در این پروژه از ۳ دیتاست متفاوت استفاده شده است که لینک‌های مرتبط با این دیتاست‌ها در ادامه درج شده است. به طور کلی ۳۰۰۰عکس داریم.

https://www.kaggle.com/datasets/andrewmvd/car-plate-detection

https://www.kaggle.com/datasets/skhalili/iraniancarnumberplate?resource=download

https://github.com/roozbehrajabi/ALPR_Dataset

با استفاده از کد rename_objects_in_xml.py تمام کلاس‌های موجود در annotation های عکس‌ها یکسان شده است و تنها یک کلاس به نام Licence وجود دارد.

annotation مربوط به بعضی دیتاست‌ها از نوع xml است که برای اینکه بتوانیم در Yolov7 استفاده کنیم باید آن‌ها را تبدیل به txt کنیم. که با استفاده از کد convert_voc_to_yolo.py این تبدیل صورت گرفته است.

دیتاست موجود به نسبت ۸۰ و۱۰و۱۰ به مجموعه train و validation و test تقسیم شده است.

در این پروژه دو قسمت وجود دارد. قسمت اول مربوط به Detaction پلاک ها است و بخش دوم مربوط به Recognition متن پلاک است.

قسمت اول را به وسیله Yolov7 پیاده‌سازی کردیم.

کد زیر مربوط به نصب Yolov7 و پیش‌نیازهای آن است.

```
%cd /content/gdrive/MyDrive
# Download YOLOv7
!git clone https://github.com/WongKinYiu/yolov7 # clone repo
%cd yolov7
# Install dependencies
%pip install -qr requirements.txt  # install dependencies
!wget "https://github.com/WongKinYiu/yolov7/releases/download/v0.1/yolov7_training.pt"
%ls
%cd ../
import torch
print(f"Setup complete. Using torch {torch.__version__})")
```

سپس دیتاست مان را دانلود میکنیم.

```
!wget --load-cookies /tmp/cookies.txt "https://docs.google.com/uc?export=download&confirm=$(wget --quiet --save-cookies /tmp/cookies.txt --keep-sessio
```

```
--2023-02-13 12:07:01--  https://docs.google.com/uc?export=download&confirm=t&id=1Vc4ahQhgP5ytJXjEtzODVP9f-u7VCvpP
Resolving docs.google.com (docs.google.com)... 142.250.145.138, 142.250.145.113, 142.250.145.102, ...
Connecting to docs.google.com (docs.google.com)|142.250.145.138|:443... connected.
HTTP request sent, awaiting response... 303 See Other
Location: https://doc-08-1s-docs.googleusercontent.com/docs/securesc/ha0ro937gcuc7l7deffksulhg5h7mbp1/ntu75f3uo2814fcllu5tak3gend9kl51/1676289975000/16
8387786892766705 00/*/1Vc4ahQhgP5ytJXjEtzODVP9f-u7VCvpP?e=download&uuid=f8cb9588-e617-4169-819c-e1eb4d4baf2d [following]
Warning: wildcards not supported in HTTP.
--2023-02-13 12:07:01--  https://doc-08-1s-docs.googleusercontent.com/docs/securesc/ha0ro937gcuc7l7deffksulhg5h7mbp1/ntu75f3uo2814fcllu5tak3gend9kl51/1
676289975000/16838778689276670500/*/1Vc4ahQhgP5ytJXjEtzODVP9f-u7VCvpP?e=download&uuid=f8cb9588-e617-4169-819c-e1eb4d4baf2d
Resolving doc-08-1s-docs.googleusercontent.com (doc-08-1s-docs.googleusercontent.com)... 173.194.79.132, 2a00:1450:4013:c05::84
Connecting to doc-08-1s-docs.googleusercontent.com (doc-08-1s-docs.googleusercontent.com)|173.194.79.132|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 403427063 (385M) [application/x-zip-compressed]
Saving to: ‘LicencePlate_yolo7.zip’

LicencePlate_yolo7. 100%[===================>] 384.74M   226MB/s    in 1.7s

2023-02-13 12:07:03 (226 MB/s) - ‘LicencePlate_yolo7.zip’ saved [403427063/403427063]
```

سپس با استفاده از کد زیر مدل را train می‌کنیم.

```
import os
#os.environ['CUDA_VISIBLE_DEVICES'] = '0'
!python /content/yolov7/train.py --batch 16 --cfg /content/yolov7/cfg/training/yolov7.yaml --epochs 30 --data /content/yolov7/LicencePlate_yolo7/data.
```

خروجی مربوط به بخش train در بخش زیر آورده شده است.

```
     Epoch   gpu_mem       box       obj       cls     total    labels  img_size
     27/29     10.8G   0.02196  0.001948         0   0.02391        31       640: 100% 160/160 [03:01<00:00,  1.14s/it]
               Class      Images      Labels           P           R      mAP@.5  mAP@.5:.95: 100% 10/10 [00:06<00:00,  1.52it/s]
                 all         320         351       0.971       0.949       0.978       0.687

     Epoch   gpu_mem       box       obj       cls     total    labels  img_size
     28/29     10.8G   0.02175   0.00194         0   0.02369        40       640: 100% 160/160 [03:03<00:00,  1.15s/it]
               Class      Images      Labels           P           R      mAP@.5  mAP@.5:.95: 100% 10/10 [00:06<00:00,  1.49it/s]
                 all         320         351       0.965       0.949       0.979       0.695

     Epoch   gpu_mem       box       obj       cls     total    labels  img_size
     29/29     10.8G   0.02111  0.001897         0     0.023        38       640: 100% 160/160 [03:02<00:00,  1.14s/it]
               Class      Images      Labels           P           R      mAP@.5  mAP@.5:.95: 100% 10/10 [00:09<00:00,  1.10it/s]
                 all         320         351       0.968       0.943       0.981       0.698
30 epochs completed in 1.623 hours.

Optimizer stripped from runs/train/exp/weights/last.pt, 74.8MB
Optimizer stripped from runs/train/exp/weights/best.pt, 74.8MB
```

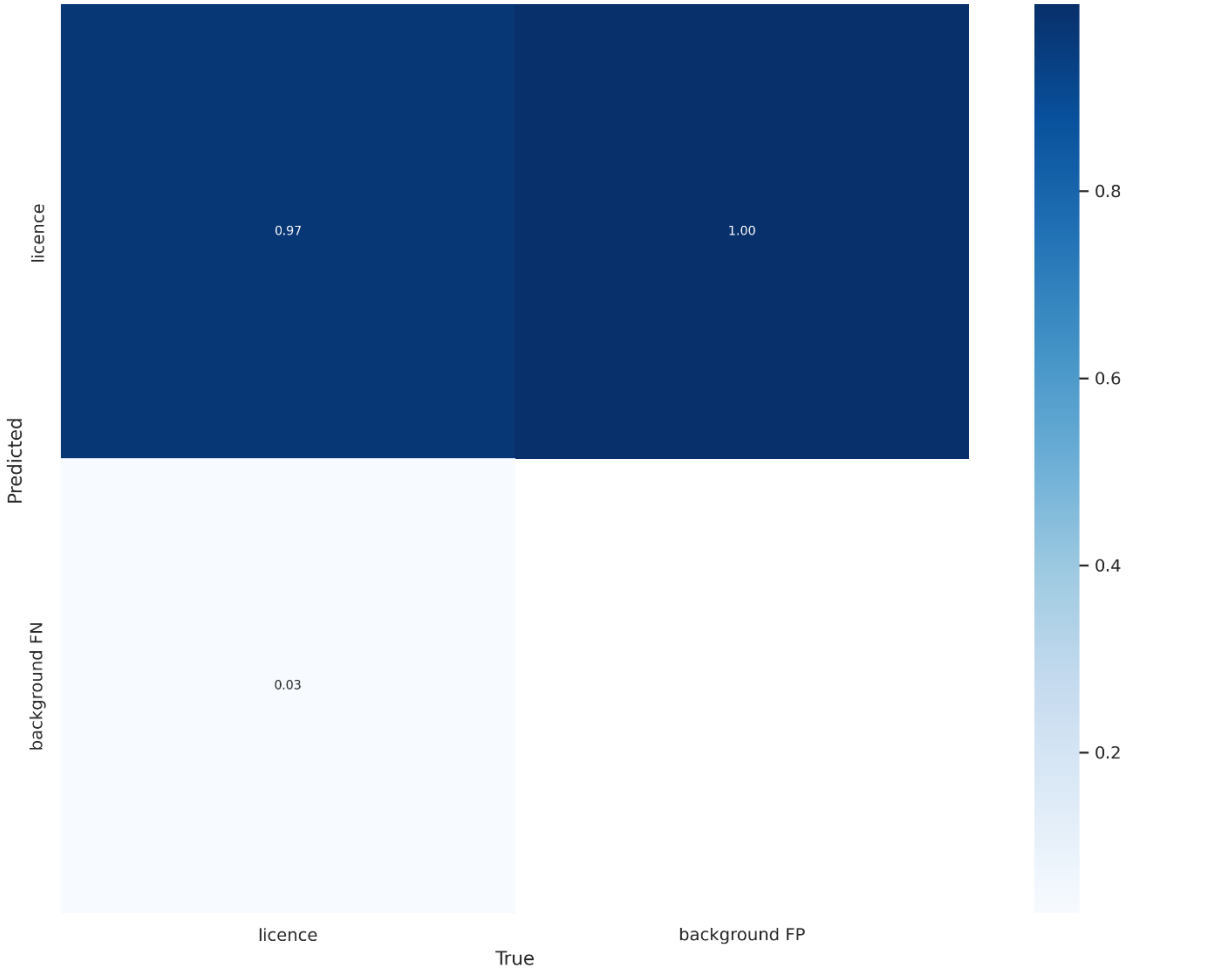
کل زمانی که برای آموزش این مدل با google colab صرف شده ۱/۶ ساعت می‌باشد و وزن مرتبط با مدل در آدرس مشخص شده ذخیره شده است.

در بخش زیر نتایج به دست آمده بارگذاری شده است.

```
display(Image("/content/yolov7/runs/train/exp2/F1_curve.png", width=400, height=400))
display(Image("/content/yolov7/runs/train/exp2/PR_curve.png", width=400, height=400))
display(Image("/content/yolov7/runs/train/exp2/P_curve.png", width=500, height=500))
display(Image("/content/yolov7/runs/train/exp2/R_curve.png", width=500, height=500))
display(Image("/content/yolov7/runs/train/exp2/confusion_matrix.png", width=500, height=500))
```

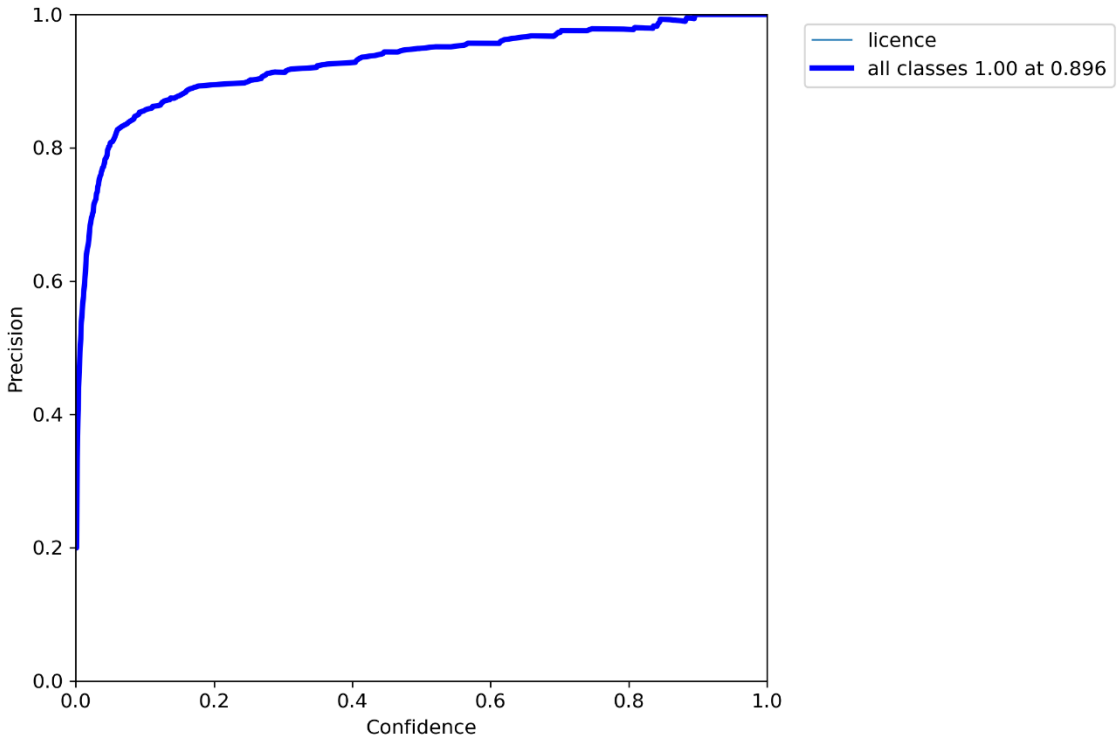
نتایج و دقت به دست آمده:

: Confusion Matrix

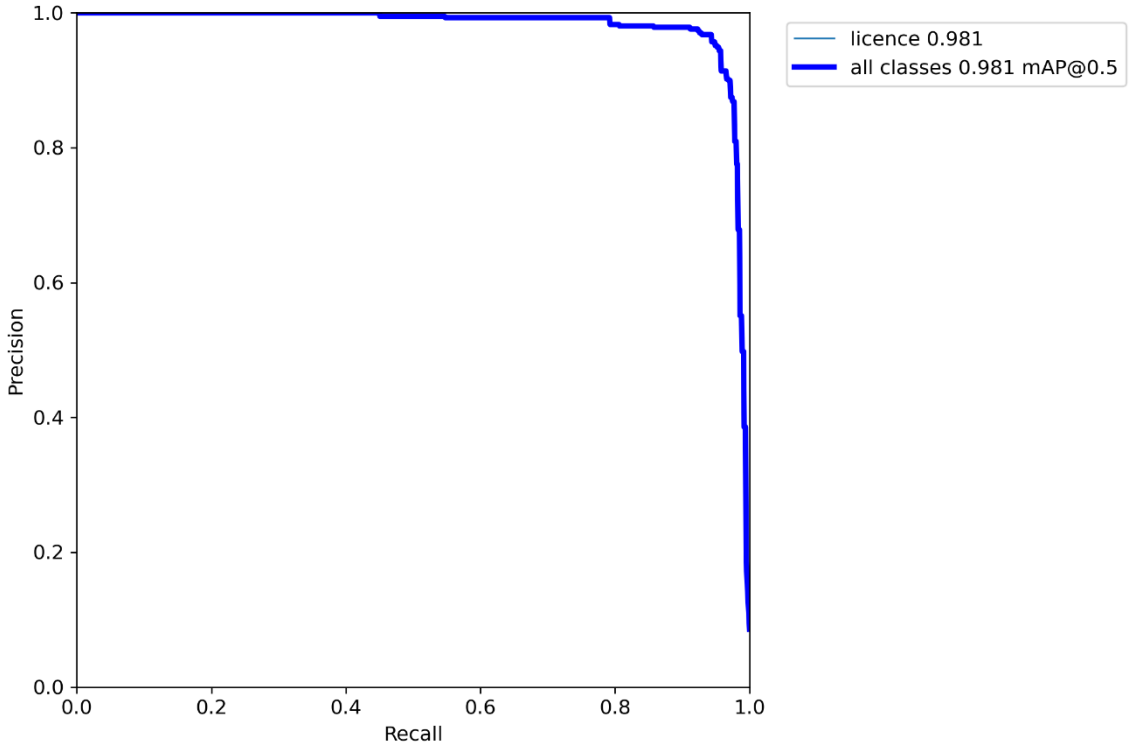


دقت برابر است با ۹۷ درصد.

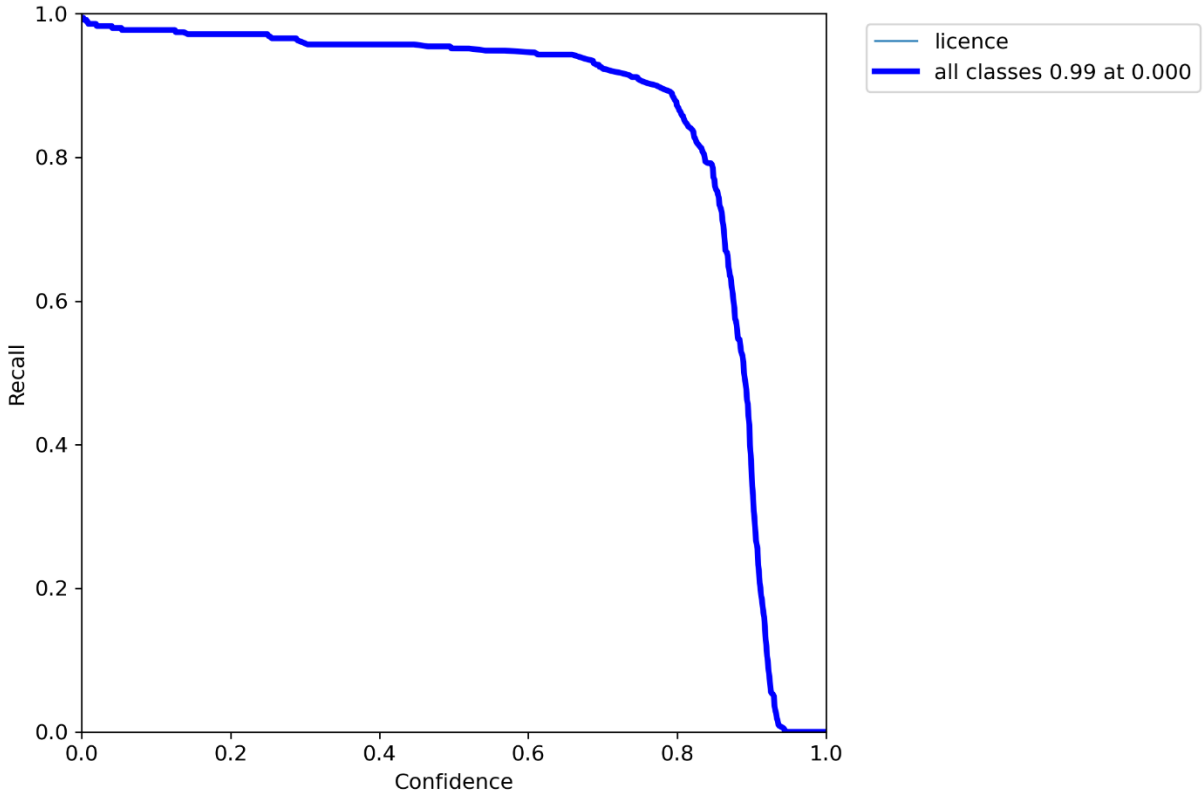
: P_Curve



: PR_Curve



: R_Curve



یک نمونه عکس خروجی:

برای قسمت دوم پروژه روش های متفاوتی برای text Recognition وجود دارد که در اینجا از easyocr استفاده شده است. در این بخش پکیج را دانلود می‌کنیم.

```
%cd /content/yolov7
```

```
/content/yolov7
```

```
!pip install easyocr
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Collecting easyocr
  Downloading easyocr-1.6.2-py3-none-any.whl (2.9 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 2.9/2.9 MB 31.4 MB/s eta 0:00:00
Collecting pyclipper
  Downloading pyclipper-1.3.0.post4-cp38-cp38-manylinux_2_5_x86_64.manylinux1_x86_64.whl (619 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 619.2/619.2 KB 45.3 MB/s eta 0:00:00
Requirement already satisfied: PyYAML in /usr/local/lib/python3.8/dist-packages (from easyocr) (6.0)
Requirement already satisfied: numpy in /usr/local/lib/python3.8/dist-packages (from easyocr) (1.21.6)
Requirement already satisfied: scikit-image in /usr/local/lib/python3.8/dist-packages (from easyocr) (0.18.3)
Requirement already satisfied: Shapely in /usr/local/lib/python3.8/dist-packages (from easyocr) (2.0.1)
Requirement already satisfied: Pillow in /usr/local/lib/python3.8/dist-packages (from easyocr) (7.1.2)
Collecting python-bidi
  Downloading python_bidi-0.4.2-py2.py3-none-any.whl (30 kB)
Requirement already satisfied: torchvision>=0.5 in /usr/local/lib/python3.8/dist-packages (from easyocr) (0.14.1+cu116)
```

و در این بخش import می‌کنیم.

```
import easyocr
reader = easyocr.Reader(['fa','en']) # this needs to run only once to load the model into memory
```

```
WARNING:easyocr.easyocr:Downloading detection model, please wait. This may take several minutes depending upon your network connection.
Progress: |████████████████████████████████████████| 100.0% Complete
WARNING:easyocr.easyocr:Downloading recognition model, please wait. This may take several minutes depending upon your network connection.
Progress: |████████████████████████████████████████| 100.0% Complete
```

```
#%cd /content/yolov7
import os
from pathlib import Path
from typing import Union
import torch
import cv2 as cv
import numpy as np
from deep_sort_realtime.deepsort_tracker import DeepSort
from models.experimental import attempt_load
from utils.general import check_img_size
from utils.torch_utils import select_device, TracedModel
from utils.datasets import letterbox
from utils.general import non_max_suppression, scale_coords
from utils.plots import plot_one_box, plot_one_box_PIL
from copy import deepcopy
#import easyocr
```

در کد زیر از وزن مدل به دست آمده در بخش قبلی استفاده می‌کنیم.

```
savepath = "/content/yolov7/Plate_Pictures"
weights = '/content/yolov7/runs/train/exp/weights/best.pt'
device_id = 'cpu'
image_size = 640
trace = True

# Initialize
device = select_device(device_id)
half = device.type != 'cpu'  # half precision only supported on CUDA

# Load model
model = attempt_load(weights, map_location=device)  # load FP32 model
stride = int(model.stride.max())  # model stride
imgsz = check_img_size(image_size, s=stride)  # check img_size

if trace:
    model = TracedModel(model, device, image_size)

if half:
    model.half()  # to FP16

if device.type != 'cpu':
    model(torch.zeros(1, 3, imgsz, imgsz).to(device).type_as(next(model.parameters())))  # run once

# Load OCR
reader = easyocr.Reader(['fa','en']) # this needs to run only once to load the model into memory

def detect_plate(source_image):
    # Padded resize
    img_size = 640
    stride = 32
    img = letterbox(source_image, img_size, stride=stride)[0]
```

خروجی این بخش:

```
from PIL import ImageFont, ImageDraw
%matplotlib inline
import matplotlib.pyplot as plt
import matplotlib.image as mpimg
import easyocr
reader = easyocr.Reader(['fa','en']) # this needs to run only once to load the model into memory
image_path = '/content/yolov7/LicencePlate_yolo7/test/images/000060.jpg'

plate_image = cv.imread(image_path)
detected_plate_image = get_plates_from_image(plate_image)
cv.imwrite(os.path.join(savepath, "my_plate_img.png"), detected_plate_image)
print("This is Input Image: ")
image_path_show = mpimg.imread('/content/yolov7/LicencePlate_yolo7/test/images/000060.jpg')
imgplot = plt.imshow(image_path_show)
plt.show()
print("This is Plate After Reognition and Cropping: ")
img_last = mpimg.imread('/content/yolov7/Plate_Pictures/my3_plate_img.png')
imgplot2 = plt.imshow(img_last)
plt.show()
import cv2
img = cv2.imread('/content/yolov7/Plate_Pictures/my_plate_img.png')
result = reader.readtext(img,detail = 0)
print("This is Plate character: ", result)
```

This is Input Image:

This is Plate After Reognition and Cropping:

This is Plate character: ['۶۲ج۲۳۷۱۳']