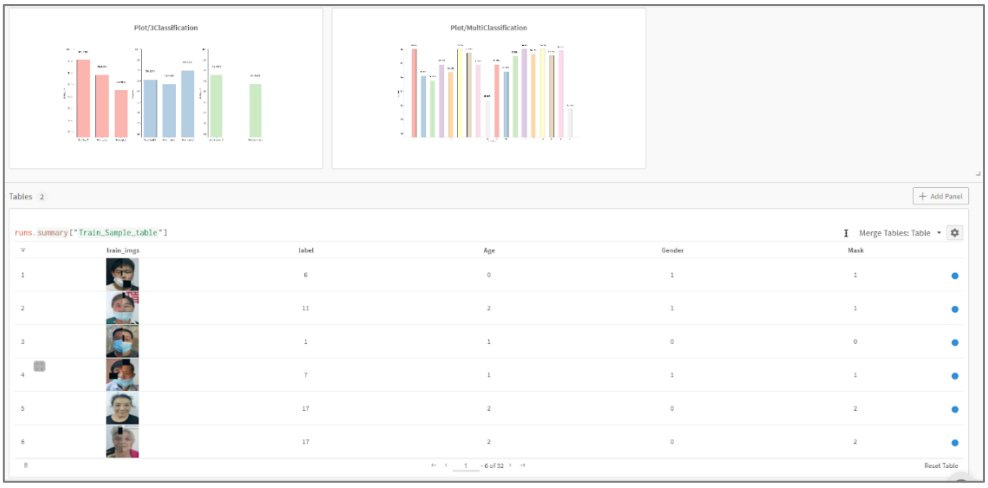
④ Task(mask, age, gender)별 accuracy 기록하고 막대그래프로 시각화 했다.

⑤ MultiClass별 acccuracy 기록하고 막대그래프로 시각화 했다.

⑥ 현재 하드웨어 상태 정보 확인 가능하다.

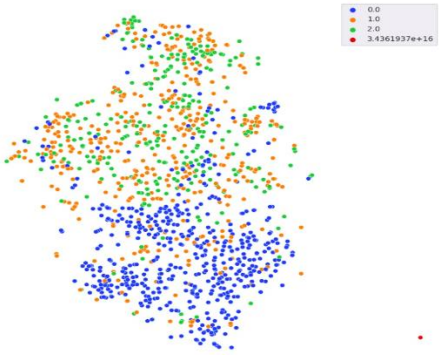
⑦ 같은 팀원들과 학습 기록/현황 공유, 비교 가능하다.





ii. T-SNE Clustering & Tensorboard Embedding

모델의 오 분류나, 데이터의 outlier가 있는지 확인하기 위해 t-SNE 임베딩 기법을 사용했다. 먼저 이미지 데이터를 t-SNE 함수를 통해 2차원 data point로 축소시킨 후, 각 레이블별로 clustering이 잘 되어 있는지 확인했다. 다음과 같은 결과 이미지를 얻을 수 있었다.



볼 수 있듯이, 레이블 0(30대 이하) 클래스는 잘 clustering 되었지만, 나머지 클래스들은 분류가 거의 안되고 있다.

t-SNE를 통해 모델들의 classifier가 잘 작동하고 있는지 대략적으로 확인할 수 있지만, 차원을 축소시키기 위해 t-SNE를 따로 학습을 시켜야 하기 때문에 clustering의 정확도를 보장하기 어려웠다.

F. 모델의 학습 결과 관리

							RandomErasing						
012	Efficientnet-b4 FaceNet	cross-entropy	Adam	StepLR	10	CutMixDataset	ToTensor Resize Normalize RandomHorizontalFlip RandomErasing	5	32	1e-4	93.53%		
013	Efficientnet-b4 FaceNet	cross-entropy	Adam	StepLR	10	CutMixDataset	ToTensor Resize Normalize RandomHorizontalFlip		32	1e-4	93.18%		Random Erasing을 제거해서 학습한 ⇒ 비슷한 성능을 유지하나 감소한 것으 다.
014	Efficientnet-b4	LabelSmoothingLoss	AdamW	Cosine-Annealing	10	CutMixDataset	ToTensor Resize Normalize RandomHorizontalFlip RandomErasing	10	32	1e-4	95.11%	80.0317%	Criterion을 labelsmoothing을 사용해보 → validation set에서 95%의 성능을 보 모습을 보여주었다. ⇒ facenet을 추가해서 한 번더 학습을
015	Efficientnet-b4 FaceNet	LabelSmoothingLoss	AdamW	Cosine-Annealing	10	CutMixDataset	ToTensor Resize Normalize RandomHorizontalFlip RandomErasing	10	32	1e-4	95.08%		
016	Efficientnet-b4 FaceNet	cross-entropy	Adam	StepLR	5	CutMixDataset	ToTensor RandomErasing	10	32	1e-4	96.25%	80.8731%	f1 = 0.7703

4. 프로젝트 수행 결과

A. 모델 평가 및 개선방향

resnet이 처음에 좋은 결과를 냈었다. 하지만 EfficientNet이 지속적으로 더 향상된 성능을 보여서 모델을 많이 바꾸기보다는, 데이터 전처리나 augmentation을 통해서 성능을 높이고자 하였다. EfficientNet의 b0~b4까지 실험을 모두 진행한 결과, b4가 가장 안정적이고 높은 성능을 보였기 때문에 b4를 이용해서 모델링을 진행했다.

분류가 애매한 부분을 제거하기 위해서 과감히 특정 나이대를 학습에 포함시키지 않는 방식을 진행하였다. 그 결과는 놀랍게도 성능이 올라감을 보여주었다. 추가로, CutMix를 직접 구현해서 augmentation 과정에 포함시킨 결과 정확도와 F1 Score가 한 번 더 향상되었다. FaceNet을 사용해서 얼굴 부분만 집중을 하여 학습을 하도록 진행해 보았지만 public dataset에서는 이전의 결과보다는 높지 못한 점수가 나왔었다. 하지만, 대회가 종료하고 나서 private data를 사용한 결과 F1 Score와 Accuracy 값이 이전의 결과보다 더 높았다.

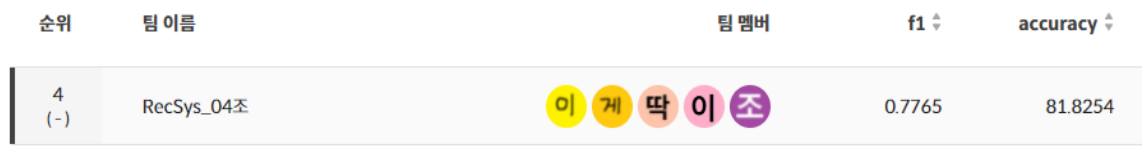
대회 종료날에는 가장 좋은 성능을 보인 모델을 가지고 Soft Voting을 진행하였으며 F1-Score는 높지 않지만, 정확도가 높게 나왔다. 대회종료 직전, 하나의 모델에서 각 task에 맞는 문제를 각각 분류하는 Multi Classification도 시도해보았지만, 시간제한으로 인하여 충분히 학습을 진행하지 못하였다.

B. 시연 결과

Model	Augmentation / Skills	F1 Score	Accuracy
ResNet34		0.7005	76.7302%
Efficientnet-b4	*BE, *RE	0.7240	78.1746%
Efficientnet-b4	BE, RE, CutMix	0.7621	81.1905%
Efficientnet-b4	BE, RE, CutMix, *HF (60’s)	0.7765	81.8254%
Efficientnet-b4	BE, RE, CutMix, FaceNet	0.7703	80.8731%
Efficientnet-b4	BE, RE, CutMix, Kfold, TTA	0.7756	82.2222%
Efficientnet-b4	Ensemble (Top3 Model)	0.7755	82.3651%
Efficientnet-b4	BE, RE, CutMix, HF (60’s), Multi Classifiaction		

*BE: Border Elimination(나이 경계 값 제거), *RE: Random Erasing, *HF: Horizontal Flip

C. 프로젝트 결과 및 사진



최종제출 모델의 Validation acc/loss

5. 자체평가 의견

A. 좋았던 점

① 프로젝트 초반에 세웠던 가설들을 증명하려고 했던 것이 좋았다.

② 역할분담에 따라, 맡은 역할을 각자 잘했고 서로 응원하고 칭찬하는 분위기와 팀워크가 좋았다.

③ 역할 분배가 잘 되었다.

④ 프로젝트 초기에 TASK 및 일정 계획을 잘 세웠다.

⑤ 프로젝트에 필요한 모듈을 모두 외부에서 가져다 쓴 것이 아니라, 직접 구현하고자 노력했다.

⑥ 프로젝트의 전반적인 과정을 쉽게 알아볼 수 있도록 notion에 문서화를 잘 시켰다.

⑦ EDA를 수행해서 데이터 분포를 먼저 확인했던 것이 후에 전략을 세우는 데에도 도움이 되었다.

⑧ 어려운 것이라고 단정 짓지 않고, 끝까지 해결하고자 했던 도전정신이 좋았다.

⑨ 이론 강의에서만 들었던 내용을 실제로 적용시키고 결과를 눈으로 확인해 보았던 것이 좋았다.

B. 아쉬웠던 점

① 최고 성능의 모델을 따라가지 않고, 더 다양한 모델로 시도를 해보았으면 좋았을 것이다.

② 학습 때마다 모든 실험에 대한 기록(성능, 결과물)들을 기록하지 못했던 것이 아쉽다. 더 체계적인 학습 케이스 관리가 필요하다고 느껴졌다.

③ 학습, 검증, 공개 데이터 셋에서 성능이 높다고 해서, 테스트 데이터 셋에 대해 성능이 높으리라는 보장이 없다. 그렇기 때문에, 가장 성능이 좋은 모델보다는 일반화가 잘 된 모델을 찾는 것이 좋을 것 같다. 랜덤 seed를 바꿔서 성능의 변화가 이루어나는지도 확인해 보면 좋을 것이라 느껴졌다.

④ 프로젝트를 진행했을 때 변수가 생겼을 때 침착하게 대응할 수 있어야 한다.

⑤ 프로젝트를 모듈화해서 파일의 중복이 없이 관리를 할 수 있으면 더 좋았을 것 같다.

⑥ 제출을 할 때는, 그 기준을 만들어서 제출 횟수를 더 소중히 다루도록 할 수 있으면 좋았을 것 같다.

6. Reference

① Irwan Bello, William Fedus, Xianzhi Du, Ekin D. Cubuk, Aravind Srinivas, EfficientNet, Rethinking Model Scaling for Convolutional Neural Networks, 2019

② Florian Schroff, Dmitry Kalenichenko, James Philbin, FaceNet: A Unified Embedding for Face Recognition and Clustering, 2015

③ Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, Youngjoon Yoo, CutMix: Regularization Strategy to Train Strong Classifiers, 2019

④ Github, EfficientNet-Pytorch, https://github.com/lukemelas/EfficientNet-PyTorch

⑤ Github, facenet-pytorch, https://github.com/timesler/facenet-pytorch

⑥ Towardsdatascience, Age Detection using Facial Images: traditional Machine Learning vs. Deep Learning

⑦ Github, pytorch_cutmix, https://github.com/hysts/pytorch_cutmix

1. 팀과 나의 학습목표

나는 처음으로 진행되는 프로젝트였기 때문에, 먼저 어떤 식으로 진행되는지 파악을 했어야 했다. 이후 전체적인 흐름을 파악하고 이번 대회에서는 역할분담을 진행하되, 모든 부분을 한 번씩 경험해보고자 하는 목표가 있었다. 그리고 이론강의에서 배웠던 내용들을 바탕으로 이번 모델링에서 적용시킬 수 있는 방법들을 편하게 브레인스토밍 하듯이 나열해보고, 최대한 많이 경험을 하는 방향으로 진행하기로 하였다.

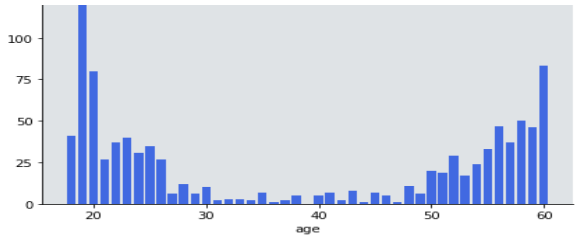
개인적으로도 위에서 목표를 정한 대로 필요에 따라 외부 블로그를 찾아다니면서 모델링과 관련된 정보를 찾고 직접 구현해보는 과정에서 AI모델링 내에서의 파이프라인을 경험해보고자 했다. 이 경험이 무조건 많은 기법을 적용시키면 좋을 것 같다는 생각과는 반대로, 데이터의 특성에 맞게 고려해서 적절한 기법들을 사용해야 한다는 것을 크게 와 닿게 해주었다.

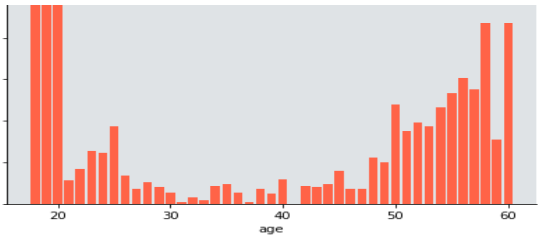
팀원 모두 마찬가지로, 멘토님이 이전에 말씀해주신 AI프로젝트의 대부분의 80%는 모델링보다 데이터를 보는 과정이 많고, 이 데이터를 직접 분석하는 시간이 길어질수록 성능을 크게 높일 수 있다고 하셨다. 그렇기에 데이터에 더 집중하자고 목표를 세웠다.

2. 모델을 개선시킨 방법

가장 처음에 기준을 삼고자 비교적 간단한 모델인 Resnet18과 Resnet34를 사용해서 별다른 augmentation과정 없이 그 결과를 제출하였다. 여기서 더 나은 성능을 내고자 이미지 분류에서 SOTA모델인 Efficientnet-b4를 사용하였다. 하지만 생각 외로, 성능이 높게 나오지 않았다.

이때부터, 성능을 높이기 위해서 데이터를 들여다보고 더 자세히 분석을 진행했다. 성능이 올라가지 못한 원인을 분석하기 위해서 어떤 클래스를 잘 분류하지 못하는지 알아보았다. 그 결과 30~59세, 60세이상의 나이대 분류를 잘 하지 못하는 것을 파악했다. 그러고 나서는 바로 데이터를 보았다. 실제로 그 분포를 보니, 57~59세와 60세의 데이터 분포가 몰려 있었던 것을 파악했다. 여기서 한가지 가설을 세웠다. "**두 클래스 모두가 동일한 feature를 학습하고 있겠구나**". 데이터를 크게 늘리지 못한다면 오히려 이런 잘못된 학습방법을 제거하기 위해서 57~59세의 데이터를 학습에 사용하지 않기로 하였다. 그 결과로 성능은 약 3%정도 향상될 수 있었다. (29위→3위)





여기에 팀원의 도움을 받아서, 얼굴부분만 학습을 진행시키기 위해서 facenet을 적용시키고, 이미지의 더 다양한 부분을 관찰하여 학습할 수 있도록 random erasing기법, horizontal flip, canny를 사용해보았고, 이미지 2개를 합치는 CutMix를 사용했다. 기본적으로 주어진 Normalize과정에 필요한 mean과 std값을 실제 학습과 평가에 사용하는 데이터로 재계산을 해서 끝까지 0.0001%라도 올리고자 노력을 했다. 최종적으로 그 결과는 private 데이터를 포함하여, 80.3651%의 정확도와, 0.7718의 F1-loss를 달성했다.

3. 내가 한 행동의 결과와 깨달음

내가 한 데이터를 제거하는 행동은 안 그래도 적은 데이터를 더 적게 사용하는 것은 어떻게 보면 일반적으로 생각할 수 있었던 것은 아니었던 것 같다. 하지만 한 번 과감히 실험해보고자 진행을 해보았는데, 그것이 성능 개선의 출발점이 되고, 모두의 사기도 올릴 수 있다고 느껴져서 큰 뿌듯함으로 돌아왔다.

대회 진행중에는 하루에 제출횟수가 제한되어 있어서 팀에서 진행한 모든 실험들을 제출하기에는 무리가 있었다. 그렇기에 Excel을 사용해서 점수가 높은 상위 5개의 제출결과를 가져와서 간이 정답을 만들었다. 이 간이 정답과 실험결과를 비교하면서 이번에 성능이 어떻게 될지 그 변화를 판단할 수 있는 기준이 되었다. 대회 후반부에는 제출하기전에 한 번씩 확인을 하고 제출을 진행했었다. 덕분에 더 좋은 실험을 많이 진행할 수 있었다.

4. 한계와 아쉬운 부분

초반에 성능이 잘 나온 모델만을 가지고 실험을 했던 것이 많이 아쉬웠다. 1, 2위를 달성한 팀의 모델을 보니 다양한 모델로 실험을 했음을 알 수 있는데, 나는 초반에 좋은 성능을 내는 모델을 결정하고 고정한 뒤, 실험을 진행한 것이 많이 아쉬웠다.

대회 기간동안 V100서버가 쉬는 시간이 없을 정도로 50번이 넘는 실험을 진행했었다. 하지만 유의미한 결과만을 내는 실험을 기록했는데, 모든 실험을 기록했으면 또 다른 인사이트를 얻었을 수 있겠다는 아쉬움이 남았다.

최종 제출로 가장 public dataset에서 가장 높은 성적을 낸 모델을 제출했는데, public dataset에서는 결과가 낮았지만 실제로 나의 생각에서 성능이 좋을 수밖에 없던 다른 모델이 private dataset에서 더 높은 성적을 기록했었다.

5. 개선사항, 새롭게 시도해볼 수 있는 것

다음에는 하나의 모델만을 고정해서 사용하는 것이 아니라, 다른 모델로도 여러 실험을 진행해보고 싶다. 생각과 그 결과가 다를 때는 그 원인을 분석하는 것이 정말 중요하다고 생각되어서 해당 부분을 다음 프로젝트나 대회에 적용시키고자 한다.

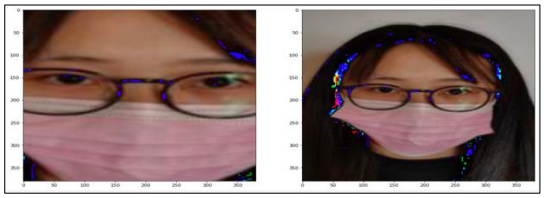
1. 학습목표

개인 학습 목표는 Pytorch 사용 및 구현 능력을 갖추는 것과, 데이터 분석의 중요도를 인지하고 다양한 방법으로 데이터를 분석해보는 것으로 정했다. 팀의 공동 학습 목표는 개개인이 최대한 여러 과정에 참여해보며 프로젝트 구조에 익숙해지는 것으로 정하고, 최종 목표는 리더보드 5위 안에 진입하는 것으로 결정했다. 개인 학습 측면에서는 Baseline 코드를 하나하나 뜯어보며 이해하려고 노력했고, 필요한 기능을 모듈화하여 기존 코드에 추가하는 과정을 통해 Pytorch 사용을 익숙히 했고, 프로젝트 구조의 전체적인 흐름을 파악할 수 있었다. 또한, EDA를 통한 학습 데이터의 분포와 모델의 학습과정을 Log기록 및 시각화 하며 면밀한 데이터 분석을 할 수 있었다. 공동 학습 측면에서는 프로젝트 초기에 역할 및 태스크를 나누고, 간트 차트를 통해 일정 계획을 세우며 우리 팀만의 프로젝트 템플릿을 구축하는 법을 배울 수 있었다. 또한, 프로젝트 그라운드 룰과 Git 버전 관리 규칙을 정함으로써 협업 능력을 키울 수 있었다.

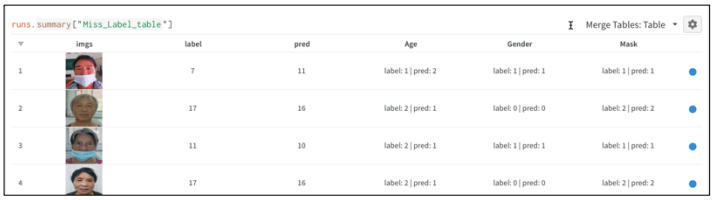
2. 모델 개선 방법

학습 목표는 사람의 마스크의 착용 여부, 성별 및 나이 예측이었으므로, 배경 정보를 제거하고 사람의 얼굴 정보만 입력 된다면 성능이 개선될 것이라 생각했다. 따라서 FaceNet의 MTCNN 모델을 사용하여 사람의 얼굴을 인식하고, 해당 부분의 이미지만 뽑아내는 Custom Transform을 구현했다. MTCNN 모델을 그냥 사용하면 오직 사람의 눈, 코, 입만 나오게 사진이 추출되는데, 머리 스타일 또는 이마의 주름 정도는 성별과 나이 구분을 하는데 중요한 Feature라 판단하였고 Crop Margin을 늘림으로써 정수리부터 턱까지 온전한 얼굴 이미지가 추출될 수 있도록 하였다. 이외에도 얼굴의 주름을 부각시킬 수 있는 Canny Edge의 Transform을 구현했다.

또한, 정확한 모델 분석을 위해 Wandb를 사용하여 마스크, 성별, 나이 각각의 Accuracy와 현재 학습 데이터의 샘플 사진 그리고 오답처리된 데이터의 사진 및 라벨링을 확인할 수 있는 Custom Wandb 클래스를 구현하였다. 직접 시각화된 이미지를 보면서 epoch이 지날수록 각 라벨의 정확도가 어떻게 변하는 지, 오답 데이터에 대해서 어떤 라벨을 잘못 추론하고 있는 지를 쉽게 분석할 수 있었다.



[FaceNet / Margin 변경 전 -> 변경 후]



[Custom Wandb - Miss Label Table]

마지막으로 멘토님께서 피드백 해주신 Multi Classifier Model을 구현했다. 마스크, 성별, 나이에 대해 각각의 Fully connected layer을 추가하여 같은 Feature을 입력으로 받고 Loss 값을 따로 계산할 수 있도록 했다. 세가지 태스크의 Loss에 각각 다른 가중치를 두어서 좀 더 추론이 약한 태스크에 비중을 두고 멀티 학습을 할 수 있도록 했다.

3. 수행 성과 및 깨달음

FaceNet의 사용이 성능은 높게 유지되면서 일반화가 잘 되는 것을 확인할 수 있었다. 팀에서 사용한 Augmentation 중 Cutmix가 성능 개선에 높은 효과를 보였는데 단순히 Cutmix만 적용했을 때 f1 스코어는 0.7765 (Public Data) -> 0.7597 (Private Data)으로 성능이 떨어졌지만, FanceNet을 통해 얼굴 부분을 추출한 후 Cutmix를 진행했을 때는 0.7703 (Public Data) -> 0.7718 (Private Data)으로 거의 성능차이가 나지 않았다.

리더보드에 최종 제출한 모델과 실제 Private 데이터 셋에서 최고 성능을 이룬 모델이 달랐었는데, 이를 통해 단순히 Public 데이터 셋의 성능에만 의존하는 것이 아니라 얼마나 일반화가 잘 된 모델인지 검증하는 것 또한 중요하다는 것을 깨달았다. 또한, Random Erasing, 경계 값 제거, Random Sampling 등 창의적인 Data Augmentation 기법들이 모여 점진적으로 성능향상이 이루어진다는 것을 알게 되었다.

4. 새롭게 시도한 변화

프로젝트 초기에 EDA를 통해 데이터의 분포와 각 라벨 간의 연관성 등을 좀 더 자세하게 파악했고 적어도 어떤 부분을 개선해야 하는지 확신할 수 있었다. 또한, 여러가지 모델을 사용하여 성능을 실험해봄으로써 각 모델의 어떤 레이어를 우리의 목적에 맞게 바꿔야 하는 지 방향을 잡을 수 있었고 새로운 모델 사용에 대한 두려움도 없어졌다. 베이스라인 코드 분석이 오래 걸렸지만 그 만큼 프로젝트 구조에 대해서 자세히 이해할 수 있었고 그 결과, 여러 기능을 적용해서 추가할 수 있었다.

5. 한계 및 아쉬웠던 점

결과 분석 툴인 Wandb를 뒤늦게 사용해서 다양한 실험들에 대한 결과를 효율적으로 관리하지 못한 점이 아쉬웠다. Multi Classifier Model을 마지막 날에 구현해서 다양한 실험을 해보지 못한 점이 아쉬움이 크게 남았다. 또한, 배경을 제거하기 위해 Semantic Segmentation을 적용해보려고 했지만 구현에 있어 한계를 느꼈었다. 다른 팀 중에서도 배경 제거를 성공한 팀이 있었는데 rembg라는 라이브러리를 통해 손쉽게 제거 한 것을 보고 다음부터는 포기하지 말고 다양한 방법을 찾아봐야겠다고 결심했다.

6. 개선사항, 새롭게 시도해볼 수 있는 것

다음 프로젝트에서는 K-fold 또는 랜덤 데이터 셋을 통해 일반화 검증을 시도해보고 싶다. 또한, 기능 구현할 때 최대한 간단하게 모듈화하여 다른 팀원이 크게 코드를 수정하지 않아도 사용할 수 있도록 구현해보고 싶다.

1. 학습목표

이미지 분류 대회가 현재까지 배워왔던 이론과 실습을 제대로 사용해 볼 수 있는 기회라고 느꼈다. 그래서 최대한 많은 것을 경험해 보는 위주로 각 단계를 깊이 파고들지는 않아도 한 번씩은 거쳐볼 수 있도록 노력하고자 하였다. 강의 내용을 따라가면서 차근차근 프로젝트를 진행하자고 정했다. 개인적으로 모델링보다는 데이터 전처리와 augmentation, data loader처럼 그전의 과정에 좀 더 관심이 있었다. 또한, notebook 형식의 코드에서 벗어나서 python 파일로 적응하고 싶어서 최대한 notebook을 사용하지 않기로 생각했다. 팀에서는 가설을 세워서 그 가설을 적용하면서 성능을 높이자는 큰 목표가 있었다. 모델을 만들고 돌아가는 코드에만 만족하지 않았고, 할 수 있는 모든 옵션들을 적용할 의지 있었다. 또한, github를 제대로 사용하여 협업하고자 했다.

2. 나는 어떤 방식으로 모델을 개선했는가?

A. 데이터 전처리

많은 사람이 반복해서 데이터에 접근할 때마다 필요한 라벨과 정보들이 추가되면 비효율적일 것 같아서 한 번에 접근하기 쉽도록 train.csv 구성을 pandas를 이용해서 추가하고 수정했다.

B. 데이터 증량

이미지의 특성을 살려 다양한 필터를 적용할 수 있다는 사실과 데이터는 많을수록 적지만 imbalance가 큰 영향을 준다는 것을 알고 있어서, 다양한 filter를 알아보았다. 가장 큰 골칫거리이던 나이 예측 문제를 해결하기 위해서 얼굴 주름이 나이와 연관되어 있을 것이라는 가설을 세우고 Canny Edge를 적용했다.

C. 교차 검증

주어진 제출 기회가 많지 않아서 결과를 accuracy와 f1-score에 의존해야 하는 것이 아쉬웠다. 그래서 최대한 일반화된 수치를 얻기 위해서 cross valiation을 알아보았다. K-fold를 적용하기 위해서 dataset 관련 코드를 보며 데이터 처리와 흐름을 공부하고 적용을 시도했다.

3. 내가 한 행동의 결과로 어떤 지점을 달성하고, 어떠한 깨달음을 얻었는가?

전까지는 모델과 파라미터에 의존해서 성능을 높이려고 했었는데, 데이터를 처리하고 로드하면서도 할 수 있는 것들도 많다는 사실을 깨달았다. 학습 중 나오는 accuracy나 f1 score을 100% 확신하기 어렵다는 생각이 들었다. 전보다 높게 나온 모델을 리더보드에 제출하면 오히려 낮아지는 결과를 마주하며 cross validation이 시간이 오래 걸리더라도 구현을 해 보아야겠다는 생각을 하게 되었고, 실제로 k-fold 모델이 private score은 더 좋게 나온 것을 보고 더욱 교차 검증의 중요성을 깨달았다. 라벨링을 하는 코드를 구현하면서 중간단계를 확인하지 않아서 특정 클래스는 아예 나오지 않는 문제가 발생했다. 값만 잘 출력된다고 구현이 잘 된 것이 아니라는 사실을 다시 한번 깨달았고, 한 단계 처리를 할 때마다 꼭 데이터를 열어서 확인해 보아야겠다고 다짐했다.

4. 전과 비교해서, 내가 새롭게 시도한 변화는 무엇이고, 어떤 효과가 있었는가?

가설을 세우고 그에 맞춰 전처리와 모델 설계를 하는 도전이 새로웠다. 학습의 결과를 사람이 해석하기 어렵다고만 한정 지어 생각했었는데, 기계가 학습하는 요인들을 직접 생각해서 주름이나 데이터양을 고려했다. 효과가 있기도 없기도 했지만, 학습에 초점을 맞추는 방법을 배운 것 같다. 그리고 sampler 적용도 새로웠다. 어차피 한 번은 마주하게 될 data들인데 그 순서가 많이 중요할까라고 생각했는데, 나은 성능이 나왔다. 같은 batch에 같은 사진이 들어가는지에 대한 에러를 고치면서 공부하며 random.sample과 choice의 차이를 알아보고, 조금 더 나은 코드로 바꾸었다.

5. 마주한 한계는 무엇이며, 아쉬웠던 점은 무엇인가?

baseline이 처음 봤을 때 한 번에 모든 것을 받아들이긴 힘들어서 알고 있던 이론들도 코드를 많이 바꾸어야 하는 것은 구현이 어려웠다. 그리고 EfficientNet이 왜 이 dataset에 좋은 결과를 내는지 등의 분석을 하지 못한 채 그저 성능이 잘 나오는 것만 골라서 조사한 것이 아쉽다. 3개의 task를 모두 다른 모델로 학습을 해서 결과를 합치는 Multi classification을 처음 문제를 접했을 때 떠올리기만 하고 3개의 모델을 모두 손을 대기에는 시간 비용이 너무 커서 진행하지 않았는데, 생각보다 단순하게 구현할 수 있던 것을 알고 조금 더 빨리해보지 않았던 것이 아쉬워졌다.

6. 한계/교훈을 바탕으로 다음 프로젝트에서 스스로 새롭게 시도해볼 것은 무엇일까?

실험 결과를 저장하고 분석하는 데에는 힘을 많이 쏟지 못했다. 최소한 wandb와 같은 도구를 제대로 사용해보고 싶다. 또한, 각 단계에서 시도해 볼 수 있는 많은 방법들이 있는데 전의 경험들이 (sampler나 최종 학습 시 validation data까지 포함해서 학습하기 등)을 잊어버리지 않게 체계화해서 나만의 flow를 만들어 다음 프로젝트 때에는 모두 시도하고 싶다. 그리고 처음부터 내가 구현한 코드만 존재할 수는 없으니, 많은 template들과 코드를 자주 접하도록 노력해야겠다고 다짐했다.

1. **나는 내(팀) 학습목표를 달성하기 위해 무엇을 어떻게 했는가?**
   나와 우리 팀의 학습 목표는 모두가 추후 프로젝트를 위해 최대한 다양한 경험을 하는 것이었다. 이를 위해 개인 측면에서는 기본 강의 내용에 충실하되 필요 시 새로운 내용을 적극적으로 찾아보고자 했고, 팀 측면에서는 자율적인 의견 제시를 장려했으며 원활한 소통을 위해 다양한 협업 툴을 활용했다.

2. **나는 어떤 방식으로 모델을 개선했는가?**
   A. **Data Transforms & Weighted Sampler**
      Gaussian blur, random erasing, 등 다양한 이미지 변형 기법을 사용했다. 성능 향상에 가장 도움이 됐던 기법은 Random Erasing 이었다. 무작위로 일부 영역을 제거하기 때문에 모델은 매 epoch 마다 이미지의 다른 영역을 학습할 수 있었다. 또한 데이터 불균형을 해소하기 위해 weighted sampler 를 만들어 사용했다.

   B. **Data Visualization by t-SNE Embedding**
      모델의 예측 결과를 시각화하기 위해 t-SNE 를 활용했다. t-SNE 로 이미지 데이터를 2 차원 data point 로 축소시킨 후 레이블과 함께 좌표 평면에 표시하여, 레이블 별로 clustering 이 잘 되었는지 확인할 수 있었다.

   C. **Implementing a Pretrained Model**
      나이를 분류하는 것이 가장 어려운 task 였기 때문에, <Deep CNN for Age Estimation based on VGG-Face Model>이라는 논문에 나오는 모델을 동일하게 구현하여 나이 분류 성능을 높이고자 했다.

   D. **Ensemble (Soft voting) & Multi Classifier Model**
      어느 정도 모델 실험이 진행된 후, 가장 성능이 좋았던 모델들을 soft voting 앙상블로 조합했다. 각 모델의 장점을 살려 성능을 높일 수 있었다. 또한 하나의 모델 안에서 feature extraction layer 는 고정한 채 말단에 있는 classifier 를 여러 개 만들어 subtask 를 할당하는 multi classifier model 을 만들었다.

3. **내가 한 행동의 결과로 어떤 지점을 달성하고, 어떠한 깨달음을 얻었는가?**
   다양한 data augmentation, 모델링 기법을 시도했고, 그 중 ensemble 과 random erasing 이 높은 성능에 기여했다. 성능을 향상시키는 과정에서 깨달은 것은, 사람이 아닌 기계가 학습하는 방식을 고려해야 된다는 점과 틀에 갇히지 않아야 한다는 점이다. 첫번째로, 기계의 관점에서 생각해야 된다. 배경을 줄이는 center crop, 얼굴을 선택하여 색을 부각시키는 기법, 등 초기 기능들은 사람의 학습 방식을 고려하여 도입했기 때문에 성능을 높이지 못했다. 시행착오를 통해 기계가 데이터를 인식하는 방법, Convolution 의 원리, 등 기계의 관점을 이해할 수록 random erasing 과 같이 더 효과적인 방법을 찾을 수 있었다. 다음으로, 틀에 갇히지 않아야 된다고 생각했다. 처음에는 최고 성능을 갱신하는 모델이 나올 때마다 그 모델을 튜닝 해야 된다고 생각했다. 물론 이 방식도 점진적으로 성능을 높여주지만, 더 유의미한 성능 향상을 위해서는 기존의 모델과는 아예 다른 접근을 여러가지 시도해야 된다는 것을 깨달았다. 팀에서 나온 face detect, cutmix, boarder line value erasing, random erasing, ensemble, 등은 모두 이런 발상의 결과물이었다.

4. **전과 비교해서, 내가 새롭게 시도한 변화는 무엇이고, 어떤 효과가 있었는가?**
   이번 프로젝트에서 함께 사용할 수 있는 툴을 만드는 것과 논문에 나온 모델을 구현하는 것을 도전해봤다. 먼저 제출 횟수 제한이 있기 때문에 우리만의 성능 검증 툴이 필요하다고 생각했다. 따라서 모델의 subtask 별 성능을 텍스트 파일로 출력하는 기능과 t-SNE 로 레이블별 clustering 을 시각화하는 기능을 구현해서 팀원에게 공유했다. t-SNE 는 차원 축소 학습이 잘 이루어지지 않아 제대로 사용하지 못했지만, subtask 별 성능 출력 기능은 모델을 평가하는 데 꾸준히 사용되어 성능 향상에 간접적으로 기여했다. 작더라도 함께 사용할 수 있는 툴을 만들면 '승수 효과'가 있다는 것을 깨달았다. 또한, 논문에 나온 나이 예측 모델을 동일하게 구현해봤다. Pretrained model 을 불러오는 것부터, architecture 와 hyper-parameter 를 조정하는 것까지 경험할 수 있었다. 코드를 뜯어보며 내용도 온전히 이해할 수 있었고, 코딩에 대한 자신감도 생겼다.

5. **마주한 한계는 무엇이며, 다음 프로젝트에서 스스로 새롭게 시도해볼 것은 무엇일까?**
   모델 평가와 고난이도 코딩 구현에서 한계를 마주했다. 우리는 정확도만 고려하여 최종 제출 모델을 선택했다. 하지만 정확도가 높더라도 그 이유를 설명할 수 없는 모델들은 결국 일반화 성능이 안 좋았고, test 성능이 오히려 떨어졌다. 다음 프로젝트에서는 단순 정확도보다 일반화 성능을 더 정확히 평가할 수 있는 지표를 연구하여 사용하고 싶다. 또한 transformer, GAN과 같이 복잡한 이론은 구현하지 못했다. 이번 프로젝트에서 코딩 구현을 경험해본 만큼, 다음 프로젝트에서는 스스로에게 한계를 두지 않고 필요하다는 생각이 들면 뭐든지 구현을 시도해보고 싶다.

1. **나는 내 학습목표를 달성하기 위해 무엇을 어떻게 했는가?**

   우리 팀과 나의 학습목표는 무엇이었나?

   - 우리 팀의 학습목표는 대회 진행과정을 전부 경험해보는 것이었다. 모델생성, 데이터로더 및 데이터셋 만들기, 데이터 전처리, 데이터 Argumentation, ensemble, 하이퍼 파라미터 튜닝 전 과정을 경험하는 것이 목표였다. 대회 시작 전 우리 팀의 목표는 5 등이였고 최종적으로 4 등을 달성할 수 있었다.

   개인 학습 측면

   - 개인적인 학습 측면으로 데이터 Argumentation (CutMix)과 CV(K-fold)이후 생성된 모델을 평균을 취하는 방법을 배울 수 있었다. 이번 프로젝트를 수행하며 학습시키는 코드를 팀에서 사용하는 방식에 맞춰 수정하고 데이터셋을 새로 만들어보면서 Pytorch 를 이전보다 더 잘 다룰 수 있게 되었다.

   공동 학습 측면

   - 공동 학습 측면으로는 모듈화 된 Baseline 을 수정하며 사용해보며 프로젝트 구조를 이해할 수 있었으며, Git 을 활용해 협업 하는 방법을 배울 수 있었다.

2. **나는 어떤 방식으로 모델을 개선했는가?**

   사용한 지식과 기술

   - Miss labeling 된 데이터를 자동으로 수정하는 코드를 작성해서 팀원들에게 공유하였다.
   - 데이터 Argumentation 의 방법 CutMix 하는 코드를 찾아보며 우리 데이터에 적용해보았다.
   - 데이터 학습과 모델 검증을 위해 CV 기법 중 하나 인 K-fold 를 적용해 보았다. 기존 데이터셋에 동일한 사람의 데이터가 7 개 들어있었기 때문에 이를 train 데이터셋과 validation 데이터셋에 섞여 들어가지 않게 K-fold 하기 위해 인덱스가 Shuffle 되지 않도록 기존 데이터셋을 상속받아 수정하였고 이 데이터 셋을 K-fold 에 적용해 보았다.
   - Inference 시에도 K-fold 를 사용해 나온 모델 5 개의 평균 값을 submission 에 반영하기 위해 평균을 취하는 OOF(Out of Fold) 방법을 사용해서 inference 하였다.

3. **내가 한 행동의 결과로 어떤 지점을 달성하고, 어떠한 깨달음을 얻었는가?**
   - Cut Mix 를 적용하여 유의미한 F1 스코어의 상승을 달성하였고 대회 중반까지 상위권을 유지할 수 있었다. 데이터의 분포가 Unbalance 한 경우에는 CutMix 를 통해 유의미한 성능 향상을 가져올 수 있음을 알 수 있었다.
   - K-fold 를 한 모델을 OOF 하여 추론했을 때 public 리더보드에서는 F1 스코어가 낮아졌다. 이후 Private 리더보드에서 확인했을 때에는 최종 제출했던 모델보다 더 나은 F1 스코어를 확인할 수 있었으며 K-fold 를 통해 모델 성능이 더 향상될 수 있다는 걸 알 수 있었다.

4. **전과 비교해서, 내가 새롭게 시도한 변화는 무엇이고, 어떤 효과가 있었는가?**
   - 딥러닝 관련 프로젝트는 처음이었지만, U-stage 에서 배웠던 내용을 모두 해보려고 시도했다. Modeling, Data Preprocessing, Data Argumentation, Training, ensemble 등 배웠던 내용을 한번씩 구현해보려 노력했다. 이 과정을 통해 이론으로 알고 있던 내용을 실제로 구현할 수 있다는 자신감을 얻을 수 있었다.

5. **마주한 한계는 무엇이며, 아쉬웠던 점은 무엇인가?**
   - 다양한 모델과 손실함수를 사용해보지 못한 것이 아쉬웠다. 처음에는 여러 모델을 사용해보았으나 나중에는 성능이 가장 잘 나오는 efficientNet 만을 사용해서 더 많은 시도를 하지 못한 것이 아쉬웠다.
   - Wandb 같은 시각화 도구를 처음부터 사용해서 학습 결과를 팀원들과 공유했으면 실험 관리를 보다 효율적으로 하지 못한 것이 아쉬웠다.

6. **한계/교훈을 바탕으로 다음 프로젝트에서 스스로 새롭게 시도해볼 것은 무엇일까?**
   - 다양한 모델과 손실함수를 사용해볼 것
   - 협업할 수 있는 환경(wandb,mlflow)을 먼저 구축하고 프로젝트를 진행할 것