

Deep Learning : Helmet Detection

딥러닝 오픈소스를 활용한 안전모 착용 감지 시스템

목차

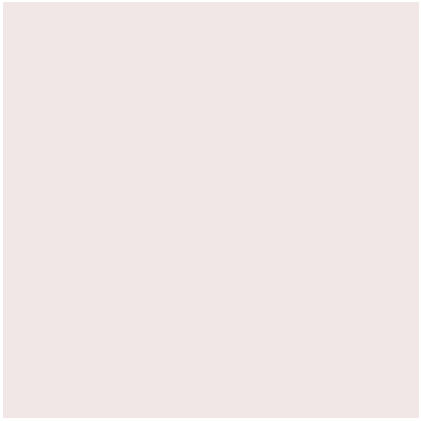
A table of contents

1 Aim & Background

2 Contents

3 Appendix

4 Reference

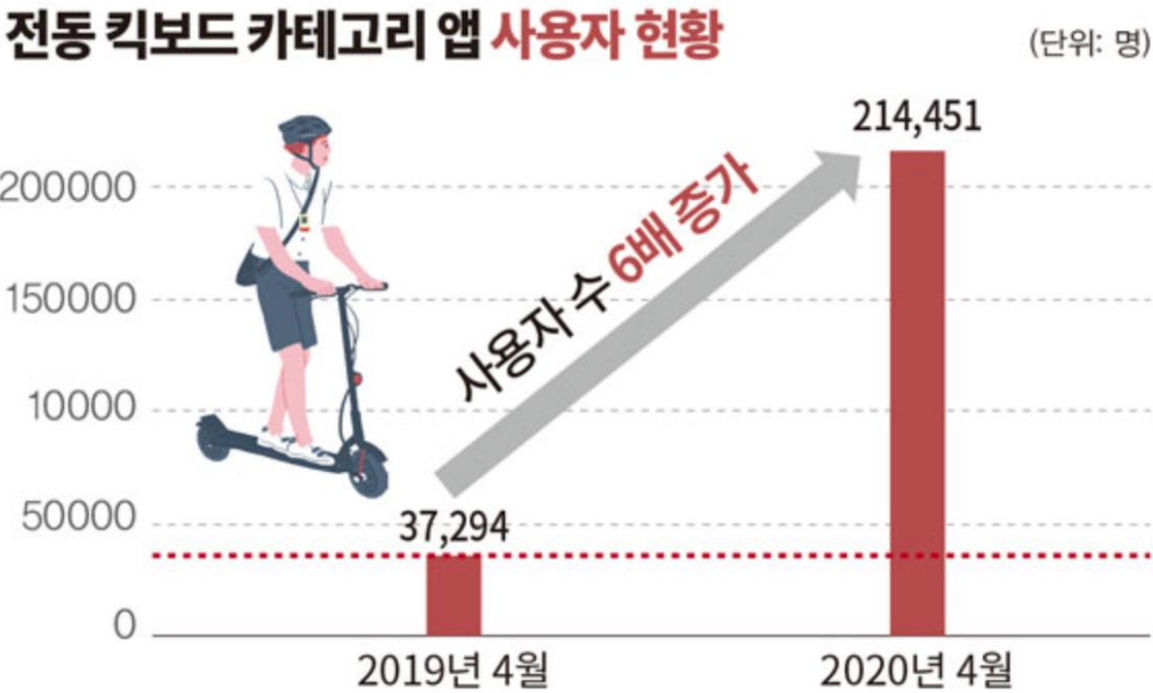


Part 1, Aim & Background

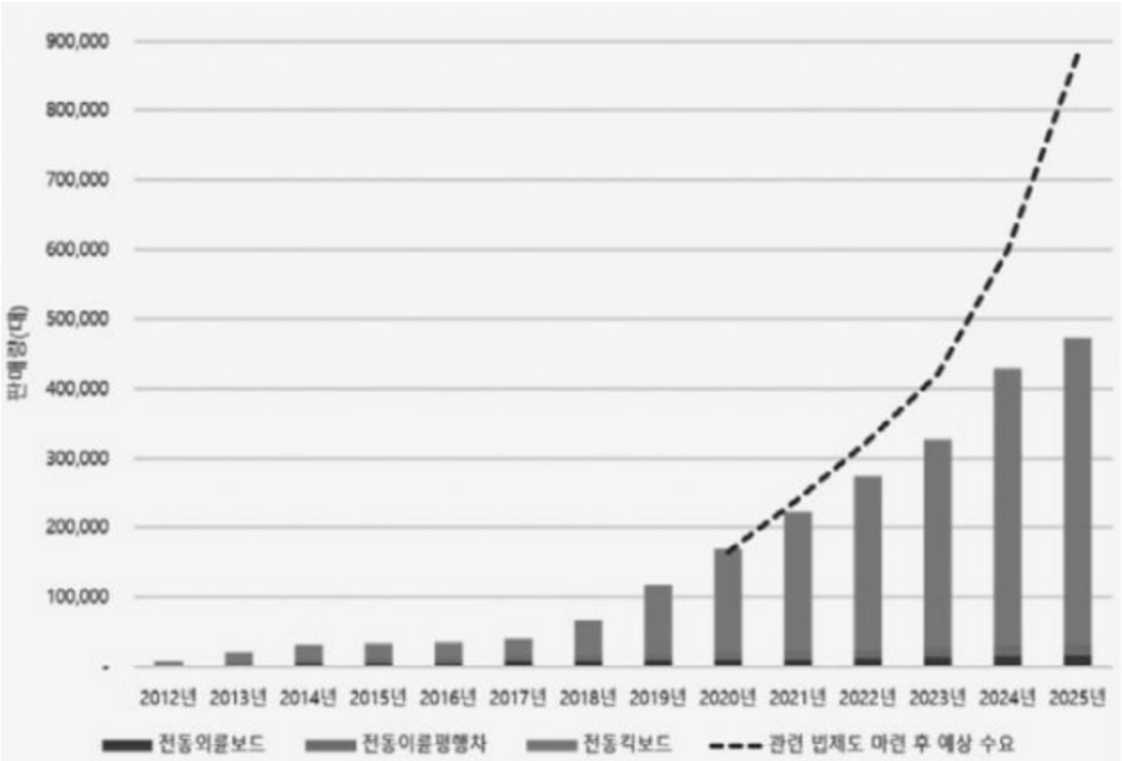
프로젝트 목표와 배경



1.1 안전모 착용의 중요성

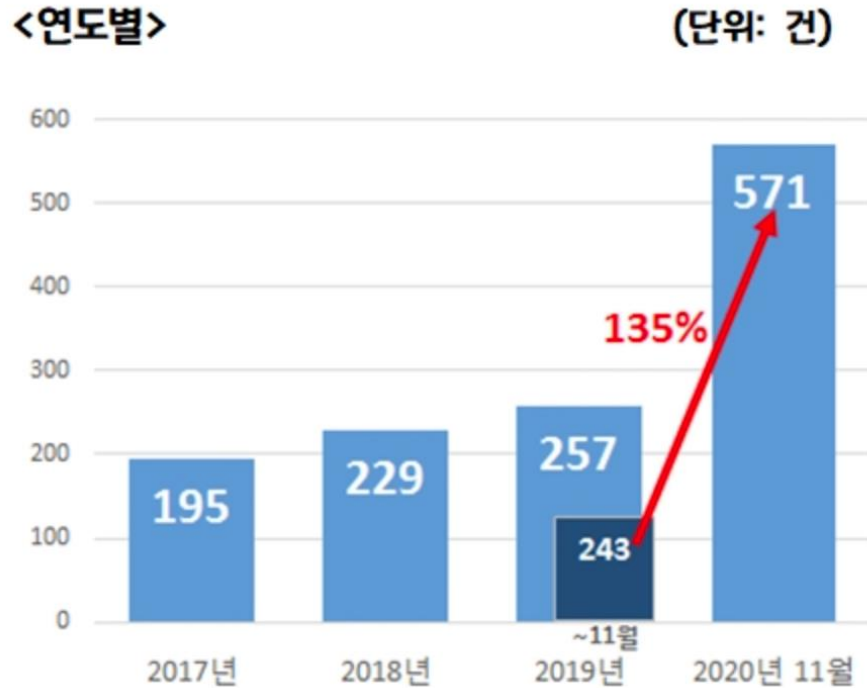


전동 킥보드 사용자 현황.



'퍼스널 모빌리티 현황 및 쟁점사항'

1.1 안전모 착용의 중요성

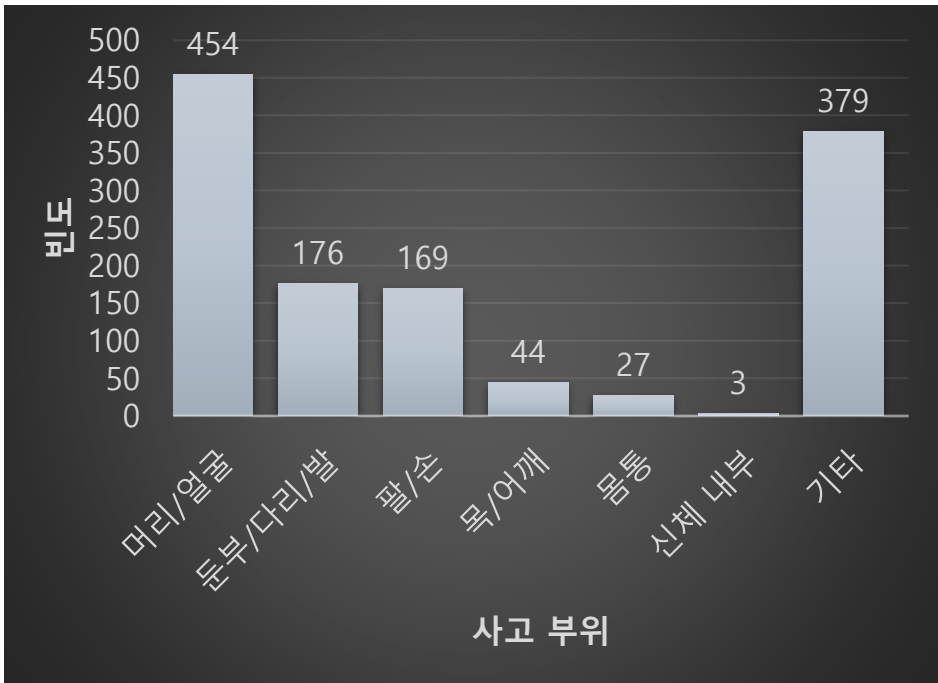


19' - 20' 수요자가 급증하며
사고율이 135% 증가

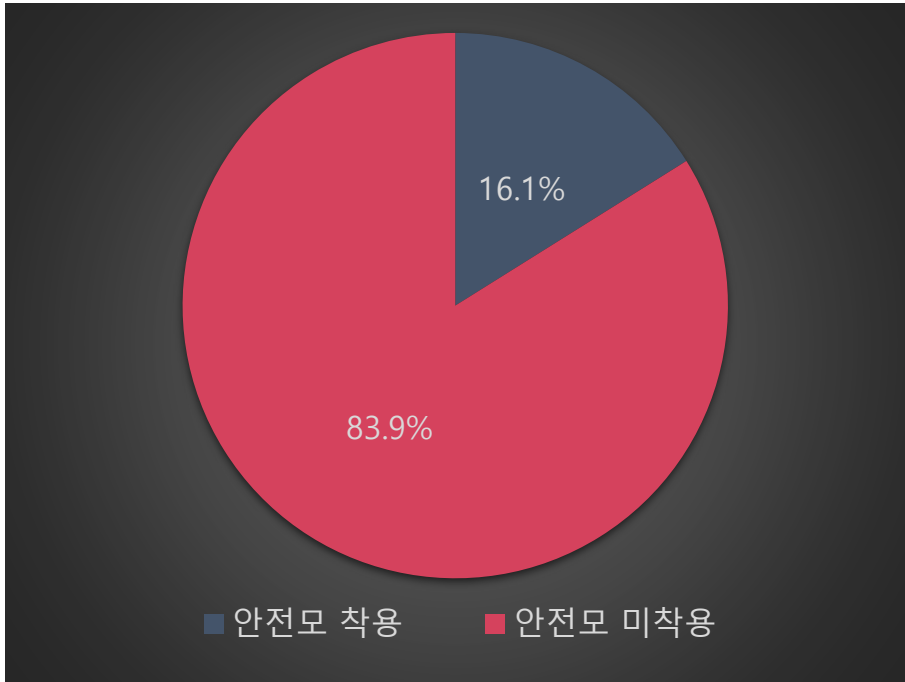
< 최근 3년 11개월('17~'20.11월)간 전동킥보드 사고 현황 >

1.1 안전모 착용의 중요성

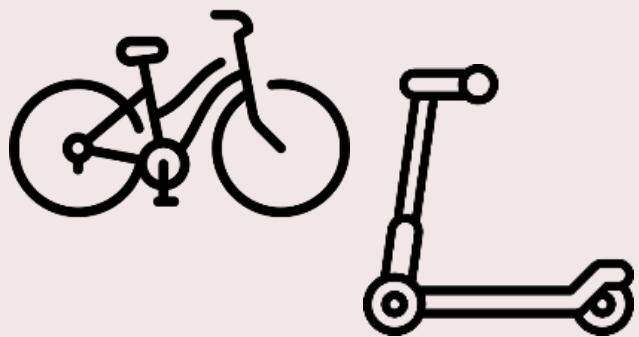
사고에 따른 위해부위별 빈도



안전모 착용 준수율(~5/26)



1



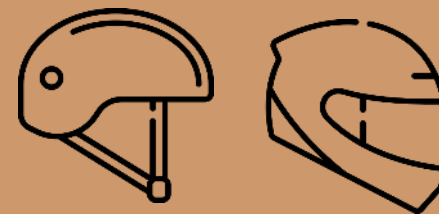
개인형 이동장치 대상

2



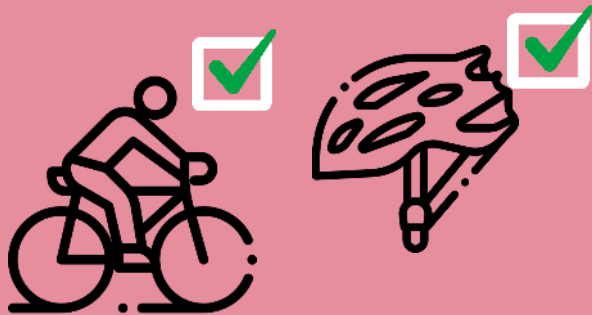
운전자 탑승

3



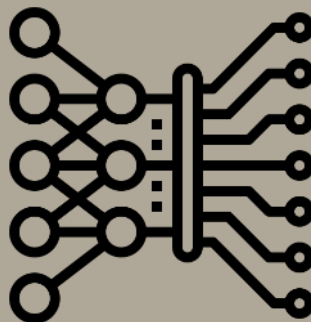
안전모 착용 확인

4



운전자와 안전모를 식별

5

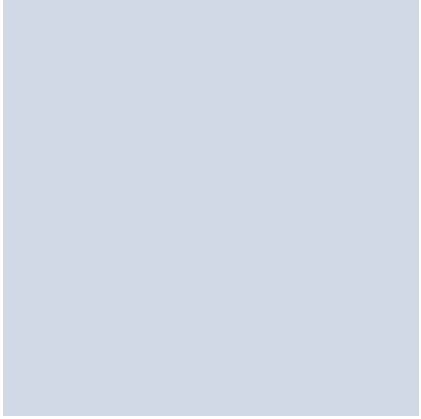


학습된 데이터셋

6

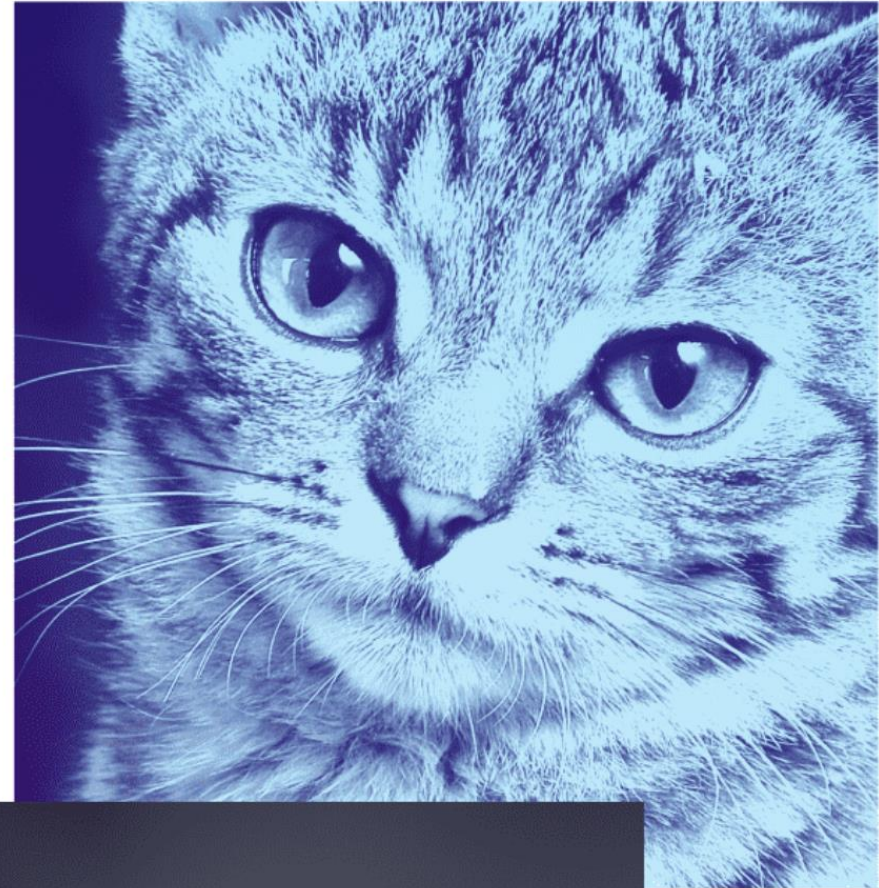


경고음 출력

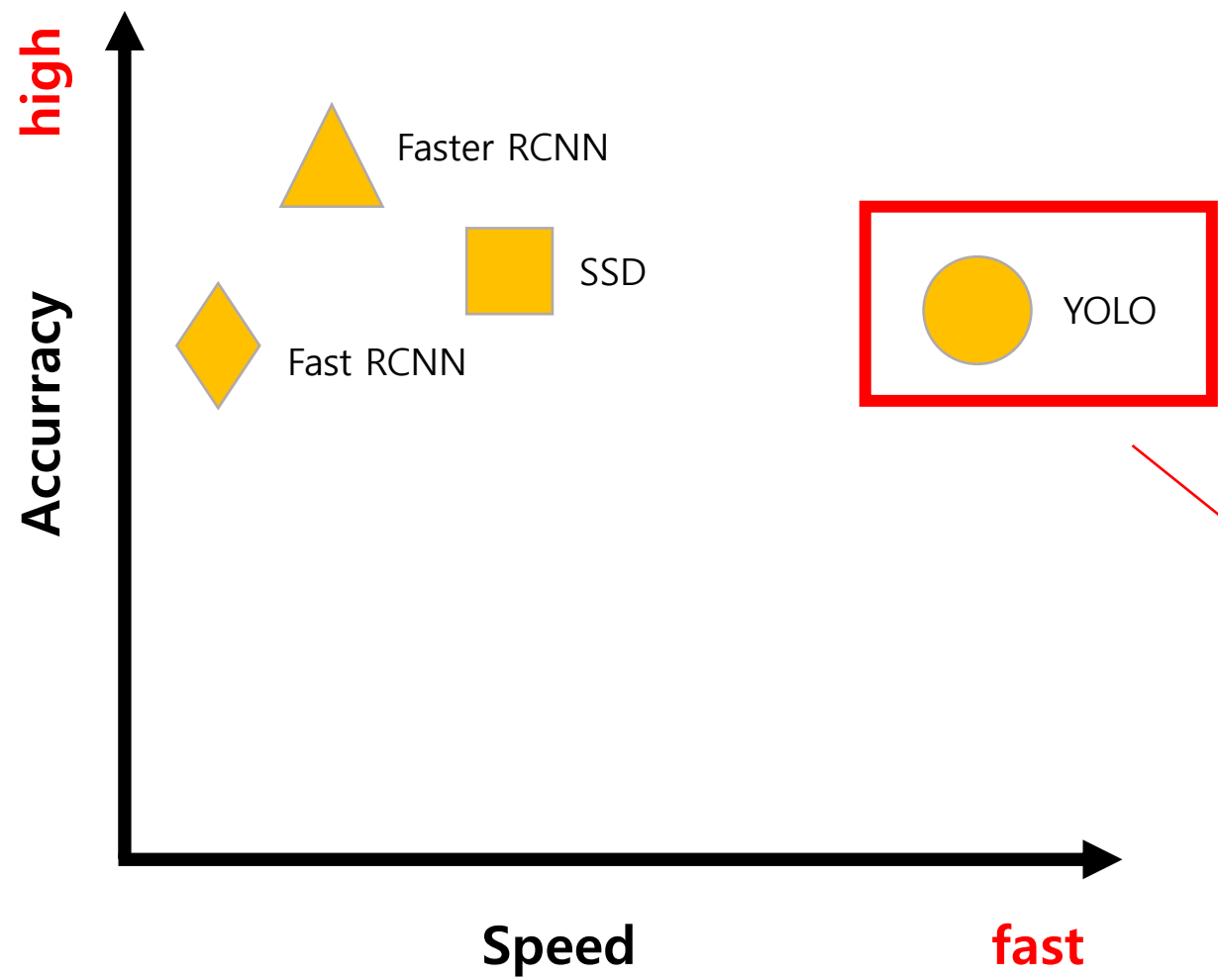


Part 2, Contents

cat



You Only Look Once
[YOLO] Implementation



Accuracy : normal
Speed : fast

Bounding image





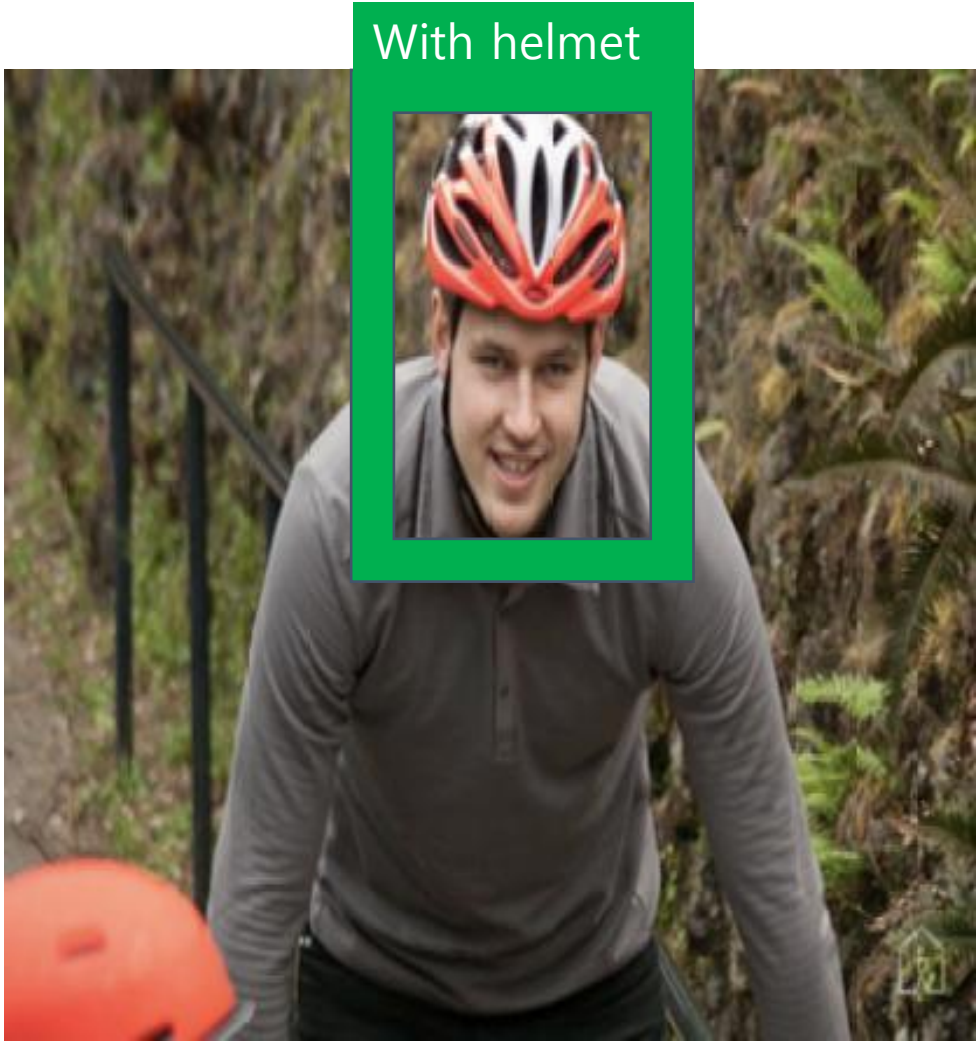




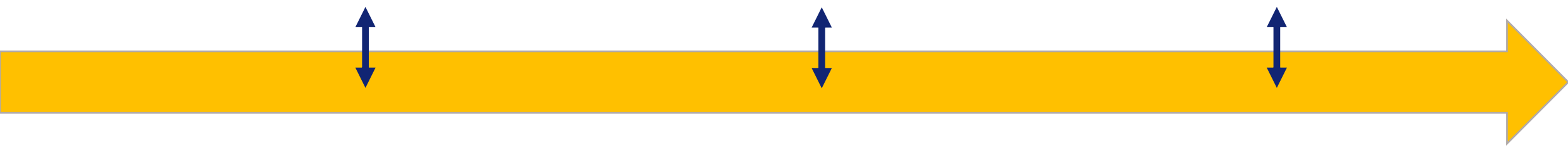
Image data
전처리



Without helmet
With helmet



Image detected
Video detected
Realtime detected





kaggle


Image data 수집



Python labeling.py 실행

Anaconda Prompt (Anaconda3) - python labeling.py
labelimg.py:1026: DeprecationWarning: an integer is required (got type float). Implicit conversion to integers using __int__ is deprecated, and may be removed in a future version of Python.
v_bar.setValue(new_v_bar_value)

Open Dir
Change Save Dir
Next Image
Prev Image
Verify Image
Save
yolo
YOLO
Create RectBox
Duplicate RectBox



☐ Use default label

helmet
person

h1 - Windows 메모장
파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)
15 0.486111 0.211806 0.618519 0.256944
1 0.506944 0.528819 0.986111 0.942361
Ln 1, Col 1 100% Unix (LF) UTF-8

File List
Object labeling 확인

Width: 660 Height: 661 / X: 6 Y: 135

Create RectBox로 Object의
영역을 Bounding

Data labeling

2.3 Train – 데이터 학습

data.yaml

```
Detection.pptx | data.yaml | README.md M
lab1 > src > ! data.yaml
1  train: ../train/images
2  val:  ../valid/images
3
4  nc: 2
5  names: [['With Helmet', 'Without Helmet']]
```

확인할 class의 개수와 이름

Dataset 경로

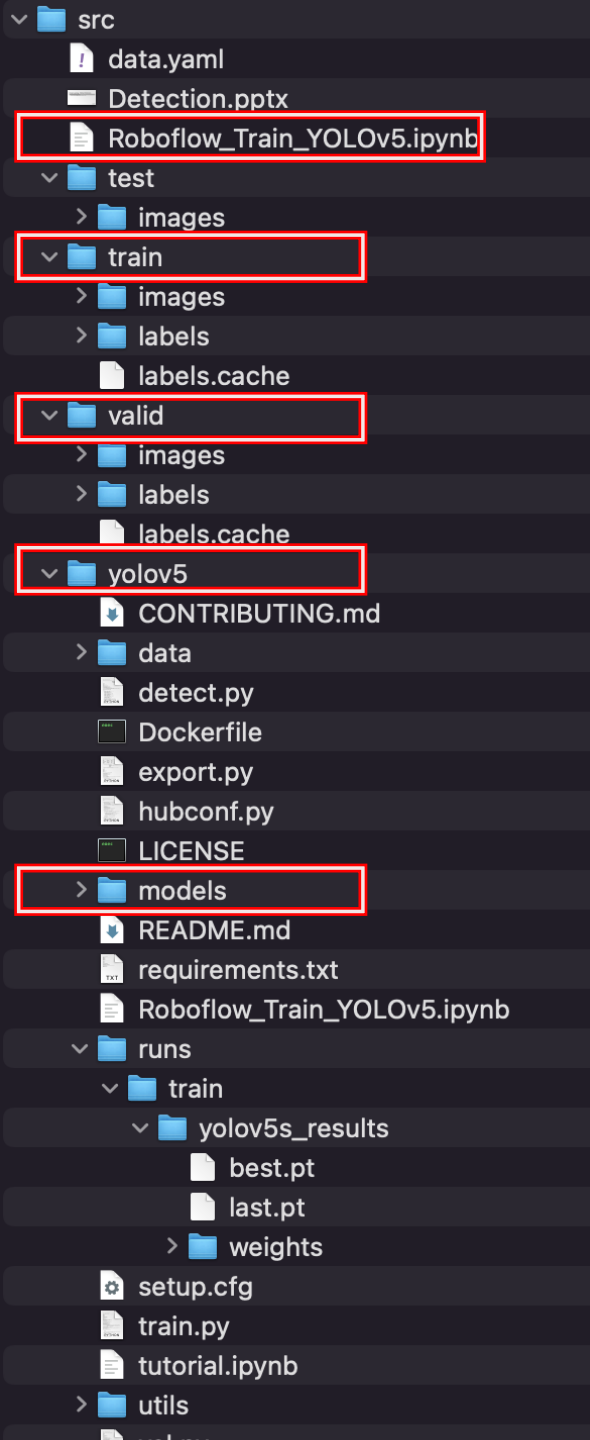
Roboflow_Train.ipynb

```
Roboflow_Train_YOLOv5.ipynb M
or > M*Next, we'll fire off training! > # train yolov5s on custom data for 100 epochs<# time its perform
+ Code + Markdown | Run All | Clear Outputs of All Cells | Outline ...
>
# train yolov5s on custom data for 100 epochs
# time its performance
%%time
%cd /content/yolov5/
!python train.py --img 416 --batch 16 --epochs 100 --data '../data.yaml' --cfg
./models/custom_yolov5s.yaml --weights '' --name yolov5s_results --cache
```

test
training
{image.jpg, label.txt}
validate
{image.jpg, label.txt}

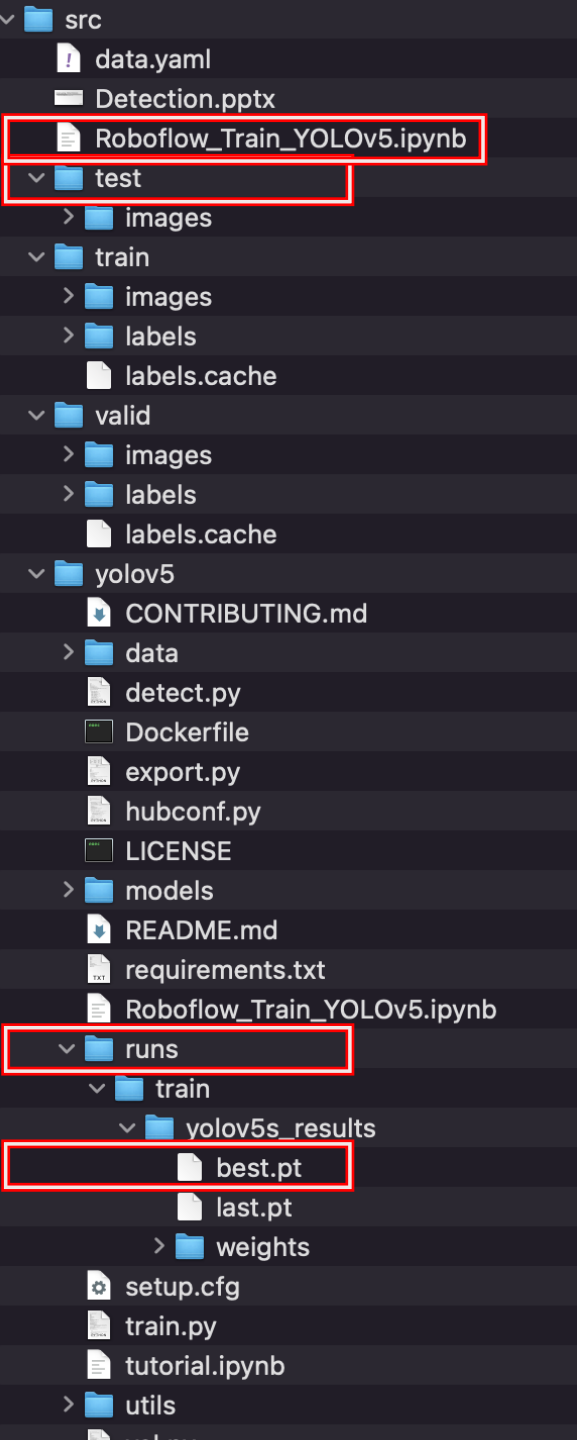
N -> S -> M -> L -> X

Trained Pytorch



Train - Result

```
# print out an augmented training example
print("GROUND TRUTH AUGMENTED TRAINING DATA:")
Image(filename='/content/yolov5/runs/train/yolov5s_results/train_batch0.jpg', width=900)
```



Test – Detecting

image Resize

train모델의 사이즈와 동일하게 test이미지의 크기를 맞춰줌

```
# image size
import glob
import cv2
path = glob.glob("../test/images/*.jpg")

for img in path:
    src = cv2.imread(img)
    dst = cv2.resize(src, dsize=(416, 416), interpolation=cv2.INTER_LINEAR)
    cv2.imwrite(img, dst)
print("finish")
```

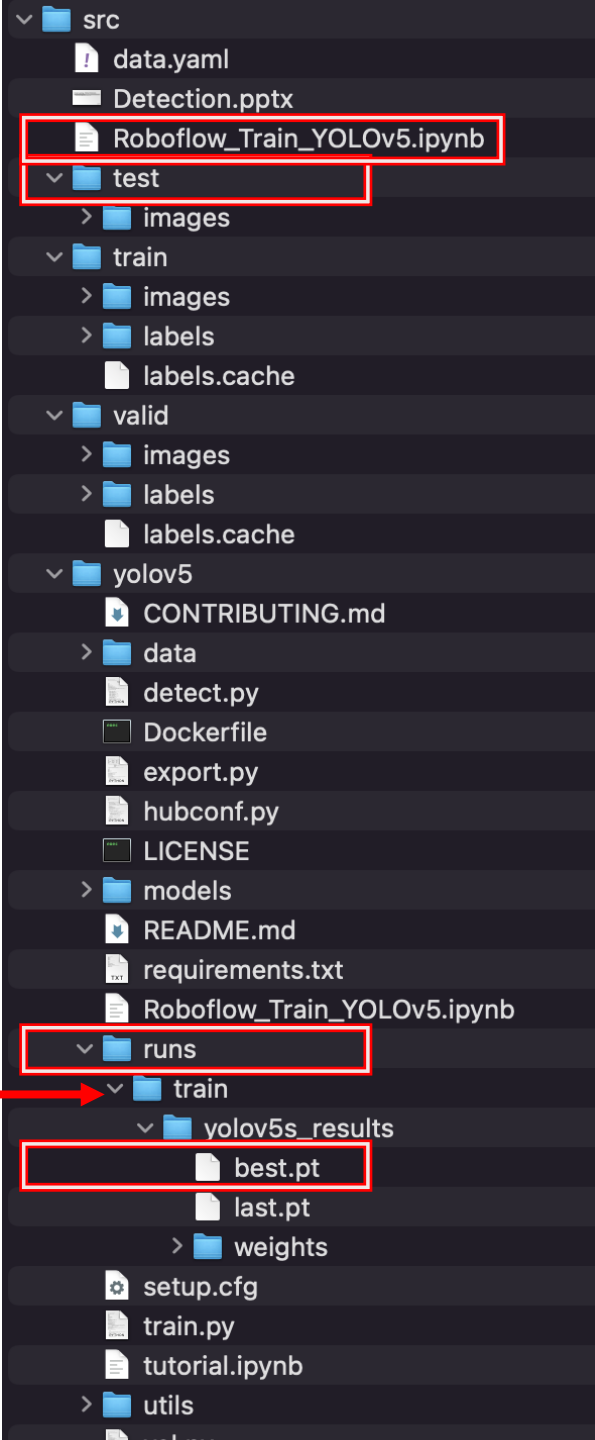
test
{image.jpg}

../test/images 존재하는 이미지 Detection

```
# when we ran this, we saw .007 second inference time. That is 140 FPS on a TESLA P100!
# use the best weights!
%cd /content/yolov5/
!python detect.py --weights runs/train/yolov5s_results/weights/best.pt --img 416 --conf 0.3 --source ../test/images
```

Detect Test image

Trained Pytorch



```
#display inference on ALL test images
#this looks much better with longer training above

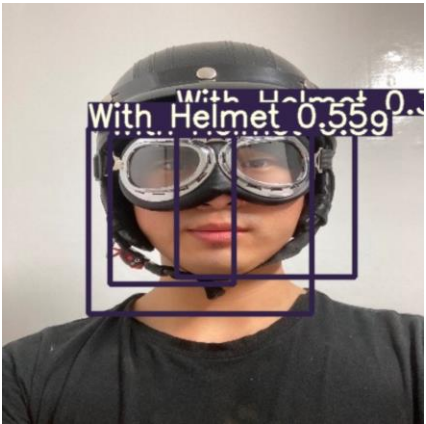
import glob
from IPython.display import Image, display

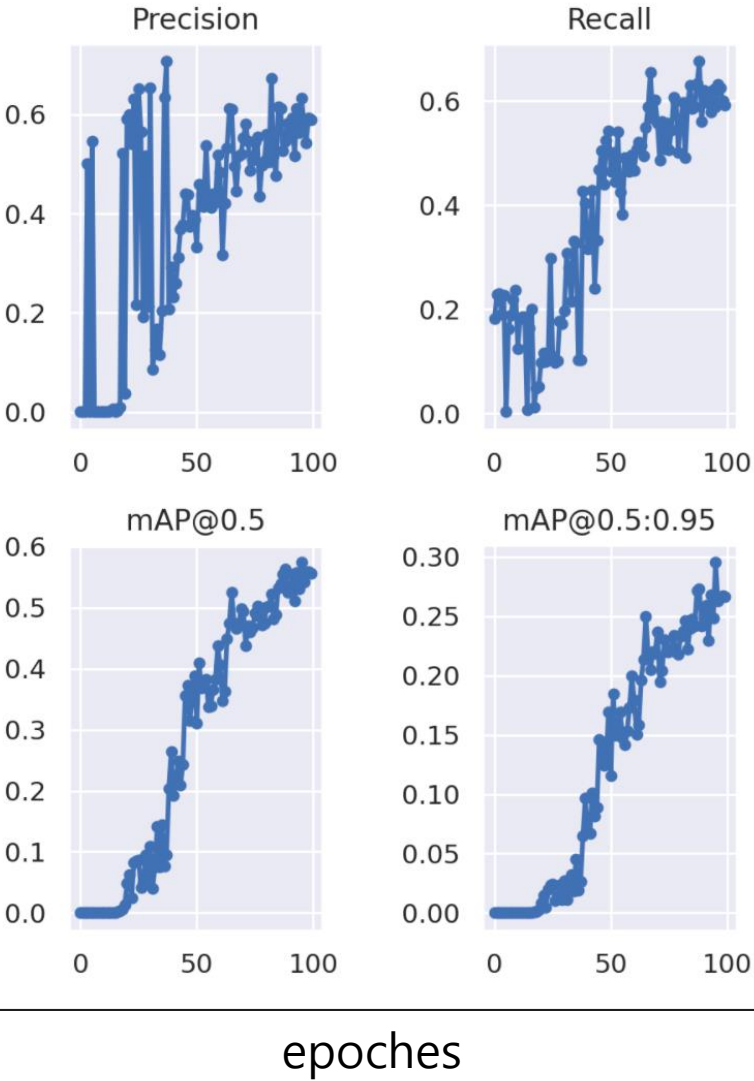
for imageName in glob.glob('/content/yolov5/runs/detect/exp/*.jpg'): #assuming JPG
    display(Image(filename=imageName))
    print("\n")
```

Bounding된 test image 표시



Detecting Result



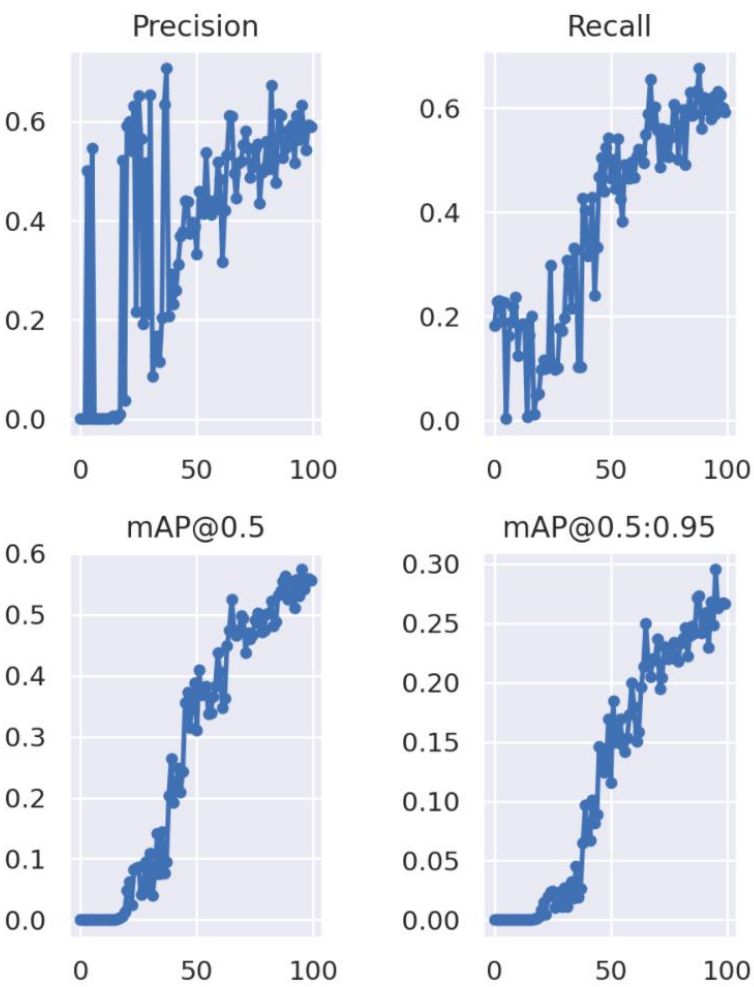


Precision : 검출 결과의 정밀도

Recall : Detection 재현율

Mean Average Precision : AP의 평균

epoches를 반복 할 수록 정확한 결과를 얻을 수 있음



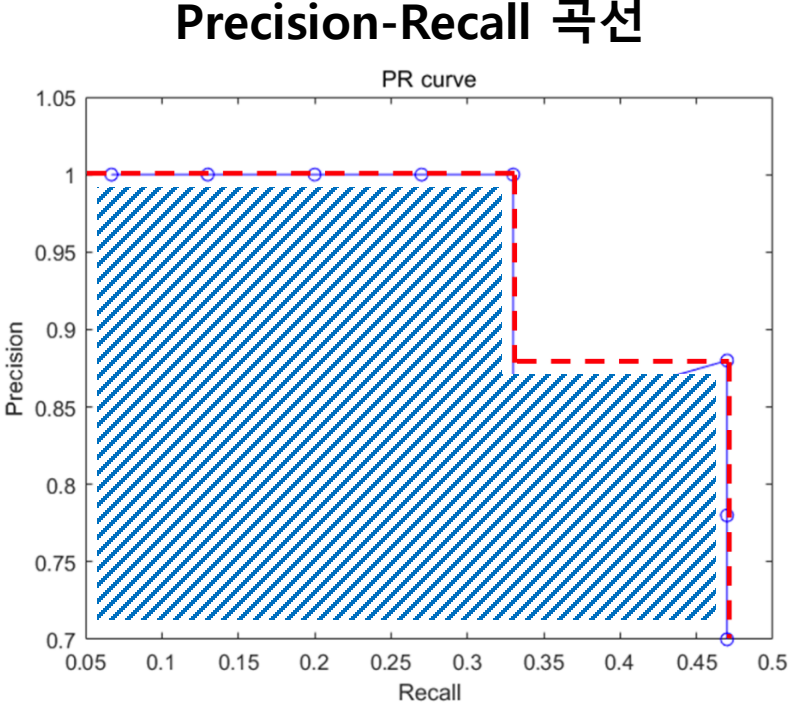
epoches

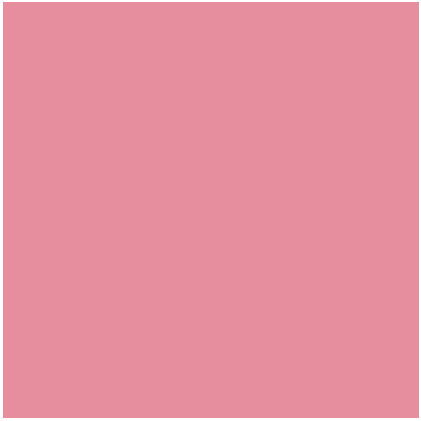
Precision : 검출 결과의 정밀!

Recall : Detection 재현율

Mean Average Precision : AP의 평균

epoches를 반복 할 수록 정확한 결과를 얻을 수 있음





Part 3, Appendix

개선방안 및 향후 계획



mAP수치가 다른 모델에
비해 다소 낮음



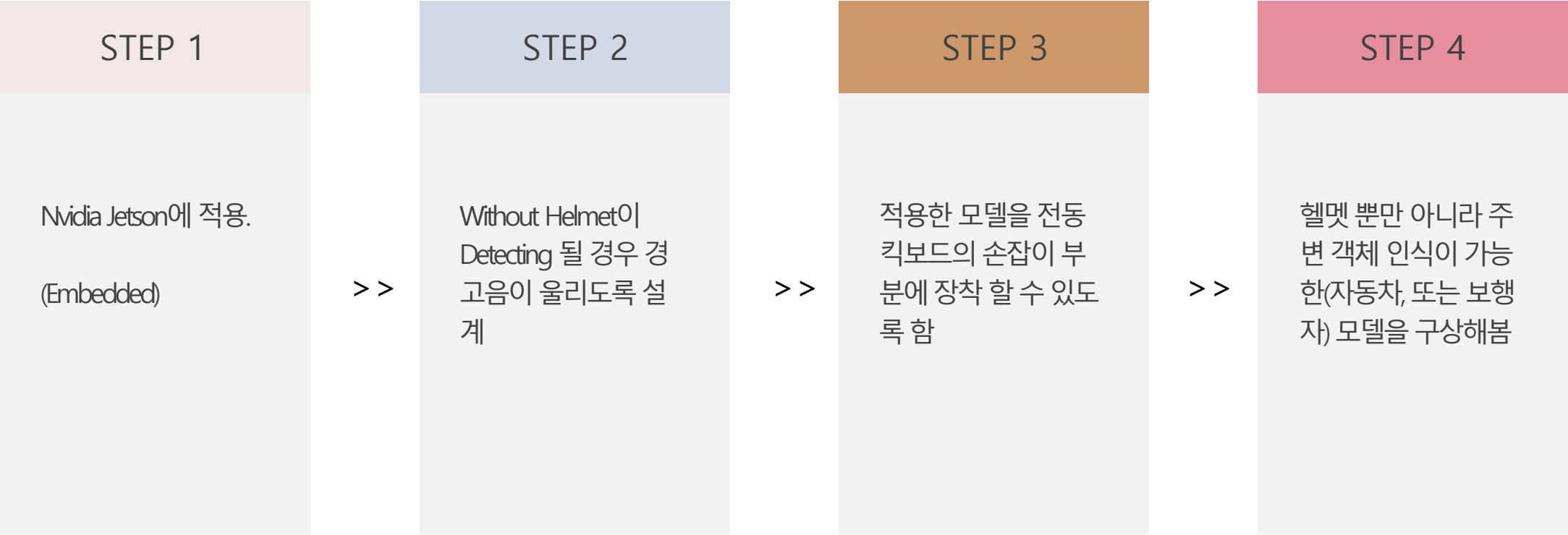
GPU사양이 높은 머신에서 정확도가 보다 높은
YOLOv5m, l, x 모델을 사용



학습 데이터를 더욱 다양하게 적용하여 여러가지
상황에서도 Detecting 할 수 있도록 함



Data Augmentation 기법으로 학습 데이터에
인위적인 노이즈를 주어 데이터 수를 늘림

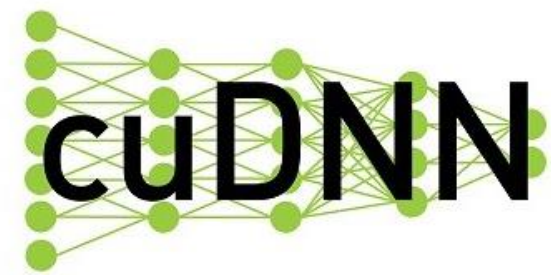




Part 4, Reference

참고자료

Required



<https://www.flaticon.com/>

<http://www.econovill.com/news/articleView.html?idxno=406417>

<https://www.donga.com/news/It/article/all/20210923/109363192/1>

<https://github.com/ultralytics/yolov5.git>

<https://github.com/tzutalin/labelImg.git>

<https://lapina.tistory.com/98>

<https://www.kaggle.com/>

<https://roboflow.com/>

<https://pytorch.org/>

<< 역할분담 >>

주우성, 최성원 : 사례 분석 및 문제 인식

주우성, 최성원 : 데이터 수집 및 전처리

주우성 : 데이터셋 학습

최성원 : 학습된 데이터를 통해 Detect

감사합니다