Image Classification Wrap-up Report

부스트캠프 AI Tech 5기 CV-08 (오기의 아이들)

김보경(T5033), 박상필(T5082), 이태순(T5165), 임현명(T5179), 천지은(T5214)

1. 팀 Wrap up

1.1 프로젝트 개요

● 프로젝트 주제

카메라로 촬영한 사람 얼굴 이미지의 마스크 착용 여부를 판단하는 프로젝트로 사람의 사진이 input으로 들어왔을 때 마스크를 착용여부, 나이, 성별을 예측하여 18개의 class로 분류하는 모델을 개발하였습니다.

COVID-19의 확산 방지를 위해 사람들은 마스크 착용, 사회적 거리 두기 등의 많은 노력을 하고 있습니다. 이 중 마스크 착용은 감염자로부터의 전파 경로를 차단하기 위한 중요한 방법으로, 올바르게 착용하는 것이 중요합니다. 하지만 넓은 공공장소에서 많은 사람들의 마스크 착용상태를 검사하는 데에는 많은 비용이 요구됩니다. 적은 비용으로 마스크 착용 여부를 판별하기 위해 마스크 착용 여부를 분류하는 모델을 개발하였습니다.

● 활용 장비 및 재료

사용언어 & 라이브러리

IDE

협업 tool

GPU: v100

● 프로젝트 구조 및 사용 데이터셋의 구조도(연관도)

1. 기본 학습 및 평가 코드 구성요소

1) dataset.py

- 마스크 데이터셋을 읽고 전처리를 진행한 후 데이터를 하나씩 꺼내주는 Dataset 클래스를 구현한 파일
- CustomAugmentation, MaskBaseDataset, MaskMultiLabelDataset 구현
- rembg_dataset.py : 배경제거한 데이터셋을 활용하기 위해 이미지를 로드할 때 .convert('RGB')를 추가한 파일

2) loss.py

- 이미지 분류에 사용될 수 있는 다양한 Loss 들을 정의한 파일
- Cross Entropy, Focal Loss, Label Smoothing Loss, F1 Loss 구현

3) model.py

- 데이터를 받아 연산을 처리한 후 결과 값을 내는 Model 클래스를 구현하는 파일

- vit_base_patch16_224, vit_small_patch16_384, vgg16_bn, resnet50, resnet101, densenet121, densenet201, efficientnet_b1, inception_resnet_v2, swin_tiny_patch4_window7_224, swin_large_patch4_window12_384 구현

4) train.py
- 실제로, 마스크 데이터셋을 통해 CNN 모델 학습을 진행하고 완성된 모델을 저장하는 파일
- skf_train.py : Stratified k-fold 적용
- train_multiclass.py : Multi-Labeling 적용
- train_optuna.py : optuna 적용
- train_cutmix_60s.py : 60세 이미지 패치로 CutMix
- train_cutmix_all_ages.py : 모든 나이대에 이미지 패치로 CutMix
- skf_train_cutmix.py : 모든 나이대에 이미지 패치 CutMix에 Stratified k-fold 적용
- skf_train_multiclass.py : Multi-Labeling에 Stratified k-fold 적용
- train_cutmix_multiclass.py : Multi-Labeling에 모든 나이대에 이미지 패치 CutMix 적용
- rembg_train.py, rembe_train_multiclass.py : 배경제거한 데이터셋을 활용하기 위해 라이브러리 선언만 변경

5) inference.py
- 학습 완료된 모델을 통해 test set 에 대한 예측 값을 구하고 이를 .csv 형식으로 저장하는 파일
- inference_multiclass.py : Multi-Labeling 적용

2. 기본 데이터셋 구성요소

1) 전체 사람 명수 : 4,500
2) 한 사람당 사진의 개수 : 7장 (마스크 착용 5, 이상하게 착용 1, 미착용 1)
   a) 마스크 착용 : mask1, mask2, mask3, mask4, mask5
   b) 이상하게 착용 : incorrect_mask
   c) 미착용 : normal
3) 한 사람당 저장된 폴더명 : id_gender_race_age
   a) gender = {Female, Male}
   b) race = {Asian}
   c) $18 \leq age \leq 60$
4) 이미지 크기 : (512, 384)
5) 전체 데이터 중 학습데이터 60%, 평가데이터 40%

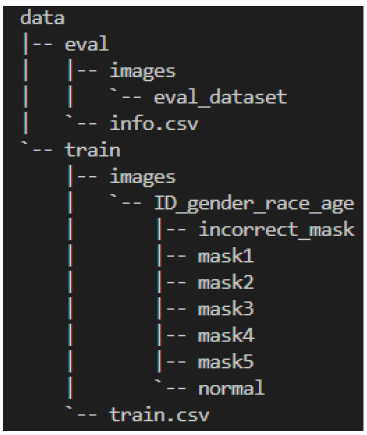
3. 증폭 데이터셋 구성요소

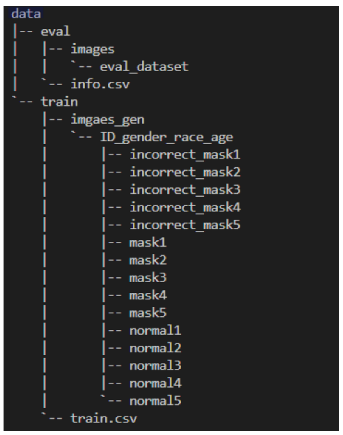
기본 데이터셋에서 albumentations 라이브러리를 활용하여 train/images datasets에 1개씩 존재하던 incorrect_mask, normal image를 4개씩 Augmentation 적용하여 증폭한 데이터셋

* 적용한 Augmentation 기법

```
trfm = Compose([
        ShiftScaleRotate(p=1.0),
        HorizontalFlip(p=0.5),
        ColorJitter(0.1, 0.1, 0.1, 0.1),
        RandomBrightnessContrast(brightness_limit=(-0.3, 0.3),
        contrast_limit=(-0.3, 0.3), p=1.0), ], p=1.0)
```

* 기본 데이터셋과 증폭 데이터셋의 디렉토리 구조도





4. 배경제거 데이터셋 구성요소

기본 데이터셋에서 rembg 라이브러리를 활용하여 train/images datasets에 remove 함수를 통해 사람을 제외한 배경제거한 데이터셋

- 일부 데이터셋에서 배경 제거가 제대로 되지 않는 현상 존재
- 배경이 투명한 이미지 형태로 PNG로 저장
- 디렉토리 구조 : images 폴더명이 images_rembg로 변경된 것 외에 기본 데이터셋과 동일

5. 배경제거 + 증폭 데이터셋 구성요소

배경이 투명했던 배경제거 데이터셋에 albumentations 라이브러리를 활용하여 train/images datasets에 1개씩 존재하던 Incorrect_mask, normal image를 4개씩 Augmentation 적용하고 배경을 검은색으로 변경하여 증폭한 데이터셋

- 디렉토리 구조 : images_gen 폴더명이 images_rembg_gen로 변경된 것 외에 증폭 데이터셋과 동일

* 적용한 Augmentation 기법

```
trfm = Compose([
        ShiftScaleRotate(p=1.0),
        HorizontalFlip(p=0.5), ], p=1.0)

# 원본 데이터는 aug기법 적용하기 않기 위한 trfm2
trfm2 = Compose([], p=1.0)
```

6. CutMix를 위한 증폭 데이터셋 구성요소

증폭 데이터셋에 적용한 Augmentation 기법에서 CutMix로 활용하기 위해 ShiftScaleRotate 변환만 제거하여 incorrect_mask, normal image를 4개씩 Augmentation 적용하여 증폭한 데이터셋

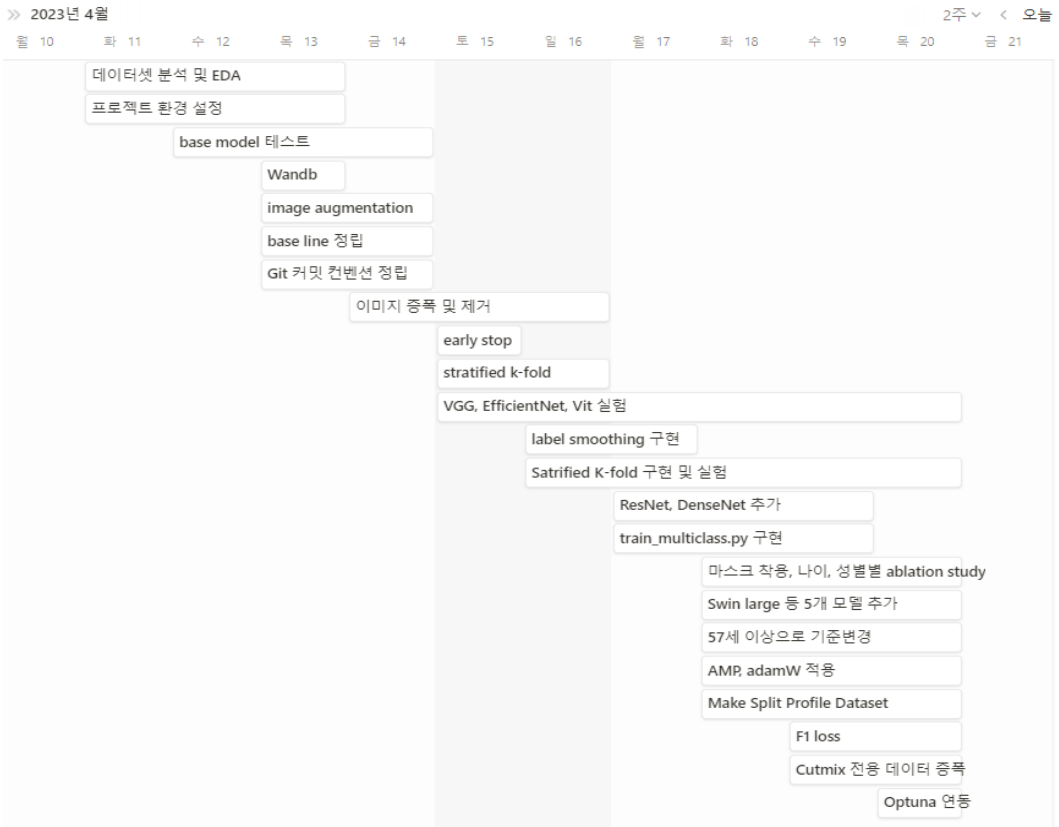
* 적용한 Augmentation 기법

```
trfm = Compose([
        HorizontalFlip(p=0.5),
        ColorJitter(0.1, 0.1, 0.1, 0.1),
        RandomBrightnessContrast(brightness_limit=(-0.3, 0.3),
            contrast_limit=(-0.3, 0.3), p=1.0), ], p=1.0)
```

1-2. 프로젝트 팀 구성 및 역할

팀원	역할
김보경	wandb 연동 및 관리, optuna 연동, base model 설계 및 실험, 가중치 실험
박상필	EDA를 통한 이상치 제거 및 불균형 분석, F1 Loss 실험, AMP 적용을 통한 컴퓨팅 리소스 개선
이태순	Github 협업 준비, multi-class 모델 설계 및 실험
임현명	vit(tiny) 실험, 마스크 착용, 나이, 성별별 모델 설계 및 실험
천지은(팀장)	데이터 증폭 및 배경제거, Stratified k-fold, CutMix 설계 및 실험, 사전학습모델 찾기

1-3. 프로젝트 수행 절차 및 방법

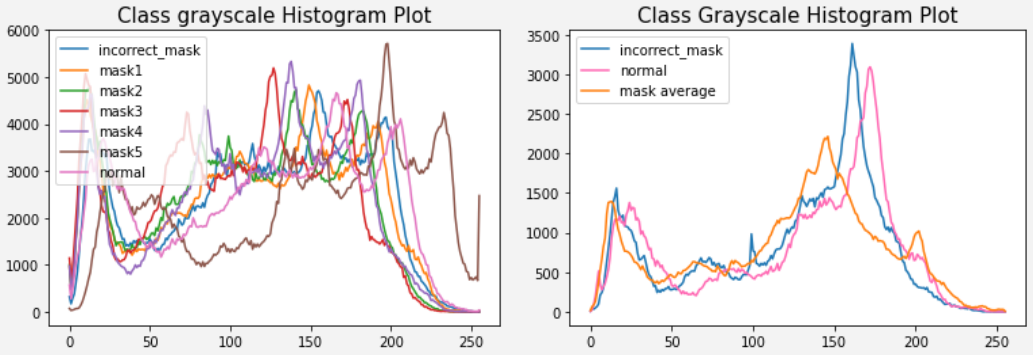


1-4. 프로젝트 수행 결과

0. EDA

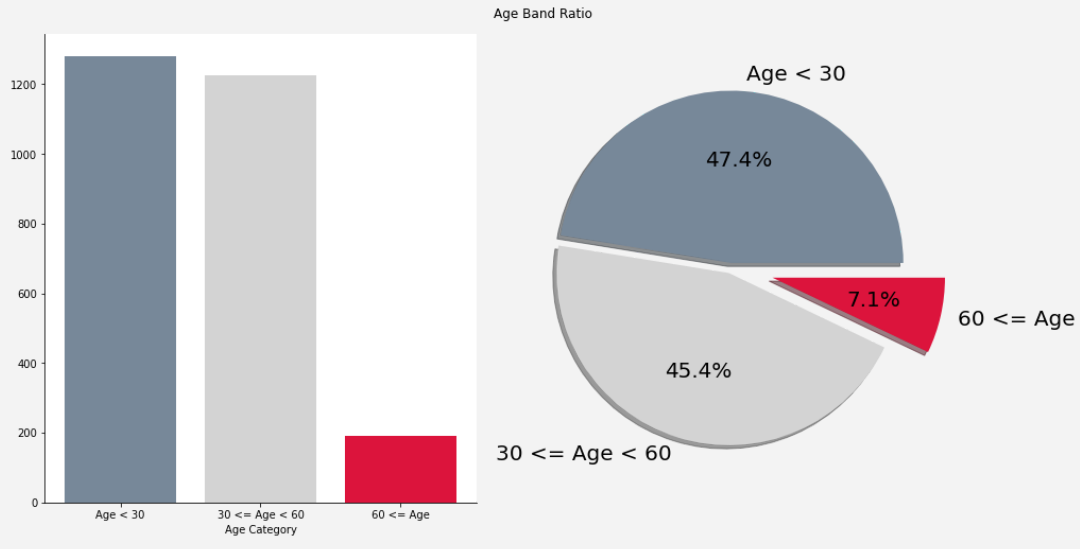
1) Train Data Distribution Analysis

● Grey(Color) Scale 분포 분석



- 마스크 5번의 특이사항을 발견할 수 있었습니다.
- 실제 데이터를 탐색해본 결과, 평범하지 않은 화려한 패턴의 마스크를 착용한 사진이 주로 마스크 5번으로 Labeling 되어 있었습니다.

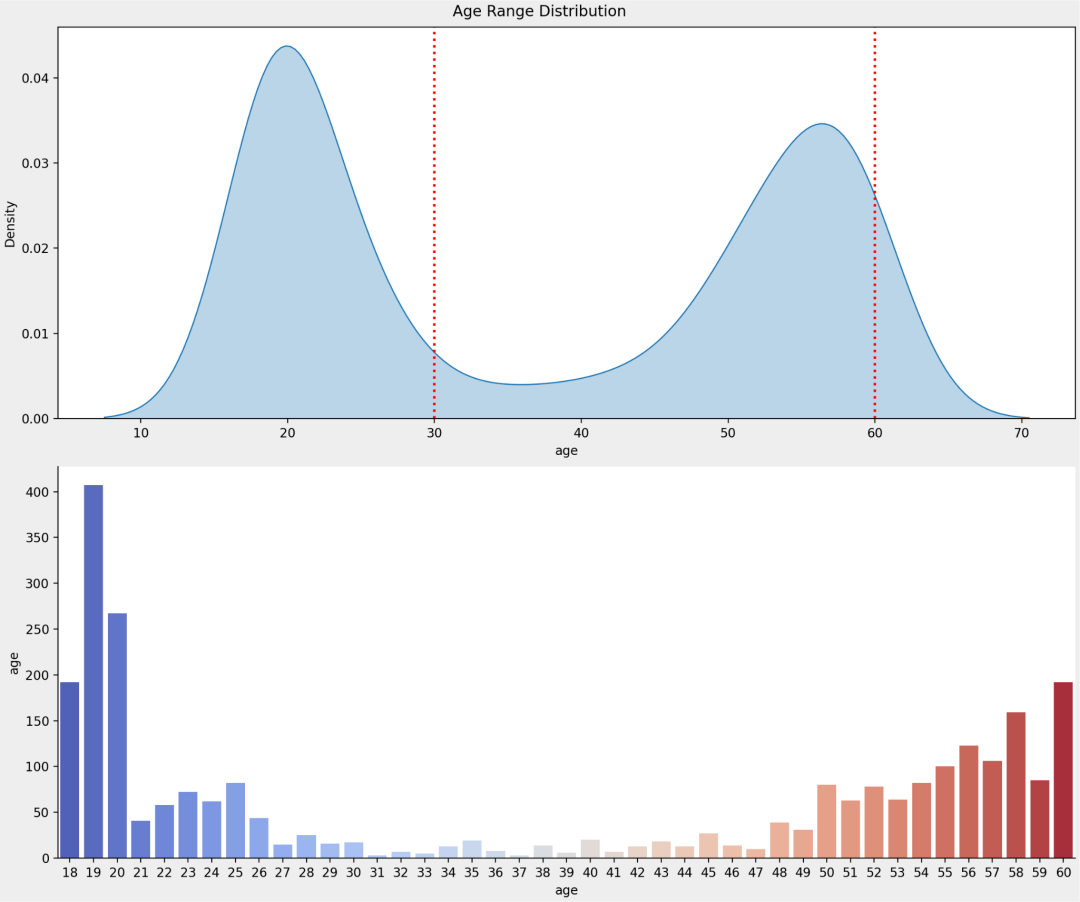
● 나이 데이터 불균형 (카테고리 불균형)



30세 미만	30세 이상 60세 미만	60세 이상
1281 명	1227 명	192 명

- 예측 해야 하는 Age Band (30세 미만, 30세 이상 & 60세 미만, 60세 이상)
- DataSet에서 60세 이상의 데이터가 다른 데이터에 비해 부족합니다.

● 나이 데이터 불균형 (카테고리 불균형)



- 전체 나이로 보면 30세와 60세의 Data가 상대적으로 부족합니다.

2) Image Raw Data Analysis

● 마스크 착용 유무 불균형

```
ID_gender_race_age
|-- incorrect_mask
|-- mask1
|-- mask2
|-- mask3
|-- mask4
|-- mask5
`-- normal
```

- 사람마다 총 7장의 사진이 존재합니다. (마스크 정상 착용 5장, 미착용 1장, 이상하게 착용 1장).

- 마스크 정상 착용의 경우 5 장이나 존재하는데 비해서 미착용과 이상하게 착용한 경우는 각각 1 장으로 상대적으로 부족해보였습니다.
- 따라서 Incorrect_mask, Normal 데이터의 증강이 필요해보였습니다.

● 노이즈 데이터

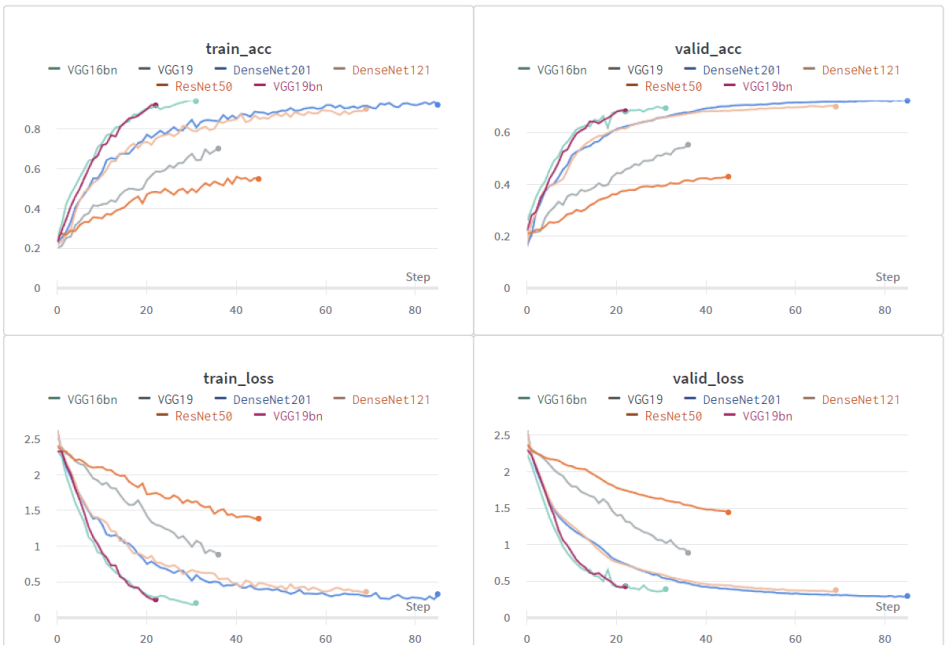
[Mislabeling]
- 남자처럼 보이지만 여자인 경우, 여자처럼 보이지만 남자인 경우
- Incorrect와 Normal 사진이 서로 반대로 Labeling 되어 있는 경우
- 제 나이처럼 안보이는 경우

[Segmentation]
- 객체의 얼굴은 대체적으로 정면 중앙에 위치하는 것으로 확인되었습니다.
- 하지만 인물의 배경에 다른 사람의 얼굴이 포함되어 있는 경우가 있습니다.
- 꽃 무늬 등 마스크에 프린팅 되어 있는 디자인과 흡사한 특징을 배경으로 가지는 데이터의 경우 모델이 마스크 착용으로 예측을 혼동할 가능성이 높았습니다.

1. VGG19, ViT, ResNet, DenseNet, MobileNet 등 기본모델 실험

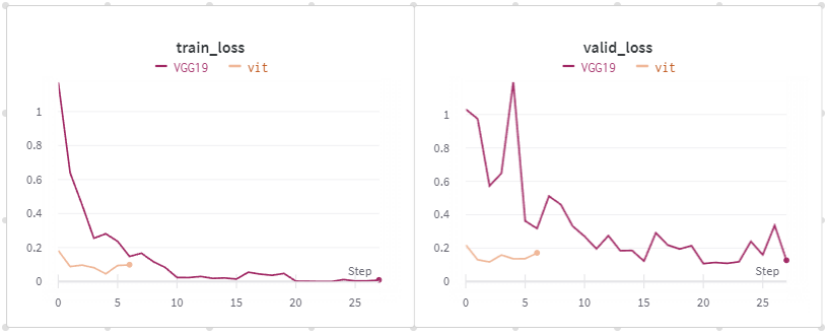
1) 여러 base model들을 비교

- VGG19_bn 모델이 DensNet201, ResNet50 등의 모델보다 더 빠르게 학습시키는 것을 확인할 수 있었습니다.



2) VGG19_bn, vit_base_patch16_224 비교

- ViT가 VGG19에 비해 학습속도도 빠르고, 정확도 또한 높게 나오는 것을 확인할 수 있었습니다.



	F1 score	Accuracy
VGG19_bn	0.4792	58.39%
ViT_base_patch16_224	0.6048	69.90%

3) Pretrained-ViT Model Freezing

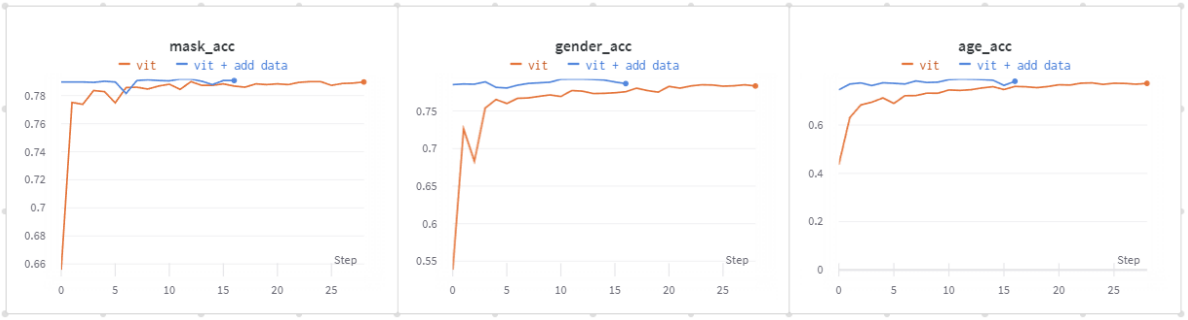
- ViT classifier만 학습시킨 ViT freeze model 과 전체 parameter를 얼리지 않고 학습한 ViT not freeze model을 비교했을 때 얼리지 않고 모든 parameter를 사용했을 때 더 좋은 결과를 보였습니다

	F1 score	Accuracy
ViT not freeze	0.6599	73.28%
ViT freeze	0.6289	69.92%

2. Data 증강 (incorrect, normal), k-fold(VGG19)

1) 데이터 증강

- 증강된 데이터셋은 기본 데이터셋 3번을 사용하였습니다.
- data 증강을 통해 mask, gender, age 의 정확도가 모두 상승한 것을 관측 할 수 있었습니다.



	F1 score	Accuracy
ViT	0.6048	69.90%
ViT + Data	0.6259	71.56%

2) k-fold

- 마찬가지로 data 증강을 했을 때, 성능이 향상된 것을 관측할 수 있습니다.

* SKF = Stratified K-Fold	F1 score	Accuracy
VGG19 + SKF	0.6039	69.59%
VGG19 + data + SKF	0.6529	74.17%

3. SwinTransformer, EfficientNet + label smoothing

- 아래의 실험결과를 보면, EfficientNet의 성능이 SwinTransformer보다 미세하게 높았어서 마지막에 둘을 앙상블을 시도해보고자 해당 2가지 모델로 선정하여 실험을 나눠서 진행하였습니다.

* ls = label smoothing	F1 score	Accuracy
SwinTransformer + ls	0.6942	75.38%
EfficientNet + ls	0.6974	74.60%

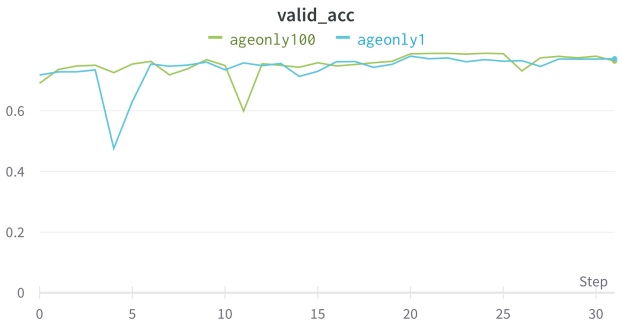
4-1. Multi-Labeling

- mask 착용 여부, age, 성별을 각기 다른 문제로 정의해보고자 multi label classification 문제로 치환
- 각 모델에 Multi-Labeling을 적용했을 때, SwinTransformer의 acc는 떨어졌고, EfficientNet의 acc는 올라갔음을 볼 수 있습니다.

* ls = label smoothing * ml = multi-labeling	F1 score	Accuracy
SwinTransformer + ls + ml	0.6980	74.94%
EfficientNet + ls + ml	0.7018	77.02%

4-2. 문제별로 모델을 세분화

- age를 맞추기위해서 출력층을 100개(0세 ~ 100세)로 갖는 모델을 생성했습니다.
- validation accuracy가 40%정도로 나왔습니다.



5. 60대 imbalance 문제 해결 -> 53 55 57age 범위실험

- age를 잘 예측을 못해서 age에 관련된 전처리를 추가해보자 했습니다.
- 60대는 60세만 존재하고 중후반 데이터가 없다는 것을 발견했습니다.
- age를 young, middle, old로 구분하는 base code에서 old를 구분하는 기준을 변경했습니다.
- 57로 수행하는 것이 가장 결과가 좋음을 발견하였습니다.

	F1 score	Accuracy

EfficientNet_b1(Age_57)	0.7311	77.22%
EfficientNet_b1(Age_55)	0.6423	71.07%
EfficientNet_b1(Age_53)	0.6104	69.66%

6. TTA

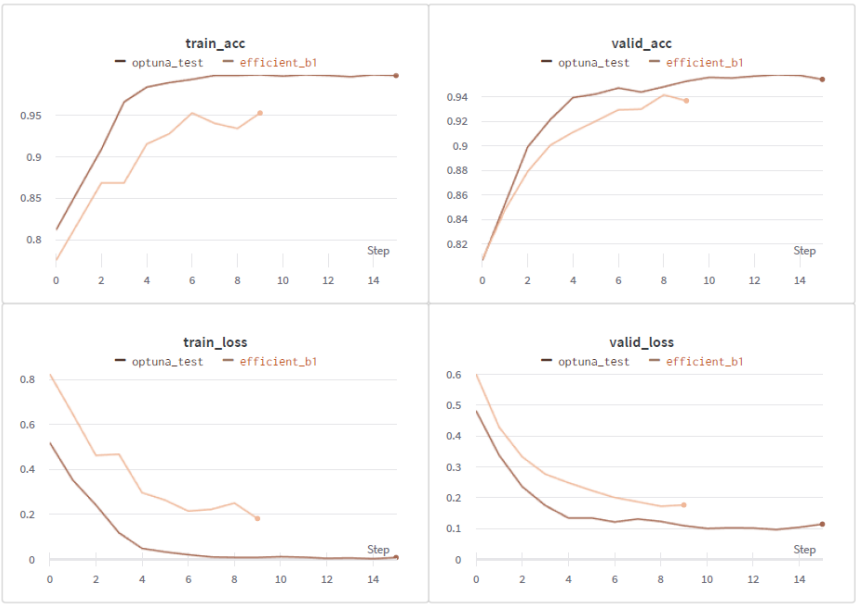
1) 적용한 4가지 Augmentation
   - original image
   - horizontal flip
   - +30 rotate
   - -30 rotate

2) EfficientNet 에서 TTA 적용 비교
   - 실험에는 EfficientNet_b1 model을 사용하였고, multi-label로 분리하여 학습을 진행했습니다.
   - TTA를 적용함으로써 accuracy가 미세하게 증가한 것을 볼 수 있습니다.

	F1 score	Accuracy
TTA 적용	0.7320	77.33%
TTA 미적용	0.7311	77.22%

7. Optuna, AMP 적용

- 성능 개선을 위해 하이퍼파라미터 튜닝을 적용해보고자 하였습니다.
- Optuna와 AMP를 적용해보았고, 그 결과 성능이 좋아짐을 확인했습니다.



- ● Mixed Precision Learning (AMP)

	Runtime	Accuracy
AMP 적용 (전)	**39m0s**	**83.40%**

AMP 적용 (후)	17m39s	83.98%

*(1 epoch 기준) AMP 적용 시 2 ~ 3배 RunTime 감소

[AMP]
- AutoCast, GradScaler
- FP16 연산 혹은 FP32 연산을 혼합하여 모델의 메모리 사용 감소 및 계산 속도 향상
- (384, 384) size를 input으로 받기 때문에 연산속도가 상대적으로 매우 느린 SwinTransformer_large_p4_384를 사용가능하게 함

8. 최종 모델 선정 및 분석

분류	내용		
아키텍처	SwinTransformer_Large_p4_384 + AMP		
LB점수		F1 Score	Accuracy
	Public	0.7600	80.14%
	Private	0.7509	79.61%
데이터	1인당 Normal, Incorrect_Mask 5배 증강 (40500 장) MaskSplitByProfileDataSet (Sampling) MisLabeling 등 이상치 제거 (Pre-Processing) Age 카테고리 기준점 변경		
Training Time Augmentation	**CustomAugmentation** ● CenterCrop(320, 256) ● Resize(384, 384) ● ColorJitter(0.1, 0.1, 0.1, 0.1) ● Normalize ∘ mean (0.47237855, 0.42983933, 0.40898423) ∘ std (0.24123258, 0.24687928, 0.49184509)		
Loss / Optimizer	F1 Loss / AdamW		
Tool Kit	wandb, AMP		
검증전략	팀내 초기 ViT 모델의 output.csv 파일의 F1 Score와 비교		
모델 평가 및 개선	데이터 불균형 문제를 해결하기위해 부족한 Category의 데이터를 증강했고, 노이즈 데이터를 처리해주었습니다. 프로젝트 초기에 준수한 성능을 보여줬던 ViT 모델을 팀내 베이스라인 모델로 선정하여 성능 개선의 지표로 설정했습니다. 하지만 ViT 모델은 Parameter 수가 무거워서 오버피팅이 현상을 발생시켰습니다. 따라서 보다 가벼운 모델인 SwinTransformer를 사용하게 되었습니다. 그리고 공간 정보의 손실이 적다는 특징이 있습니다. SwinTransformer는 Tiny(224) 버전과 Large(384) 버전이 존재했는데, Large 버전은 Input Image의 사이즈가 384로 많은 학습 시간을 요구합니다. 해당 문제는 AMP를 사용하여 학습시간을 2.5배 단축시켜 해결하였다. 높은 Validation Acc에 비해 상대적으로 낮은 Test Acc를 개선하기 위해 Raw Data의 배경을 제거하고, Optimizer로 AdamW 를		

	사용했다. 그리고 SKF 등 앙상블 기법을 적용하여 일반화 성능을 높일 수 있었다.

1-5. 자체 평가 의견

○ 잘한 점들

- baseline의 구조를 PyTorch template에 맞춰서 빠르게 구성하여, 코드 병합이 쉬웠습니다.
- 각자 하고 싶은 실험을 맡아서 수행하였습니다.
- Data 불균형을 해결해보고자 많은 시도들을 수행하였습니다.
- Wandb를 통해 실험 공유가 잘 되었습니다.

○ 시도 했으나 잘 되지 않았던 것

- Augmentation 시도 했으나 성능이 잘 오르지 않았습니다.
- 코드리뷰와 같은 Github 규칙이 잘 지켜지지 않았습니다.

○ 아쉬웠던 점들

- 초반 단계에서 각각 EDA한 것을 종합하는 과정이 없어서 아쉬웠습니다.
- 소통하는 방법을 잘 몰라, 소통이 완벽하게 되지 않은 점이 존재하여 아쉬웠습니다.
- Multi Label Class 에 어울리는 Loss 함수 개조를 해보지 못했습니다.
- 독특한 컬러 분포를 가지는 마스크 5 번 사진들의 유의미한 전처리를 진행해보지 못했습니다.

○ 프로젝트를 통해 배운 점 또는 시사점

- PyTorch template에 따라 ML project의 구조를 짜는 것을 배웠다.
- WandB와 Optuna를 통해 Hyper Parameters Tuning하는 법을 배웠다.
- 로그 분석을 통해 모델의 성능을 개선하는 법을 배웠다.

2. 개인 회고

김보경 (T5033)

○ **나는 내 학습목표를 달성하기 위해 무엇을 어떻게 했는가?**

저희 팀은 private test dataset에 대해서 f1 score와 accuracy 높이기라는 목표를 가지고 다양한 개선 시도를 해보는 것이었습니다. 저는 목표를 달성하기 위해 크게 아래 3가지를 시도해 봤습니다.

1. **wandb 및 optuna 연동**
   wandb 연동 코드를 추가하여 실험 공유에 도움이 되고자 하였습니다. 또 하이퍼 파라미터 튜닝을 위해 optuna를 연동하였습니다. 두 프로그램 덕분에 실험 공유 및 실시간 학습 상황을 볼 수 있었고, 업무의 효율성을 높일 수 있었습니다. 다만 두 프로그램 모두 보다 일찍 코드를 추가하였으면 하는 아쉬움이 남았습니다.

2. **baseline pretrained 모델 테스트 및 freeze 여부 실험**
   학습 데이터 셋이 적은 만큼 pretrained 모델을 사용하고자 했고, 여러 pretrained 모델을 돌려보면서 어떤 모델이 좋을지에 대해 분석해 보았습니다. 이 과정에서 vgg19와 vgg19_bn, vgg16과 vgg16_bn의 성능을 비교하면서 batch normalization의 성능을 보게 되었습니다.
   또 pretrained model을 freeze를 하는 것이 성능이 좋을까라는 궁금증이 생겨, 실험해 본 결과 freeze를 하지 않는 것이 성능이 더 좋음을 확인하였습니다. 실험 결과를 공유하여 freeze를 활용하지 않는 방향으로 추후 실험들을 진행하였습니다.

3. **60대 augmentation 수행 후 실험**
   초중반의 데이터가 모두 존재하는 10~50대 데이터 셋과 달리 60살의 데이터만 존재하는 60대 데이터 셋을 증폭시켜 성능을 향상해 보고자 하였습니다. 그 결과 성능이 좋아지지 않았습니다. 좋아지지 않은 이유에 대해 다음 2가지를 생각해 보았습니다.
   a. 데이터 양이 적은 것이 영향을 준만큼 중후반의 데이터가 없었던 것이 성능 개선에 제약을 준 것 같다.
   b. 60살의 데이터를 너무 많이 늘려서 오히려 다른 데이터의 비율을 줄이게 되었고 결과적으로 성능 저하를 일으킨 것 같다.
   위 실험을 통해 성능 개선을 이루지는 못했으나, 데이터의 양, 질 외에도 데이터의 분포와 같은 경향성도 중요하다고 생각하게 된 계기가 되었습니다.

○ **마주한 한계는 무엇이며, 아쉬웠던 점은 무엇인가?**

1. 서버 환경의 문제가 생겨 학습이 제대로 수행되지 않았습니다. 그 결과 대회 후반부에 테스트를 거의 못하여 아쉬움이 남습니다.
2. 성능 외 구체적인 목표를 세우지 않고, 하고 싶은 것을 다 해보자는 마인드로 대회에 참여하였습니다. 그렇기에 더욱더 코드, 실험, 의견 공유가 중요했었다고 생각했는데, 제대로 수행되지 않은 것 같습니다.
3. 적극적으로 성능 개선에 임하지 못했던 것 같습니다. 남이 짜놓은 코드를 사용하기만 했지, 제가 원하는 기능을 위한 코드를 직접 짜지 않는 등 적극적으로 실험에 임하지 못해 아쉬움이 남습니다.
4. 제공된 문서와 코드를 제대로 톺아보고 실험 해보기보단 무작정 실험 먼저 수행해야겠다는 맘이 앞서서 코드 분석을 제대로 수행 못했습니다.

○ **한계/교훈을 바탕으로 다음 프로젝트에서 시도해볼 것은 무엇일까?**

1. **강의를 적극적으로 듣기**
   대회를 접하기 전까진 강의를 듣고 개념에 대해서만 알고 넘어가곤 했었습니다. 이번 대회를 진행하면서 강의의 중요성을 다시 한번 느끼게 되었고 강의에서 배운 내용을 단순히 알고 넘기기보단 어떻게 활용해 볼 건지, 실전에서 어떻게 사용되는지 찾아보기 등 보다 적극적으로 강의를 활용해야겠다는 생각이 들었습니다.

2. **의견 공유를 보다 적극적으로 하기**
   이번 대회에서 각자 EDA를 수행하고, 원하는 개선 point를 정해 각자 개선하는 방향으로 진행하였습니다. 해당 방식으로 진행하고 난 후 생각해 보니 각자 제공된 코드 분석, EDA 수행 후 이해한 바를 공유한 시간이 있었어야겠다는 생각이 들었습니다. 또 깨달은 바를 노션에 적어놓고 공유를 안 하니 팀원들이 해당 트러블이 있다는 점을 인지 못하는 것을 보고 보다 적극적인 의견 공유가 필요하다 생각하였습니다.

3. **github 무서워하지 말고 적극적으로 활용하기**
   github으로 협업 프로그램을 하는 것이 처음이라 두려움이 앞서 제대로 활용을 하지 못했습니다. 그래서 팀원들에게 많이 의지를 했는데, 다음 프로젝트 전까지 스스로 github을 잘 다룰 수 있도록 하고, 적극적으로 활용할 수 있도록 할 것입니다.

4. **강의 외적으로도 기술들을 찾아보고 적극적으로 적용해보기**
   제공되는 강의도 좋지만 보다 다양한 경험을 하기 위해선 제가 직접 찾아보고 활용해야 한다는 것을 몸소 깨달았습니다. 개선 point와 개선 방향을 보다 구체적으로 세운 후 구글링, 논문 등을 활용해 다양한 개선 방법을 적용해 보는 등 적극적으로 모델 개선에 임해야겠다는 생각이 들었습니다.

박상필 (T5082)

[Introduction]

데이터 마이닝 절차는 **문제 정의 – EDA – 전처리 – 모델링 – 평가 – 피드백 사이클(반복)**의 순서로 진행됩니다. 이번 대회는 2주라는 짧은 시간으로 제한된 상황이었기 때문에, 멀리 갈수록 파급효과가 큰 Long-Term Cycle을 예방하기 위해, 주어진 데이터를 이해하는 **EDA 단계**에 집중하였습니다. 그리고 **데이터 전처리 단계**에서 여러 시도를 해보고 다양한 Intuition을 쌓기 위해 노력했습니다.

[Experience Summary]

EDA는 데이터가 가진 특징을 파악하고 어떻게 전처리할 것인지 이해하는 단계입니다. 데이터에 숨겨져있는 Feature Attribution를 파악하기 위해서 엑셀과 Python의 통계 및 시각화 툴을 활용하여 데이터의 분포를 분석하고, 원본 데이터를 시각적으로 파악하고 어떻게 생겼는지 이해하는 과정을 가졌습니다. 해당 과정을 통해서 다음과 같은 문제점과 해결 방안을 마련 하게 되었습니다.

문제점		세부 문제	설명	해결 방안
데이터 불균형	색상 분포	5번 마스크의 특이점	평범하지 않은 화려한 패턴의 마스크를 착용한 사진이 주로 마스크 5번, bias를 유발	색 보정 및 정규화 처리
	카테고리 클래스 분포	특정 나이대 부족	30세와 60세 등 특정 카테고리의 데이터 부족	Augmentation 증강, CutMix, 준지도 학습
		데이터 부족	균형 있는 데이터 증강, Cross Validation 필요	데이터 증강, 클래스 분포를 고려한 SKF 사용
노이즈 데이터	배경 노이즈	꽃무늬, 사람 두명	학습 시 예측에 혼동 유발 가능성 마스크 5번과 연관	크롭 및, 배경 삭제
	Mislabeling	Labeling 오표기	노이즈 데이터	수작업으로 수정
기타	리소스 부족	1 epoch 당 30분 이상 소요	복잡한 연산을 요구하는 학습을 실험해보기에는 시간이 부족	AMP 적용

[Conclusion]

평가 지표인 F1-Score를 개선하기 위해서는 데이터 불균형 문제를 해결하는 것이 우선순위일 것으로 생각되었습니다. 따라서 데이터의 샘플링 방식부터 전처리와 모델링까지 고려해보면 좋을 점들이 다양했고, 전반적으로 어떻게하면 일반화 성능을 높일 수 있을지 고민했던 것 같습니다.

그리고 평소에 저는 주피터 노트북만 사용해왔었는데 VSCode를 서버와 연동시켜서 다양한 실험을 편리하게 경험해볼 수 있었던 것 같습니다. 특히, 이번 프로젝트를 통해 Wandb 와 Tensorboard 같은 로그 & 디버깅 툴을 다뤄본 점이 좋았습니다. 또한, 인터넷에서 좋다라는 기법들이 해당모델에 적용해도 좋을지 판별하기가 어려웠던 것 같습니다. 때문에 성능의 지표로 삼으면 좋을만한 기준이 되는 모델이나 지표 같은 것들을 파악해두면 좋을 것 같다는 생각이 들었습니다. 그리고 시도해보면 좋을만한 전처리 기법들을 구상만 하고 응용은 못해본 점이 많이 아쉬웠던 것같습니다. 추가적으로 인공지능을 기반으로하는 실전 Task 의 경우 컴퓨팅 자원이 매우 중요하다는 것을 알게 되었고, 현업에서는 주어진 시간적, 물리적 자원을 효율성있게 활용하는 것에 대한 중요도를 높게 염두한다는 것을 알게 되었던 것 같습니다.

이태순 (T5165)

○ 나는 내 학습목표를 달성하기 위해 무엇을 어떻게 했는가?

1. Git, Github baseline 제시

Git, Github을 사용하기 위해 Repository를 만들고 컨벤션을 같이 정했습니다. 같은 Baseline 코드라고 해도 개인이 실험하는 과정에서 코드에 변형이 있을 것이고 5명의 코드가 다 달라질 것이었습니다. 따라서 나중에 같은 코드를 사용하고 싶을 때 통일된 코드가 없다면 이것을 합치기 위해 불필요하게 시간이 많이 들 것 같았고, Github을 먼저 정리함으로써 더 많은 실험을 하려고 했습니다.

2. Trouble shooting 공유

팀 notion 에 Trouble shooting 게시판을 따로 만들었습니다. 그리고 먼저 서버에서 Git을 이용하면서 겪었던 Error 해결 방법을 공유하였습니다.

○ 나는 어떤 방식으로 모델을 개선했는가?

처음에는 주로 VGG19와 ViT 2가지 모델을 사용하였습니다. 실험을 통해 VGG19 보다 ViT가 학습이 빠른 경향이 있었고 정확도 또한 좋게 나왔습니다. 또 pretrain된 ViT의 전체 parameter를 학습하는 것과 freeze한 parameter를 학습하는 것을 비교했었습니다.

그리고 single-class문제 예측을 mask, gender, age로 나누어서 training하는 multi-label로 문제를 치환하였습니다. 하지만 예측 성능은 생각보다 좋지 않게 나와 헤메고 있을 때 동료 캠퍼분들이 제시한 방법들을 조합하여 성능이 좋은 모델을 발견할 수 있었습니다.

EfficientNet, smooth loss와 57세까지 training시 60대에 포함시키며 실험했습니다. 또 어느 나이대까지 포함시켜야 성능이 좋을지 각각 53세, 55세, 57세로 나누어 실험을 진행하였습니다. 그리고 horizontal flip, rotate 이미지 등의 augmentation을 활용한 TTA로 미세하게 성능을 올릴 수 있었습니다. 이를 통해 팀제출 중 2등의 모델을 찾았습니다.

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

1. SSH 접속을 통해 서버를 끌어와 local에서 작업 – 더 편하게 작업 가능
2. Git, Github – 강의에서 배운 것을 활용할 수 있었고, Git, Github 공포증 극복
3. 프로젝트 구조 파악 – 직접 코드를 짜보고, 돌리고, error를 해결함으로써 학습, 추론의 전체적인 구조를 익힐 수 있었습니다.
4. 올바른 개념 파악 – stratified k-fold, TTA, EDA 등 잘못 알고 있었던 개념을 확실히 알게 되었습니다.

○ 마주한 한계는 무엇이며, 아쉬웠던 점은 무엇인가?

아쉬웠던 것은 stratified k-fold의 개념을 완전히 잘못 알아서 잘못된 코드를 짰다는 것입니다. 프로젝트가 끝나고 이 사실을 알게 되어 성능향상 기회를 잃어버려서 아쉬웠습니다. 협업에서의 아쉬운 점은 최대한 공유를 통해 능률을 올리려고 했는데, 신경 쓰지 못하는 부분이 있었다는 것입니다. 혼자 다른 dataset을 사용했던 것과 코드에 잘못된 부분이 있었던 것은 대회를 진행하며 아쉬운 점이었습니다..

○ 한계/교훈을 바탕으로 다음 프로젝트에서 시도해볼 것은 무엇일까?

이번에는 branch 전략을 시간이 없고, Git에 대해 익숙하지 않기 때문에 2개로 가져갔는데 다음에는 main, dev, feat 3가지 branch를 사용하여 협업해보고 싶습니다. 그리고 Git graph를 더 깔끔하게 가져가는 것을 해보고 싶습니다.

그리고 또 크게 얻어가는 것은 바로 Data 분석입니다. 저는 주로 model에 신경을 쓰며 초반에 진행했는데 팀원분들께서 data의 결함을 해결해오시면서 성능향상을 이루시는 것을 보고 data의 중요성을 아주 깊게 깨달았습니다. 따라서 프로젝트 시작 전 EDA를 제대로 해야겠다고 느꼈습니다.

또 코드리뷰 문화를 확실히 정해서 가져가야할 것 같습니다. 이번에는 리뷰를 조금 소홀히 한 느낌이 있는데 서로의 코드를 리뷰함으로써 프로젝트의 이해도도 향상할 수 있고, 실수가 있다면 잡아줄 수 있기 때문입니다.

임현명 (T5179)

1. 나는 내 학습목표 달성을 위해 무엇을 어떻게 했는가?(학습목표)

모델 개선 방식을 생각해 보고 프로젝트 중간에 겪게되는 작은 오류, 궁금한 점 등을 확인하려고 했습니다.

2. 나는 어떤 방식으로 모델을 개선했는가?(모델 개선 방식)

어떤 문제(마스크, 성별, 나이 등등)가 잘 안 풀리는가 확인해 보기 위해서 3개의 모델을 만들었습니다.(마스크 착용을 확인하는 모델, 성별을 맞추는 모델, 나이를 맞추는 모델)
잘 안 풀리는 문제가 있으면 그 문제에 초점을 맞춰서 모델을 개선하려고 하였습니다. 3개의 모델을 만든 것도 일종의 모델 개선이라고 생각하였습니다.
나이를 맞추는 모델을 만들때는 출력층을 100개로 만들어(100개로 만들면 58세이상을 장년층으로 조정할 수 도있습니다.) 학습을 진행하고 validation_data로 검증할때는 출력을 3개의 클래스로 나누어서 검증했습니다.
마스크 문제도 2개의 문제로 나누어서 생각해보려하였지만 다양한 데이터가 있기때문에 1개의 문제로 생각하였습니다.

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

나이 예측 모델에서 아주 작은 성능의 향상이 있었습니다.(1%) 효과가 크지 않구나 느꼈습니다. 실험이 잘못된것은 아닌지도 생각하게되었습니다.

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

처음으로 협업을 진행하였습니다. 다른 사람들은 어떻게 문제에 접근하는지 알 수 있었습니다.

5. 마주한 한계는 무엇이며, 아쉬웠던 점은 무엇인가?

1. 여러가지 기법, 패키지들을 알게된것은 좋았지만 그걸로 실험을 해보지는 못해서 아쉬웠습니다.
2. 몇가지 가설을 가지고 개선을 시도하였습니다. 가설이 개인적으로 충분히 타당하다고 생각하였는데 실험 결과는 좋지않았습니다. 실험을 많이 해봤으면 좋겠다는 아쉬운 점도 있고 실험이 제대로 된것인지도 확인해볼 수 있었으면 좋았을거같습니다.
3. 구현하고 싶은 것이 있었는데 구현하는 법을 몰랐던 것
4. 장년층의 기준(60세 이상)을 조정하지 않고 실험한 것, 장년층의 기준을 조정하는 것이 데이터 불균형 문제를 해결한 것인지 확인해 보지 못한 것

한계/교훈을 바탕으로 다음 프로젝트에서 시도해볼 것은 무엇인가?

1. 이번에 알게된 기법들을 사용할것입니다.
2. 가설을 조금 더 여러번의 실험으로 확인할것입니다.

천지은 (T5214)

우리 팀의 공통 목표는 Mask Image Classification의 모델 성능을 개선하기 위해 다양한 시도를 해보는 것이었습니다. 저는 목표를 달성하기 위해 앞서 EDA에서 데이터 불균형 개선으로 데이터 증폭과 Stratified k-fold 방법 2가지를 활용했고, mask 5번 이미지에는 특이한 마스크의 존재를 확인하고 해당 마스크 패턴이 배경과 비슷하다면 영향을 받을 것이라 생각하여 사람 주위의 배경을 제거하였습니다. 그리고, 해당 데이터에서 가장 높은 성능을 내는 pretrained 모델을 찾아보았으며, 데이터에서 얼굴이 모두 중앙에 위치하므로 이를 반씩 나누어서 학습하는 CutMix를 시도해보았습니다.

1. 데이터증폭은 다양한 패턴을 학습할 수 있도록 데이터셋 크기를 늘려주는 기법입니다. 주어진 데이터셋은 한 사람당 mask 이미지는 5개이지만, incorrect_mask, normal 이미지가 1개씩만 존재하는 문제가 있습니다. 따라서 incorrect_mask, normal 이미지에 Augmentation 기법을 적용하여 데이터 크기를 늘려주었고, 그 결과로 일반화 성능을 급격하게 향상시키는 시점이 되었습니다. 이를 통해 데이터셋의 크기가 학습에 매우 중요한 역할을 한다는 것을 몸소 깨달았습니다.

2. Stratified k-fold는 클래스 불균형 문제가 있는 데이터셋을 학습할 때, 클래스 비율을 고려해서 데이터를 나누어 학습하는 방법입니다. 주어진 데이터셋은 나이, 성별, 마스크 등 클래스 불균형 문제가 크게 존재합니다. 따라서 일반 학습보다 Stratified k-fold를 통해 학습하여 성능 향상을 기대해보았고, 직접 실험하고 비교해본 결과, 예상대로 일반 학습보다 성능이 조금 더 향상된 모습을 볼 수 있었습니다.

3. 이미지 배경 제거는 사람 주위의 배경을 제거하여 불필요한 정보를 학습하는 것을 방지하고, 사람 위주로 집중해서 학습할 수 있도록 하여 성능 향상을 기대했던 방법입니다. 배경 제거한 데이터셋에서 주어진 데이터셋보다 모델의 성능이 향상된 것을 보았고, 이를 증폭시켜 학습시킨다면 더 좋은 모델을 만들 수 있을거라 생각했습니다. 하지만, rembg 라이브러리를 사용해서 배경을 제거할 때 일부 Augmentation이 적용되지 않는 현상이 있었습니다. 그래서 rotate, horizontalflip 함수만 적용되어 오히려 데이터 증폭만 되었을 때 보다 성능이 떨어졌습니다. 배경 제거하는 시간에만 8시간이 소요되어, 짧은 기간으로 차마 시도하지는 못했지만 이미 데이터가 증폭된 상태에서 배경 제거를 하면 어땠을까 하는 아쉬움이 남았습니다.

4. pretrained 모델 찾기는 paper with code를 참고하여여 image classification 대회에서 우승했던 모델들을 찾아보았습니다. 이때 EfficientNet과 SwinTransformer 모델이 label smoothing loss와 함께 쓸 때 가장 높은 성능을 보였습니다.

5. CutMix는 사람의 얼굴이 모두 중앙에 있는 것을 보고, 이를 세로로 절반씩 자르고 붙인 새로운 이미지를 학습시킨다면 일반화 성능이 오를지 않을까 예상하여 시도하였습니다. 하지만, 구현한 CutMix는 Multi-labeling과 적용했을 때, GPU 부족 에러가 떠서 제대로 실험하지 못했고 성능 향상에도 크게 기여하지못해 아쉽습니다.

짧은 기간의 프로젝트로 더욱 다양한 시도를 해보지 못했지만, 이론으로 배웠던 여러가지 방법을 대회를 통해 직접 코드를 구현해보고 적용해보며 지식을 많이 쌓아갈 수 있었습니다. 또한, 각 팀원마다 새로운 시각이나 관점에서 문제를 바라봄으로써 다양한 아이디어가 많이 도출되었습니다. 하지만, 소통이 원활하게 이어지지 않아 구현 상황 및 코드 리뷰가 제대로 되지 않은 점이 아쉽습니다. 이번 프로젝트를 통해 소통의 중요성을 깨닫고 다음 프로젝트에서 소통을 우선시하여 더 나은 결과물을 만들어 내고싶습니다.