

```
import os
import time
import random
import numpy as np
import pandas as pd
import cv2, torch
from tqdm.auto import tqdm
import shutil as sh
```



```
from IPython.display import Image, clear_output
import matplotlib.pyplot as plt
%matplotlib inline
```

```
!git clone https://github.com/ultralytics/yolov5 # clone repo
!pip install -U pycocotools
!pip install -qr yolov5/requirements.txt # install dependencies
!cp yolov5/requirements.txt ./
```

```
Cloning into 'yolov5'...
remote: Enumerating objects: 16522, done.
remote: Counting objects: 100% (120/120), done.
remote: Compressing objects: 100% (104/104), done.
remote: Total 16522 (delta 50), reused 52 (delta 16), pack-reused 16402
Receiving objects: 100% (16522/16522), 15.18 MiB | 19.03 MiB/s, done.
Resolving deltas: 100% (11310/11310), done.
Requirement already satisfied: pycocotools in /usr/local/lib/python3.10/dist-packages (2.0.7)
Requirement already satisfied: matplotlib>=2.1.0 in /usr/local/lib/python3.10/dist-packages (from pycocotools) (3.7.1)
Requirement already satisfied: numpy in /usr/local/lib/python3.10/dist-packages (from pycocotools) (1.25.2)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib>=2.1.0->pycocotools) (1.2.0)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.10/dist-packages (from matplotlib>=2.1.0->pycocotools) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib>=2.1.0->pycocotools) (4.50.0)
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib>=2.1.0->pycocotools) (1.4.5)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib>=2.1.0->pycocotools) (24.0)
Requirement already satisfied: pillow>=6.2.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib>=2.1.0->pycocotools) (9.4.0)
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib>=2.1.0->pycocotools) (3.1.2)
Requirement already satisfied: python-dateutil>=2.7 in /usr/local/lib/python3.10/dist-packages (from matplotlib>=2.1.0->pycocotools) (2.8.2)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-packages (from python-dateutil>=2.7->matplotlib>=2.1.0->pycocotools) (1.16.0)

195.4/195.4 kB 4.0 MB/s eta 0:00:00
723.1/723.1 kB 9.7 MB/s eta 0:00:00
62.7/62.7 kB 6.9 MB/s eta 0:00:00
23.7/23.7 MB 36.6 MB/s eta 0:00:00
823.6/823.6 kB 49.2 MB/s eta 0:00:00
14.1/14.1 MB 64.3 MB/s eta 0:00:00
731.7/731.7 MB 1.1 MB/s eta 0:00:00
410.6/410.6 MB 2.6 MB/s eta 0:00:00
121.6/121.6 MB 8.4 MB/s eta 0:00:00
56.5/56.5 MB 14.2 MB/s eta 0:00:00
124.2/124.2 MB 8.5 MB/s eta 0:00:00
196.0/196.0 MB 2.9 MB/s eta 0:00:00
166.0/166.0 MB 7.5 MB/s eta 0:00:00
99.1/99.1 kB 11.5 MB/s eta 0:00:00
21.1/21.1 MB 58.9 MB/s eta 0:00:00
```

```
img_h, img_w, num_channels = (380, 676, 3)
df = pd.read_csv('../input/car-object-detection/data/train_solution_bounding_boxes (1).csv')
df.rename(columns={'image': 'image_id'}, inplace=True) #rename the image column as the convention in yolo5
df['image_id'] = df['image_id'].apply(lambda x: x.split('.')[0])
# remove the file extension from the image id column
df['x_center'] = (df['xmin'] + df['xmax'])/2 # center coordinate calculations
df['y_center'] = (df['ymin'] + df['ymax'])/2
df['w'] = df['xmax'] - df['xmin'] # width and height calculation
df['h'] = df['ymax'] - df['ymin']
df['classes'] = 0
df['x_center'] = df['x_center']/img_w # Normalization of coordinates they should be between 0 and 1
df['w'] = df['w']/img_w
df['y_center'] = df['y_center']/img_h
df['h'] = df['h']/img_h
df.head()
```

	image_id	xmin	ymin	xmax	ymax	x_center	y_center	w	h	classes		
0	vid_4_1000	281.259045	187.035071	327.727931	223.225547	0.450434	0.539817	0.068741	0.095238	0		
1	vid_4_10000	15.163531	187.035071	120.329957	236.430180	0.100217	0.557191	0.155572	0.129987	0		
2	vid_4_10040	239.192475	176.764801	361.968162	236.430180	0.444645	0.543678	0.181621	0.157014	0		
3	vid_4_10020	496.483358	172.363256	630.020260	231.539575	0.833213	0.531451	0.197540	0.155727	0		
4	vid_4_10060	16.630970	186.546010	132.558611	238.386422	0.110347	0.559122	0.171491	0.136422	0		

Next steps: [Generate code with df](#) [View recommended plots](#)

lets display a random imgae

```
index = list(set(df.image_id))
image = random.choice(index)
print("Image ID: %s"%(image))
img = cv2.imread(f'/kaggle/input/car-object-detection/data/training_images/{image}.jpg')
img.shape

Image ID: vid_4_600
(380, 676, 3)

image = random.choice(index)
Image(filename=f'/kaggle/input/car-object-detection/data/training_images/{image}.jpg',width=600)
```



```
source = 'training_images'
if True:
    for fold in [0]:
        val_index = index[len(index)*fold//5:len(index)*(fold+1)//5]
        for name,mini in tqdm(df.groupby('image_id')):
            if name in val_index:
                path2save = 'val2017/'
            else:
                path2save = 'train2017/'
            if not os.path.exists('/tmp/convertor/fold{}/labels/'.format(fold)+path2save):
                os.makedirs('/tmp/convertor/fold{}/labels/'.format(fold)+path2save)
            with open('/tmp/convertor/fold{}/labels/'.format(fold)+path2save+name+".txt", 'w+') as f:
                row = mini[['classes','x_center','y_center','w','h']].astype(float).values
                row = row.astype(str)
                for j in range(len(row)):
                    text = ' '.join(row[j])
                    f.write(text)
                    f.write("\n")
            if not os.path.exists('/tmp/convertor/fold{}/images/'.format(fold,path2save)):
                os.makedirs('/tmp/convertor/fold{}/images/'.format(fold,path2save))
            sh.copy("/kaggle/input/car-object-detection/data/{}/{}.jpg".format(source,name), '/tmp/convertor/fold{}/images/{}/{}.jpg'.format(
100% 355/355 [00:01<00:00, 352.73it/s]
```

```
!python yolov5/detect.py --weights yolov5/yolov5s.pt --img 676 --conf 0.4 --source /kaggle/input/car-object-detection/data/testing_images
# this part of the code determines the directories for the yolofile the weights to be loaded,
# setting the input image sizes, and confidence score setup
```

```
detect: weights=['yolov5/yolov5s.pt'], source=/kaggle/input/car-object-detection/data/testing_images, data=yolov5/data/coco128.yaml,
YOLOv5 v7.0-295-gac6c4383 Python-3.10.12 torch-2.2.1+cu121 CPU
```

```
Downloading https://github.com/ultralytics/yolov5/releases/download/v7.0/yolov5s.pt to yolov5/yolov5s.pt...
100% 14.1M/14.1M [00:00<00:00, 126MB/s]
```

```
Fusing layers...
```

```
YOLOv5s summary: 213 layers, 7225885 parameters, 0 gradients, 16.4 GFLOPs
```

```
WARNING ⚠ --img-size [676, 676] must be multiple of max stride 32, updating to [704, 704]
```

```
image 1/175 /kaggle/input/car-object-detection/data/testing_images/vid_5_25100.jpg: 416x704 (no detections), 315.9ms
image 2/175 /kaggle/input/car-object-detection/data/testing_images/vid_5_25120.jpg: 416x704 (no detections), 303.4ms
image 3/175 /kaggle/input/car-object-detection/data/testing_images/vid_5_25140.jpg: 416x704 (no detections), 282.2ms
image 4/175 /kaggle/input/car-object-detection/data/testing_images/vid_5_25160.jpg: 416x704 (no detections), 304.4ms
image 5/175 /kaggle/input/car-object-detection/data/testing_images/vid_5_25180.jpg: 416x704 (no detections), 298.8ms
image 6/175 /kaggle/input/car-object-detection/data/testing_images/vid_5_25200.jpg: 416x704 (no detections), 309.9ms
image 7/175 /kaggle/input/car-object-detection/data/testing_images/vid_5_25220.jpg: 416x704 (no detections), 302.5ms
image 8/175 /kaggle/input/car-object-detection/data/testing_images/vid_5_25240.jpg: 416x704 (no detections), 297.3ms
image 9/175 /kaggle/input/car-object-detection/data/testing_images/vid_5_25260.jpg: 416x704 (no detections), 311.1ms
image 10/175 /kaggle/input/car-object-detection/data/testing_images/vid_5_26320.jpg: 416x704 (no detections), 289.1ms
image 11/175 /kaggle/input/car-object-detection/data/testing_images/vid_5_26400.jpg: 416x704 (no detections), 295.1ms
image 12/175 /kaggle/input/car-object-detection/data/testing_images/vid_5_26420.jpg: 416x704 (no detections), 308.6ms
image 13/175 /kaggle/input/car-object-detection/data/testing_images/vid_5_26560.jpg: 416x704 1 car, 302.1ms
image 14/175 /kaggle/input/car-object-detection/data/testing_images/vid_5_26580.jpg: 416x704 1 car, 290.3ms
image 15/175 /kaggle/input/car-object-detection/data/testing_images/vid_5_26600.jpg: 416x704 2 cars, 456.8ms
image 16/175 /kaggle/input/car-object-detection/data/testing_images/vid_5_26620.jpg: 416x704 1 car, 467.5ms
image 17/175 /kaggle/input/car-object-detection/data/testing_images/vid_5_26640.jpg: 416x704 3 cars, 462.9ms
image 18/175 /kaggle/input/car-object-detection/data/testing_images/vid_5_26660.jpg: 416x704 1 car, 466.8ms
image 19/175 /kaggle/input/car-object-detection/data/testing_images/vid_5_26680.jpg: 416x704 3 cars, 1 truck, 480.6ms
image 20/175 /kaggle/input/car-object-detection/data/testing_images/vid_5_26700.jpg: 416x704 2 cars, 393.4ms
image 21/175 /kaggle/input/car-object-detection/data/testing_images/vid_5_26720.jpg: 416x704 3 cars, 291.5ms
image 22/175 /kaggle/input/car-object-detection/data/testing_images/vid_5_26740.jpg: 416x704 3 cars, 303.8ms
image 23/175 /kaggle/input/car-object-detection/data/testing_images/vid_5_26760.jpg: 416x704 3 cars, 299.7ms
image 24/175 /kaggle/input/car-object-detection/data/testing_images/vid_5_26780.jpg: 416x704 2 cars, 1 truck, 299.3ms
image 25/175 /kaggle/input/car-object-detection/data/testing_images/vid_5_26800.jpg: 416x704 2 cars, 1 truck, 303.9ms
image 26/175 /kaggle/input/car-object-detection/data/testing_images/vid_5_26820.jpg: 416x704 2 cars, 296.2ms
image 27/175 /kaggle/input/car-object-detection/data/testing_images/vid_5_26840.jpg: 416x704 1 car, 294.5ms
image 28/175 /kaggle/input/car-object-detection/data/testing_images/vid_5_26860.jpg: 416x704 1 car, 297.6ms
image 29/175 /kaggle/input/car-object-detection/data/testing_images/vid_5_26880.jpg: 416x704 1 car, 317.9ms
image 30/175 /kaggle/input/car-object-detection/data/testing_images/vid_5_26900.jpg: 416x704 3 cars, 302.2ms
image 31/175 /kaggle/input/car-object-detection/data/testing_images/vid_5_26920.jpg: 416x704 2 cars, 295.6ms
image 32/175 /kaggle/input/car-object-detection/data/testing_images/vid_5_26940.jpg: 416x704 2 cars, 321.7ms
image 33/175 /kaggle/input/car-object-detection/data/testing_images/vid_5_26960.jpg: 416x704 1 car, 299.2ms
image 34/175 /kaggle/input/car-object-detection/data/testing_images/vid_5_26980.jpg: 416x704 (no detections), 297.9ms
image 35/175 /kaggle/input/car-object-detection/data/testing_images/vid_5_27240.jpg: 416x704 1 car, 311.2ms
image 36/175 /kaggle/input/car-object-detection/data/testing_images/vid_5_27260.jpg: 416x704 (no detections), 293.7ms
image 37/175 /kaggle/input/car-object-detection/data/testing_images/vid_5_27280.jpg: 416x704 (no detections), 300.8ms
image 38/175 /kaggle/input/car-object-detection/data/testing_images/vid_5_27300.jpg: 416x704 3 cars, 3 traffic lights, 310.8ms
image 39/175 /kaggle/input/car-object-detection/data/testing_images/vid_5_27320.jpg: 416x704 3 cars, 4 traffic lights, 305.4ms
image 40/175 /kaggle/input/car-object-detection/data/testing_images/vid_5_27360.jpg: 416x704 3 cars, 5 traffic lights, 291.5ms
image 41/175 /kaggle/input/car-object-detection/data/testing_images/vid_5_27380.jpg: 416x704 3 cars, 5 traffic lights, 296.3ms
image 42/175 /kaggle/input/car-object-detection/data/testing_images/vid_5_27400.jpg: 416x704 3 cars, 4 traffic lights, 295.6ms
image 43/175 /kaggle/input/car-object-detection/data/testing_images/vid_5_27420.jpg: 416x704 3 cars, 3 traffic lights, 290.8ms
image 44/175 /kaggle/input/car-object-detection/data/testing_images/vid_5_27440.jpg: 416x704 3 cars, 2 traffic lights, 289.8ms
image 45/175 /kaggle/input/car-object-detection/data/testing_images/vid_5_27460.jpg: 416x704 2 cars, 2 traffic lights, 297.6ms
image 46/175 /kaggle/input/car-object-detection/data/testing_images/vid_5_27480.jpg: 416x704 4 cars, 1 traffic light, 288.1ms
image 47/175 /kaggle/input/car-object-detection/data/testing_images/vid_5_27500.jpg: 416x704 1 car, 286.0ms
image 48/175 /kaggle/input/car-object-detection/data/testing_images/vid_5_27520.jpg: 416x704 2 cars, 307.9ms
```

```
predicted_files = []
for (dirpath, dirnames, filenames) in os.walk("runs/detect/py"):
    predicted_files.extend(filenames)
```

```
Image(filename='/content/yolov5/runs/detect/exp/vid_5_26640.jpg')
```



Image(filename='/content/yolov5/runs/detect/exp/vid_5_26560.jpg')



Image(filename='/content/yolov5/runs/detect/exp/vid_5_25120.jpg')



Image(filename='/content/yolov5/runs/detect/exp/vid_5_26920.jpg')



Image(filename='/_content/yolov5/runs/detect/exp/vid_5_26880.jpg')

