노트북

Face Recognition

- Tasks
  1. Data Preprocessing
     - 모델 사용을 위해 데이터를 일관성 있게 정리해야 합니다.
  2. Object Detection
     - 전처리 된 데이터를 이용하여 학습과 추론을 진행하세요.
  3. Extra
     - 조원들의 얼굴 이미지를 수집한 후, 조원들을 구분하는 모델을 만들어보세요.

Google Drive 연동

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

데이터셋 압축 해제

- Face_Recognition.zip : 이미지 데이터 압축 파일

```
import os, glob, shutil, zipfile
from tqdm import tqdm
```

```
def dataset_extract(file_name) :
    with zipfile.ZipFile(file_name, 'r') as zip_ref :
        file_list = zip_ref.namelist()

        if os.path.exists(f'./{file_name[-20:-4]}/') :
            print(f'데이터셋 폴더가 이미 존재합니다.')
            return

        else :
            for f in tqdm(file_list, desc='Extracting'
                zip_ref.extract(member=f, path=f'./{fi


dataset_path = '/content/drive/MyDrive/program/miniPJT


dataset_extract(dataset_path)
```

```
Extracting: 100%|██████████| 11960/11960 [00:06<00
```

데이터셋 살펴보기

```
# 데이터셋 폴더 내에 있는 모든 파일의 경로 가져오기
dataset_folder = '/content/Face_Recognition/train' # 실
num_files = len(os.listdir(dataset_folder))
print("데이터셋 이미지파일의 개수:", num_files//2)
```

```
데이터셋 이미지파일의 개수: 5214
```

폴더 생성

- YOLO 모델에서 요구하는 폴더의 형식이 있습니다.
- 해당 형식에 맞춰 폴더를 만드세요.

```
dataset_img = '/content/Face_Recognition/train/images/
dataset_label = '/content/Face_Recognition/train/label
datavlid_img = '/content/Face_Recognition/valid/images
datavlid_label = '/content/Face_Recognition/valid/labe

data_path = [dataset_img, dataset_label, datavlid_img,

# 폴더 생성
for path in data_path:
  os.makedirs(path)
```

폴더에 맞게 파일 이동

- 이전 단계에서 만든 하위 폴더에 이미지 파일과 텍스트 파일을 이동시키세요.

```
dataset= '/content/Face_Recognition/train/'
datavlid = '/content/Face_Recognition/valid/'

#glob.glob('/content/Face_Recognition/train/*jpg') # j
#shutil.move(os.path.join(dataset, img), os.path.join(
# train-이미지
mg_files_jpg = glob.glob(os.path.join(dataset, '*.jpg'
mg_files_txt = glob.glob(os.path.join(dataset, '*.txt'
mg_files_jpg_valid = glob.glob(os.path.join(datavlid,
mg_files_txt_valid = glob.glob(os.path.join(datavlid,

# dataset_img로 각 .jpg 파일을 이동
for img_file in mg_files_jpg:
    shutil.move(img_file, dataset_img)

# dataset_label로 각 .txt 파일을 이동
for img_file in mg_files_txt:
    shutil.move(img_file, dataset_label)

# valid_img로 각 .jpg 파일을 이동
for img_file in mg_files_jpg_valid:
    shutil.move(img_file, datavlid_img)

# valid_label로 각 .txt 파일을 이동
for img_file in mg_files_txt_valid:
    shutil.move(img_file, datavlid_label)
```

YOLO 모델에 적용할 YAML 생성하기

- 클래스는 총 13개입니다. 클래스 순서에 주의하세요.
- 'Barack Obama', 'Che Guevara', 'Cristiano Ronaldo', 'David Beckham', 'Donald Trump', 'Elon Musk', 'Joe Biden', 'Lionel Messi', 'Mark Zuckerberg', 'Oprah Winfrey', 'Steve Harvey', 'Steve Jobs', 'Zaha Hadid'

```
import yaml
```

```
## yaml 파일 읽고 수정 필요
# yaml 형식에 맞는 데이터
data = {
    'train': './train/images',
    'val': './valid/images',
    'nc': 13,
    'names': [
        'Barack Obama',
        'Che Guevara',
        'Cristiano Ronaldo',
        'David Beckham',
        'Donald Trump',
        'Elon Musk',
        'Joe Biden',
        'Lionel Messi',
        'Mark Zuckerberg',
        'Oprah Winfrey',
        'Steve Harvey',
        'Steve Jobs',
        'Zaha Hadid'
    ]
}

# yaml 파일 생성
with open('/content/Face_Recognition/famous_human.yaml
    yaml.dump(data, f) # data를 yaml 형식으로 변환하여
```

YOLO v8 모델 사용

- yaml 파일의 경로 설정에 주의하세요.

```
!pip install ultralytics
```

```
Collecting ultralytics
  Downloading ultralytics-8.2.5-py3-none-any.wh
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 7
Requirement already satisfied: matplotlib>=3.3.
Requirement already satisfied: opencv-python>=4
Requirement already satisfied: pillow>=7.1.2 ir
Requirement already satisfied: pyyaml>=5.3.1 ir
Requirement already satisfied: requests>=2.23.0
Requirement already satisfied: scipy>=1.4.1 in
Requirement already satisfied: torch>=1.8.0 in
Requirement already satisfied: torchvision>=0.9
Requirement already satisfied: tqdm>=4.64.0 in
Requirement already satisfied: psutil in /usr/l
Requirement already satisfied: py-cpuinfo in /u
Collecting thop>=0.1.1 (from ultralytics)
  Downloading thop-0.1.1.post2209072238-py3-nor
Requirement already satisfied: pandas>=1.1.4 ir
```

```
Requirement already satisfied: seaborn>=0.11.0
Requirement already satisfied: contourpy>=1.0.1
Requirement already satisfied: cycler>=0.10 in
Requirement already satisfied: fonttools>=4.22.
Requirement already satisfied: kiwisolver>=1.0.
Requirement already satisfied: numpy>=1.20 in /
Requirement already satisfied: packaging>=20.0
Requirement already satisfied: pyparsing>=2.3.1
Requirement already satisfied: python-dateutil>
Requirement already satisfied: pytz>=2020.1 in
Requirement already satisfied: tzdata>=2022.1 i
Requirement already satisfied: charset-normaliz
Requirement already satisfied: idna<4,>=2.5 in
Requirement already satisfied: urllib3<3,>=1.21
Requirement already satisfied: certifi>=2017.4.
Requirement already satisfied: filelock in /usr
Requirement already satisfied: typing-extensior
Requirement already satisfied: sympy in /usr/lo
Requirement already satisfied: networkx in /usr
Requirement already satisfied: jinja2 in /usr/l
Requirement already satisfied: fsspec in /usr/l
Collecting nvidia-cuda-nvrtc-cu12==12.1.105 (fr
  Using cached nvidia_cuda_nvrtc_cu12-12.1.105-
Collecting nvidia-cuda-runtime-cu12==12.1.105 (
  Using cached nvidia_cuda_runtime_cu12-12.1.10
Collecting nvidia-cuda-cupti-cu12==12.1.105 (fr
  Using cached nvidia_cuda_cupti_cu12-12.1.105-
Collecting nvidia-cudnn-cu12==8.9.2.26 (from to
  Using cached nvidia_cudnn_cu12-8.9.2.26-py3-r
Collecting nvidia-cublas-cu12==12.1.3.1 (from t
  Using cached nvidia_cublas_cu12-12.1.3.1-py3-
Collecting nvidia-cufft-cu12==11.0.2.54 (from t
  Using cached nvidia_cufft_cu12-11.0.2.54-py3-
Collecting nvidia-curand-cu12==10.3.2.106 (from
  Using cached nvidia_curand_cu12-10.3.2.106-py
Collecting nvidia-cusolver-cu12==11.4.5.107 (fr
  Using cached nvidia_cusolver_cu12-11.4.5.107-
Collecting nvidia-cusparse-cu12==12.1.0.106 (fr
  Using cached nvidia_cusparse_cu12-12.1.0.106-
Collecting nvidia-nccl-cu12==2.19.3 (from torch
```

```
from ultralytics import settings, YOLO
```

```
settings
```

```
{'settings_version': '0.0.4',
 'datasets_dir': '/content/datasets',
 'weights_dir': 'weights',
 'runs_dir': 'runs',
 'uuid':
'569f3ba64b326db489132663f79cd37279811de477381b83a
 'sync': True,
 'api_key': '',
 'openai_api_key': '',
```

```
 'clearml': True,
 'comet': True,
 'dvc': True,
 'hub': True,
 'mlflow': True,
 'neptune': True,
 'raytune': True,
 'tensorboard': True,
 'wandb': True}
```

```
settings['datasets_dir'] = '/content/'
```

```
settings
```

```
{'settings_version': '0.0.4',
 'datasets_dir': '/content/',
 'weights_dir': 'weights',
 'runs_dir': 'runs',
 'uuid':
'569f3ba64b326db489132663f79cd37279811de477381b83a
 'sync': True,
 'api_key': '',
 'openai_api_key': '',
 'clearml': True,
 'comet': True,
 'dvc': True,
 'hub': True,
 'mlflow': True,
 'neptune': True,
 'raytune': True,
 'tensorboard': True,
 'wandb': True}
```

```
from keras.callbacks import EarlyStopping, ModelCheckp

model = YOLO(model='yolov8s.pt', task='detect') # task
                                                # mode

es = EarlyStopping(monitor='val_loss', patience=5)
mc = ModelCheckpoint('best.pt', monitor='val_loss', sa
```

```
Downloading https://github.com/ultralytics/assets/
100%|██████████| 21.5M/21.5M [00:00<00:00, 343MB/s
```

```
model.train(model = '/content/yolov8s.pt',
            data='/content/Face_Recognition/famous_hum
            epochs=100,
            verbose=True,
            patience=15,       # mAP[:50] : mAP[50:95]
            seed=2024,
            pretrained=True,
            hsv_h=0,
            hsv_s=0,
            hsv_v=0,
            translate=0,
            scale=0,
            fliplr=0,
            mosaic=0,
            erasing=0,
            crop_fraction=0
            )
```

```
Ultralytics YOLOv8.2.5 🚀 Python-3.10.12 torch
engine/trainer: task=detect, mode=train, model=
Downloading https://ultralytics.com/assets/Aria
100%|██████████| 755k/755k [00:00<00:00, 137MB/

                   from  n    params  module
  0                  -1  1       928  ultralyti
  1                  -1  1     18560  ultralyti
  2                  -1  1     29056  ultralyti
  3                  -1  1     73984  ultralyti
  4                  -1  2    197632  ultralyti
  5                  -1  1    295424  ultralyti
  6                  -1  2    788480  ultralyti
  7                  -1  1   1180672  ultralyti
  8                  -1  1   1838080  ultralyti
  9                  -1  1    656896  ultralyti
 10                  -1  1         0  torch.nn.
 11             [-1, 6]  1         0  ultralyti
 12                  -1  1    591360  ultralyti
 13                  -1  1         0  torch.nn.
 14             [-1, 4]  1         0  ultralyti
 15                  -1  1    148224  ultralyti
 16                  -1  1    147712  ultralyti
 17            [-1, 12]  1         0  ultralyti
 18                  -1  1    493056  ultralyti
 19                  -1  1    590336  ultralyti
 20             [-1, 9]  1         0  ultralyti
 21                  -1  1   1969152  ultralyti

 22        [15, 18, 21]  1   2121079  ultralyti
Model summary: 225 layers, 11140631 parameters,

Transferred 349/355 items from pretrained weigh
TensorBoard: Start with 'tensorboard --logdir r
Freezing layer 'model.22.dfl.conv.weight'
AMP: running Automatic Mixed Precision (AMP) ch
```

```
Downloading https://github.com/ultralytics/asse
100%|██████████| 6.23M/6.23M [00:00<00:00, 305M
AMP: checks passed ✅
train: Scanning /content/Face_Recognition/train
train: New cache created: /content/Face_Recogni
albumentations: Blur(p=0.01, blur_limit=(3, 7))
/usr/lib/python3.10/multiprocessing/popen_fork.
  self.pid = os.fork()
val: Scanning /content/Face_Recognition/valid/l

Plotting labels to runs/detect/train/labels.jpg
optimizer: 'optimizer=auto' found, ignoring 'lr
optimizer: AdamW(lr=0.000588, momentum=0.9) wit
TensorBoard: model graph visualization added ✅
Image sizes 640 train, 640 val
Using 8 dataloader workers
Logging results to runs/detect/train
Starting training for 100 epochs...

      Epoch    GPU_mem   box_loss   cls_loss
```

```
import locale
locale.getpreferredencoding = lambda: "UTF=8"
```

```
result = model.predict(source='/content/Face_Recogniti
                       conf=0.5,          # confidence
                       iou=0.6,           # IoU가 0.7
                       save=True,         # 예측된 이미
                       line_width=2       # 그려지는 박
                       )
```

```
for r in result :
    boxes = r.boxes
```

Extra) 조원들의 얼굴을 구분하는지 테스트 해보자!

1. 데이터 수집 : 조원들의 얼굴 사진을 수집한다! -> 이미지 파일
2. Annotation 작업 : 얼굴 부위에 대해 라벨링을 한다! -> txt 파일
3. 본인 구글 드라이브에 업로드 및 Colab과의 연동!
4. YAML 파일 생성 : 조원들의 사진이 있는 경로와 클래스 수, 클래스 이름 정보가 있는 YAML을 새로 생성!

5. 추가 학습 : 위의 단계에서 생성된 최적의 가중치를 가진 모델을 불러와 추가 학습을 시킨다!
6. 테스트 : 조원들을 잘 인식하는지 predict 해본다!

YAML 새로 생성하기

- 클래스의 수는 조원의 수입니다.
- Annotation 작업 과정에서 정해진 클래스의 순서 규칙을 따라주세요.

```
import os, glob, shutil, zipfile
from tqdm import tqdm


def dataset_extract(file_name) :
    with zipfile.ZipFile(file_name, 'r') as zip_ref :
        file_list = zip_ref.namelist()

        if os.path.exists(f'./{file_name[-20:-4]}/') :
            print(f'데이터셋 폴더가 이미 존재합니다.')
            return

        else :
            for f in tqdm(file_list, desc='Extracting'
                zip_ref.extract(member=f, path=f'./{fi


dataset_path = '/content/drive/MyDrive/program/miniPJT



dataset_extract(dataset_path)
```

```
Extracting: 100%|██████████| 2348/2348 [00:34<00:0
```

```
# 데이터셋 폴더 내에 있는 모든 파일의 경로 가져오기
dataset_folder = '/content/— 5차_4일차/16조/16non/trai
num_files = len(os.listdir(dataset_folder))
print("데이터셋 이미지파일의 개수:", num_files)
```

```
데이터셋 이미지파일의 개수: 941
```

```
# 데이터셋 폴더 내에 있는 모든 파일의 경로 가져오기
dataset_folder = '/content/— 5차_4일차/16조/16ㅐ/test
num_files = len(os.listdir(dataset_folder))

import yaml
```

```
데이터셋 이미지파일의 개수: 230
```

```
## yaml 파일 읽고 수정 필요
# yaml 형식에 맞는 데이터
data = {
    'train': './train/images',
    'val': './test/images',
    'nc': 6,
    'names': [
        'SIJ',
        'LCH',
        'KJE',
        'KHS',
        'YMS',
        'YBH'
    ]
}

# yaml 파일 생성
with open('/content/— 5차_4일차/16조/16ㅐ/16_team.yam
    yaml.dump(data, f) # data를 yaml 형식으로 변환하여


!pip install ultralytics
```

```
Successfully installed nvidia-cublas-cu12-12.1.3.1
```

```
from ultralytics import settings, YOLO


settings
```