Toward the Reduction of Class Imbalances

Level1. Team 9th 구해조. Mask Classification Wrap-up Report

김민윤(T6017), 배종욱(T6071), 신호준(T6091), 전병관(T6152), 최수진(T6174)

2023.12.13 ~ 2023.12.21

Abstract

COVID-19 팬데믹 상황에서 공공 보건 조치의 필요성을 감안하여 카메라로 촬영된 얼굴 이미지를 통해 마스크 착용 상태, 나이, 성별을 분류하는 모델을 개발하였다. 모델 개발에 앞서 EDA(Exploratory Data Analysis) 및 시각화로 시작하였고, 기존 베이스라인 코드에 있던 문제점을 해결한 코드를 제작하였다. 이후, 데이터 내의 레이블의 오류와 나이 클래스 간 불균형을 발견하고 이를 해결하기 위해 레이블의 부정확성을 해결하고 대표성이 부족한 나이 범주에 대한 추가 데이터를 생성하여 학습 데이터의 불균형 완화에 초점을 맞추었다. 또한 모델의 분류 집중도를 높이기 위해 배경 제거 및 얼굴 탐지 기술을 적용하고, MTL(Multi Task Learning) 및 모델 앙상블 전략을 통해 모델의 정확도와 견고함을 향상시켰다. 레이블, 클래스 별 F1-score와 정확도 지표를 사용한 성능 평가에서는 다양한 방법을 통해 F1-score가 향상됨을 확인할 수 있었다.

1. Problem Definition

1.1 Task

COVID-19 팬데믹은 전 세계에 큰 영향을 미쳤으며, 특히 공공 보건과 안전에 중대한 도전을 제시했다. 이 바이러스의 전파는 주로 호흡기 비말을 통해 이루어지기 때문에, 마스크 착용은 전파를 방지하는 데 중요한 역할을 한다. 이 연구의 주요 목표는 카메라로 촬영된 얼굴 이미지를 분석하여 마스크 착용 상태, 나이, 성별을 자동으로 분류하는 인공지능 시스템을 개발하는 것이다. 이 시스템은 공공장소에서의 마스크 착용을 모니터링하고, COVID-19의 확산을 예방하는 데 중요한 역할을 할 수 있다.

1.2 Mask, Age, Gender Classification

입력받은 데이터로부터 크게 3가지의 분류를 정확히 하는 것을 목표로 한다. 우선 마스크를 정확히 착용했는지 (Mask), 마스크를 착용하지 않았는지 (Normal), 그리고 마스크를 올바르지 않게 착용했는지 (Incorrect_Mask) 판별한다. 그리고 해당 이미지가 남성인지, 여성인지 구분한다. 또한 해당 이미지에 있는 사람의 나이가 0살에서 30살 사이인지 (Young), 30살에서 60살 사이인지 (Middle), 60살 이상인지 (Old) 판단하는 것을 목표로 한다. 따라서 한 장의 이미지가 주어졌을 때 나이, 성별, 마스크 착용 여부를 모두 판단하여 총 18가지의 클래스에 대한 분류를 시행한다.

1.3 Dataset

해당 분류 문제를 해결하기 위해 주어진 데이터셋은 이미지를 포함하는 폴더와 train.csv 파일이 제공된다. train.csv에는 각 테스트 이미지에 대한 성별, 나이 그리고 이미지가 저장된 경로 정보를 다음과 같이 포함한다.

```
id, gender, race, age, path
000001,female, Asian, 45, 00001_female_Asian_45
000002, female, Asian, 52, 000002_female_Asian_52
000004, male, Asian, 54, 000004_male_Asian_54
```

< train.csv file>

각 이미지들을 포함하는 폴더에는 아래와 같은 이미지들을 포함한다. 마스크를 쓰지 않은 normal, 5종류의 서로 다른 마스크를 쓴 이미지들인 mask1,2,3,4,5, 그리고 마스크를 올바르게 착용하지 않은 incorrect_mask 이미지들로 총 7개의 이미지를 포함한다.

Incorrect Mask Mask1 Mask2 Mask3 Mask4 Mask5 Normal

< Profile Data Example>

1.4 Metric

본 프로젝트에서는 모델의 성능 평가를 위해 F1-score와 Top-1 accuracy를 사용하였다. Top-1 accuracy는 모델이 가장 높은 확률로 예측한 클래스가 실제 클래스와 일치하는 비율을 나타낸다. F1-score는 아래 수식으로 계산되며, Precision과 Recall에 대한 설명은 다음과 같다. Precision은 모델이 참(True)으로 분류한 데이터 중에서 실제로 참인 데이터의 비율을 나타내며, Recall은 실제로 참인 데이터 중에서 모델이 참으로 판단한 데이터의 비율을 나타낸다. 일반적으로 F1-score는 모든 클래스의 평균 성능으로 평가한다.

$$F1\ Score = 2 \times \frac{recall \times precision}{recall + precision}$$

< F1-score Expression >

하지만, 학습된 모델의 레이블 별 난이도, 데이터 불균형 또는 중요도를 분석하기 위해 각 레이블(나이, 성별, 마스크 착용 여부) 별 평균 F1-score, 각 레이블(나이, 성별, 마스크 착용 여부)의 클래스 별 F1-score 그리고 마스크 착용 여부에 따른 나이별 F1-score를 계산하였다.

2. Foundation Work

2.1 Baseline Code

주최 측에서 제공되었던 베이스라인 코드를 활용하여 기존에 제작했던 Resnet 모델 학습을 해본 결과 아래와 같은 문제점을 확인하였다.

첫 번째로는 기존 베이스라인 코드에서의 학습을 진행하는 train.py에서 실험에 필요한 대부분의 코드를 변경해야 했다. 이때 코드의 복잡성 증가하여 코드의 가독성, 유지 보수, 재사용성 감소 그리고 실험 관리의 어려움 등이 발생했다. 이를 해결하기 위해 각각의 컴포넌트(model, loss, optimizer, metric, transforms, datasets, utils)를 별도의 파일로 분리하고 각각의 컴포넌트를 train.py에서 “get_컴포넌트이름”라는 함수를 만들어 설정 부분에서 이름을 입력하면 해당하는 컴포넌트 함수를 불러올 수 있도록 구현하였다.

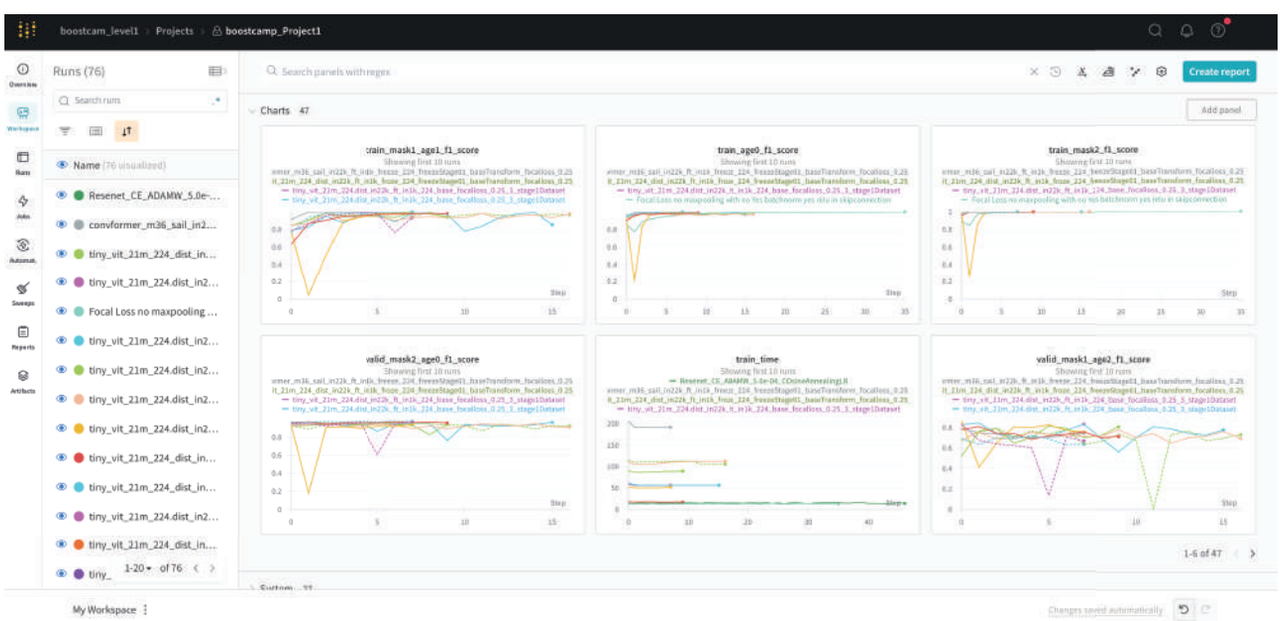
두 번째로 인자 파싱 방법으로 하이퍼 파라미터를 설정하는 방식이었다. 인자 파싱 방법은 실행을 할 때 직관적인 장점을 갖고 있지만 본 프로젝트는 모델, 손실 함수, Learning rate, 학습 Epoch 수 그리고

W&B 설정 등 40개가 넘는 하이퍼 파라미터를 설정 및 수정하고 기록 및 공유하는데 많은 어려움을 겪었다. 그리하여 Config(Yaml) 파일방식의 설정 파일 기반 방식으로 수정하였다. 즉, 실험 시 필요한 모든 하이퍼 파라미터를 Yaml 파일을 변경하는 것을 통해 실험이 가능하도록 설계하였다.

세 번째로는 모델 weight 관리가 어려웠다. 모델이 같은 폴더에 저장되거나 해당 모델이 어떠한 실험인지 알 수 없었다. 그리고 서버의 데이터 저장 용량이 100GB밖에 되지 않아 모든 epoch마다 weight가 저장 시 용량에 문제가 일어났다. 그리하여 실험마다 실험 시리얼을 생성하고 results 폴더 아래 시리얼 이름을 갖는 폴더에 Config 파일을 저장하고 원하는 Metric이 최고점이 나올 때마다 해당 epoch의 weight를 저장하도록 구현하였다. 또한, Early Stopping을 구현하여 최대한 중요한 weight를 저장되도록 설계했다.

2.2 W&B

모델 성능과 문제점 및 문제점을 개선되었는지 확인하기 위해 세분화된 Metric이 필요하다고 판단했다. 그리하여 Train, Valid 마다의 레이블별, 클래스 그리고 마스크 착용 여부에 따른 나이 별 F1-score, Loss, Accuracy 등을 계산하였다. 이를 분석하고 기록하기 위해 W&B에 저장했다. 많은 실험이 예상되어 Runs 이름 컨벤션을 지정하여 실험을 잘 구분할 수 있도록 하였다.



< W & B Log Page >

2.3 Activation Map

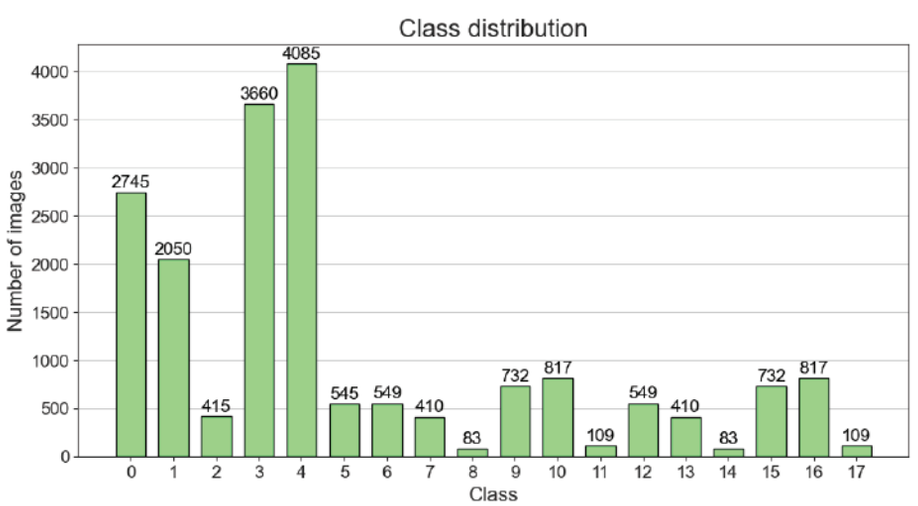
이미지를 분석한 결과 핵심 부분인 얼굴 이외의 Noise가 많이 존재했다. 그리하여 모델이 해당 사진에서 집중하고 있는 부분이 얼굴에 집중하고 있는지 그리고 얼굴 내에 옳은 부분을 집중하고 있는지 확인할 필요가 있다고 판단했다. Torch-cam을 사용하였고 SmootGradCAMpp를 활용하여 준비한 Valid의 이미지를 모델마다 Activation Map을 생성하여 모델이 잘 집중하고 있는지 확인했다.

< Not Background Remove Activation Map >

3. EDA(Exploratory Data Analysis)

3.1 Visualization of Dataset

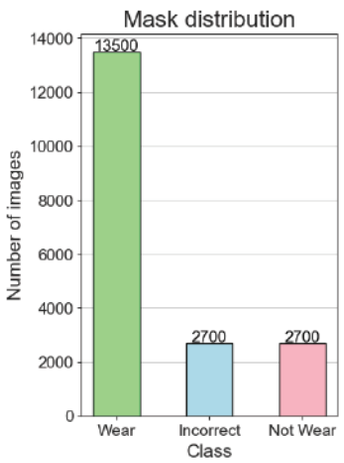
이 프로젝트에서는 각 클래스별 학습 데이터의 분포를 정확히 이해하기 위해 전체 데이터 구성을 시각화하여 분석하였다. 이를 통해, 학습 데이터의 클래스별 분포에 불균형이 크게 나타나는 것을 확인하였다.



< Number of Data per Class >

3.2 Distribution of Mask Class

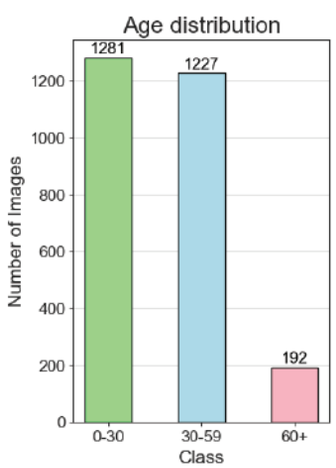
다음은 주어진 데이터셋을 마스크 착용 상태에 따라 나누어 분석하였다. 한 사람 개체에 대해 마스크를 정상적으로 착용한 상태(Mask)에 해당하는 이미지 5장, 마스크를 비정상적으로 착용한 상태(Incorrect)에 해당하는 이미지 1장, 그리고 마스크를 착용하지 않은 상태(Normal)에 해당하는 이미지 1장이 포함되어 있다. 이에 따라, 마스크 착용 상태 별 데이터의 비율은 각각 약 72%, 14%, 14%를 차지한다.

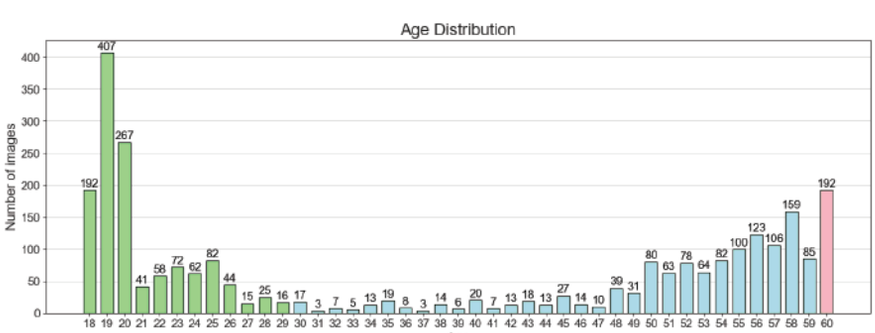


< Mask Class Distribution >

3.3 Distribution of Age Class

주어진 데이터셋을 나이(Age)에 따라 분류하여 분석하였다. 연령대는 30세 미만을 'Young', 30세 이상 60세 미만을 'Middle', 그리고 60세 이상을 'Old'로 각각 구분하였다. 각 연령대별 데이터 분포를 조사한 결과, 'Old' 구간의 데이터가 가장 적었음을 확인하였다. 각 연령대별 데이터 분포를 분석하였을 때, 30세와 40대의 데이터는 상대적으로 부족하였고, 70세 이상의 데이터는 존재하지 않았다. 이러한 결과는 연령대별로 데이터 분포에 불균형이 존재함을 보여준다.

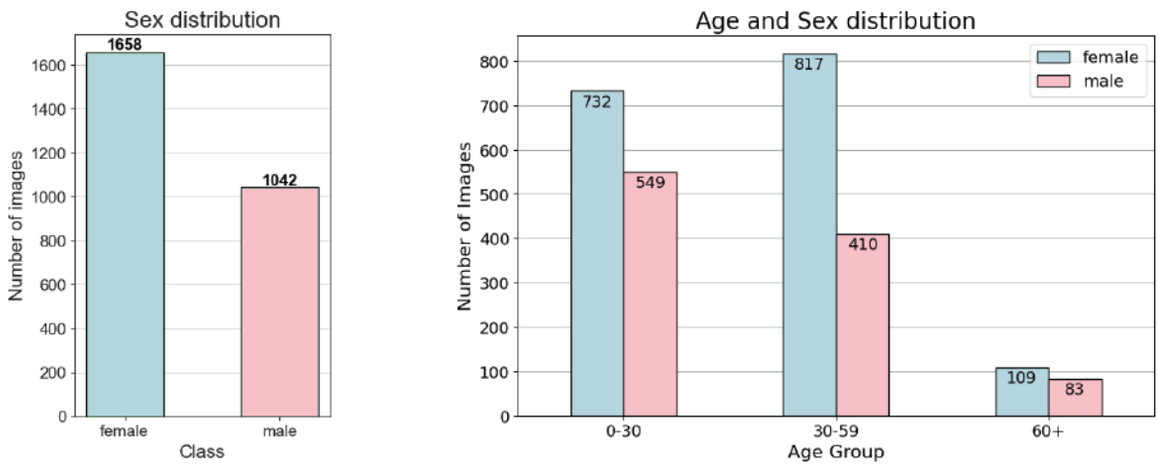




< Age Class Distribution >

3.4 Distribution of Gender(Male, Female) Class

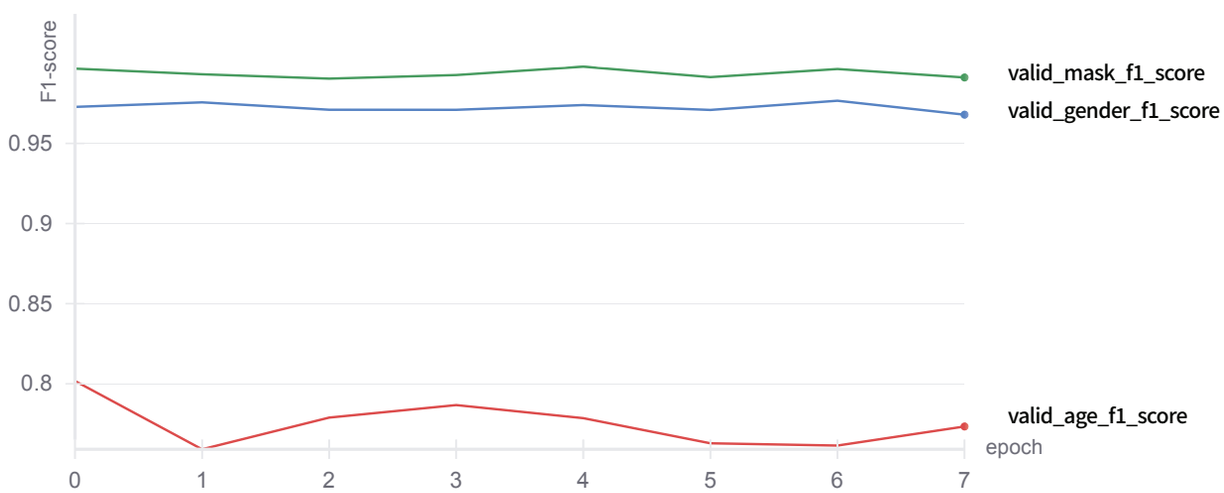
주어진 데이터셋을 성별(Gender)에 따라 분류하여 분석하였다. 여성(Female) 클래스의 데이터가 남성(Male) 클래스의 데이터보다 많은 것을 확인하였으며, 이는 나이(Age) 클래스 별로 분포를 확인하였을 때에도 동일하게 나타났다. 이러한 결과는 데이터셋 내에 성별 클래스별 불균형이 크게 나타나고 있음을 보여준다.



< Sex Class Distribution >

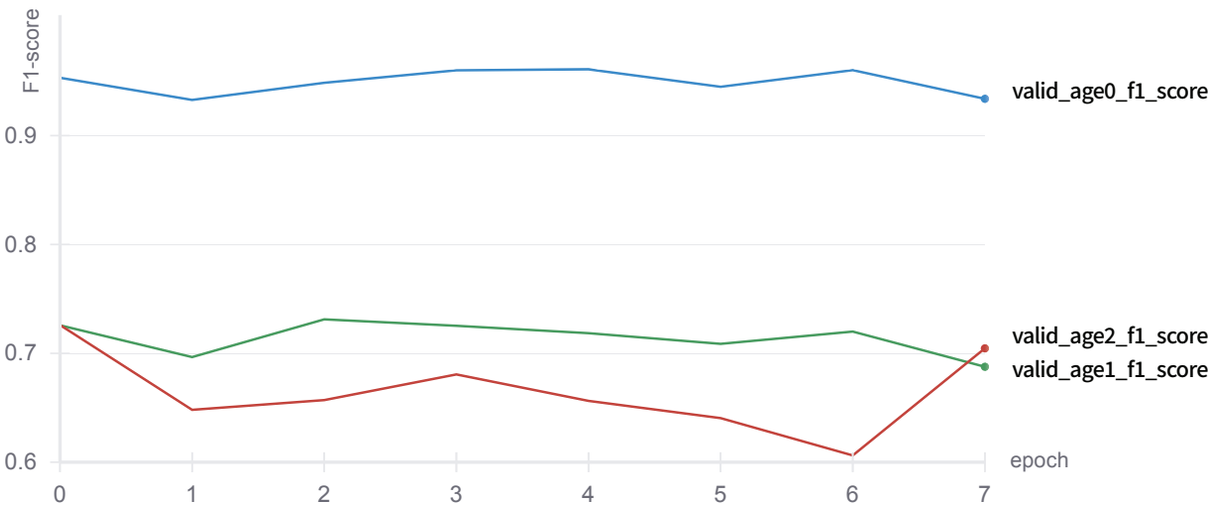
3.5 Analyzing Base Model Results

EDA를 통해 데이터 불균형을 발견한 다음, 기본 모델을 학습하여 모델 결과를 분석하였고 이때 W&B를 통해 확인한 결과 나이, 성별, 마스크 착용 여부 중 나이에 문제점이 있음을 확인하였다.



< Valid Class F1-score >

문제점이 있는 나이에 해당하는 각 클래스를 분석한 결과 나이 중 Middle과 Old에 문제점이 있음을 확인했다. Middle과 Old 간에 데이터 불균형이 존재했고, Middle과 Old 사이의 경계값인 50대 후반에 Noise가 많이 존재했다. 그리하여 Data Cleansing과 Data Remove를 진행하였다.



< Valid Age Class F1-score >

4. Data Cleansing

4.1 Overview

제공된 데이터셋의 시각화 과정에서 잘못된 레이블을 다수 발견하였다. 특히, 주어진 데이터셋이 일반적인 이미지 분류 학습에 사용되는 데이터양에 비해 1/10에서 1/100 정도로 현저하게 적은 양임에도 불구하고 잘못된 레이블 데이터가 존재하였다. 따라서 모델 학습에 노이즈로 작용하여 성능에 좋지 않은 영향을 미칠 것으로 판단하여 이를 다시 정답 레이블로 수정 하거나 데이터를 제거하는 과정을 거쳤다.

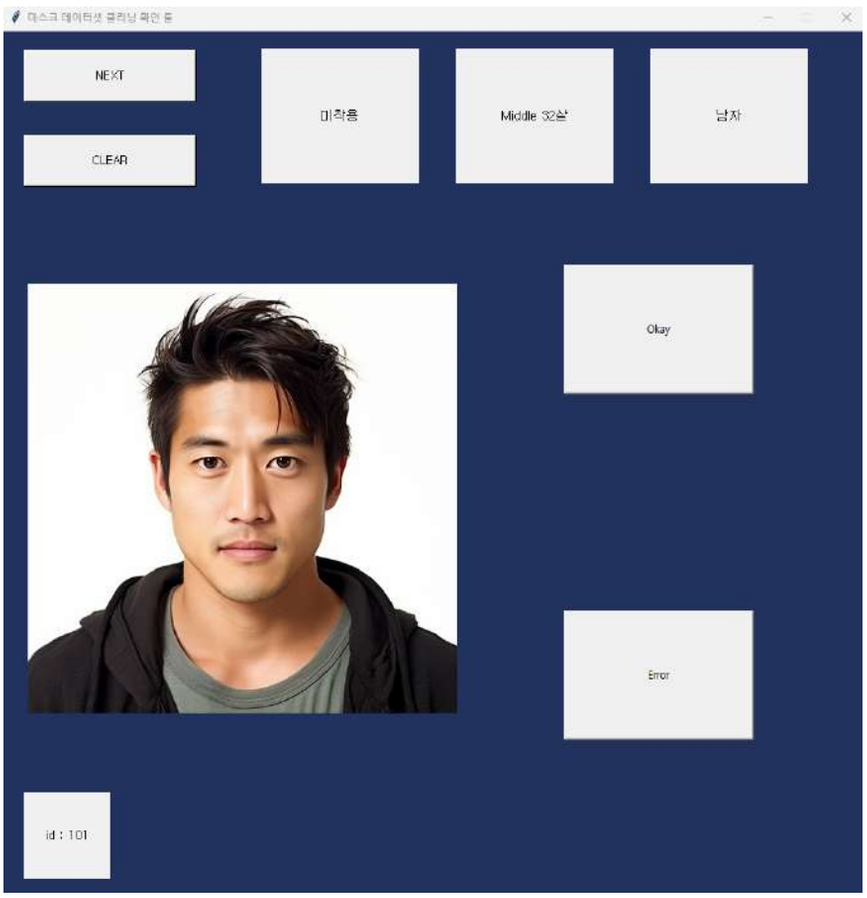
4.2 Methods and Techniques

주어진 학습 데이터는 2700개의 하위 폴더가 포함되어 있으며 각 하위 폴더에는 7개의 이미지가 포함되어 총 18900장의 파일이 존재한다. 이를 팀원 수대로 분배하여 전수조사를 실시하였다.

4.2.1 Tool for Data Inspection

폴더에 Label이 표기되어 있고 7장의 이미지들이 각 폴더 내에 위치한다. 기존에는 개별 폴더를 확인하여 파일명과 이미지를 매칭하였을 때 100장 당 50분이 소요되었다. 이에 문제점을 인지하여 효율

적으로 검수할 수 있는 Data Checking 툴을 다음과 같이 개발하였다.



< Data Error Checking Tool >

검수를 시작할 이미지의 ID를 입력하면 해당 ID에 대한 이미지 및 레이블이 시각화되며 차례로 레이블과 이미지 간 일치 여부를 확인한 후, 일치하지 않을 경우 해당 이미지의 레이블을 csv 형태로 저장한다. 개개인이 수작업으로 각 폴더를 열어 파일을 확인하는 번거로운 과정을 Data Checking 툴을 개발함으로써 데이터 검수 시간이 100장 당 50분에서 10분으로 줄었다.

4.2.2 Tool for Relabeling

4.2.1에서 제안한 Data Checking 툴을 통해 레이블과 이미지가 일치하지 않는 이미지들을 다수 확인한 후 팀원들과 회의를 통해 잘못된 레이블의 기준을 설정하고 이를 어떤 레이블로 수정할 것인지를 의사 결정하였다. 이를 바탕으로 기존의 레이블에서 성별, 마스크 여부, 나이를 수정하였다. 특히 나이의 경우 시각화 결과 정확한 나이를 설정하기 어려워 0-30, 30-60, 60+로 분류해야 하는 클래스 단위로 레이블을 설정한 다음 해당 나이대 기준으로 임의의 값을 부여하는 방식으로 레이블을 수정하였다.

< Data Labeling Tool >

4.2.3 Data Remove

앞선 의사 결정을 통해 기존 데이터셋의 레이블을 수정하였으나 분류 모델을 통한 성능 평가 시 Age F1-score가 낮게 나오는 현상이 발생하였다. 이는 50대 후반의 이미지 데이터들이 60세들과 비교 시 뚜렷하게 구분되지 않았고, 50대 후반과 60대 이상의 데이터 양을 비교했을 때 50대 후반의 데이터양이 2배 가까이 되어 학습에 방해가 될 것 같다는 판단을 했다. 이를 해결하고자 50대 후반에 해당 데이터들을 제거하는 과정을 거쳤다.

4.3 Results and Impact

4.3.1 Experimental Result after Relabeling

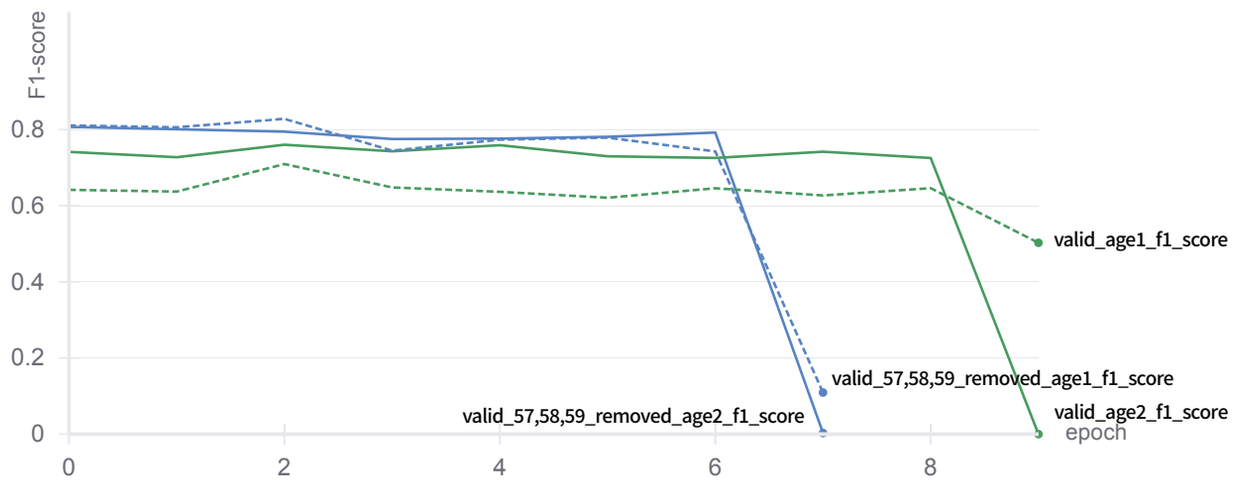
Model	F1-score (val)
ViT (tiny)	0.7193
ViT (tiny) + Data cleansing	0.7543

< Original data vs Cleaned data comparison Table >

데이터 클렌징 후의 성능을 살펴보면, 데이터 클렌징 전에 비해 F1-score가 향상된 것을 확인할 수 있다. 표에서 볼 수 있듯이, 데이터 정제 전의 F1-score는 0.7193이지만, 데이터 정제 후에는 F1-score가 0.7543으로 증가하였다. 이 결과는 데이터 클렌징 과정이 중요한 전처리 과정임을 보여준다.

4.3.2 Experimental Result after Data Remove

앞선 4.2.3에서 언급한 바와 같이 문제가 되는 50대 후반의 데이터, 즉 나이 태스크의 레이블인 57, 58, 59를 제거한 뒤 전후의 성능 비교를 위한 실험을 진행하였다. 데이터를 제거한 결과, 나이 태스크의 검증(validation) F1-score가 크게 향상되었음을 아래 그래프를 통해 알 수 있다.



< Base vs Age Removed Valid Middle Old Class F1-score >

검증 데이터셋 뿐만 아니라 테스트 데이터셋에서도 뚜렷한 성능 향상을 보였다.

Model	F1-score (test)
ViT (tiny)	0.7193
ViT (tiny) + Label remove	0.7356

< Original data vs Label removed data comparison Table >

5. Validation Dataset

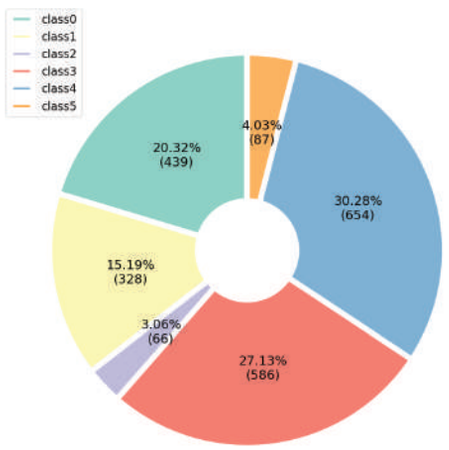
5.1 Overview

베이스라인 코드로 실험한 결과를 통해 테스트 데이터셋과 검증 데이터셋의 성능 차이가 큼을 확인하였다. 이는 전체 이미지 기준으로 random split 되기 때문에 학습 데이터셋에 있는 사람이 검증 데이

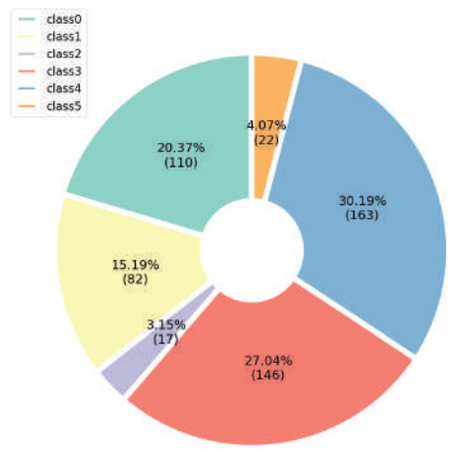
터셋에도 포함될 소지가 있어 과적합을 유발할 수 있다고 판단하였다.

5.2 Methods and Techniques

데이터를 profile 기준으로 split 하되 sklearn의 stratify를 사용하여 학습과 검증의 각 클래스 비율이 비슷하게 random split 되도록 함수를 구현하여 추가하였다.

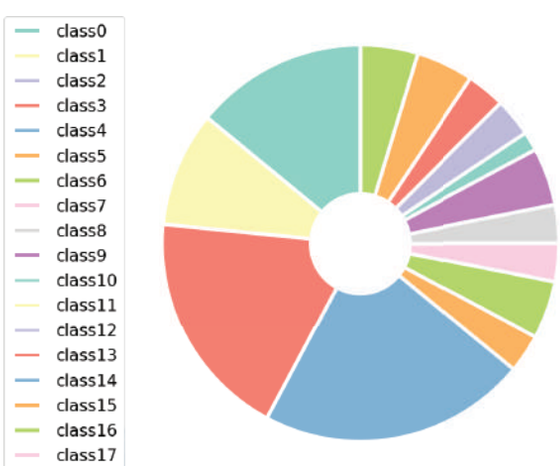


< Train dataset : 2160 >

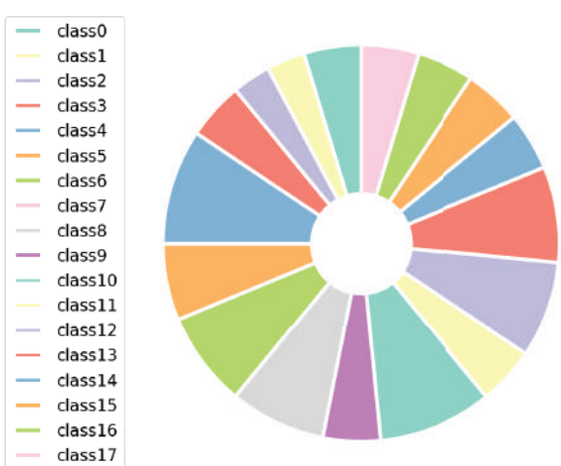


< Validation dataset : 540 >

또한, 데이터 불균형으로 인해 학습 중 batch에 들어가는 클래스의 개수가 균등하게 들어가지 않는 문제를 해결하기 위해 각 클래스 별로 가중치를 다르게 부여하여 batch 내의 클래스의 개수가 비슷하게 유지될 수 있도록 WeightedRandomSampler를 사용하였다.



< Without Applying WeightedRandomSampler >



< Applying WeightedRandomSampler >

5.3 Results and Impact

Validation dataset	F1 (val)	Acc (val)	F1 (test)	Acc (test)
Random sampling	0.92	96.82	0.69	74.93
Custom validation dataset	0.65	70.78	0.64	72.98

< Random Sampling vs Custom Validation dataset comparison Table >

이 표는 검증 데이터셋과 테스트 데이터셋에서의 성능 차이를 감소하기 위한 실험 결과를 나타낸다. 실험은 random 샘플링으로 구축된 데이터셋과 Stratify와 WeightedRandomSampler를 적용시킨 custom 검증 데이터셋으로 진행하였다.

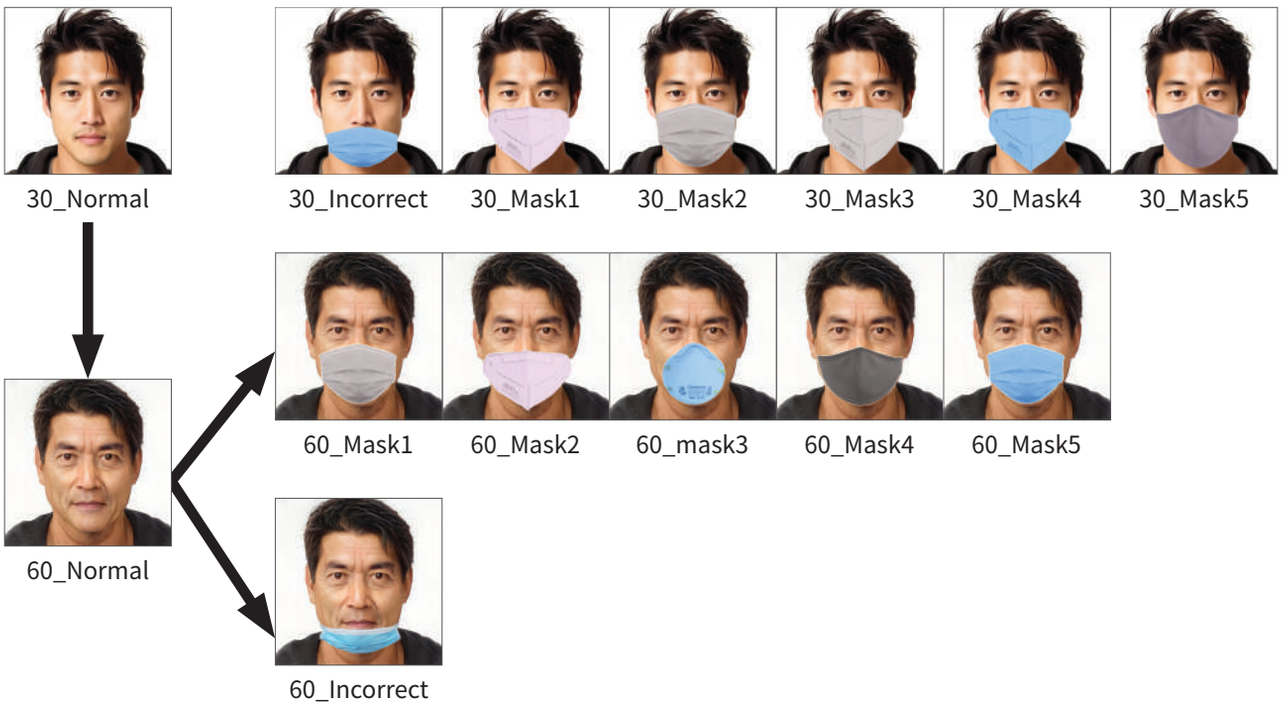
random 샘플링은 이미지를 무작위로 선택하기 때문에 학습 과정에서 사용한 데이터 중 일부가 검증 과정에서도 사용될 수 있다. 그 결과, 검증 성능은 높게 나타나지만, 테스트 데이터셋에서의 성능은 검증 성능에 비해 떨어진다. 이는 랜덤 샘플링의 경우 검증 데이터셋과 테스트 데이터셋 사이에 공통적으로 포함된 데이터가 있어서 발생하는 문제일 수 있다.

반면에, custom 검증 데이터셋의 경우, 각 클래스의 샘플 비율을 조정하여 불균형 데이터셋에서의 학습 효율을 개선한다. 또한, 학습 데이터셋과 검증 데이터셋이 서로 독립적이며 실험을 통해 검증 성능과 테스트 성능 사이에 큰 차이가 없음을 알 수 있다. 이는 학습과 검증, 테스트 과정에서 동일한 데이터가 사용되지 않아 과적합을 방지하고, 모델의 일반화 성능을 더 잘 평가할 수 있다.

6. Data Generation

6.1 Overview

3. EDA를 통해 주어진 데이터셋의 경우 나이 클래스 데이터의 불균형 문제가 존재함을 파악하였다. 특히, Old 클래스(60+)가 다른 클래스 대비 상대적으로 데이터가 많이 부족하며 실험을 통해 해당 클래스의 F1-sccore가 낮게 나오는 것을 확인할 수 있었다. 이를 해결하고자 아래와 같이 Age transformation, masked, incorrected wearing mask 데이터를 생성하여 전체 데이터셋이 나이대 별로 균형을 이루게 재구성하였다.

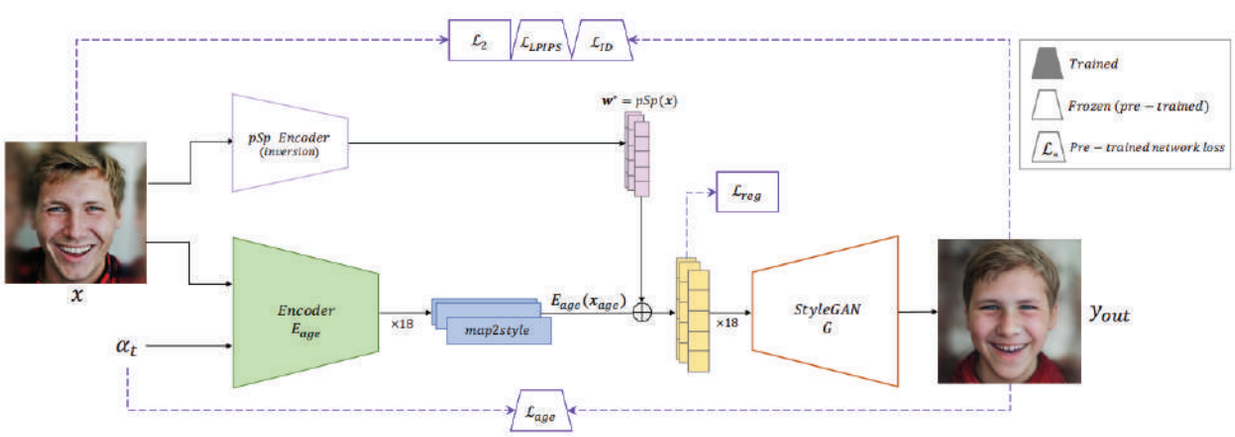


< Convert face dataset to masked dataset >

6.2 Age transformation

6.2.1 Generation Technique for Age Transformation

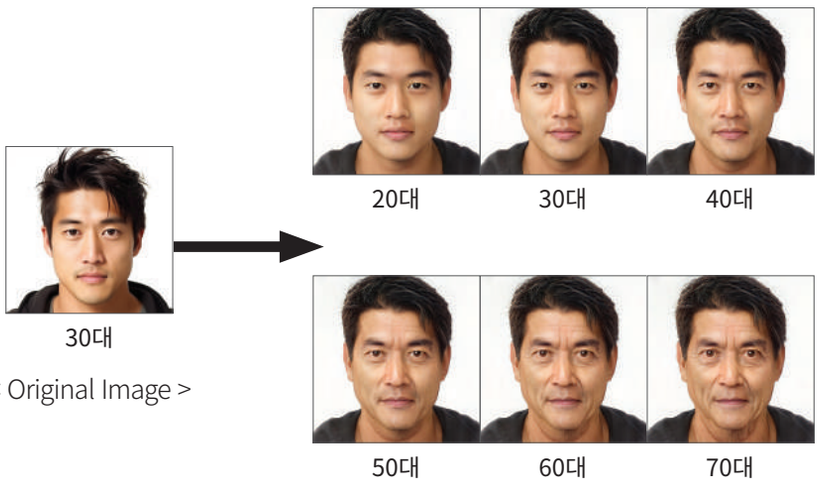
Only a Matter of Style : Age Transformation Using a Style-Based Regression Model (SIGGRAPH 2021)에서 제안한 방법을 통해 데이터셋에 포함된 이미지들 중 마스크를 쓰고 있지 않은 이미지(normal)를 이용하여 다양한 연령 이미지 생성 후 이를 마스크 합성을 통해 데이터 생성하였다.



< Only a Matter of Style >

6.2.2 Results of age transformation Data

주어진 학습용 데이터셋을 4. Data Cleansing을 통해 데이터 정제 과정을 거친 뒤 해당 이미지를 이용하여 다양한 나이대의 이미지를 다음과 같이 생성하였다.



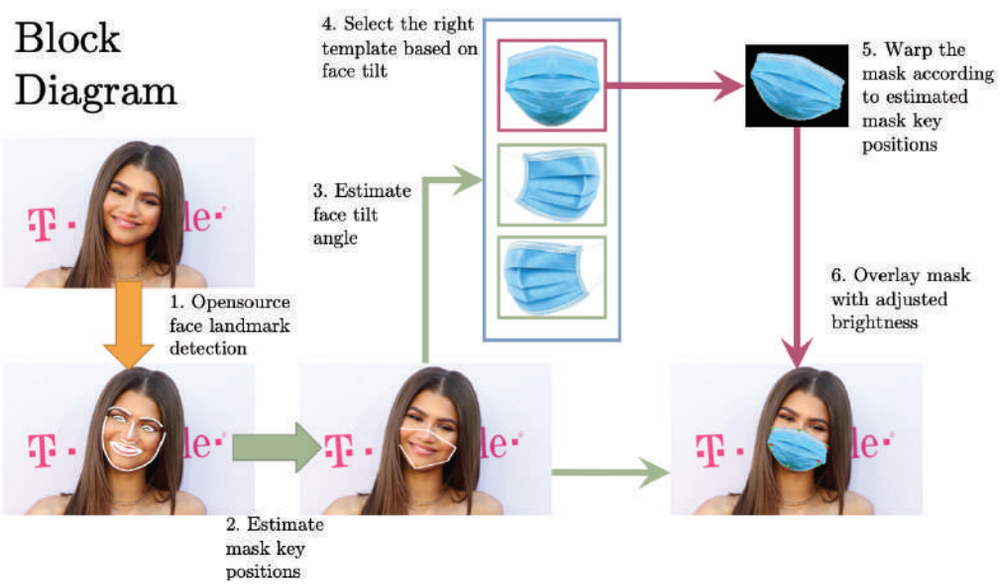
< Original Image >

< Generated Images >

6.3 Masked Image Generation

6.3.1 Generation Technique for masked image generation

마스크 합성 데이터를 생성하기 위해 아래와 같이 MaskTheFace에서 제안된 기법을 적용하였다.



< Block Diagram >

마스크 합성 데이터 생성을 위해 4.3 Evaluation of Age Transformation Data에서 생성한 normal 이미지에 얼굴의 특징(landmark)을 추출한 다음 마스크 키포인트를 검출한다. 그다음 EDA를 통해 제공된 데이터에서 주로 사용되는 마스크와 유사한 4 종류의 마스크를 다음과 같이 선정하였다.

< Selected Masks>

6.3.2 Results of Masked Image Generation

Age transformation을 통해 생성된 normal 이미지에 12 종류의 마스크를 합성한 결과는 다음과 같다.

< Synthesized Masks on Age-Transformed Image >

해당 모델의 경우 입력한 이미지에서 얼굴 특징을 추출하지 못할 경우 합성을 진행하지 않고 다음 이미지에 대한 합성을 진행하도록 코드를 수정하였다. 그 결과 총 입력 이미지들의 1% 내외에 대해 합성이 이루어지지 않았으며 이외의 이미지에는 모두 정상적인 마스크 합성 데이터를 얻을 수 있었다.

6.4 Incorrected Mask Image Generation

6.3에 제안된 마스크 합성 모델의 경우 마스크를 제대로 착용한 경우에 대해서만 생성이 가능한 한계가 있다. 따라서 마스크 오착용 이미지 생성을 위해 Rapid Object Detection using a Boosted

Cascade of Simple Features에서 제안된 방식을 이용하여 얼굴의 좌표값을 구하고 얼굴 비율을 하이퍼 파라미터 값으로 인물 간 구분 없이 일괄 적용하여 합성하였다. 마스크 합성에 쓰인 마스크 이미지는 제공받거나 생성한 이미지 파일에서 직접 추출하였다. 제공된 학습용 데이터의 오착용 이미지들 중 턱에 착용한 마스크 이미지 일부와 4.3에서 제안된 과정에서 생산된 마스크 이미지 중 Surgial, KN95, Cloth 총 3종류의 마스크를 사용하였다.

6.4.1 Results of incorrected mask image generation

마스크 incorrect 이미지 생성을 수행한 결과는 다음과 같다.

< Synthesized Incorrect Masks on Age-Transformed Image >

이처럼 기존 데이터셋에 포함된 다양한 incorrect 케이스들을 최대한 반영할 수 있는 합성 데이터를 생성하였다.

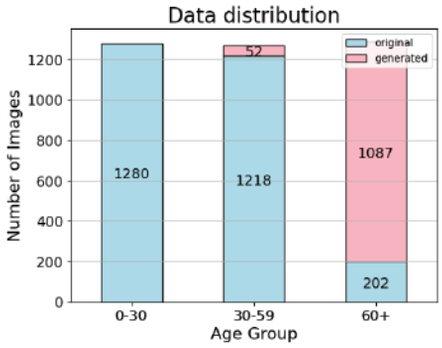
6.5 Generated Dataset Grouping

앞서 3. EDA에서 분석한 바와 같이 Age 클래스 불균형 해소를 위해 생성한 Aged image, masked image, incorrected mask image들을 기존의 데이터셋의 구성에 맞춰서 폴더별로 7장의 사진을 무작위 추출하여 구성한다.

6.5.1 Stage-1

Age 클래스 중 가장 데이터가 많은 Young 클래스의 개수인 1281장을 기준으로 Middle, Old 클래스의 데이터를 생성된 데이터셋에서 추출한다. 기존 Middle 클래스의 1227장의 이미지에 생성된 데이터에서 54장을 추출하고 Old 클래스의 경우 기존의 192장의 이미지에 1089장의 생성된 이미지를 추가하여 다음과 같이 나이 클래스 간의 데이터 불균형을 해소했다.

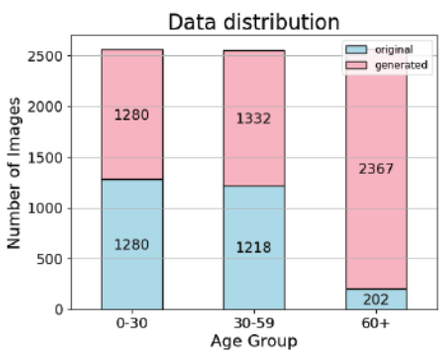
추가 데이터는 Age transformation 및 12가지의 다양한 마스크에서 무작위로 비복원 추출하여 사용하였다.



< Stage-1 Data distribution >

6.5.2 Stage-2

Stage-1에서 Young 클래스 개수로 나머지 클래스의 데이터를 생성 데이터에서 추출하여 합쳤다면 Stage-2에서는 age 클래스 중 가장 데이터가 많은 Young 클래스의 2배를 기준으로 Middle, Old 클래스의 데이터를 생성된 데이터셋에서 추출하였다.



< Stage-2 Data distribution >

6.6 Evaluation of Generated Dataset

Model	F1-score (val)
ViT (tiny)	0.7130
ViT (tiny) + generated data 1	0.8520
ViT (tiny) + generated data 2	0.8364

< Original data vs Generated data comparison Table >

이 표는 생성된 데이터를 추가하여 학습한 모델의 성능을 비교한다. ‘generated data 1’은 클래스 비율에 따라 불균형이 없도록 생성한 데이터셋을, ‘generated data 2’는 클래스 비율을 고려하여 불균형

이 없도록 생성하고, 원래의 데이터 양보다 더 많이 생성한 데이터셋을 의미한다.

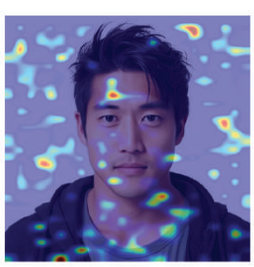
ViT (tiny) 모델만을 사용하여 학습한 경우의 F1-score는 0.7130으로 측정된다. 반면에 ‘generated data 1’을 추가로 활용하여 학습한 경우 F1-score는 0.8520으로 측정되고, ‘generated data 2’를 추가로 활용한 경우는 0.8364로 측정된다.

이 결과를 통해, 생성된 데이터를 추가로 활용하여 학습하는 것이 모델의 성능을 개선하는데 효과적임을 알 수 있다. 그러나 ‘generated data 2’의 경우처럼 너무 많은 데이터를 생성하면 과적합의 위험이 있음을 확인할 수 있다. 따라서, 데이터를 생성할 때는 적절한 양을 고려하는 것이 중요하다.

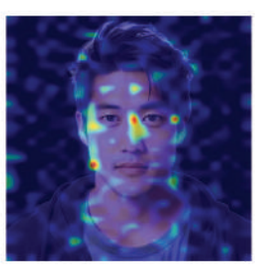
7. Background Removal & Face Detection

7.1 Importance of Background Removal & Face Detection

이미지 전처리에 앞서 Activation Map을 통해 모델이 얼굴보다는 옷이나 배경에 더 많이 집중하는 것을 확인하였다. 이는 얼굴을 학습하여 클래스를 분류하는 데 있어 noise로 작용할 가능성이 커 모델이 얼굴에 더욱 잘 집중할 수 있도록 이미지에서 배경을 제거하였다. 그 결과 모델이 배경을 지우기 전보다 얼굴에 더 집중하긴 했지만 여전히 옷과 배경에도 집중이 분산되는 것이 확인되어 Face Zoom in 까지 추가적으로 진행했다.



< Not Background remove Activation Map >

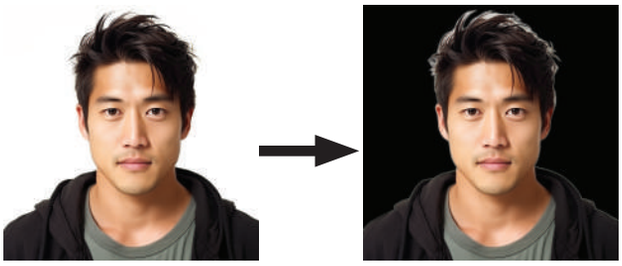


< Background remove Activation Map >

< Background remove Face Zoom in Activation Map >

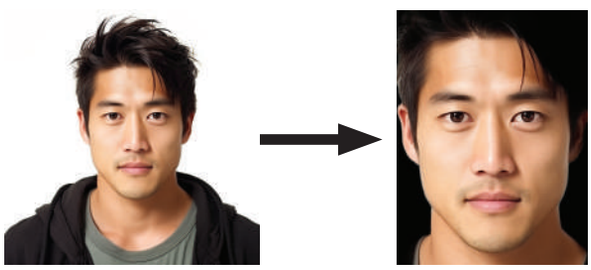
7.2 Applied Methods

Background Removal를 위해 처음에는 기본적인 segmentation 방법을 고려했다. 하지만 복잡한 배경 조건에서도 높은 정확도와 효율성을 기대할 수 있는 solution이 필요하다고 판단하여 딥러닝 기술을 활용하여 U2-Net 기반의 Rembg 모델을 사용해 Background Removal을 진행했다.



< Base Image > < Background remove Image >

Face Zoom in을 하기 위하여 초반에는 torchvision의 CenterCrop을 이용하려고 했으나 얼굴의 크기와 위치가 일정하지 않아 일괄적으로 Crop을 진행하면 얼굴 정보에 손실이 있을 수 있다고 판단했다. 따라서 Face detection 라이브러리를 활용하고자 하였고, 속도, 성능 부분에서 모두 준수하다고 알려진 MTCNN을 사용해 얼굴 검출을 한 후 Crop을 진행했다. 얼굴이 검출되지 않는 경우는 CenterCrop으로 처리했다.



< Base Image > < Background remove
Face Zoom in Image >

최종적인 데이터셋을 만들 때는 Rembg의 처리 속도를 고려해 MTCNN으로 얼굴을 검출하여 Crop 된 이미지에서 Background Removal을 적용했고, 모든 profile 데이터에 대해 전처리를 진행했다.

Incorrect Mask Mask1 Mask2 Mask3 Mask4 Mask5 Normal

< Background Remove & Face Zoom In Images >

7.3 Performance Analysis

Model	F1-score (val)
ViT (tiny)	0.7130
ViT (tiny) + detected data	0.7833
ViT (tiny) + remove bg	0.6962
ViT (tiny) + remove bg + detected data	0.7502

< Background Remove & Face Zoom In Images Experiment Table >

위 표는 face zoom in만 처리한 detected dataset, background remove만 처리한 remove bg dataset, face zoom in과 background remove 모두 처리한 dataset으로 성능을 F1-score로 분석한 결과를 나타낸다. 예상했던 것과는 다르게 background remove만 처리한 dataset이 F1-score 0.7833으로 가장 성능이 높았다.

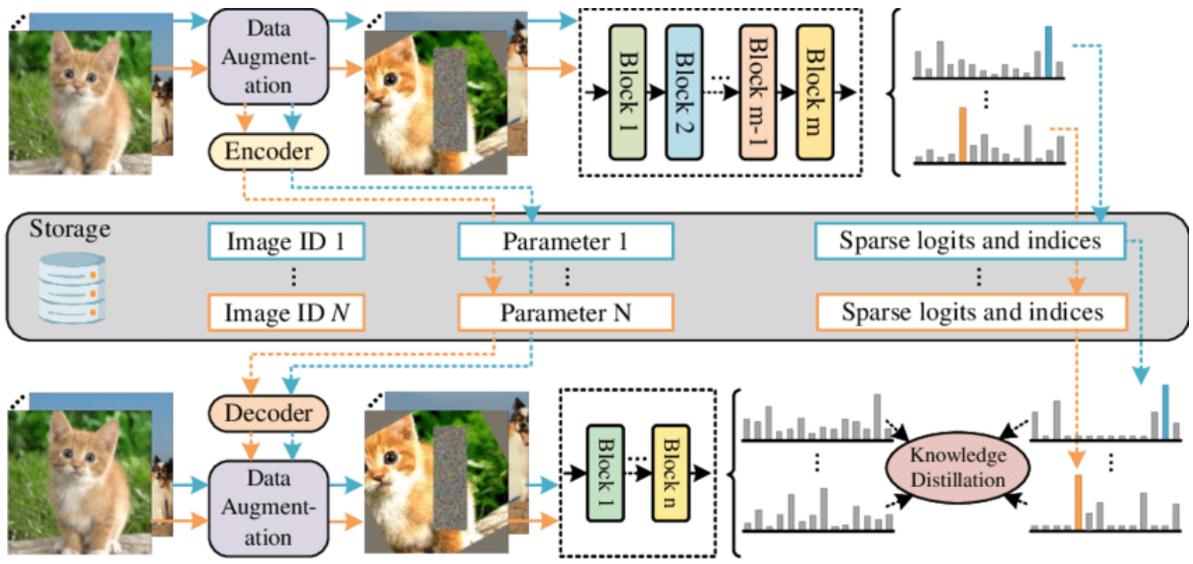
8. Backbone Models

Model	F1-score (val)	F1-score (test)	Parameters (M)
ResNet34	0.6754	0.5742	21.8
EfficientNet(B5)	0.7029	0.6928	30.0
Caformer (B36)	0.7230	0.6693	98.7
Vision Transformer (Tiny)	0.7130	0.7193	21.2

< Backbone Experiment Table >

위 표는 주어진 데이터셋에 대한 다양한 모델의 성능을 F1-score로 분석한 결과를 나타낸다. CNN 기반의 ResNet34와 EfficientNet(B5), 그리고 Transformer 기반의 Caformer(B36)와 Tiny Vision Transformer(ViT)를 사용했다. 이 모델들은 모두 ImageNet 데이터셋에 대해 사전 학습된 모델이다.

ResNet34를 학습한 결과 검증 점수가 낮게 나왔다. 그리하여 EfficientNet(B5)를 학습하였고 검증 점수와 테스트 점수가 대폭 상승하였다. Sota 급 모델 중 하나인 Caformer 모델을 학습한 결과 검증 점수가 높게 나왔지만 테스트 점수는 검증 점수보다 소폭 하락하였다. 이러한 이유를 분석한 결과 모델의 파라미터가 크고 데이터셋이 작아 과적합이 의심되었고, 약 30M 이상의 파리미터를 가진 모델에서는 과적합이 될 확률이 높다 판단하여 30M 이하의 모델 중 적은 데이터셋에 적합한 모델을 찾아보았다. 그 중 파라미터가 적고 작은 데이터셋에 적합한 Knowledge Distillation 모델인 Tiny Vit가 주어진 태스크에 적합한 모델로 판단하였다.



< Tiny VIT Architecture>

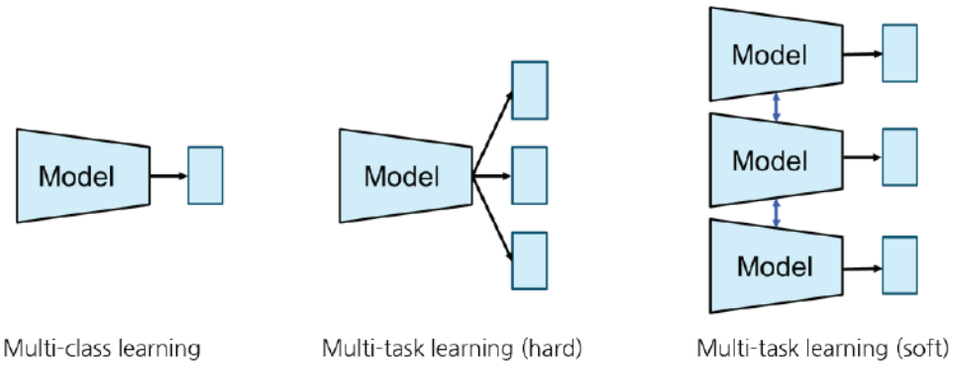
실험 결과, Tiny ViT가 가장 높은 테스트 F1-score인 0.7193을 보여주었다. 이 결과는 가장 큰 모델이 항상 최상의 성능을 내지는 않음을 보여준다. 따라서, 앞으로의 실험에서는 ImageNet 데이터셋에 대해 사전 학습된 Tiny ViT를 사용하여 실험을 진행하였다.

9. Multi-Task Learning (MTL)

9.1 MTL

앞에서 언급한 것처럼 이 프로젝트는 마스크 착용 여부, 성별, 나이를 판별하는 세 가지 태스크에 집중하며, 이들은 각각 3개, 2개, 3개의 클래스로 구성되어 총 18개의 클래스를 갖고 있다. 다양한 클래스로 이루어진 복잡한 태스크를 동시에 처리하는 것은 어렵기 때문에, 각 태스크를 별도로 처리하기 위해 멀티태스크 학습(Multi-task learning, MTL)을 도입하였다. 멀티태스크 학습은 여러 태스크를 동시에 학습하면서, 태스크 간의 연관성을 활용해 일반화 성능을 향상시키는 방법이다.

9.2 Implementation



< Multi-task learning Architecture >

위 그림은 멀티 클래스 학습과 멀티태스크 학습의 '하드 파라미터 공유'와 '소프트 파라미터 공유'를 비교한 것이다. 멀티태스크 학습을 구현하기 위해 두 가지 방법이 적용되었다. 하드 파라미터 공유는 단일 네트워크에서 여러 태스크를 동시에 학습하며, 태스크 별로 세 개의 분류기를 포함한다. 이 방법은 여러 태스크 간의 파라미터를 효과적으로 공유한다. 소프트 파라미터 공유는 각 태스크 별로 독립적인 네트워크를 구축한다. 이때 l2 norm을 적용해 각 네트워크의 파라미터 간 거리를 최소화하고, 파라미터를 공유하도록 한다. 이 방법은 각 네트워크가 독립적으로 작동하면서도 공통의 특성을 학습하게 하여, 다양한 태스크에 대한 학습을 가능하게 한다

9.3 Results

Learning type	F1 (test)	Acc (test)
Multi-class learning	0.72	78.80
Multi-task learning (Hard)	0.72	78.20
Multi-task learning (Soft)	0.72	79.47

< Multi-task learning Experiment Table >

위 표는 멀티 클래스 학습과 멀티태스크 학습의 성능을 비교한 결과를 보여준다. 멀티 클래스 학습은 마스크 착용 여부, 나이, 성별이라는 18개의 클래스를 하나의 태스크로 처리하는 방식을 의미한다. 멀티태스크 학습은 하드 파라미터 공유와 소프트 파라미터 공유 두 가지 방식을 사용하였다.

표의 결과를 살펴보면, 멀티 클래스 학습과 하드 파라미터 공유 방식의 멀티태스크 학습은 F1-score에서 동일한 성능을 보이나, 정확도 면에서는 멀티 클래스 학습이 약간 더 높은 값을 나타냈다. 그러나, 소프트 파라미터 공유 방식의 멀티태스크 학습은 F1-score에서 동일한 성능을 유지하면서도 정확도 면

에서는 가장 높은 성능을 보였다. 이 결과는 소프트 파라미터 공유 방식의 멀티태스크 학습이 이 문제에 대해 가장 효과적인 방법임을 나타낸다.

10. Loss Function

손실 함수는 모델의 예측 결과와 실제 값 간의 차이를 측정하는 함수이다. 손실 함수를 사용하는 이유는 모델의 성능을 평가하고 학습하는 과정에서 모델의 가중치를 조정하기 위해서이다. 손실 함수는 모델이 얼마나 좋은 예측을 하는지 파악하는 지표로 사용된다. 일반적으로 분류 문제에 크로스 엔트로피 손실 함수가 사용되므로 먼저 사용해 보았으나, 특정 클래스에서 문제를 잘 맞추지 못하는 문제가 발생하였고 이는 클래스 불균형에서 오는 문제로 판단하였다. 해당 문제를 극복하기 위해 비교적 쉽게 맞추는 문제에는 패널티를 부여하고 어려운 문제에는 가중치를 부여하는 Focal loss function을 도입했다. 아래 식에서 p_t는 예측확률을 의미하며 (1-p_t)는 예측한 정도가 강할수록 그 값이 작아지게 되고 이는 못맞추는 문제에 대해 가중치를 부여하는 것을 의미한다. 여기에 감마값을 부여해서 그 가중치의 정도를 정하게 된다.

$$\mathrm{FL}(p_\mathrm{t}) = -\alpha_\mathrm{t}(1-p_\mathrm{t})^\gamma \log(p_\mathrm{t}).$$

< Focal Loss Expression >

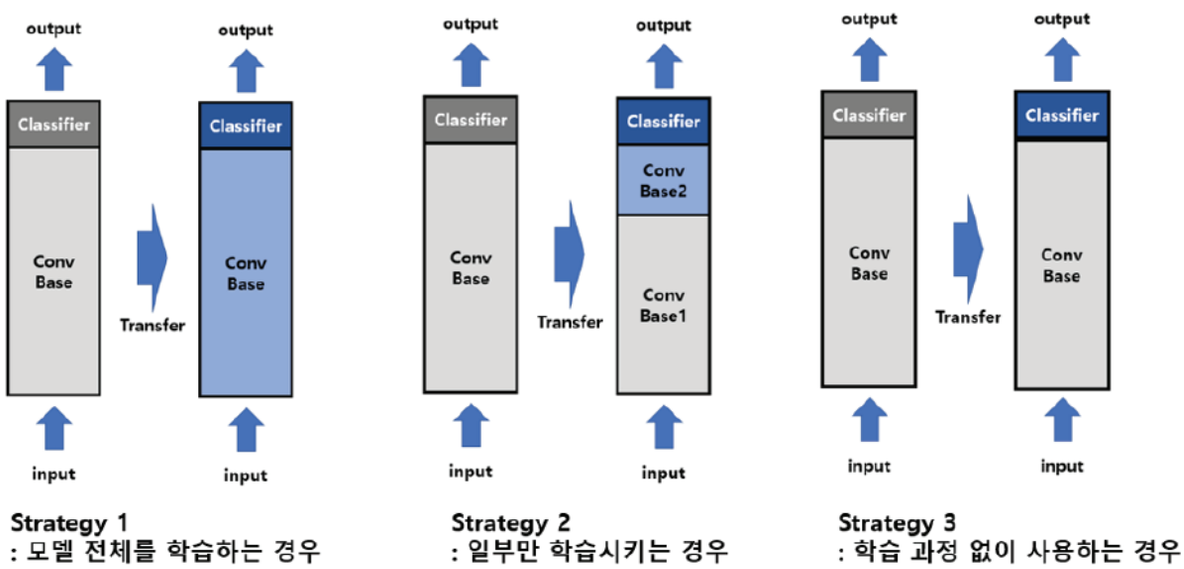
11. Fine-tune

Fine tuning은 사전 학습된 모델의 가중치를 미세하게 조정하여 새로운 태스크에 적용하는 방법이다. 이 프로젝트에서는 ImageNet에서 사전 학습된 모델을 사용하였으므로, 이를 통해 주어진 태스크를 수행하고자 한다.

11.1 Fine-tune Methods

Fine-tuning 방법은 크게 세 가지로 구분할 수 있다. 첫 번째는 모델 전체를 학습시키는 방법이고, 두

번째는 모델의 일부 레이어를 고정(freeze) 한 후 나머지 레이어를 학습시키는 방법이다. 세 번째 방법은 분류기(classifier)를 제외한 모든 레이어를 고정한 후, 분류기(classifier)만 학습시키는 방식이다. 본 프로젝트에서는 얼굴 데이터셋을 사용하였으므로, 일부 레이어만 학습시키는 방법을 사용하였다. 사전 학습된 모델의 초기 레이어는 일반적인 특징을 학습하는 반면, 후반의 레이어는 더 복잡하고 고차원적인 특징을 학습한다. 따라서 얼굴 데이터셋과 같이 특정 도메인에 특화된 특징이 중요한 경우, 후반의 레이어만을 학습시켜 특화된 특징을 더 잘 반영하도록 하는 것이 효과적일 것으로 기대한다.



< Fine Tuning Strategy >

11.2 Performance Improvement

Model	Freeze layer	F1-score (val)
ViT (tiny)	None	0.7193
ViT (tiny)	0, 1	0.7452
ViT (tiny)	0, 1, 2	0.7306

< Layer Freeze Experiment Table >

이 표는 Vision Transformer (ViT)의 'tiny' 모델을 사용하면서, 고정시킨 레이어(Freeze layer)에 따른 성능을 나타낸다. 'None'은 모든 레이어가 학습한 것을 나타내며, '0, 1'은 첫 번째와 두 번째 레이어가 학습에서 제외되었음을, '0, 1, 2'는 첫 번째, 두 번째, 그리고 세 번째 레이어가 학습에서 제외되었음을 나타낸다.

이 중에서 첫 번째와 두 번째 레이어를 고정시킨 '0, 1' 이 가장 높은 F1-score를 달성하였다. 이 결과

는 고정된 레이어가 사전 학습된 모델에서 얻은 정보를 유지하면서, 나머지 레이어가 추가적인 학습을 통해 더욱 복잡한 특징을 추출할 수 있음을 보여준다. 이를 통해, 모델이 효과적으로 패턴을 학습하는 것을 보여준다.

12. Model Ensemble

실험을 통해 Top 1 모델이 Train 데이터셋에 과적합된다고 판단하여 이를 해결하고자 모델 앙상블을 적용하였다.

12.1 Ensemble Methods

기존 데이터셋, 데이터 제거 데이터셋, 수정된 레이블링 데이터셋, stage-2 데이터셋으로 총 4개의 데이터셋을 사용하여 Top 1 모델 아키텍처에 학습을 시킨 후 4개의 모델을 Soft Voting Ensemble을 수행하였다.

12.2 Performance Improvement

Model	F1-score (test)
Top1 ViT (tiny)	0.7517
Top4 ViT (tiny) Ensemble	0.7558

< Voting Ensemble Table >

이 표는 앙상블 방식을 사용한 모델의 성능을 비교한다. Top1 ViT (tiny)는 단일 모델을 의미하며, 반면에 Top4 ViT(tiny)는 4개의 다른 데이터셋으로 학습된 모델들을 앙상블한 것을 나타낸다. 이때, 앙상블은 soft voting 방법을 사용하여 진행하였다. 각 모델은 다양한 조합으로 생성된 데이터셋을 이용하여 학습하였다.

Top4 모델들의 앙상블 결과, F1-score는 0.7558로, Top1 모델의 0.7517에 비해 약간의 성능 향상을 보인다. 이 결과는 다양한 데이터셋으로 학습한 모델을 앙상블하는 것이 성능 향상에 효과적인 방법임을 보인다.

13. Conclusion

본 프로젝트에서는 COVID-19 팬데믹 대응을 위한 공공 보건 조치의 일환으로, 카메라로 촬영된 얼굴 이미지를 통해 마스크 착용 상태, 나이, 성별을 분류하는 모델을 개발하였다. EDA를 통한 데이터 분석과 이를 바탕으로 수행한 레이블 오류 수정, 추가 데이터 생성을 통한 균형 잡힌 데이터셋을 생성하였다. 또한, 배경 제거와 얼굴 인식을 적용하여 모델의 정확도를 높였다. 뿐만 아니라 MTL 및 모델 앙상블을 도입하여 모델의 정확성 및 견고성을 강화했다. 이러한 전략들을 통해 복잡한 분류 문제에 대한 해결책을 제시하였다.

다만 Midjourny와 같은 Text-to-Image 생성 소프트웨어로 이미지 생성에 많은 시간이 걸려 원하는 양 만큼이 데이터 생성이 어려워 사용하지 못한 점, TTA(Test Time Augmentation), K-fold cross validation, 멀티 태스크 학습을 적용 등과 같은 성능 개선 기법들을 적용하지 못한 한계가 있다.

14. Review

14.1 Team Roles and Responsibilities

- 김민윤(T6017) : Metric 구현, Validation Dataset 구축, Background Removal & Face zoom in, Data Remove 함수 구현
- 배종욱(T6071) : Age transformation, Masked Image generation, Re-labeling 툴 제작
- 신호준(T6091) : 일정 스케줄링, BaseLine Code 제작, Git 관리, W&B연동, Metric 구현, Activation Map 구현, Data Checking Tool 제작, Background Removal & Face zoom in, Backbone, Loss, Finetuning, Ensemble 실험 및 분석
- 전병관(T6152) : Incorrect Masked Image generation, Dataset grouping
- 최수진(T6174) : Multi-task Modeling 구현, Backbone & Method Application 실험 및 결과 분석

14.2 Collaboration Tools

14.2.1 Notion, Google Calandar

프로젝트 일정 관리, 업무 정리 및 공유

https://www.notion.so/eddyyy/classification-7ea2926faff04537ab8e31716c151f9f?pvs=4

14.2.2 WandB

모델 실험 기록 관리, 공유 및 시각화

https://wandb.ai/boostcam_level1/projects

14.2.3 Github

코드 버전 기록 및 코드 공유

https://github.com/boostcampaitech6/level1-imageclassification-cv-09

14.2.4 Discord

실시간 커뮤니케이션 및 자료 공유