YOLO 객체 인식 실습

Smart Embedded System Lab
Kookmin University

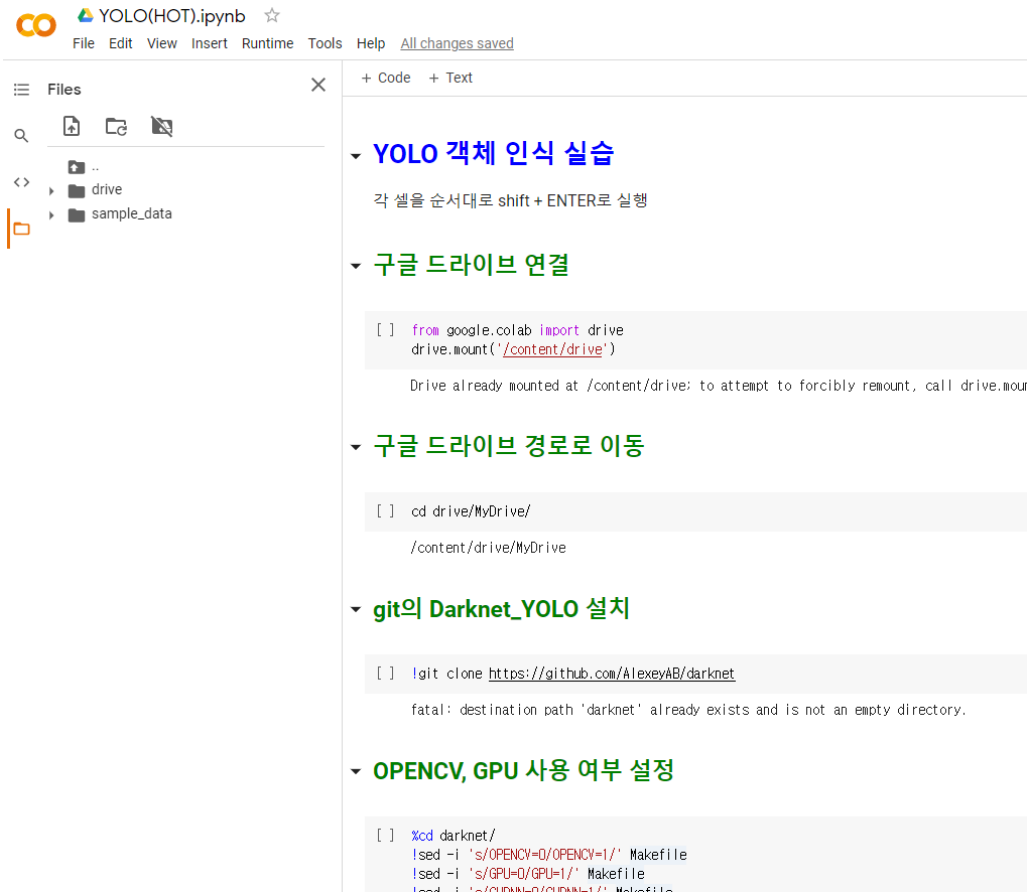
YOLO 시작

➢ Colab에서 YOLO 시작하기

- **주소창에 https://colab.research.google.com/ 입력**
  - 업로드 클릭 -> 파일선택
  - YOLO(HOT).ipynb 열기

➢ Colab에서 YOLO 시작하기

- YOLO 구성 Colab화면으로 진입



YOLO 실습 진행 방식

➢ YOLO 실습 진행 방식

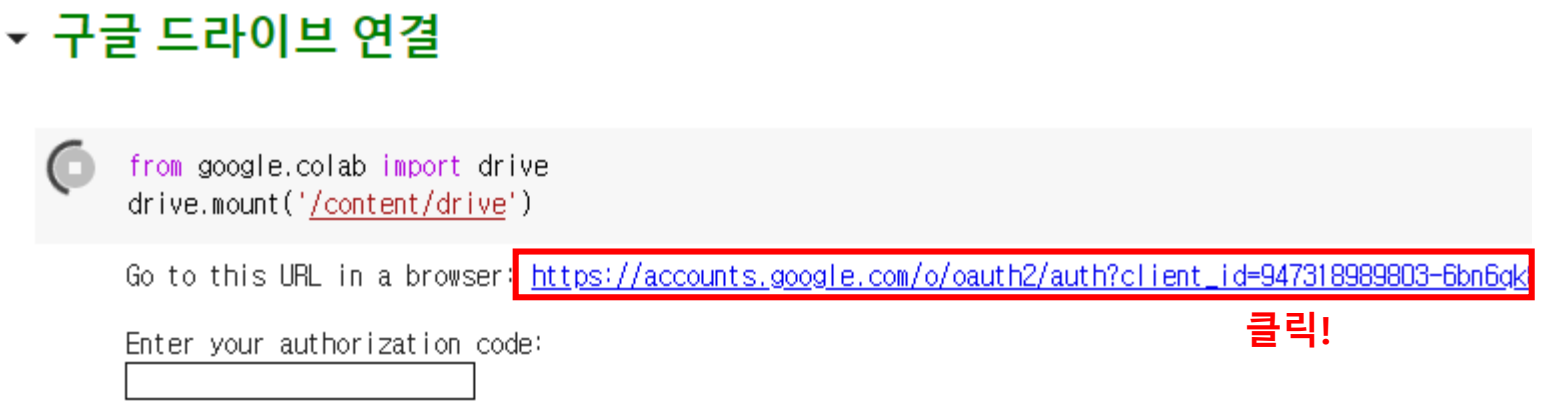
- **가장 위에 있는 라이브러리 설치 진행 cell 부터 순서대로 진행**
  - 각 셀의 좌측의 ▶ 버튼 클릭 또는 Shift + ENTER 클릭!
  - 각 cell이 다 돌아가는지 확인하여 다음 cell 실행

✓Cell 실행 중 : ◼ cell 실행 완료 및 대기 중 : ▶

YOLO 환경 구성

➢ 구글 드라이브 연결

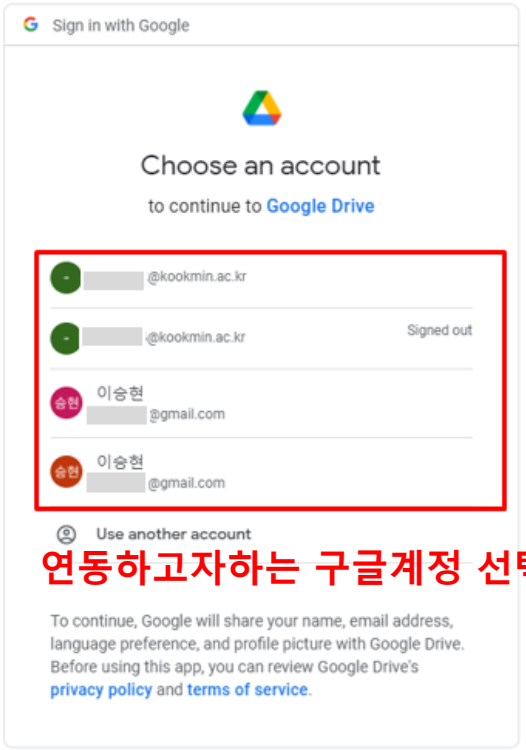
- **content 폴더 안의 drive라는 이름으로 구글 드라이브 연결**
- **실행 시 URL 링크 생성**
  - URL 링크 클릭



YOLO 환경 구성

➢ 구글 드라이브 연결

- 연동하고싶은 구글 드라이브 선택하여 클릭!
- 제공되는 코드를 복사



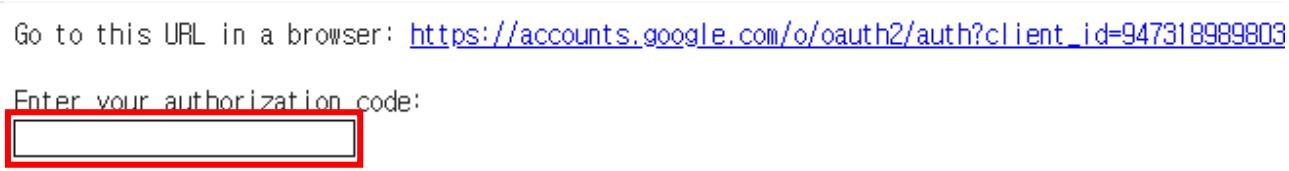
연동하고자하는 구글계정 선택!

클릭하여 복사

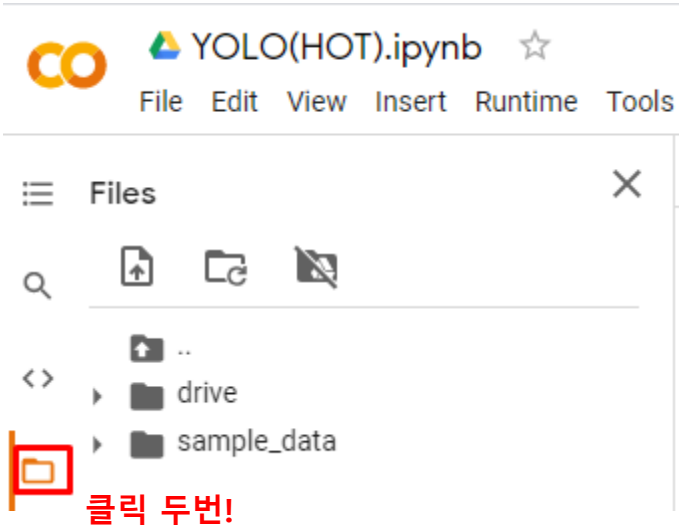
YOLO 환경 구성

➢ 구글 드라이브 연결

- 복사한 코드를 구글 드라이브 연결 cell에 입력 + ENTER



- 좌측의 폴더아이콘 두번 클릭하여 drive폴더 생성 확인



YOLO 환경 구성

➢ 구글 드라이브로 경로 이동

- **YoLo 폴더를 구글 드라이브 내에 설치**
  - 기본 제공경로에 설치시 Colab Runtime이 끊기면 설치한 폴더도 사라짐
  - 구글 드라이브 내에 설치하여 설치 폴더 유지
- **구글 드라이브로 경로 이동**

```
cd drive/MyDrive/
```

▾ 구글 드라이브 경로로 이동

```
[ ]  cd drive/MyDrive/

     /content/drive/MyDrive
```

YOLO 환경 구성

➢ git의 YOLO 설치

- **github에 있는 YOLO 실습코드 설치 진행**

```
!git clone https://github.com/AlexeyAB/darknet
```

▾ git의 Darknet_YOLO 설치

```
[ ]  !git clone https://github.com/AlexeyAB/darknet
```

➢ Makefile 설정

- **OPENCV, GPU 사용 여부 설정**
  - OPENCV 사용 시 :

```
!sed -i 's/OPENCV=0/OPENCV=1/' Makefile
```

  - GPU 사용 시 :

```
!sed -i 's/GPU=0/GPU=1/' Makefile
!sed -i 's/CUDNN=0/CUDNN=1/' Makefile
```

▾ OPENCV, GPU 사용 여부 설정

```
%cd darknet/
!sed -i 's/OPENCV=0/OPENCV=1/' Makefile
!sed -i 's/GPU=0/GPU=1/' Makefile
!sed -i 's/CUDNN=0/CUDNN=1/' Makefile
```

OPENCV, GPU 사용

YOLO 환경 구성

➢ YOLO 빌드

- **Makefile로 구성된 YOLO 설정 빌드**

```
!make
```

▾ YOLO 빌드

```
!make

chmod +x *.sh
g++ -std=c++11 -std=c++11 -Iinclude/ -I3rdparty/stb/include -DOPENCV `pkg-config --cflags opencv4 2> /dev/null || pkg-config --cflags opencv` -DGPU -I/usr/local/cud
./src/image_opencv.cpp: In function ‘void draw_detections_cv_v3(void**, detection*, int, float, char**, image**, int, int)’:
./src/image_opencv.cpp:926:23: warning: variable ‘rgb’ set but not used [-Wunused-but-set-variable]
                float rgb[3];
                      ^~~
./src/image_opencv.cpp: In function ‘void draw_train_loss(char*, void**, int, float, float, int, int, float, int, char*, float, int, int, double)’:
./src/image_opencv.cpp:1127:13: warning: this ‘if’ clause does not guard... [-Wmisleading-indentation]
            if (iteration_old == 0)
            ^~
./src/image_opencv.cpp:1130:10: note: ...this statement, but the latter is misleadingly indented as if it were guarded by the ‘if’
         if (iteration_old != 0){
         ^~
 /src/image_opencv.cpp: In function ‘void cv_draw_object(image, float*, int, int, int*, float*, int*, int, char**)’:
```

YOLO 환경 구성

➢ YOLO 활용에 유용한 함수 구성

- **이미지 출력, 업로드, 다운로드 함수 구현**
  - 함수활용을 통해 간단하게 이미지 출력 및 업로드, 다운로드 가능

▾ 유용한 함수 구성

```
[ ]  # define helper functions
     def imShow(path):
       import cv2
       import matplotlib.pyplot as plt
       %matplotlib inline

       image = cv2.imread(path)
       height, width = image.shape[:2]
       resized_image = cv2.resize(image,(3*width, 3*height), interpolation = cv2.INTER_CUBIC)

       fig = plt.gcf()
       fig.set_size_inches(18, 10)
       plt.axis("off")
       plt.imshow(cv2.cvtColor(resized_image, cv2.COLOR_BGR2RGB))
       plt.show()

     # use this to upload files
     def upload():
       from google.colab import files
       uploaded = files.upload()
       for name, data in uploaded.items():
         with open(name, 'wb') as f:
           f.write(data)
           print ('saved file', name)

     # use this to download a file
     def download(path):
       from google.colab import files
       files.download(path)
```

YOLO 환경 구성

➢ YOLO weight 설치

- YOLO 활용을 하기 위한 미리 학습된 모델 weight 설치
- yolov3 weight 설치

```
!wget https://pjreddie.com/media/files/yolov3.weights
```

- yolov4 weight 설치

```
!wget https://github.com/AlexeyAB/darknet/releases/download/darknet_yolo_v3_optimal/yolov4.weights
```

▾ yolov3 weight 설치

```
!wget https://pjreddie.com/media/files/yolov3.weights

--2021-01-11 20:11:41--  https://pjreddie.com/media/files/yolov3.weights
Resolving pjreddie.com (pjreddie.com)... 128.208.4.108
Connecting to pjreddie.com (pjreddie.com)|128.208.4.108|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 248007048 (237M) [application/octet-stream]
Saving to: 'yolov3.weights'

yolov3.weights      100%[===================>] 236.52M  9.21MB/s    in 25s

2021-01-11 20:12:06 (9.35 MB/s) - 'yolov3.weights' saved [248007048/248007048]
```

▾ yolov4 weight 설치

```
!wget https://github.com/AlexeyAB/darknet/releases/download/darknet_yolo_v3_optimal/yolov4.weights

--2021-01-11 20:30:14--  https://github.com/AlexeyAB/darknet/releases/download/darknet_yolo_v3_optimal/yolo
Resolving github.com (github.com)... 140.82.114.3
Connecting to github.com (github.com)|140.82.114.3|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://github-production-release-asset-2e65be.s3.amazonaws.com/75388965/ba4b6380-889c-11ea-9751-
--2021-01-11 20:30:14--  https://github-production-release-asset-2e65be.s3.amazonaws.com/75388965/ba4b6380-
Resolving github-production-release-asset-2e65be.s3.amazonaws.com (github-production-release-asset-2e65be.s
Connecting to github-production-release-asset-2e65be.s3.amazonaws.com (github-production-release-asset-2e65
HTTP request sent, awaiting response... 200 OK
Length: 257717640 (246M) [application/octet-stream]
Saving to: 'yolov4.weights'

yolov4.weights      100%[===================>] 245.78M  71.5MB/s    in 3.5s

2021-01-11 20:30:18 (70.0 MB/s) - 'yolov4.weights' saved [257717640/257717640]
```

➢ YOLO 이미지 실습

- Colab에서 이미지 출력을 위해 미리 함수 구성된 imShow를 통한 이미지 출력
- yolov3 이미지 실습

```
!./darknet detect cfg/yolov3.cfg yolov3.weights data/dog.jpg
imShow('predictions.jpg')
```

- yolov4 이미지 실습

```
!./darknet detect cfg/yolov4.cfg yolov4.weights data/dog.jpg
imShow('predictions.jpg')
```

yolov3 이미지 실습

```
!./darknet detect cfg/yolov3.cfg yolov3.weights data/dog.jpg
imShow('predictions.jpg')
```

```
 CUDA-version: 10010 (10010), cuDNN: 7.6.5, GPU count: 1
 OpenCV version: 3.2.0
 0 : compute_capability = 750, cudnn_half = 0, GPU: Tesla T4
net.optimized_memory = 0
mini_batch = 1, batch = 1, time_steps = 1, train = 0
   layer   filters  size/strd(dil)      input                output
   0 conv     32       3 x 3/ 1    416 x 416 x   3 ->  416 x 416 x  32 0.299 BF
   1 conv     64       3 x 3/ 2    416 x 416 x  32 ->  208 x 208 x  64 1.595 BF
```

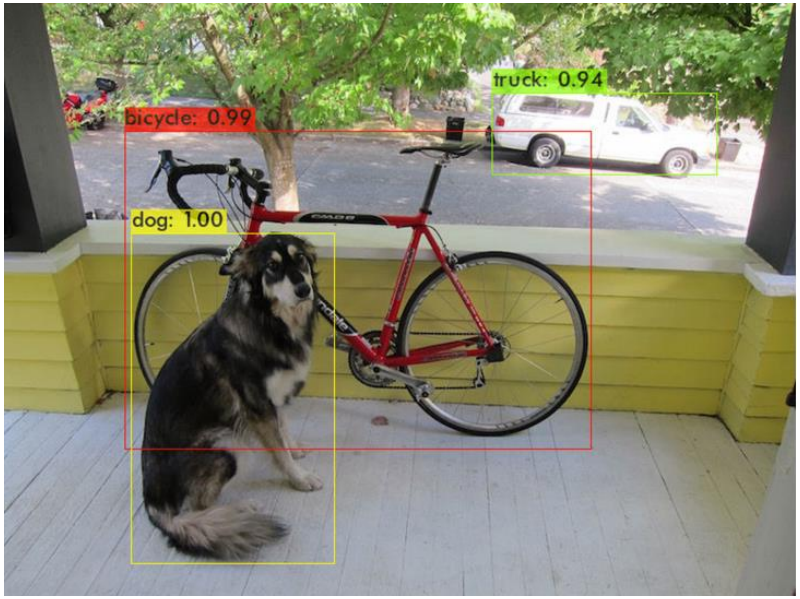
yolov4 이미지 실습

```
!./darknet detect cfg/yolov4.cfg yolov4.weights data/dog.jpg
imShow('predictions.jpg')
```

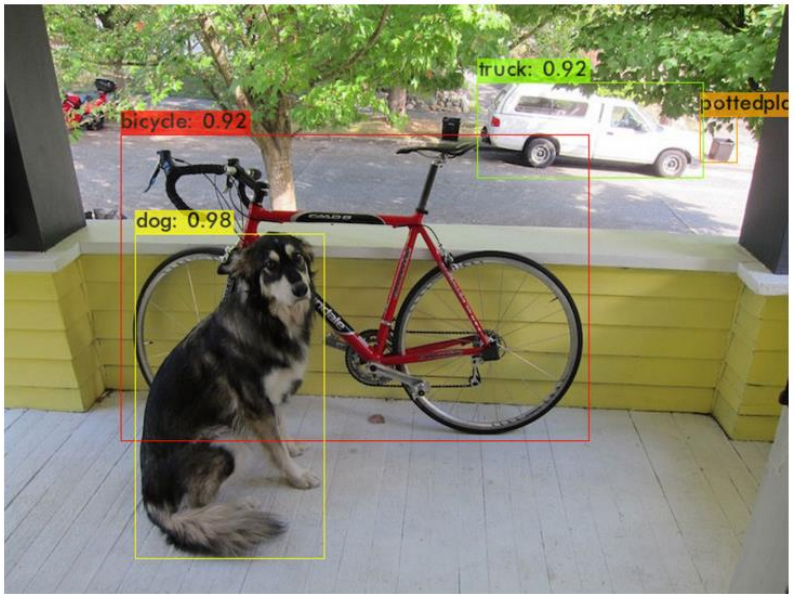
```
 CUDA-version: 10010 (10010), cuDNN: 7.6.5, GPU count: 1
 OpenCV version: 3.2.0
 0 : compute_capability = 750, cudnn_half = 0, GPU: Tesla T4
net.optimized_memory = 0
mini_batch = 1, batch = 8, time_steps = 1, train = 0
   layer   filters  size/strd(dil)      input                output
   0 conv     32       3 x 3/ 1    608 x 608 x   3 ->  608 x 608 x  32 0.639 BF
   1 conv     64       3 x 3/ 2    608 x 608 x  32 ->  304 x 304 x  64 3.407 BF
   2 conv     64       1 x 1/ 1    304 x 304 x  64 ->  304 x 304 x  64 0.757 BF
   3 route  1                                     ->  304 x 304 x  64
```

➢ YOLO 이미지 실습

- 이미지 실습 결과



Yolov3 결과



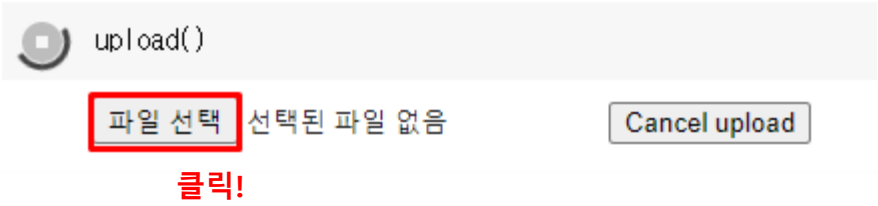
Yolov4 결과

YOLO 영상 실습

➢ 데이터 업로드 진행

- https://drive.google.com/file/d/1V68xfo5-YrpHgh0IPuAPWD21rM6U17hW/view?usp=sharing 에서 KITTI 영상 다운로드
  - 이전 실습에서 설치 진행 시 설치를 하지않아도됨
- **업로드 함수 구성 및 업로드 진행**
  - '파일 선택'을 클릭하여 다운로드한 KITTI영상 업로드

```
[ ]  def upload():
       from google.colab import files
       uploaded = files.upload()
       for name, data in uploaded.items():
         with open(name, 'wb') as f:
           f.write(data)
           print('saved file', name)
```



- **영상 업로드 결과 확인**

```
[13]  upload()
```

파일 선택 KITTI_data.mp4

- **KITTI_data.mp4**(video/mp4) - 13302152 bytes, last modified: 2021. 1. 13. - 100% done

Saving KITTI_data.mp4 to KITTI_data.mp4
saved file KITTI_data.mp4

YOLO 영상 데이터 실습

➢ Yolov3 영상 데이터 실습

- **Colab에서는 실시간으로 YOLO 변환된 영상데이터 출력이 안됨**
  - 영상 출력 시도 시 영상 출력이 안되며 YOLO변환 중단
  - -dont_show를 추가하여 YOLO 적용 중 영상 출력 시도 중단
- **YOLO가 적용된 영상 데이터 저장**
  - -out_filename <영상 데이터 이름>
  - 실행 경로에 yolo가 적용된 yolo_video.avi 데이터 생성

```
!./darknet detector demo cfg/coco.data cfg/yolov3.cfg yolov3.weights -dont_show 0015_fps_10.mp4 -out_filename yolo_video.avi
```

▾ yolov3 영상데이터 실습

```
[ ]  !./darknet detector demo cfg/coco.data cfg/yolov3.cfg yolov3.weights -dont_show 0015_fps_10.mp4 -out_filename yolo_video.avi

     Streaming output truncated to the last 5000 lines.
     car: 55%
     car: 45%
     car: 33%
     car: 27%

     FPS:36.0        AVG_FPS:38.8

      cvWriteFrame
     Objects:
```

YOLO 영상 데이터 실습

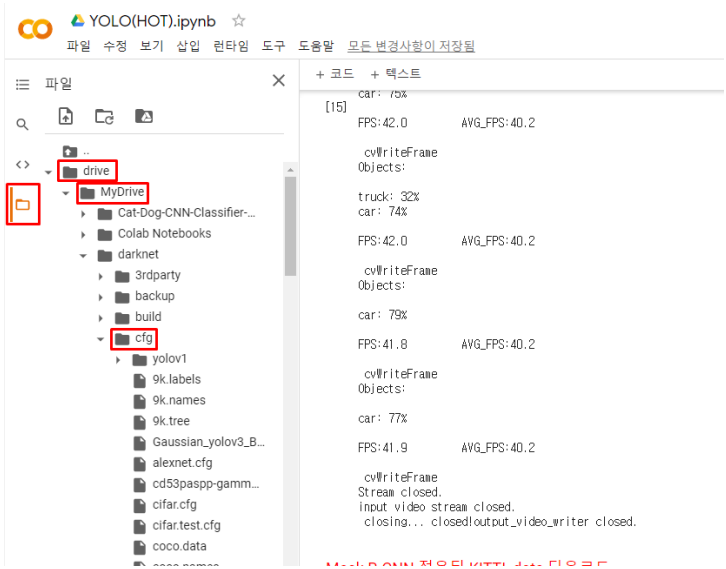
➢ YOLOv3 영상 데이터 실습

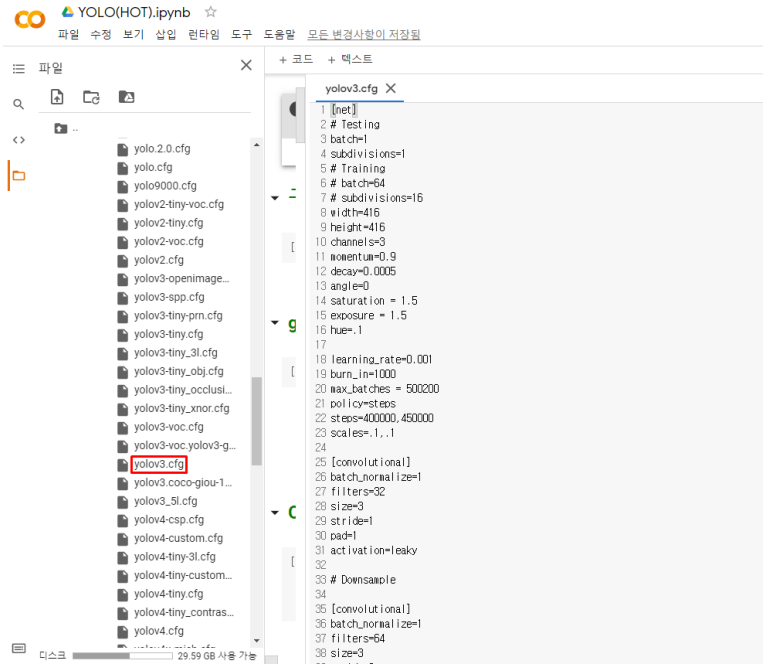
- 영상 데이터에 Yolo 적용 결과

YOLO Custom weight 학습

➢ YOLOv3 custom weight 학습

- **Cfg파일 설정**
  - drive/MyDrive/darknet/cfg/yolov3.cfg 더블클릭





YOLO Custom weight 학습

➢ YOLOv3 custom weight 학습

▪ Cfg파일 설정

- drive/MyDrive/darknet/cfg/yolov3.cfg 더블클릭
- width, height는 32의 배수로 설정
  ✓학습 시 Gpu 메모리 관련 에러 발생하는 경우에는 값을 줄여야 함
- max_batches = 4000*class 수로 설정
  ✓수식이 아닌 계산값 작성
- Batch, subdivisions, learning_rate 등 원하는 설정 값 입력

yolov3.cfg

```
[net]
# Testing
batch=1
subdivisions=1
# Training
# batch=64
# subdivisions=16
width=416
height=416
channels=3
momentum=0.9
decay=0.0005
angle=0
saturation = 1.5
exposure = 1.5
hue=.1

learning_rate=0.001
burn_in=1000
max_batches = 500200
policy=steps
steps=400000,450000
scales=.1,.1

[convolutional]
batch_normalize=1
filters=32
size=3
stride=1
pad=1
activation=leaky

# Downsample

[convolutional]
batch_normalize=1
filters=64
size=3
```

YOLO Custom weight 학습

➢ YOLOv3 custom weight 학습

▪ Cfg파일 설정

- yolov3.cfg 파일 내의 [yolo] 부분 찾기
- Classes의 값을 본인이 학습시킬 class의 수만큼 입력
- [yolo] 위의 [convolutional]의 filters 값을 (classes+5)*3 값으로 수정
  ✓Ex)classes가 20인 경우 filters=75
- [yolo]는 총 3개이므로 해당 부분 모두 수정

```
[convolutional]
size=1
stride=1
pad=1
filters=255
activation=linear


[yolo]
mask = 3,4,5
anchors = 10,13,  16,30,  33,23,  30,61,  62,45,  59,119,  116,90,  156,198,  373,326
classes=80
num=9
jitter=.3
ignore_thresh = .7
truth_thresh = 1
random=1
```

YOLO Custom weight 학습

➢ YOLOv3 custom weight 학습

▪ names 파일 설정

- drive/MyDrive/darknet/cfg/coco.names 파일 수정
  ✓분류할 class명 기입

coco.names

```
1 person
2 bicycle
3 car
4 motorbike
5 aeroplane
6 bus
7 train
8 truck
9 boat
10 traffic light
11 fire hydrant
12 stop sign
13 parking meter
14 bench
15 bird
16 cat
17 dog
18 horse
19 sheep
20 cow
21 elephant
22 bear
23 zebra
24 giraffe
25 backpack
26 umbrella
27 handbag
28 tie
29 suitcase
30 frisbee
31 skis
32 snowboard
33 sports ball
34 kite
35 baseball bat
36 baseball glove
37 skateboard
38 surfboard
```

YOLO Custom weight 학습

➢ YOLOv3 custom weight 학습

▪ data 파일 설정

- drive/MyDrive/darknet/cfg/coco.data 파일 수정
  - ✓classes : class 수
  - ✓train : train data의 경로를 담고있는 파일의 경로
    - train data는 yolo mark를 사용하여 생성 가능
  - ✓valid : validation data의 경로를 담고있는 파일의 경로
  - ✓names : names 파일의 경로
  - ✓backup : 학습한 weight가 저장될 경로

coco.data

```
1 classes= 80
2 train  = data/train.txt
3 #valid  = data/valid.txt
4 #valid = data/coco_val_5k.list
5 names = data/coco.names
6 backup = /home/pjreddie/backup/
7 #eval=coco
8
```