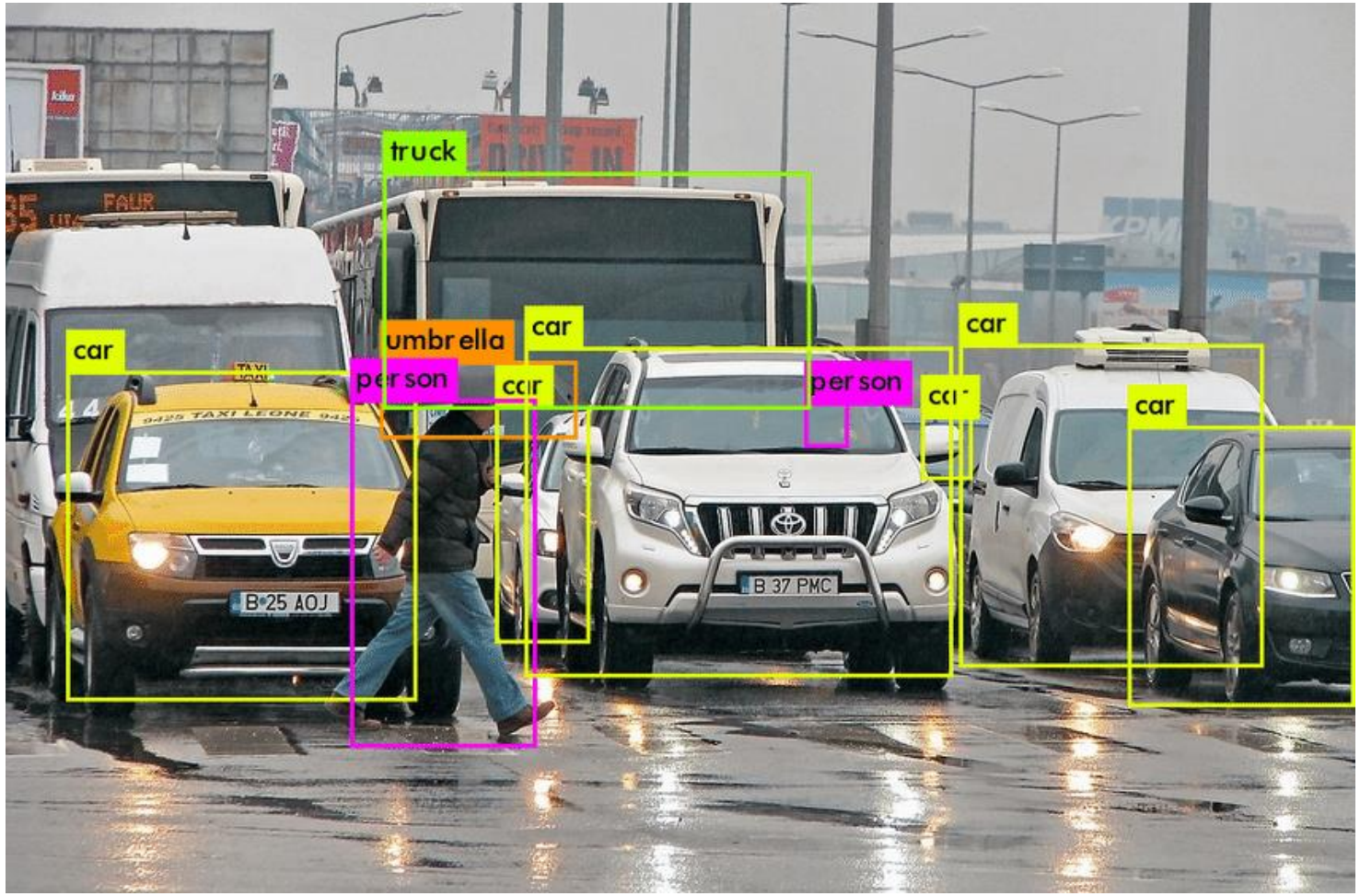
객체 탐지 (Object Detection)

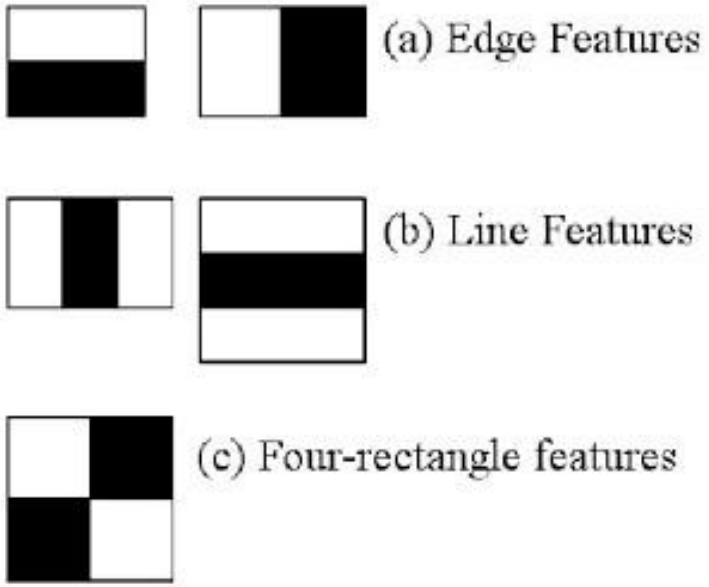


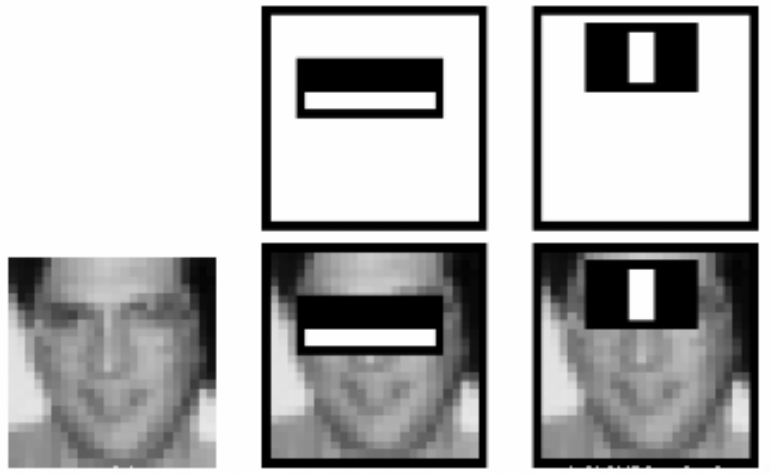
박경규

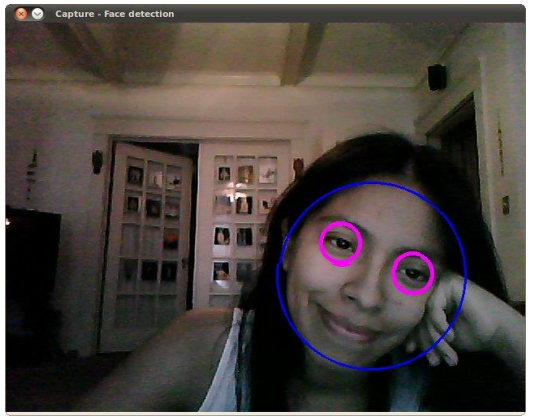
Haar feature-based cascade classifiers

2001년에 Paul Viola와 Michael Jones이 제안된 객체 검출 방법으로, 검출 대상 객체가 있는 이미지와 없는 이미지 (Positive Image, Negative Image)을 최대한 많이 활용해서 다단계 함수를 훈련시키는 기계학습 방식입니다.

Haar feature







참조 : http://www.gisdeveloper.co.kr/?p=7208

Haar feature-based cascade classifiers

haar_cascade.py

```python
from __future__ import print_function
import cv2 as cv
import argparse

def detectAndDisplay(frame):
    frame_gray = cv.cvtColor(frame, cv.COLOR_BGR2GRAY)
    frame_gray = cv.equalizeHist(frame_gray)
    #-- Detect faces
    faces = face_cascade.detectMultiScale(frame_gray)
    for (x,y,w,h) in faces:
        center = (x + w//2, y + h//2)
        frame = cv.ellipse(frame, center, (w//2, h//2), 0, 0, 360, (255, 0, 255), 4)
        faceROI = frame_gray[y:y+h,x:x+w]
        #-- In each face, detect eyes
        eyes = eyes_cascade.detectMultiScale(faceROI)
        for (x2,y2,w2,h2) in eyes:
            eye_center = (x + x2 + w2//2, y + y2 + h2//2)
            radius = int(round((w2 + h2)*0.25))
            frame = cv.circle(frame, eye_center, radius, (255, 0, 0 ), 4)
    cv.imshow('Capture - Face detection', frame)

```

https://docs.opencv.org/master/db/d28/tutorial_cascade_classifier.html

Haar feature-based cascade classifiers

```
22  parser = argparse.ArgumentParser(description='Cascade Classifier tutorial')
23  parser.add_argument('--face_cascade', help='Path to face cascade.',
24      default='haarcascade_frontalface_alt.xml')
25  parser.add_argument('--eyes_cascade', help='Path to eyes cascade.', d
26      efault='haarcascade_eye_tree_eyeglasses.xml')
27  parser.add_argument('--camera', help='Camera divide number.',
28      type=int, default=0)
29
30  args = parser.parse_args()
31  face_cascade_name = args.face_cascade
32  eyes_cascade_name = args.eyes_cascade
33  face_cascade = cv.CascadeClassifier()
34  eyes_cascade = cv.CascadeClassifier()
35
```

https://docs.opencv.org/master/db/d28/tutorial_cascade_classifier.html

Haar feature-based cascade classifiers

```
36  #-- 1. Load the cascades
37  if not face_cascade.load(cv.samples.findFile(face_cascade_name)):
38      print('--(!)Error loading face cascade')
39      exit(0)
40  if not eyes_cascade.load(cv.samples.findFile(eyes_cascade_name)):
41      print('--(!)Error loading eyes cascade')
42      exit(0)
43
44  camera_device = args.camera
45  #-- 2. Read the video stream
46  cap = cv.VideoCapture(camera_device)
47
48  if not cap.isOpened:
49      print('--(!)Error opening video capture')
50      exit(0)
51  while True:
52      ret, frame = cap.read()
53      if frame is None:
54          print('--(!) No captured frame -- Break!')
55          break
56      detectAndDisplay(frame)
57      if cv.waitKey(10) == 27:
58          break
```

https://docs.opencv.org/master/db/d28/tutorial_cascade_classifier.html

객체 탐지(이미지) – haar cascade

object_detection_haar_cascade.ipynb

```
face_cascade_file = 'haarcascade_frontalface_alt.xml'
eyes_cascade_file = 'haarcascade_eye_tree_eyeglasses.xml'

face_cascade = cv2.CascadeClassifier()
eyes_cascade = cv2.CascadeClassifier()

if not face_cascade.load(cv2.samples.findFile(face_cascade_file)):
    print('Error loading face cascade')
    exit(0)
if not eyes_cascade.load(cv2.samples.findFile(eyes_cascade_file)):
    print('Error loading eyes cascade')
    exit(0)
```

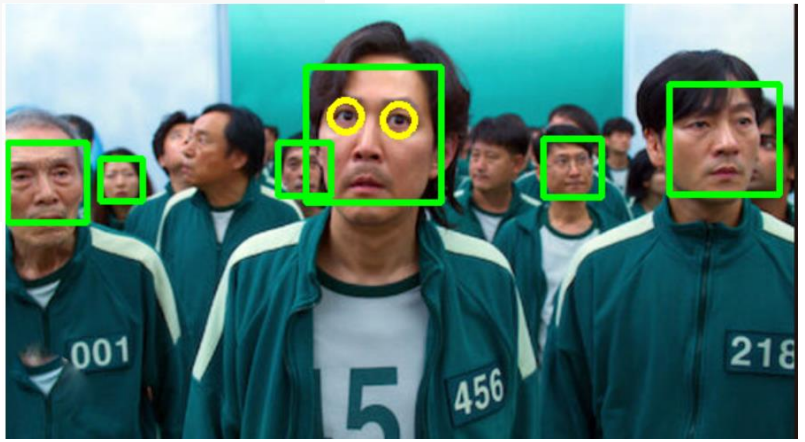
객체 탐지(이미지) – haar cascade

object_detection_haar_cascade.ipynb

```
faces = face_cascade.detectMultiScale(gray)

for (x,y,w,h) in faces:
    center = (x + w//2, y + h//2)
    img = cv2.rectangle(img, (x, y), (x+w, y+h), (0, 255, 0), 3)
    face_rg = gray[y:y+h, x:x+w]
    eyes = eyes_cascade.detectMultiScale(face_rg)
    for (x2, y2, w2, h2) in eyes:
        eye_center = (x + x2 + w2//2, y + y2 + h2//2)
        radius = int(round((w2+h2)*0.25))
        img = cv2.circle(img, eye_center, radius, (0, 255,

cv2_imshow(img)
```



객체 탐지(이미지) – haar cascade

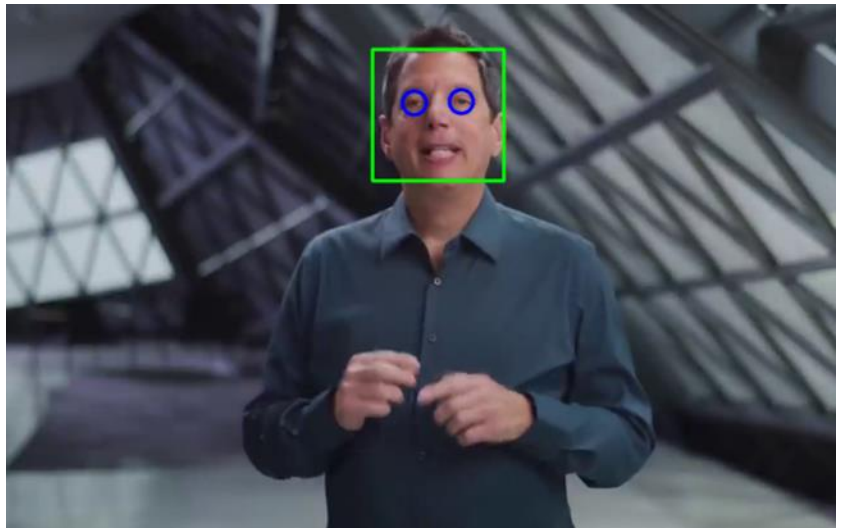
object_detection_haar_cascade.ipynb

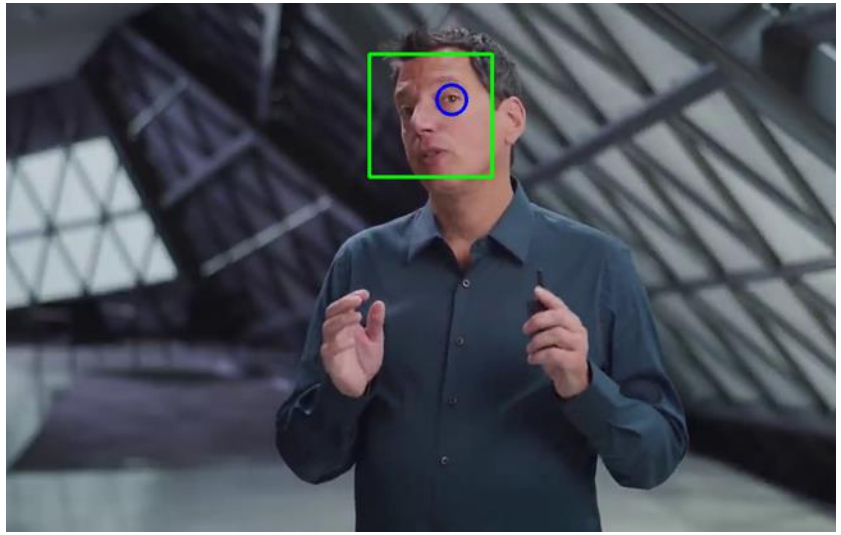
```
def detect(frame):
    IPython.display.clear_output(wait = True)
    frame_gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    frame_gray = cv2.equalizeHist(frame_gray)
    faces = face_cascade.detectMultiScale(frame_gray)
    for (x,y,w,h) in faces:
        center = (x + w//2, y + h//2)
        frame = cv2.rectangle(frame, (x, y), (x+w, y+h), (0, 255, 0), 4)
        face_rg = frame_gray[y:y+h,x:x+w]
        eyes = eyes_cascade.detectMultiScale(face_rg)
        for (x2,y2,w2,h2) in eyes:
            eye_center = (x + x2 + w2//2, y + y2 + h2//2)
            radius = int(round((w2 + h2)*0.25))
            frame = cv2.circle(frame, eye_center, radius, (255, 0, 0 ), 4)
    cv2_imshow(frame)
```

객체 탐지(동영상) – haar cascade

object_detection_haar_cascade.ipynb

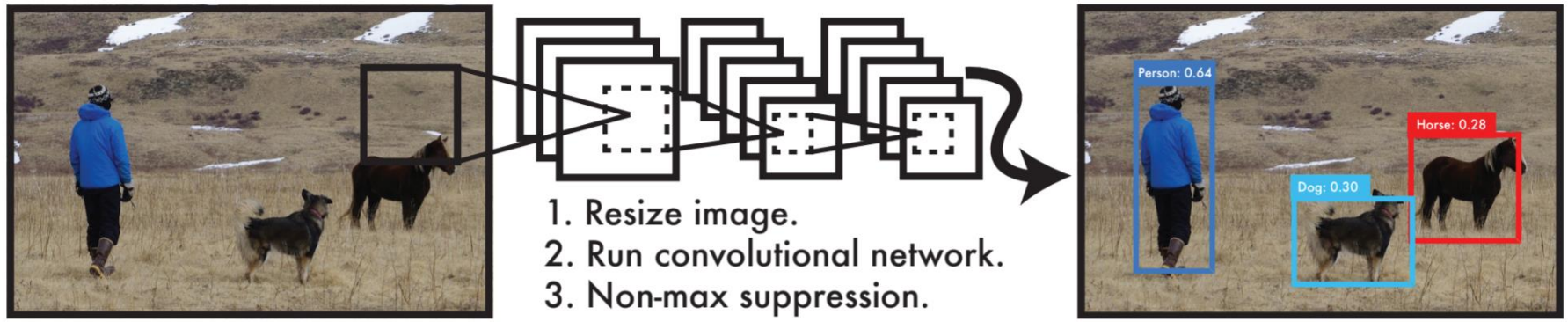
```
vedio_file = 'video.mp4'
cap = cv2.VideoCapture(vedio_file)
if not cap.isOpened:
    print('Error opening video file')
    exit(0)
while True:
    ret, frame = cap.read()
    if frame is None:
        print('End')
        break
    detect(frame)
```





YOLO (You Look Only Once)

하나의 컨볼루션 네트워크가 여러 bounding box와 그 bounding box의 클래스 확률을 동시에 계산해 줍니다.
YOLO는 이미지 전체를 학습하여 곧바로 검출 성능(detection performance)을 최적화합니다.



- YOLO 연구진은 객체 검출(object detection)에 새로운 접근방식을 적용했습니다.
- YOLO는 이미지 전체에 대해서 하나의 신경망(a single neural network)이
  한 번의 계산만으로 bounding box와 클래스 확률(class probability)을 예측합니다.
- bounding box란 객체의 위치를 알려주기 위해 객체의 둘레를 감싼 직사각형 박스를 말합니다.
- 클래스 확률이란 bounding box로 둘러싸인 객체가 어떤 클래스에 해당하는지에 관한 확률을 의미합니다.
- 객체 검출 파이프라인이 하나의 신경망으로 구성되어 있으므로 end-to-end 형식입니다.
- 기존의 multi-task 문제를 하나의 회귀(regression) 문제로 재정의했습니다.

https://bit.ly/30o69Pk

COCO Dataset

COCO dataset은 object detection, segmentation, keypoint detection 등을 위한 데이터셋으로,
매년 다른 데이터셋으로 전 세계의 여러 대학/기업이 참가하는 대회에 사용되고 있습니다.


```
"images": [
  ...
  {
    "license": 1,
    "file_name": "000000324158.jpg",
    "coco_url": "http://images.cocodataset.org/val2017/000000324158.jpg",
    "height": 334,
    "width": 500,
    "date_captured": "2013-11-19 23:54:06",
    "flickr_url": "http://farm1.staticflickr.com/169/417836491_5bf8762150_z.jpg",
    "id": 324158
  },
  ...
],
```



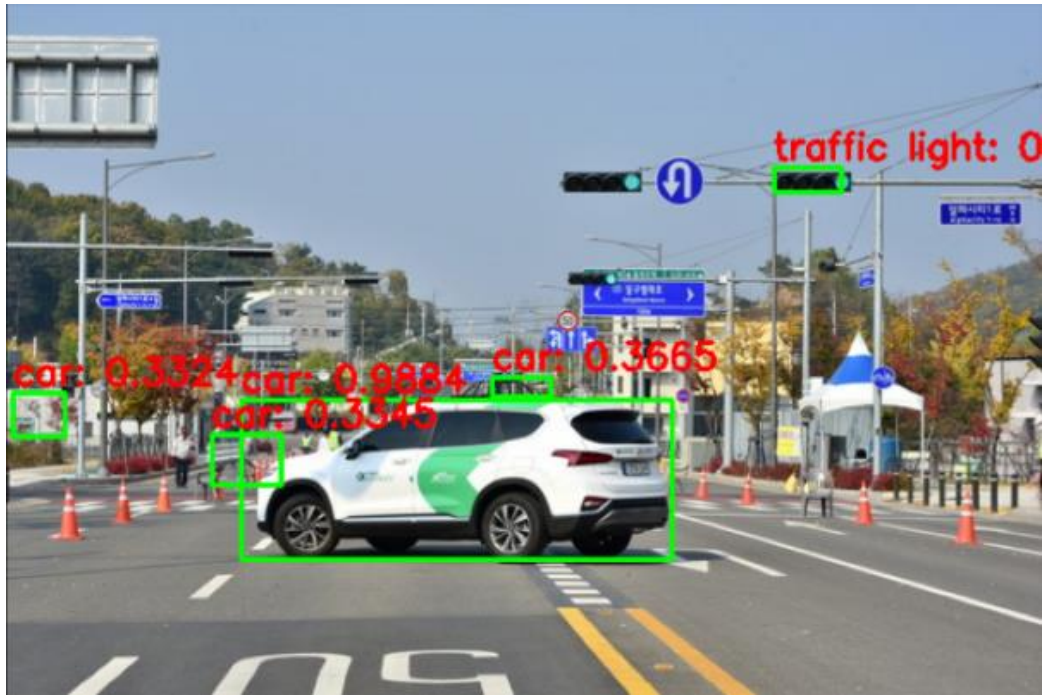
```
"annotations": [
  ...
  {
    "segmentation": [
      [
        216.7,
        211.89,
        216.16,
        217.81,
        215.89,
        220.77,

        ...
        212.16
      ]
    ],
    "area": 759.3375500000002,
    "iscrowd": 0,
    "image_id": 324158,
    "bbox": [
      196.51,
      183.36,
```


https://ukayzm.github.io/cocodataset/

OpenCV DNN Tensorflow

object_detection_dnn.ipynb

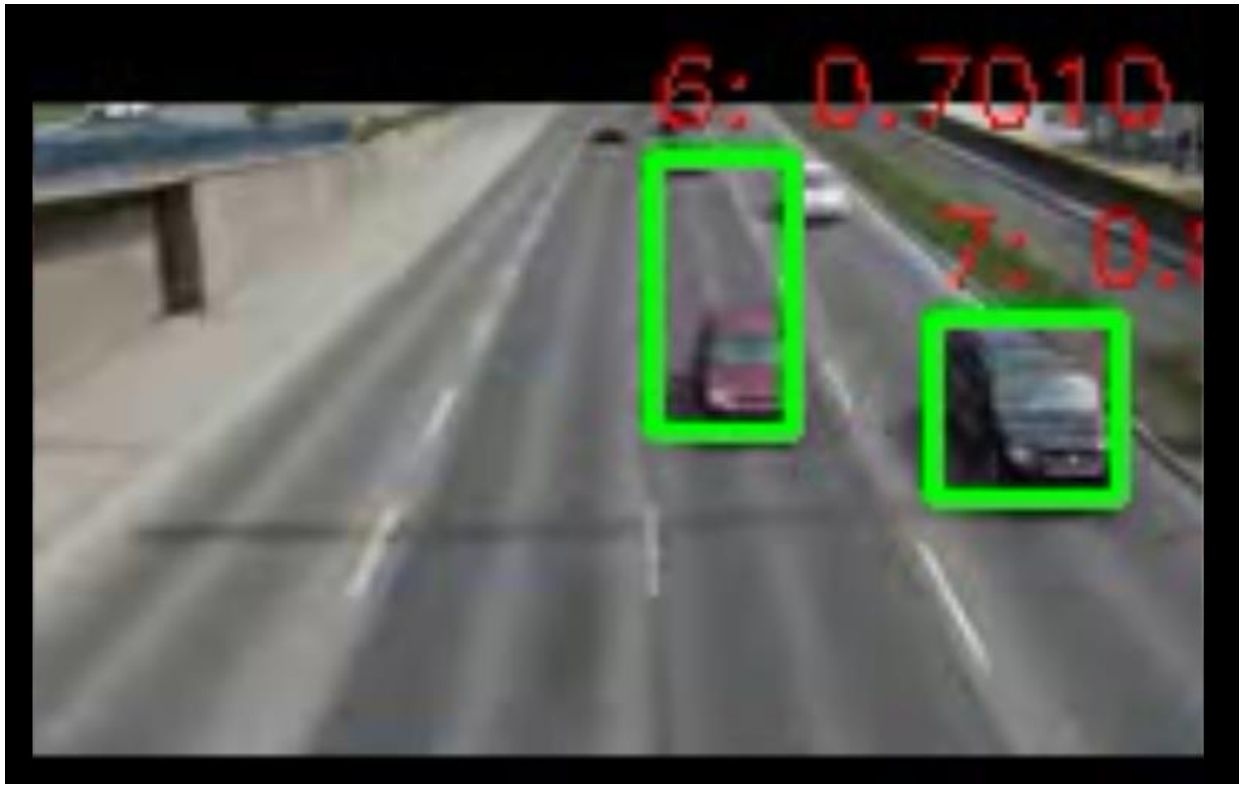


OpenCV DNN Yolo

object_detection_yolo.ipynb

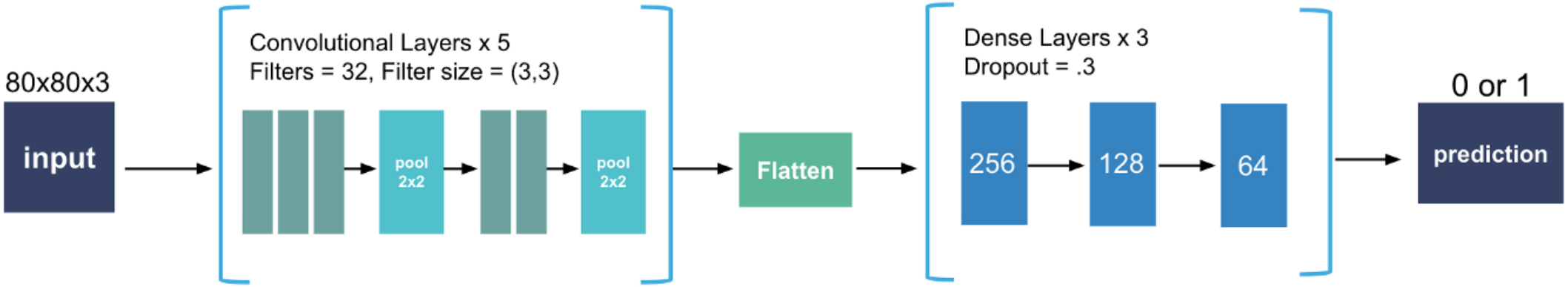
■ 이미지 객체 탐지

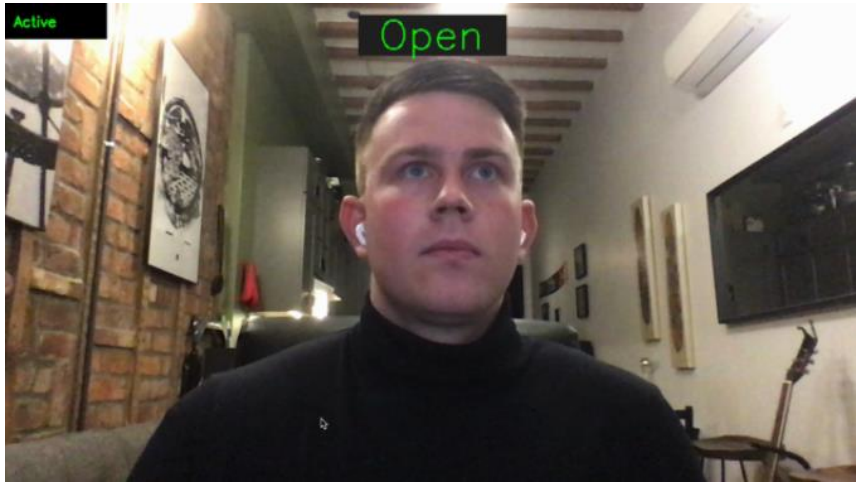
■ 동영상 객체 탐지

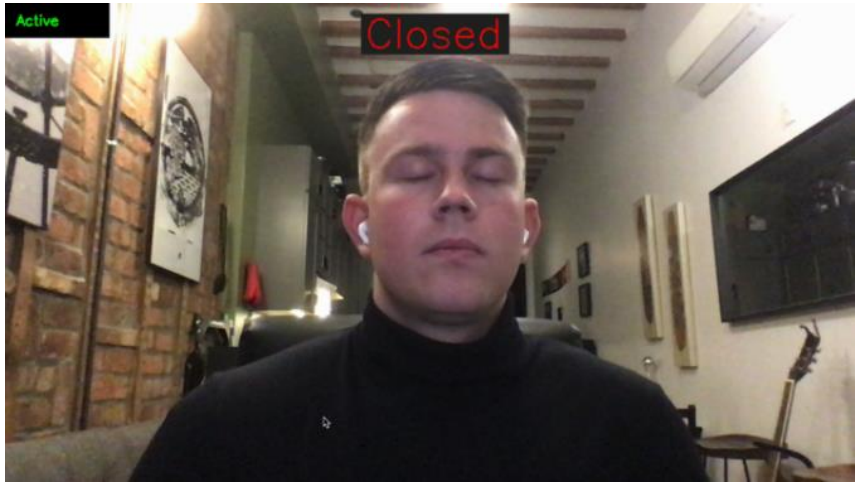


참고 : Drowsiness Detection

https://towardsdatascience.com/drowsiness-detection-using-convolutional-neural-networks-face-recognition-and-tensorflow-56cdfc8315ad







Thank you

kgpark88@gmail.com