((실습)) YOLO5를 이용한 마스크 인식

[1] Mask 데이터 준비

ailab.silla.ac.kr/lec/pipaai/roboflow.zip 마스크데이터셋 다운로드 하기

(원본데이터 출처: https://public.roboflow.com/object-detection/mask-wearing/)

윈도우즈 탐색기로 c:\ai\mask\datasets\roboflow 폴더에 압축 풀기

[2] Anaconda Prompt에서 가상환경 설정

c:

cd \ai

md mask

cd mask

conda create -n yolov5mask git

(base) conda activate yolov5mask

[3] 관련 패키지 설치

(yolov5mask) c:\ai\mask> git clone https://github.com/ultralytics/yolov5.git

(((※ ai\mask 폴더 아래에 datasets 폴더와 새로 다운받은 yolov5 폴더가 같이 존재해야 함)))

install Lib for YOLOv5

(yolov5mask) c:\ai\mask>conda install -c anaconda cython numpy pillow scipy seaborn pandas requests

(yolov5mask) c:\ai\mask>conda install -c conda-forge matplotlib pyyaml tensorboard tqdm opencv

install pytorch (GPU 설치시)

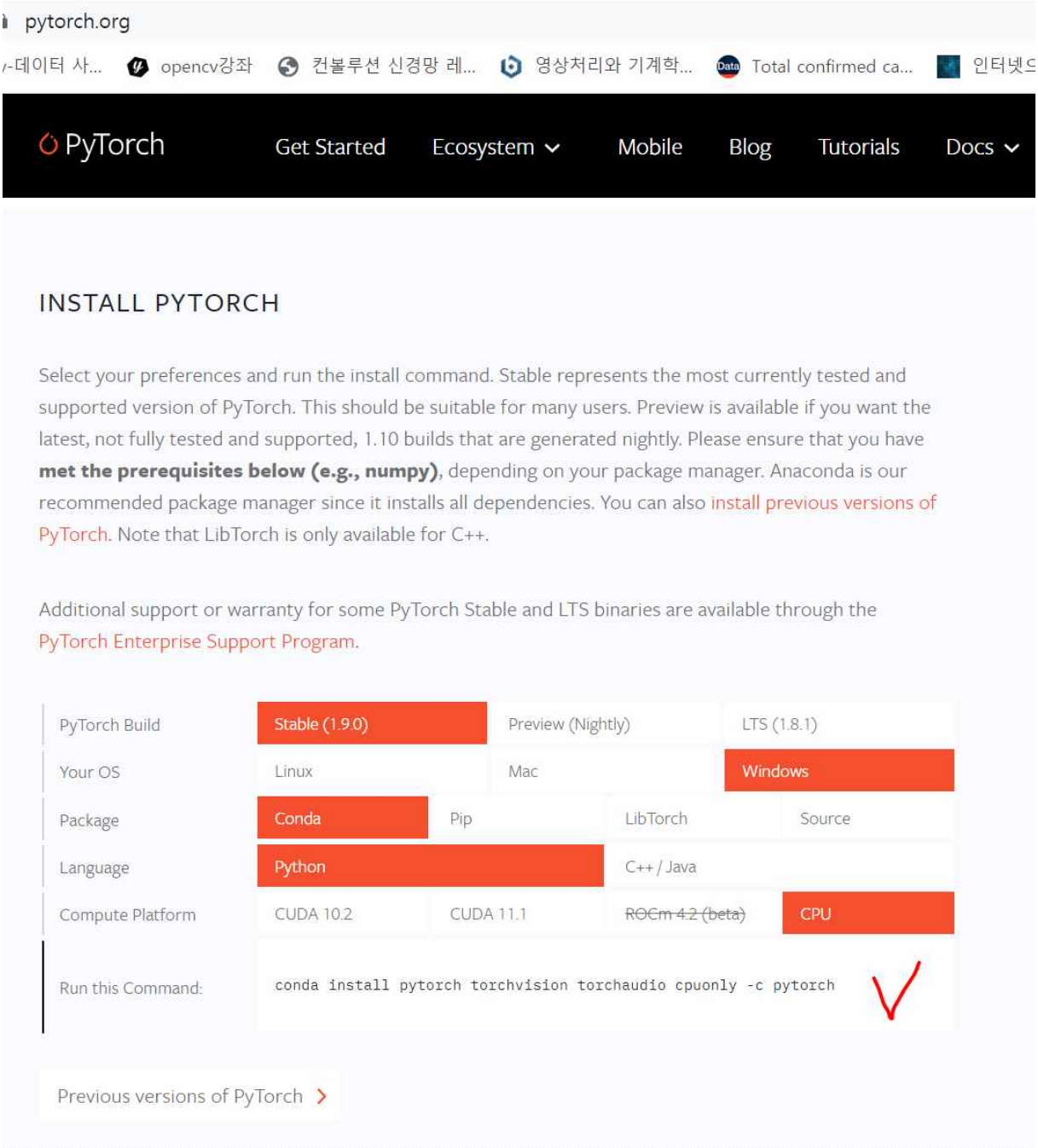
(yolov5mask) c:\ai\mask>conda install pytorch torchvision torchaudio **cudatoolkit=10.2** -c pytorch

install pytorch (GPU 미설치시)

(yolov5mask) c:\ai\mask>conda install pytorch torchvision torchaudio **cpuonly** -c pytorch

Extra

(yolov5mask) c:\ai\mask>conda install -c conda-forge onnx



[4] roboflow.yaml 파일 다운받아 설정

학습용과 검증용 이미지 파일위치정보를 담은 yaml 파일 다운받아 설정

ailab 사이트에서 다운받은 roboflow.yaml 파일을, ai\mask\yolov5\data 폴더에 저장한다.

roboflow.yaml | coco128.yaml | train.py 5

D: > myprojects > yolov51 > data > roboflow.yaml

```yaml
# COCO 2017 dataset http://cocodataset.org - first 128 training images
# Train command: python train.py --data coco128.yaml
# Default dataset location is next to YOLOv5:
#   /parent
#     /datasets/coco128
#     /yolov5


# Train/val/test sets as 1) dir: path/to/imgs, 2) file: path/to/imgs.txt, or 3) li
path: ../datasets/roboflow  # dataset root dir
train: images/train  # train images (relative to 'path') 128 images
val: images/valid  # val images (relative to 'path') 128 images
test:  images/test # test images (optional)

# Classes
nc: 2  # number of classes
names: [ 'NO mask', 'Mask' ]  # class names


# Download script/URL (optional)
#download: https://github.com/ultralytics/yolov5/releases/download/v1.0/coco128.zi
```

[5] 학습하기

python train.py --img 640 --batch 4 --epochs 40 --data data/roboflow.yaml --cfg models/yolov5s.yaml --weights yolov5s.pt

관리자: Anaconda Prompt (Anaconda30) - conda install -c anaconda cython numpy pillow scipy seaborn pandas requests - conda install -c conda...

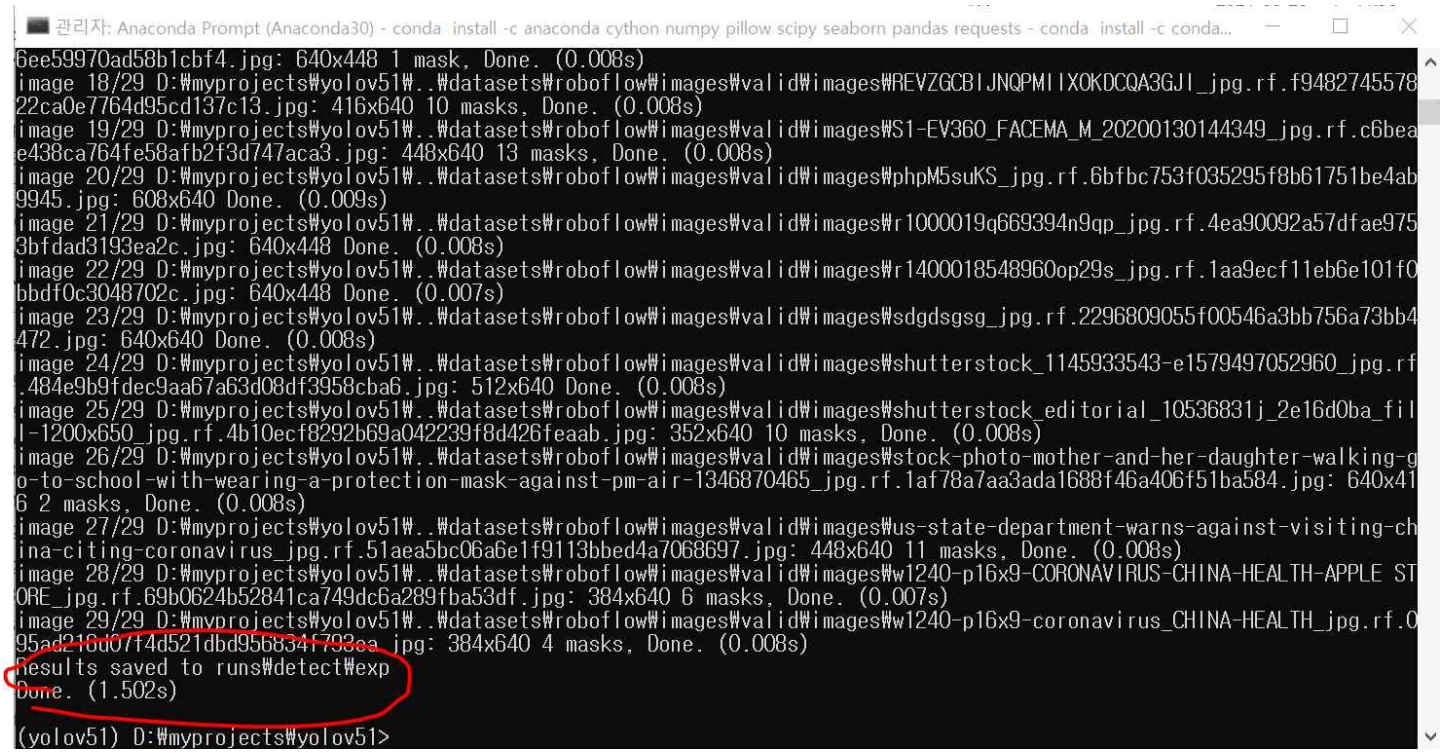
```
               Class     Images     Labels          P          R     mAP@.5 mAP@.5:.95: 100%|| 4/4 [00:00<00:00,  8.32
                 all         29        162      0.109      0.144     0.0754     0.0182

     Epoch   gpu_mem       box       obj       cls     total    labels  img_size
       6/9     1.26G   0.08786   0.09218   0.02163    0.2017         2       640: 100%| 27/27 [00:02<00:00,  9.99it/s
               Class     Images     Labels          P          R     mAP@.5 mAP@.5:.95: 100%| 4/4 [00:00<00:00,  8.30
                 all         29        162      0.128       0.18     0.0971     0.0213

     Epoch   gpu_mem       box       obj       cls     total    labels  img_size
       7/9     1.26G   0.08282   0.08994   0.02103    0.1938         3       640: 100%| 27/27 [00:02<00:00,  9.92it/s
               Class     Images     Labels          P          R     mAP@.5 mAP@.5:.95: 100%| 4/4 [00:00<00:00,  8.52
                 all         29        162      0.685      0.162      0.131      0.031

     Epoch   gpu_mem       box       obj       cls     total    labels  img_size
       8/9     1.26G   0.07965   0.09079   0.02022    0.1907        13       640: 100%| 27/27 [00:02<00:00, 10.01it/s
               Class     Images     Labels          P          R     mAP@.5 mAP@.5:.95: 100%| 4/4 [00:00<00:00,  8.89
                 all         29        162      0.698      0.183      0.154     0.0409

     Epoch   gpu_mem       box       obj       cls     total    labels  img_size
       9/9     1.26G   0.07709   0.08271   0.01959    0.1794        11       640: 100%| 27/27 [00:02<00:00,  9.84it/s
               Class     Images     Labels          P          R     mAP@.5 mAP@.5:.95: 100%| 4/4 [00:00<00:00,  5.37
                 all         29        162      0.726      0.162      0.174     0.0492
                mask         29        142      0.453      0.324      0.311     0.0879
              NoMask         29         20          1          0     0.0361     0.0106
10 epochs completed in 0.011 hours.

Optimizer stripped from runs\train\exp2\weights\last.pt, 14.4MB
Optimizer stripped from runs\train\exp2\weights\best.pt, 14.4MB
```

[6-1] 기존(다운로드) 학습한 정보로 실행

python detect.py —source ./inference/images/ --weights yolov5s.pt —conf 0.2

[6-2] 새로이 학습한 정보로 실행 (폴더내 이미지 입력시)

python detect.py --source ../datasets/roboflow/images/valid/images/ --weights runs/train/exp?/weights/best.pt --conf 0.2



결과는 runs\detect\exp? 폴더에 저장

[6-3] 새로이 학습한 정보로 실행 (웹캠 입력시)

python detect.py --source 0 --weights runs/train/exp?/weights/best.pt --conf 0.2

※ 마스크 착용한 데이터의 비중이 큼

((실습)) YOLO5를 이용한 차량 및 번호판 인식

[1] 차량 데이터 학습

학습사이트에서 cars.zip 차량 이미지데이터와 annotation파일들을 다운로드 하여 datasets/car 폴더에 압축을 푼다

학습사이트에서 car.yaml을 다운로드하여 yolov5\data폴더에 저장한다

car.yaml × train.py ●

D: > myprojects > yolov51 > data > ! car.yaml

```
1   # COCO 2017 dataset http://cocodataset.org - first 128 training images
2   # Train command: python train.py --data coco128.yaml
3   # Default dataset location is next to YOLOv5:
4   #   /parent
5   #     /datasets/coco128
6   #     /yolov5
7
8
9   # Train/val/test sets as 1) dir: path/to/imgs, 2) file: path/to/imgs.txt, or 3) list: [pat
10  path: ../datasets/car  # dataset root dir
11  train: images/train  # train images (relative to 'path') 128 images
12  val: images/valid  # val images (relative to 'path') 128 images
13  test:  images/test # test images (optional)
14
15  # Classes
16  nc: 2  # number of classes
17  names: [ 'car', 'plate' ]  # class names
18
19
20  # Download script/URL (optional)
21  #download: https://github.com/ultralytics/yolov5/releases/download/v1.0/coco128.zip
```

labelimg로 데이터에 대한 annotation파일을 구한다

아래 명령어로 학습한다.
python train.py --img 416 --batch 4 --epochs 20 --data data/car.yaml --cfg models/yolov5m.yaml —weights yolov5m.pt

아래 명령어로 valid 폴더아래 이미지에 대하여 Test한다
python detect.py --source ../datasets/car/images/valid --weights runs/train/exp14/weights/best.pt --conf 0.1