마스크 착용 상태 분류 프로젝트 Wrap up Report

Recsys-15 (BEST CHOICE)

김원섭(T3044), 김진수(T3058), 민태원(T3080), 이상목(T3146), 조민재(T3204)

1. 프로젝트 개요

"카메라로 촬영한 사람 얼굴 이미지의 마스크 착용 여부를 판단하는 Task"

COVID-19 의 확산으로 우리나라는 물론 전 세계 사람들은 경제적, 생산적인 활동에 많은 제약을 가지게 되었다. 감염자의 입, 호흡기로부터 나오는 비말, 침 등으로 인해 다른 사람에게 쉽게 전파가 될 수 있기 때문에 감염 확산 방지를 위해 무엇보다 중요한 것은 모든 사람이 마스크로 코와 입을 가려서 감염자로부터의 전파 경로를 원천 차단하는 것이다. 하지만 모든 사람들의 올바른 마스크 착용 상태를 검사하기 위해서는 추가적인 인적자원이 필요하기 때문에, 우리는 카메라로 비춰진 사람 얼굴 이미지 만으로 이 사람이 마스크를 쓰고 있는지, 쓰지 않았는지, 정확히 쓴 것이 맞는지 자동으로 가려낼 수 있는 시스템이 필요하다. 이 시스템이 공공장소 입구에 갖춰져 있다면 적은 인적자원으로도 충분히 검사가 가능할 것으로 기대할 수 있다

1.1 활용 장비 및 개발환경

OS : Ubuntu 20.04 LTS, Windows

IDE : VS Code

GPU : Tesla V100 *Boostcamp 로부터 제공받은 서버

주 사용 언어 : Python 3.8.5

Frameworks : Pytorch 1.7.1, torchvision, albumentations, etc.

Co-op tools : github, notion, slack

1.2 프로젝트 구조 및 사용 데이터셋의 구조도(연관도)

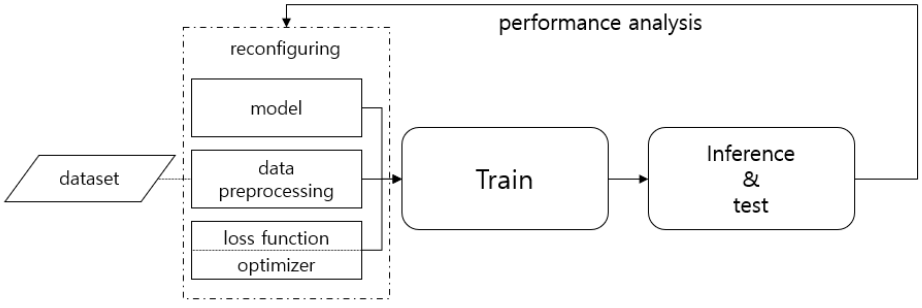

그림 1 프로젝트 구조도

성별, 인종, 나이 등 label 정보로 구성된 1 인 1directory 와 사람 한 명당 총 7 장의 서로 다른 image (올바른 착용 5 장, 잘못된 착용 1 장, 미착용 1 장)

그림 2 Train Dataset 구조

마스크 착용여부, 성별, 나이로 분류된 총 18개의 클래스

Class 1	Mask	Gender	Age
0	Wear	Male	< 30
1	Wear	Male	>= 30 and < 60
2	Wear	Male	>= 60
3	Wear	Female	< 30
4	Wear	Female	>= 30 and < 60
5	Wear	Female	>= 60
6	Incorrect	Male	< 30
7	Incorrect	Male	>= 30 and < 60
8	Incorrect	Male	>= 60
9	Incorrect	Female	< 30
10	Incorrect	Female	>= 30 and < 60
11	Incorrect	Female	>= 60
12	Not Wear	Male	< 30
13	Not Wear	Male	>= 30 and < 60
14	Not Wear	Male	>= 60
15	Not Wear	Female	< 30
16	Not Wear	Female	>= 30 and < 60
17	Not Wear	Female	>= 60

그림 3 분류에 따른 label class

2. 프로젝트 팀 구성 및 역할

팀 구성 : 조민재, 김진수, 이상목, 민태원, 김원섭

프로젝트 목적의 달성과 동시에 팀원 개개인의 성장을 목표로, U stage에서 학습했던 다양한 방법론들과 추가적으로 습득한 지식을 실습하였다. 팀원 개인이 모델링이 가능한 상황이므로, 각자 시도한 방법과 결과를 공유하여 성능을 높이는 형태로 협업하였다.

3. 프로젝트 수행 절차 및 방법

팀원 개인의 DL 프로젝트 숙련을 겸하여 각자의 작업환경에서 다양한 조건 하에 모델을 학습하고, 개선방안을 탐색하는 과정에서 성능 향상의 실마리를 찾아 나가는 방법을 택하였다. google, github, kaggle 등에서 reference 를 참조하여 다양한 방법론으로 실험해 보았다.

가장 먼저 EDA 를 통해 데이터의 분포를 파악하고 상대적으로 비중이 적은 데이터에 Data Augmentation 을 적용하였다. 이 후 학습에 사용할 Model 을 탐색하고 성능 향상을 위하여 label 별 분할 학습을 하거나, validation 에도 cross validation, K-fold 등 다양한 방법을 적용해 보았다. Ensemble 과(voting, Out-Of-Fold), 등을 시도하였다.

4. 프로젝트 수행 결과

4.1 탐색적 분석 및 전처리

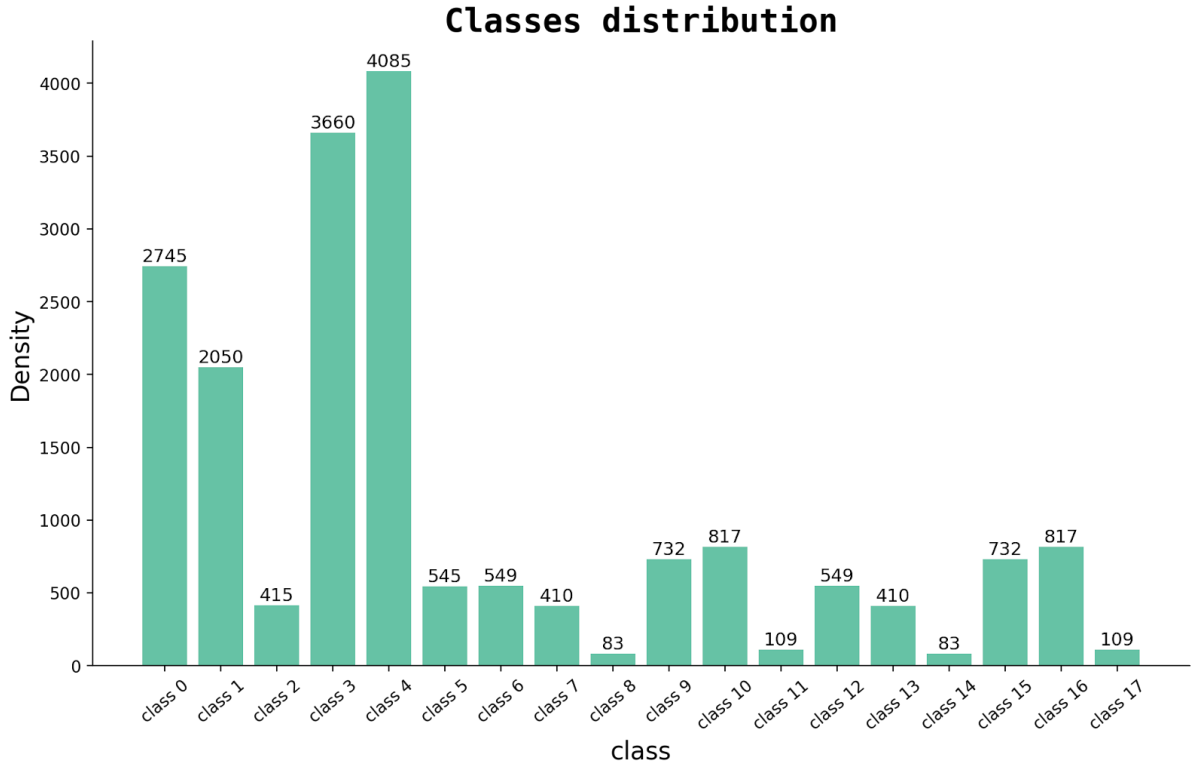

그림 4 전체 class 별 분포

각 class 간 데이터 수의 imbalance 가 심함을 눈으로 확인할 수 있다. 특히, 60 세 이상으로 분류되는 class {8, 11, 14, 17}의 데이터는 현저히 부족하다는 것을 알 수 있는데, 이 문제로 인해 데이터 수가 상대적으로 적은 class 의 데이터가 제대로 학습되지 못하고 상대적으로 비중이 큰 class 의 데이터는 과도하게 반영되는 문제가 발생할 수 있다.

데이터 수가 비교적 적은 class 들의 이미지를 복제하고, 이 후 augmentation 을 적용하여 전체 데이터셋의 균형을 맞춤으로써 data Imbalance 문제를 해결할 수 있다.

1) Gender & Age 분류 관점에서, 상대적으로 적은 측의 데이터들을 복제하여 비율을 맞춘다.

male	< 30	549
	>= 30 and < 60	410
	>= 60	83
female	< 30	732
	>= 30 and < 60	817
	>= 60	109

before

male	< 30	1000
	>= 30 and < 60	1000
	>= 60	1000
female	< 30	1000
	>= 30 and < 60	1000
	>= 60	1000

after

그림 5 gender & age 분포의 전처리 후 변화

2) Mask 클래스에 대해 'incorrect', 'normal' 데이터들을 복제하여 Mask 클래스 비율을 맞춘다

그림 6 전처리 이전 mask 착용 상태의 분포

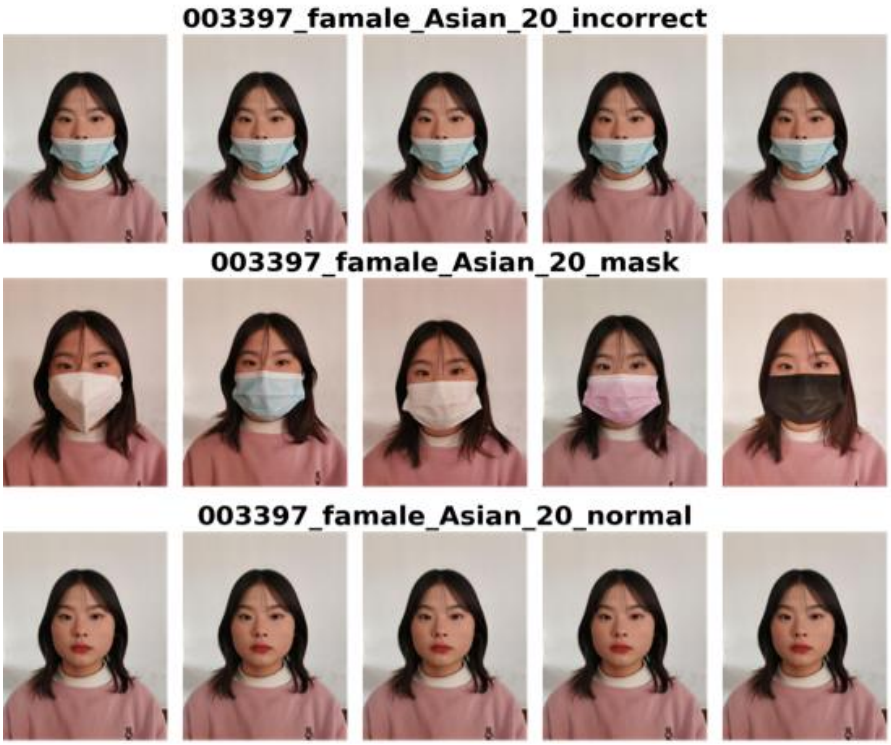

그림 7 전처리 이후 mask 착용 상태의 분포

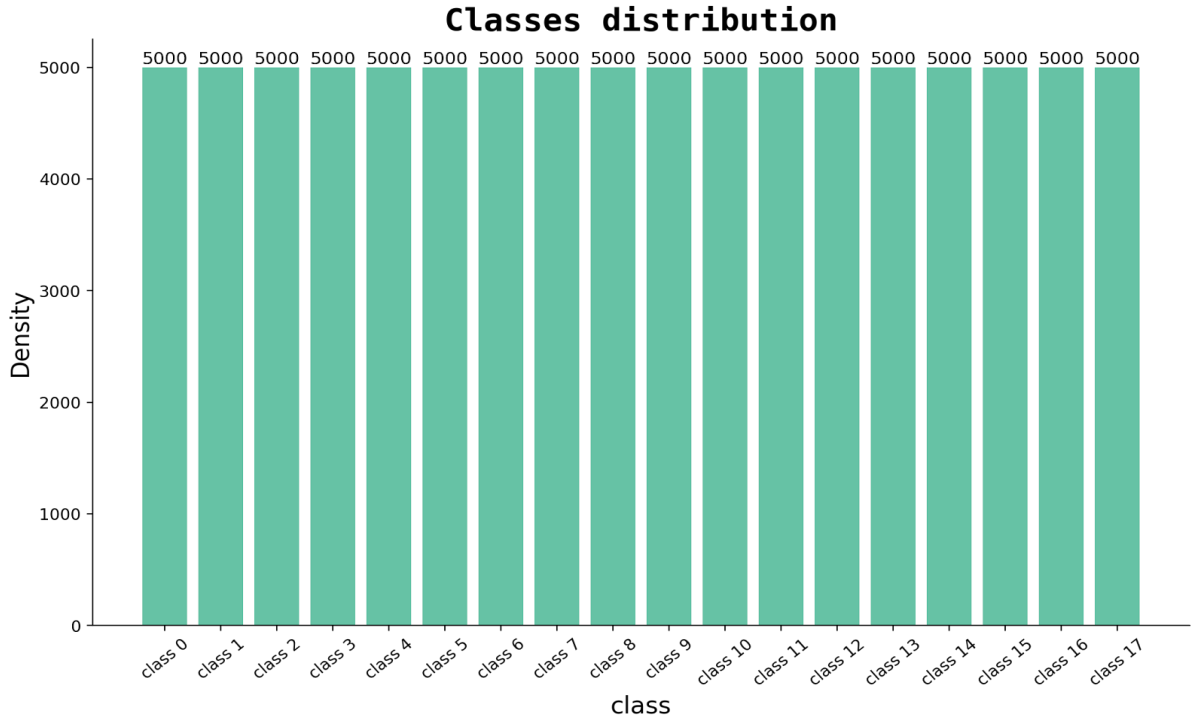

그림 8 전처리 이후 전체 class 분포

4.2 모델 개요

Pretrained 상태의 Resnet 18, 34, 50, 101, 152 를 주로 활용하였다.

layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
conv2_x	56×56	3×3 max pool, stride 2				
		$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$
conv3_x	28×28	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times8$
conv4_x	14×14	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times23$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times36$
conv5_x	7×7	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$
	1×1	average pool, 1000-d fc, softmax				
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9

그림 9 Resnet 18 ~ 152 내부 구조

4.3 모델 선정 및 분석

아래 표는 유의미한 결과를 보인 실험들 중 대표값을 선별했다.

backbone	resnet18	efficientnet b3	resnet18 + efficientnet b0	resnet34	resnet152	resnet18
augmentation	aug_0	aug_1	aug_0	aug_0	aug_0	aug_0
img_size	(512,384)	(512,384)	(512,384)	(512,384)	(512,384)	(512,384)
optimizer	Adam	Adamax	AdamW	Adam	Adam	AdamW
loss function	Focal	label smoothing	*CE	label smoothing	CE	CE sum (CE_by_labels)
learning rate	0.001	0.0005 ~0.00001	0.001	0.001	0.001	0.001
main feature	hard-voting ensemble	StepLR	layer concat classification	**A+G+M 개별모델 ensemble		multi_label classification
LB F1 score	0.7097	0.6770	0.6620	0.7128	0.7200	0.7005

***CE : Cross Entropy**

**** A : Age, G : Gender, M : Mask**

AUGMENTATION TYPE	FUNCTIONS
AUG_0	HorizontalFlip(), ShiftScaleRotate(), RandomBrightnessContrast(limit = (-0.3,0.3)), GaussNoise(), ColorJitter(), Solarize()
AUG_1	Random Augmentation

Efficientnet, Resnet, Mnasnet 등 다양한 모델을 활용하여 Fine-Tuning 하고, 제출 결과 중 f1-score 가 가장 높은 모습을 보인 Resnet 계열 모델을 집중적으로 활용하였다.

또한, 목적함수로 label smoothing, f1-loss, focal loss, cross-entropy loss 를 적용했다. f1-loss 를 사용했을 때 LB 점수가 눈에 띄게 나쁜 결과를 보였고, 나머지 목적함수들은 대체로 비슷한 수준의 성능 개선을 보였다.

4.4 모델 평가 및 개선

유의미한 결과를 기대하며 활용했던 방법들이 큰 소득을 보지 못했다. 다양한 원인들을 짚어볼 수 있는데, 우선 데이터 간 격차를 줄이기 위해 빈약한 60 대 이상의 데이터들을 증강하는 과정에서, 비슷한 정보를 가진 augmentation 결과물들이 오히려 overfitting 을 야기했을 가능성이다. 학습에 사용된 resnet 이 2014 년에 나온 구형 모델이라는 점도 빼놓을 수 없고, train 에 사용된 데이터셋과 test 에 사용되는 데이터셋의 분포가 지나치게 불균형을 가졌을 수도 있다. TTA 를 사용하여 이 점을 극복할 수 있었을 것이다.

결과적으로는 추가적인 작업을 거의 가하지 않고 resnet152 에 기본적인 augmentation 만 사용한 데이터셋으로 학습한 모델이 가장 좋은 점수를 냈다. 실질적으로 모델을 거의 개선하지 못했음을 방증하는 결과다. 다만 이후에 참여할 competition 에서 최신 동향의 논문들을 이해하고 적용할 수 있게끔 고전 방법론들을 충분히 실험해볼 수 있었다.

5. 자체 평가 의견

1 인 당 한 개씩 주어진 GPU 로 각자 시도하고 싶은 작업을 할 수 있어 보다 다양한 방법론을 시도해볼 수 있었던 점은 긍정적인 영향을 주었다. 하지만 팀원별로 작업하며 방향을 찾아가다 보니, 오히려 유연한 협업을 방해한 요인 중 하나가 되기도 하였다. 시도해 본 방법론에 대해 의견을 공유하고 결과를 나누었지만, 겹친 작업이 다수 발생하는 등 비효율적인 분업이 되었던 점이 특히 협업 과정에서의 아쉬움으로 남는다. 또한 resnet, effnet 등 torchvision 라이브러리가 제공하지 않는 모델을 활용하려는 시도가 적었던 점이 아쉬웠다.

6. 팀원 개인 회고

조민재(T_3204)

목표

주어진 사진에서 인물의 age, gender, mask 착용 여부를 분류하는 model 구현

모델 개선을 위해 시도했던 방법

가장 먼저 데이터 EDA 를 시도하였다. 분류에 핵심적인 영향을 줄 수 있는 train data 의 class 별 분포를 살피고, age, gender, mask 중에서 age 의 불균형이 가장 심함을 파악하였다.
데이터의 preprocessing 에 공을 들였다. 데이터의 imbalance 를 해소하기 위해 augmentation 을 통한 oversampling 을 수행하고, mislabeling 된 데이터가 많아 수작업으로 선별 후 수정하였다.
데이터의 양 자체가 적었기에, validation 을 하면서도 train data 를 최대한으로 활용하기 위해 k-fold cross validation 을 실행한 5 개의 모델을 hard voting 방식으로 ensemble 하였다.
Focal loss, label smoothing, cross entropy loss 등의 다양한 목적함수를 적용시켜보았고, 가장 준수한 성적을 내는 optimizer 와 learning rate 를 고정하였다.
Age, gender, mask 을 각각 분류하는 3 개의 model 로 분리하여 그 결과를 묶어 classify 하였다.
위의 시도들을 다양한 방법으로 조합하여 가장 우수한 성능을 내는 pretrained model 을 선별하였다.

회고

위에서 언급한 방법들은 전혀 의도한 결과를 보지 못했으며, 시도에 따라 리더보드 점수가 랜덤하게 오르락내리락 했다. 기존에 시도하던 방식의 조합을 바꿔가며 운에 의지하여 높은 점수를 내보려던 것이 가장 큰 패인이었다. 시도한 방법들 역시 Base line 코드에서 제시한 가이드라인과 고전적인 방법에서 크게 벗어나지 못했다.

단순히 주어진 문제를 겉으로만 훑고 1 차원적인 해결책을 강구하는 것은 유의미한 성능향상을 꾀할 수 없었다. 문제를 세부적으로 바라보지 못했던 것이다. 계속된 제자리걸음에 좁아진 시야를 떨치고 다양한 가능성을 고민해 봤어야 했다. 대회 종료 후 우수팀들의 방식을 보며, 내가 얼마나 틀에 갇혀 의미 없는 반복을 해왔는지 깨달았다. 단순히 주어진 데이터의 불균형을 해결해야한다! 모델이 많을수록 좋은 결과를 낼 것이다! 에 몰두해 평가 기준인 F1-score 를 높이려면 어떻게 접근해야 하는지, 모델이 주어진 데이터를 온전히 학습하고 있는지, 쉽게 접근할 수 있는 torchvision model 들 외에 현재 좋은 평가를 받는 모델이 무엇인지, 내가 시행착오로 고정한 hyperparameter 들에 더 좋은 선택지가 있었는지, 단순히 oversampling 한 데이터들이 오히려 과적합을 일으키지는 않는지에 대한 어쩌면 당연히 해봤어야 했던 고민들을 외면했다.

앞으로의 방향

모델을 손수 구현하는 것이 처음이었기 때문에, 이론으로만 알고 있던 지식들을 실험하는 데에 시간을 모두 소비한 것이 이번 대회의 주요 패착이라고 볼 수 있었다. 그러나 성장 측면에서는 유의미했는데, 이번 대회를 통해 강의로 배운 고전적인 method 들을 원 없이 실험하고 익혀 볼 수 있었고, 우수 팀들의 사례를 보며 모델 성능의 향상을 위해 어떤 방법을 모색해야 하는지 잘 배웠기 때문이다. 또한 최신 트렌드의 논문들과 자료들을 찾아보는 일을 게을리하지 않아야 함을 상기했다.

김진수(T_3058)

목표

PyTorch 프로젝트에 대한 숙련도 향상 및 협업 경험 쌓기

개인 실험

데이터 EDA, 베이스라인 코드 개선

Albumentations 를 통한 Data Augmentation : randomAug 를 사용하여 다른 레이블의 수와 Imbalance 문제를 해결하게끔 시도하였음. 결과적으로 학습정확도 향상이 이루어졌다.
또한 기존 Mask 를 사람별로 나누는 코드를 폴더별로 나누고, K-fold 를 적용시킬 수 있도록 indexing 을 하는 등 기존 코드를 개량하였다.

베이스라인코드의 Loss, 커스텀모델, 학습률 조정 등의 하이퍼 파라미터 조정 : F1 Loss, label_smoothing, Focal, CE Loss 등의 Loss 와 Resnet, Mnasnet, Effnet 계열 모델을 사용해보고 결과를 통해 성능을 측정했다.

마스크, 성별, 나이를 따로 예측하는 3 개 모델을 통해 출력값을 도출하는 모델 개발 : 나이를 잘 예측하지 못한다는 점에서 생각해낸 방법이었으나, 결과적으로 큰 성능향상은 이루어지지 않았다.

K-fold Validation 및 Out-Of-Fold 와 Test Time Augmentation 구현 : K-fold Validataion 을 직접 구현하였으나, 애초에 데이터자체에서 Overfitting 이 일어날 수 있는 상황이었기에 검증에서 큰 효과를 보지 못했다.

Out-Of-Fold 기법을 통해 여러 모듈을 앙상블하여 0.2 수준의 성능향상을 이루어낼 수 있었다.
TTA 를 통해서도 일반화가 잘 이루어졌다.

Dropout 층 추가 : 모델의 정확도가 100%을 찍었기에 Overfitting 이 일어나고 있음을 파악했다. Robust 한 모델을 위하여 Dropout 층을 Pretrained 모델의 classifier 앞에 추가해보았다. 하지만 효과는 미미했다.

아쉬웠던 점

협업 : 처음에 계획했을 때는 1 주일간 각자 다양한 실험을 해보고 2 주차에 하나의 모델로 개발을 해보기로 했으나, 하다보니 각자의 프로젝트를 진행하게 되었고, 서로가 실험해봤던 방법론을 다시 실험해보는 상황이 발생하게 되었다. Git 에 대한 활용법을 배워서 프로젝트에서 활용해보거나 체계적으로 표를 만들어 실험과정을 관리했다면 하는 아쉬움이 컸다.

다른 방법론들

Grad-cam : Grad-Cam 을 마지막에 알게 되었는데, 이를 사용해보지 못해서 아쉽다. 이것을 적용해본다면 모델이 사진의 어떤 부분을 보는지 알 수 있다. 다음에 하게된다면 우선적으로 구현해야할 방법으로 파악된다.

HyperParameter Tuning : 강의에서도 효과가 미미하다고 하여 제일 마지막 날에 해보려 했으나, 결국 시간이 부족하여 해보지 못했다. 이를 지원해주는 NNI 나 ML flow 같은 툴들이 있는데, 못해보아서 아쉽다.

조금 더 다양한 모델 실험 : 너무 Resnet 과 Efficientnet 계열 모델들만 실험해보았던 것과 상위팀들이 발견했던 Swin 모델을 찾아내지 못했던 것은 아쉽다. 너무 timm 라이브러리쪽으로 쉽게 개발하려

했던 것이 실수였고
다음에는 Github 의 모델들을 활용해볼 수 있도록 해야겠다.

이상목 (T_3146)

목표
모델의 성능 개선
효율 좋은 dataset, model, loss...etc 코드 작성

고민해본 문제들
Task 분해에 대한 고민
해당 문제는 독립적으로는 18 개의 class 를 분류하는 문제이지만, Age, Gender, Mask 각각에 대한 분류 문제로 생각할 수 있다.
하지만, 독립적인 18 개 분류 문제로 접근해도 그 수가 많지 않으므로 충분히 모델이 잘 분류할 것으로 기대할 수 있고 이에 따른 성능 비교가 필요하다.
또한, 한 개 모델이 3 가지를 한번에 분류하는 경우와 각각의 모델이 Task 에 대해 분류하는 경우에 대한 비교 역시 필요하다.
불균형 처리에 대한 고민
나이브하게 생각하였을 때, 불균형를 해결해 주는 방법을 통해 모델의 성능 개선을 꾀할 수 있을 것이다.
하지만, 가장 큰 불균형을 보인 데이터는 나이에 대한 데이터였으며, 나이에 대한 3 개 클래스 중 2 개 클래스는 비슷한 충분한 숫자를 가지고 있었다.
이에 따라 오히려 충분히 학습시키면, 오히려 나머지 2 개 클래스의 디시전 바운더리가 더 명확해져 데이터가 부족한 클래스도 잘 예측할 수 있으리라 생각하였다.

해결하지 못한 문제
오버피팅에 대한 제어
Train Set 과 Valid Set 을 구분하여 학습시켰지만, 어느정도 학습된 이후 학습 정확도와 검증 정확도의 갭이 발생하며 오버피팅되는 모습을 관찰할 수 있기를 기대하였다.
하지만 오히려 학습을 시킬수록 검증 정확도까지 100%에 가까워지는 모습만 관찰할 수 있었고, 결국 오버피팅 문제는 대회 내내 전혀 관리하지 못했다.
분류 레이어에 Dropout 을 추가하는 정도로는 해결되지 않았고, 사람을 기준으로 Train 과 Valid Set 을 구분하였으니, 사람 자체에 대한 치팅은 일어날 수 없는 상황이었다.
해당 문제의 원인으로는 같은 배경에서 찍힌 비슷한 성별/연령의 사람들이 많았기 때문으로 추정하고 있으며, 사람이 아닌 배경을 보고 성별/연령을 구분하는 오류를 일으킨 것으로 생각한다.
이를 해결하기 위해 이미지의 중간을 잘라내서 학습하는 형태로 배경의 영향력을 줄여보려 시도하였지만, 큰 효과를 얻진 못했다.

시도하지 못한 것

이미지 로딩 과정의 정교한 설계

이미지는 기본적으로 디스크에 저장되어 있으며, baseline 에서는 디스크에서 그때 그때 읽어 feeding 시키는 형태로 구현되어 있었다.

이를 약간 개선하여 이미지를 메모리에 올린 후 feeding 시키면 좀 더 빠르게 학습시킬 수 있을 것으로 생각하였지만, 디스크에서 그때 그때 읽어 feeding 시키는 것의 학습 시간 차이를 직접 느끼지 못했다.

하지만 이 과정에서 약간 안일하게 접근하여, 시간을 측정하는 코드를 추가하여 개선된 정도를 수치화 하지 직접 이득을 비교해 보는 작업을 진행하지는 않았다.

테스트 코드 작성

Augmentation 코드를 작성하거나, dataset 관련 부분을 작성할 때, 테스트 코드를 작성하여 작성하면 개발 속도 및 좀 더 강건하고 효율적인 코드를 작성하는데 효과적으로 작용하였을 수 있을 것으로 생각되지만, 미처 작성하지 못했다.

차후 목표사항

dataset 을 작성하는 과정에서, 처음에는 선형적으로 메모리에 이미지를 하나씩 올리는 형태로 구현하였었는데, 이 때 너무 오랜 시간이 필요로 했다. 결국은 None 으로 이미지 리스트를 초기화한 후 dataloader 의 worker 가 첫 에포크를 돌 때 채워 넣도록 구현하였는데, 이 역시 정교한 설계로 생각되지는 않아 아쉽다. worker 의 작동 원리 파악하고 GIL 등 파이썬의 특징에 대해 좀 더 공부하여 활용할 수 있도록 정리해보려 한다.

실험과 공부도 좋지만, 협업 과정에 있어 약간 아쉬운 부분이 있었다. 특히 일부 모델을 작성하는 과정에서 전용 loss 가 필요했던 부분 등이 협업을 방해했던 부분이라 생각한다. 이를 더 쉽게 공유할 수 있도록 컨벤션 개선 및 쉽게 문서화할 수 있도록 노션 및 슬랙을 좀 더 적극적으로 활용할 필요가 있음을 느꼈다.

민태원(T_3080)

목표

baseline 코드 이해와 score 의 상승을 위한 다양한 방법론 적용

목표 달성을 위해 시도해본 방법

<개인 학습>

다양한 model 들을 추가해 사용 : torchvision 에 있는 다양한 model 을 사용해 보면서 model 별로 score 에 미치는 영향을 확인, torchvision 외에도 image classification 을 검색했을 때 나오는 model 들을 github 에서 받아와서 사용.

(결과) 결과적으로는 모두 좋지 않았음. model 의 사용 방법이 잘못된건지 hyperparameter 설정에 문제가 있었는지는 의문. 추후 학습을 계속 진행하며 잘못된 부분은 수정하기로 판단.

oversampling 기법 적용 : Imbalanced data 로 인해 성능이 나오지 않는다고 판단해 같은 이미지를 여러번 사용하는 oversampling 을 수행한 후 다양한 transform 을 적용하면 Imbalance 문제를 해결할 수 있다고 생각.

(결과) training 과정에서는 성능이 많이 올라갔지만 막상 LB score 는 급감하는 현상 발생. overfitting 의 문제가 발생한다고 판단.

overfitting 을 해결하기 위한 다양한 방법을 고민 : oversampling 한 데이터들의 다양성을 주기 위해 transform 을 더 다양하게 주거나 dropout layer 를 추가하는 방법을 생각하며 관련 정보를 찾아봄. (결과) transform 의 추가는 유의미한 효과를 주지 못했음. dropout layer 추가는 layer 구성을 어떻게 변경해야 할 지 몰랐기 때문에 관련 정보를 찾아보다가 대회 기간 종료.

<공동 학습>

개개인의 역량 향상 : 팀원들 대부분 딥러닝 프로젝트나 competition 에 대한 경험이 부족했기 때문에 우선 적응하기 위해 각자 자신의 아이디어를 적용해보며 역량 향상을 목표로함.

방법론이나 아이디어에 대한 공유 : 프로젝트를 진행하면서 어떤 model 을 사용해봤고 성능은 어떤지에 대해 공유하고 oversampling 이나 data augmentation 에 대한 생각을 공유함.

아쉬운 점

competition 참여가 처음이었기 때문에 baseline 을 처음 받았을 때, 코드 분석을 하는데 시간을 너무 많이 사용했던 것이 아쉬웠음.

혼자서 문제 해결을 해온 경험이 더 많았기 때문에 질문할 수 있는 다양한 방법들을 활용하지 못하고 비효율적으로 진행한 것 같아 아쉬움.

팀 단위로 진행된 프로젝트였지만 대부분의 시간을 개인학습에 사용했고 팀의 이점을 살리지 못한 것이 아쉬움.

상위권 팀들의 발표에서 다양한 Loss Function 들에 대해 의문을 가지고 분석한 것을 보고 이론적인 부분에 대해 고민을 못해봤다는 것이 아쉬움.

마주한 한계

코드 구현 능력 : 이론적으로는 어느정도 이해했지만 이를 구현 및 적용하기 위한 코딩 능력 부분이 부족했다고 생각.

딥러닝 이론에 대한 이해 : 애매하게 이론적인 부분을 알고 있는 경우, '왜 이렇게 하는지', '왜 이런 결과가 나오는지'에 대한 이해가 부족하다고 생각. (ex. pre-trained model 을 사용할 때, 어떤 부분의 weight, bias 를 어떤 방법으로 초기화 해야되는지를 정확하게 이해하지 못함.)

교훈 및 앞으로의 방향

너무 완벽함을 추구하려다 보니 비효율적인 시간분배로 인해 좋지않은 결과를 가져옴. 좀 더 전략적인 방법으로 접근해야겠다고 느낌.

부스트캠프의 장점 중 하나는 좋은 동료들과 팀 워크를 발휘할 수 있는 경험을 할 수 있는 것인데, 이것을 잘 살리지 못했다고 생각하기 때문에 다음 프로젝트 부터는 어떻게 하면 팀원들과 시너지를 낼 수 있을지 고민해봐야 겠다고 생각함.

여유가 된다면 다양한 competition 을 참여해 보는 것이 다른 사람들의 노하우를 배울 수 있고, 좀 더 생각을 확장해 효율적으로 진행할 수 있는 방법을 모색하는데에 도움이 될 것이라고 생각함.

김원섭(T_3044)

목표

(팀) AI 모델링 파이프라인에 대한 이해를 위해 각자 모델링을 진행하고, 시도한 방법과 결과를 공유하며 성능을 높이는 형태로 협업을 진행한다.

(개인) 나만의 모델링 코드를 구축하고, 다양한 실험 사이클(가설-검증)를 구성한다.

나는 내 목표를 달성하기 위해 무엇을 어떻게 했는가? (타임라인)

baseline code 에 대한 이해 및 EDA

-> [baseline 제출] resnet18 을 fine-tuning 하여 학습시킨 모델을 제출하여 이후에 있을 실험에 대한 비교 기준(baseline)을 생성

결과: <accuracy> val: 90% | test: 20%

실패 원인 분석: train/val set 을 나눌 때, 동일한 사람의 사진이 각 set 에 나눠져서 들어가기 때문에 val 에 치팅과 과적합이 있을 것이라고 판단.

-> [가설: early stopping, train/val set split 구현] early stopping 을 구현하고, 한 사람이 train/val set 중 하나에만 들어가게끔 하면 기존 문제점들을 해결할 수 있을 것이다.

결과: accuracy 63%의 baseline 모델을 얻게 됨.

결과 분석: accuracy 가 63%인 모델을 얻게 되었으나, train/val 의 accuracy 점수가 80%에 임박하는 것에 비해 너무 떨어지는 점수를 얻는다는 것을 발견.

-> [가설: k-fold cv 구현] k-fold CV 를 통해 좀 더 일반화된 모델 검증을 한다.

결과: cv 에 대한 결과를 평균내어 좀 더 일반화된 모델 검증을 할 수 있었음

결과 분석: k-fold 를 그저 일반화된 모델 검증용도로 만들었으므로, 직접적인 모델 성능 향상은 기대할 수 없었고, 오히려 당장의 실험 속도가 저하되어 학습목표달성에 안 좋게 작용한다고 판단 -> 보류

-> [가설: 데이터셋 증강] 1. imbalanced sampler 라이브러리를 사용 / 2. 상대적으로 적은 데이터들을 증강시키어(Copy & Paste) class 비율을 비슷하게 맞추고, 중복되는 데이터에 대해 적절히 transform 시키면 모델 성능이 올라갈 것이다.

결과: 1. accuracy 65% / 2. accuracy 69%의 모델을 얻게 됨

결과 분석 1: imbalanced sampler 라이브러리는 minor 한 클래스는 oversampling 을 하지만, major 클래스에 대해서는 undersampling 이 적용됨. 이 때문에 큰 성능 향상이 일어나지 않음.

결과 분석 2: 성능은 향상되었지만, 과적합 문제가 발생하였을 수 있다고 판단.

-> [가설: albumentation, cutmix] albumentation 라이브러리를 통해 더 다양한 transform 을 적용시키고, Cutmix 를 구현하면, 일반화 성능이 올라갈 것이다.

결과: 성능 개선이 되지 않음

실패 원인 분석: Cutmix 가 사람이 아닌 배경 부분에 적용됐을 경우에도 학습을 하기 때문에 오류가 발생할 수 있음

제대로 된 원인 분석을 하지 않고, 다음 실험을 진행

-> [가설: Task 분해, nni] 팀원이 Age, Gender, Mask 별로 따로 모델을 구현하여 ensemble 하는 것을 보고 따라함과 동시에 nni 를 통해 하이퍼파라미터 서칭을 진행

결과: 성능 개선이 되지 않음

실패 원인 분석: 이전 가설에 대한 실패 원인 분석과 피드백을 정확히 하지 않아, Cutmix 에서 동일한 오류가 발생했다고 판단됨

마주한 한계는 무엇이며, 아쉬웠던 점은 무엇인가?

계속되는 실험에도 성능이 나오지 않아 스스로 너무 조급해했던 것 같다. 이 때문에 실험 이후 제대로 된 피드백과 원인 분석을 하지 않았던 것 같다.

팀 단위로 협업을 제대로 못했던 점이 아쉽다.

시간이 부족하여 마지막 가설에 대한 실패 원인을 개선하지 못한 점과 다양한 모델을 실험해보지 못했던 점이 아쉽다.

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

계속해서 다음 실험을 계획하는 것보다 한 실험에 대해 정확한 원인 분석과 피드백을 하는 것이 더 중요하다는 것을 알았다. 다음 프로젝트에선 실험 별로 글이나 대시보드를 작성하며 이러한 문제를 해결해보도록 해봐야겠다.

효율성과 효과성을 높이기 위해 팀 내 분업, 대시보드 생성 및 관리 등 팀 단위 전략을 세세하게 세우고 프로젝트에 임해보고자 한다.