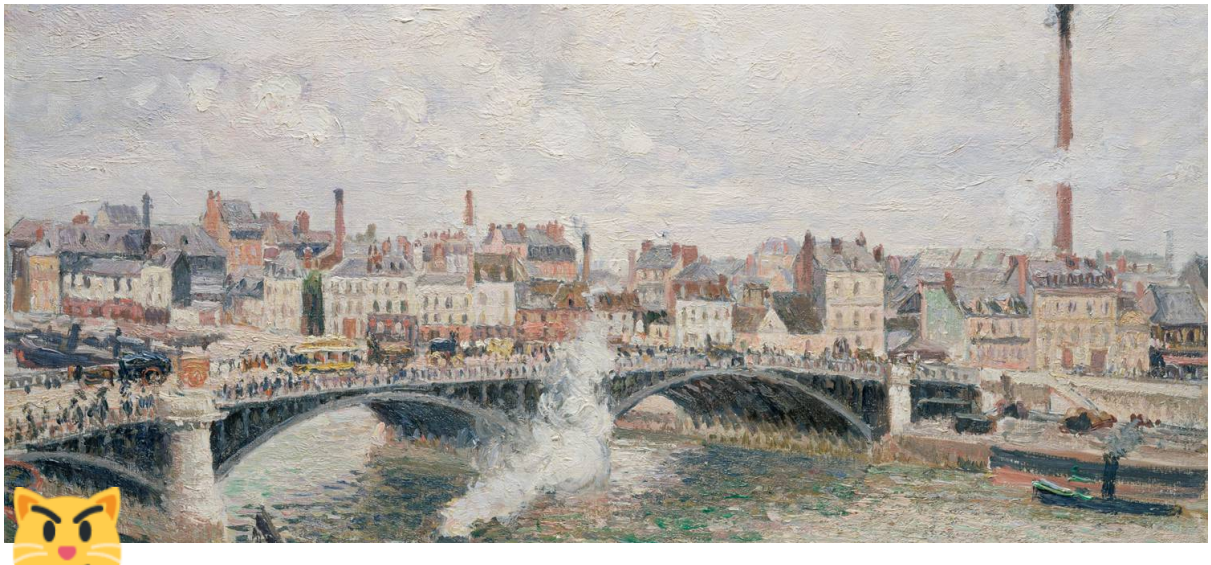


# 랩웹리포트

| | |
|---|---|
| 강의 번호 | |
| 복습 | ☐ |
| 유형 | |
| 자료 | |
| 작성일시 | @2021년 4월 8일 오후 3:34 |

# [Pstage_Competition]-Image Classify

> Public Score: 0.7835 4등
> Private Score: 0.7726 3등

---

## MASK | AGE | Gender Classification

- ✔️**Competition Data 개요**
  1. 전체 사람 명 수 : 4500
  2. 한 사람당 사진의 개수 : 7( 마스크 착용 6, 이상하게 착용(턱스크,코스크) 1장, 미착용 1장 )

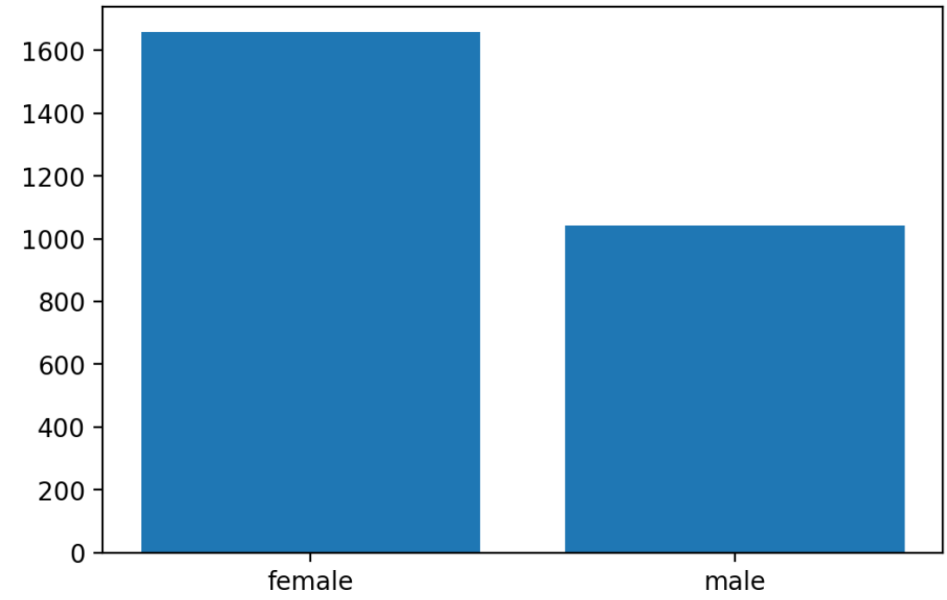
3. 이미지 크기: (384,512)
4. Evaluation : F1_score (Macro)

마스크 착용여부, 성별, 나이를 기준으로 총 18개의 클래스가 있습니다.

| 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 |

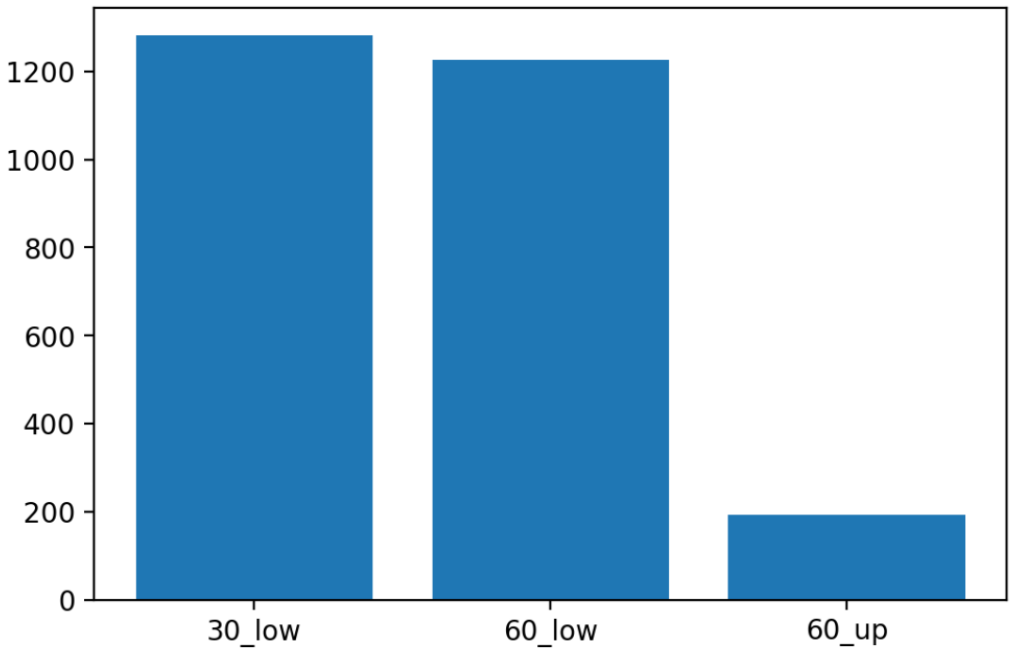
## EDA (Exploratory Data Analysis,탐색적 데이터 분석)

- **성별 분포**




> 여성 1600 남성 1100명 정도로 분포하고 있으며 많은 차이는 아니기 때문에 data imbalance 라고 보기엔 어렵다

- **Age 분포**




> 0~30 과 30~60 의 비율은 비슷하지만 60대의 비율이 압도적으로 적은 것을 볼 수 있다. 즉 data imbalance가 심하다.

- **Mask 분포**
  - [5: 1 :1]로 완전히 같은 비율을 가지고 있다.
  - 마스크마다 색이 다른 것도 존재 하며 , 마스크 대신 손수건을 착용한 이미지도 존재
  - 턱스크, 코스크 ,눈스크 등 마스크를 잘 못 착용한 다양한 이미지 존재
  - 마스크를 아예 착용하지 않은 클린한 이미지

## [학습]

1. Efficient_Net B3
2. Learning Rate:
    1. Age Label : 1e-4
    2. Mask Label: 3e-4
    3. Gender Label: 1e-4
3. Optimizer : Adam
4. Loss :
    1. Crossentropy_loss: Mask, Gender
    2. Focal_loss: age
5. Epoch:
    1. Age: 4 Epoch
    2. Mask , Gender : 10 Epoch
6. Batch_size: 64

## [전처리]

1. Facenet_pytorch : MTCNN 을 이용하여 face부분 crop (Train img, Test_img )
2. MTCNN을 통해 face가 detect 되지 않은 이미지는 Center_Crop [300,300]
3. Labeling :

1. Age→ [0~30 : 0 |  30~58:1 | 59~60: 2 |
2. Mask → 0 | 1 | 2
3. Gender: Female: 0 | Male : 1

## [Augmentation]

1. Mask:
    - Perspective, Rotate,RadomBrightness, HueSaturationValue, RandomContrast
2. Age:
    - Rotate, RandomGridshuffle(2,2), Perspective, GaussNoise
3. Gender
    - Rotate, Perspective, GaussNoise
4. ALL→ Resize(224,224) , Normalize()

## [Train & Validation]

1. 5_Fold 중 1개의 Fold를 선택하여 Train Set, Valid set 분할
2. Best F1-Score를 이용하여 model_parameter() update
3. Seed : 777 고정
4. Mask, Age, Gender 별로 다른 Learning Rate, Loss, augmentation 적용

## [Environment]

- Window 10
- VsCode
- GPU P-40

## 😊[Better Score & Method]

- **Age를 60이 아닌 59~60을 하나의 Class로 구분**

```
# 60 age의 개수는 192정도로 다른 class에 비해서 너무 양이 적으며 , 55~60세는 사람의 눈
으로도 구분하기 힘들정도로 개인별로 더 늙어보이거나, 더 젊어 보이는 차이가 존재 한다.
따라서 59,60을 모두 하나의 class로 바꾸어도 괜찮다고 생각하여 class를 나누었다
```

---

- **Mask, Age, Model 별로 다른 Train**

```
- `MASK`를 분리하는데 있어서 얼굴과 마스크의 위치를 제외하고는 모든게 Noise가 된다고 생각했다
따라서 RGB, HusSaturation Value, Brightness 등의  Augmentation을 적용해 주었고
일부 사진들은 정면이 아닌 관점의 변화도 있고, 얼굴을 기울인 사진들도 있었기에 Perspective()
Rotate(limit=20) 을 넣어 주었다 .

- `Age` 분류가 가장 어렵고 맞추기 힘든 부분이라고 생각한다. 아무리 높은 layer의 resolution이
높은 이미지를 사용한다 하더라도 , 5~10살 정도의 차이가 아닌 1 년 차이를 계산하는 것은 힘들다
즉 라벨의 기준이 되는 [29~30/ 59~60] 이 사이의 차이를 정확하게 예측하기란 힘들기 때문에
어느 정도의 오차는 불가피하다.
또한 age label의 경우 data imbalance가 심하기 때문에 crossentropy_loss가 아닌 focal_loss를
사용하였다.
또한 전체적인 형태가 아니라 주름, 피부색, 등 작은 feature에서도 특징을 뽑을 수 있도록 ,
RandomGridShuffle()을 적용하였다.

- 'Gender' mask와 마찬가지로 model은 test/valid set을 나눴을때(사람기준) 99%이상의 확률로
gender를 분류해 낸다. gender에 있어서 중요한 feature는 머리길이, 피부색, 얼굴형태, 이목구비,
등 낮은 layer에서 뽑아낼 수 있는 feature와 높은 layer에서 뽑을 수 있는 feature들이 둘다
존재 한다고 생각했다.
```

---

- **Using All K_fold**

```
처음에는 5_K-fold를 사용하여 train_set과 valid_set를 구분하여 학습하였고 ,
label별로 k_fold를 했을때와 사람 별로 K_fold를 했을때 성능이 label별로 나누었을때가 더
높았다.

사람별로 K_fold를 train하면 overfitting이 정확히 언제 되는지 알 수 있다는 장점이 있다.
하지만 Public F1_score는 0.7660이 한계 였다.

label을 기준으로 학습을 하게되면 f1_score는 계속 올라가게 된다. 즉 같은 사람의 이미지가
존재하기 때문에 overfitting이 되어도 valid_Test에서 검증이 제대로 되지 않는 단점이 있었다

따라서 시도한 방법은 사람을 기준으로 idx를 나눠서 학습을 할때 어느 epoch에서 가장 학습이
잘 되는지를 기록하고 같은 epoch로  label기준으로 된 train_set을 학습하는 방법이었다.

Epoch : 4일때 최고의 성능을 보였고 이 epoch를 label 기준 trainset에 적용하여 학습시켰다.

이때 최고 F1_score는 0.77까지 오르게 되었다 .

또한 train_img 와 test_img 에는 겹치는 사람들이 생각보다 많이 존재 하는데 모든 train data를
사용할 경우 generalize는 덜 되겠지만.. Test_img에 대해서는 더 정확한 classify를 할 수 있다는
생각으로 valid_set없이 오직 train_set만으로 4epoch만큼 학습 시켯을때 가장 좋은 성능을
```

보였다 .

- **Using TTA(Test Time Augmentation)**
  - [ TTA Kaggle ](https://www.kaggle.com/andrewkh/test-time-augmentation-tta-worth-it)
  - TTA를 이용한 test_set augmentation 사용
  - 0.6정도의 f1_score 상승

---

- **Wrong Labels modify**
  - Train_img에 잘못된 라벨들이 존재하여 수정

---

- **Using Focal_loss**
  - age의 경우 `data imbalance`가 심하기 때문에 `focal_loss`를 사용
  - loss에 weight을 추가해주었지만 성능 상승에 영향을 주지 않음

---

- **Crop img with Facenet**
  - `Facenet`을 이용하여 얼굴 부분 crop→ center crop보다 높은 성능을 보였음

---

## 😢[Failed & Method]

- input img size의 크기를 크게 했었지만 성능 향상 x

- Using larger Model → B4,B5,B7의 더 큰 모델들을 사용하였지만 성능 향상 x

- 5가지 trained model  앙상블 → Kfold별로 모델을 따로 학습 후 test set 에서 voting→ 성능 하락

- Loss Weight→ age의 데이터 불균형을 위해 사용 → 성능 향상 x

## 👍[피어세션&토론]

- 토론게시판과 피어세션을 통해서 정말 많은 것을 배울 수 있었습니다.
  - Train_set 과 Valid_set를 사람별로 구분하는 법
  - Data 불균형 시각화에 대한 다양한 글
  - TTA의 존재→ 피어세션분께서 TTA에 대한 언급을 하였고 어떻게 적용하는지 등 어려워 하는 부분에 대해서 세심하게 알려주시고 코드를 통해서 어떻게 적용하는지 까지 친절하게 알려주셧다.
  - 여러 loss들에 대한 견해 와 분석
  - Data 불균형을 해결 할 수 있는 여러가지 기법들
  - 정말 주옥 같았던 Master Session (보찬님, 태진님 너무 감사드립니다)
  - Vscode, pycharm을 이용한 py파일 사용 → production 증가 (공감중입니다!)
  - seed를 이용한 reproduction의 중요성
  - 여러가지 앙상블 방법들에 대한 토론글 → voting
  - 새로 알게된 Augmentation → cutmix( 어떻게 적용하고 사용하는지 )
- [Augmentation에 대해 내가 분석한글 ](http://boostcamp.stages.ai/competitions/1/discussion/post/40)

---

## ✔[Insight]

✓ 0.5s

| | Hong_psudo_label | songbae_tta |
|---|---|---|
| 0 | 2346 | 2299 |
| 1 | 1738 | 1310 |
| 2 | 1435 | 1829 |
| 3 | 1504 | 1520 |
| 4 | 1653 | 1482 |
| 5 | 312 | 551 |
| 6 | 477 | 468 |
| 7 | 344 | 281 |
| 8 | 295 | 362 |
| 9 | 305 | 301 |
| 10 | 320 | 297 |
| 11 | 69 | 100 |
| 12 | 474 | 465 |
| 13 | 309 | 281 |
| 14 | 323 | 361 |
| 15 | 305 | 301 |
| 16 | 326 | 288 |
| 17 | 65 | 104 |

- submission을 통해서 결과값을 계속해서 관찰했었다 위 사진은 acc가 제일 높았던 submission 과 f1_score가 제일 높았던 submission의 비교이다 . 또한 이전에도 계속해서 비교를 했었는데 점수가 acc가 거의 동일한데 f1_score가 0.05 정도의 차이를 내는 것이 `30 60의 비율 이었다` 그래서 60대를 조금더 예측하도록 age를 살짝 오버피팅 시켜서 제출했다

## 🤢[반성할점]

- TensorBoard를 사용해서 acc/ lr /loss등을 시각화하고 분석하려고 했으나 아직 익숙하지 않아서 제대로 시행하지 못했음.. 다음 컴페티션엔 꼭 해볼 것 !
- Ipynb만 쓰다가 py로 작업을 하려니 여러 문제들이 많았음.. 리눅스 명령어에 익숙하지 않았고 , 계속해서 오류가 발생해서 너무 힘들었음
- 여러 Arg.parser들을 추가했지만 아직 경험이 부족한건지 생산성이 그렇게 높아지지 않았음 좀 더 config파일을 다듬고 코드를 짤 때 좀 더 생각해서 코드를 짤 것
- 처음에 만들었던 코드들이 서버 문제로 날아갔는데 .. 복구하는데 5일이나 걸렸음.. 이 기간 동안 한번도 제출하지 못하고 도전해보지 못한 것이 아쉬움
- 여러 피어세션들의 아이디어를 듣고 시도해볼만하다고 생각은 했지만 막상 시도해본 것이 많이 없음.. 더 시도하고 도전했다면 더 높은 성과를 얻었을 것이라고 생각

## 🎁[정말 고마운 피어분]

- 김홍엽님께서 정말 다양한 방법들과 code들을 공유해 주셧고 여러가지 아이디어들도 공유해주셧습니다.
- TTA, AGE Filter, Ensamble ,Focal_loss, Arc Face_loss ,Psudo_lableing등 다양한 의견을 제시해주시고 어려워 하는 부분에 대해서 코드도 알려주시고 많은 도움을 받았습니다 .

## [Requirements]

| Package | Version |
|---|---|
| absl-py | 0.12.0 |
| albumentations | 0.5.2 |
| anyio | 2.2.0 |
| argon2-cffi | 20.1.0 |
| async-generator | 1.10 |
| attrs | 20.3.0 |
| autopep8 | 1.5.6 |
| Babel | 2.9.0 |
| backcall | 0.2.0 |
| bcrypt | 3.2.0 |
| beautifulsoup4 | 4.9.1 |
| bleach | 3.3.0 |
| boto3 | 1.17.39 |
| botocore | 1.20.39 |
| cachetools | 4.2.1 |
| certifi | 2020.12.5 |
| cffi | 1.14.0 |
| chardet | 3.0.4 |
| conda | 4.9.2 |
| conda-build | 3.18.11 |
| conda-package-handling | 1.7.0 |
| cryptography | 2.9.2 |
| cycler | 0.10.0 |
| decorator | 4.4.2 |
| defusedxml | 0.7.1 |
| efficientnet-pytorch | 0.7.0 |
| entrypoints | 0.3 |

```
facenet-pytorch         2.5.2
filelock                3.0.12
future                  0.18.2
gevent                  21.1.2
glob2                   0.7
google-auth             1.28.0
google-auth-oauthlib    0.4.3
graphviz                0.16
greenlet                1.0.0
grpcio                  1.36.1
idna                    2.9
imageio                 2.9.0
imgaug                  0.4.0
importlib-metadata      3.9.0
inotify-simple          1.2.1
ipykernel               5.5.0
ipython                 7.16.1
ipython-genutils        0.2.0
ipywidgets              7.6.3
jedi                    0.17.1
Jinja2                  2.11.2
jmespath                0.10.0
joblib                  1.0.1
json5                   0.9.5
jsonschema              3.2.0
jupyter-client          6.1.12
jupyter-core            4.7.1
jupyter-packaging       0.7.12
jupyter-server          1.5.1
jupyterlab              3.0.12
jupyterlab-pygments     0.1.2
jupyterlab-server       2.3.0
jupyterlab-widgets      1.0.0
kiwisolver              1.3.1
libarchive-c            2.9
madgrad                 1.1
Markdown                3.3.4
MarkupSafe              1.1.1
matplotlib              3.2.1
mistune                 0.8.4
```

```
mkl-fft                 1.1.0
mkl-random              1.1.1
mkl-service             2.3.0
nbclassic               0.2.6
nbclient                0.5.3
nbconvert               6.0.7
nbformat                5.1.2
nest-asyncio            1.5.1
networkx                2.5
notebook                6.3.0
numpy                   1.18.5
oauthlib                3.1.0
olefile                 0.46
opencv-python           4.5.1.48
packaging               20.9
pandas                  1.1.5
pandocfilters           1.4.3
paramiko                2.7.2
parso                   0.7.0
pexpect                 4.8.0
pickleshare             0.7.5
Pillow                  7.2.0
pip                     20.0.2
pkginfo                 1.5.0.1
prometheus-client       0.9.0
prompt-toolkit          3.0.5
protobuf                3.15.6
psutil                  5.7.0
ptyprocess              0.6.0
pyasn1                  0.4.8
pyasn1-modules          0.2.8
pycodestyle             2.7.0
pycosat                 0.6.3
pycparser               2.20
Pygments                2.6.1
PyNaCl                  1.4.0
pyOpenSSL               19.1.0
pyparsing               2.4.7
pyrsistent              0.17.3
PySocks                 1.7.1
```

```
python-dateutil         2.8.1
pytz                    2020.1
PyWavelets              1.1.1
PyYAML                  5.3.1
pyzmq                   22.0.3
requests                2.23.0
requests-oauthlib       1.3.0
retrying                1.3.3
rope                    0.18.0
rsa                     4.7.2
ruamel-yaml             0.15.87
s3transfer              0.3.6
sagemaker-training      3.7.3
scikit-image            0.18.1
scikit-learn            0.24.1
scipy                   1.6.2
seaborn                 0.11.1
Send2Trash              1.5.0
setuptools              46.4.0.post20200518
Shapely                 1.7.1
six                     1.14.0
sniffio                 1.2.0
soupsieve               2.0.1
tensorboard             2.4.1
tensorboard-plugin-wit 1.8.0
terminado               0.9.3
testpath                0.4.4
threadpoolctl           2.1.0
tifffile                2021.3.17
toml                    0.10.2
torch                   1.6.0
torch-tb-profiler       0.1.0
torchvision             0.7.0
tornado                 6.1
tqdm                    4.46.0
traitlets               4.3.3
typing-extensions       3.7.4.3
urllib3                 1.25.8
wcwidth                 0.2.5
webencodings            0.5.1
```

```
Werkzeug                1.0.1
wheel                   0.34.2
widgetsnbextension      3.5.1
zipp                    3.4.1
zope.event              4.5.0
zope.interface          5.3.0
```