```
بنام خدا
نام و نا مخانوادگی: پرهام پرخیال
امتحان میانترم هوش مصنوعی
مدرس درس: دکتر ذبیحی فر



import os
HOME = os.getcwd()
print(HOME)
```

ابتدا با ایمپرت کردن او اس برای هدایت کردن دایرکتوری به قسمت کانتنت در کولب استفاده میکنیم

```
!pip install ultralytics==8.0.20

from IPython import display
display.clear_output()

import ultralytics
ultralytics.checks()
```

سپس کتابخانه ی ultralytics دانلود و نصب میکنیم و با فراخوانی display از کتابخانه IPython خروجی این سل از کولب را پاک میکنیم تا مشکلی در سل های دیگر پیش نیاید و بعد کتابخانه را فراخوانی میکنیم و با متد checks() بررسی میکنیم که بدرستی دانلود و ایمپرت شده است

```
from ultralytics import YOLO

from IPython.display import display, Image
```

یولو را فراخوانی میکنیم و دستورات displayو Image را فراخوانی میکنیم در صورت نیاز به نشان دادن تصویر یا وارد کردن تصویری به داخل کد.

```
from IPython.display import clear_output
!pip install kaggle
clear_output()
from google.colab import files

# Upload kaggle.json
uploaded = files.upload()

!mkdir -p ~/.kaggle/ && mv kaggle.json ~/.kaggle/ && chmod 600
~/.kaggle/kaggle.json
```

فایل جیسون کگل را بارگذاری میکنیم در این مقطعکه باید از سایت کگل بصورت کلید اختصاصی در قسمت تنظیمات برداریم و دانلود کنیم و در صفحه ای که با ران کردن این سل باز میشود اپلود کنیم.

```
!kaggle datasets download -d dasmehdixtr/drone-dataset-uav
!unzip /content/drone-dataset-uav.zip
```

دیتا ست را دانلود و انزیپ میکنیم.

```
import os
import shutil
from sklearn.model_selection import train_test_split

# Define the paths
data_dir = '/content/drone_dataset_yolo/dataset_txt'
train_dir = '/content/datasets/train'
test_dir = '/content/datasets/test'
val_dir = '/content/datasets/val'

# Create train, test, and validation directories
os.makedirs(train_dir, exist_ok=True)
os.makedirs(test_dir, exist_ok=True)
os.makedirs(val_dir, exist_ok=True)


# Get the list of image and label files
image_files = [file for file in os.listdir(data_dir) if
file.endswith('.jpg')]  # Adjust the extension as needed
label_files = [file for file in os.listdir(data_dir) if
file.endswith('.txt')]
print(len(image_files), len(label_files))

# Identify label files with corresponding images
valid_label_files = [label_file for label_file in label_files if
label_file.replace('.txt', '.jpg') in image_files]

# Remove invalid label files
invalid_label_files = set(label_files) - set(valid_label_files)

# Split the data into train, test, and validation sets
```

```
train_images, test_val_images, train_labels, test_val_labels =
train_test_split(
    image_files, valid_label_files, test_size=0.3, random_state=42
)

test_images, val_images, test_labels, val_labels = train_test_split(
    test_val_images, test_val_labels, test_size=0.17, random_state=42
)  # 0.2 of the total data for validation and 0.1 for testing

# Move images and labels to their respective directories
def move_files(image_files, label_files, destination_dir):
    os.makedirs(destination_dir, exist_ok=True)
    for img_file, label_file in zip(image_files, label_files):
        shutil.move(os.path.join(data_dir, img_file),
os.path.join(destination_dir, img_file))
        shutil.move(os.path.join(data_dir, label_file),
os.path.join(destination_dir, label_file))

# Move files to train directory
move_files(train_images, train_labels, train_dir)

# Move files to test directory
move_files(test_images, test_labels, test_dir)

# Move files to validation directory
move_files(val_images, val_labels, val_dir)

# Optionally, remove invalid label files from the original directory
for invalid_label_file in invalid_label_files:
    os.remove(os.path.join(data_dir, invalid_label_file))
```

بدلیل انکه دیتا ست تمیز نیست و مشکل دارد و بخش های ان تشکیل نشده است با دستورات بالا ابتدا مسیر را به کولب میدهیم و 3 پوشه مجزا با مسیر دلخوه برای test,valid,train ایجاد میکنیم و با درصدی که مشخص کردیم داده ها را تقسیم میکنیم همچنین چون دیتا 1358 فایل عکس دارد و 1360 لیبل 2 لیبلی که عکس ندارند را پاک میکنیم و در انتها داده ها را به پوشه های ایجاد شده انتقال میدهیم

```
!pwd
%cd {HOME}
```

```
!yolo task=detect mode=train model='/content/best_2.pt'
data='/content/data.yaml' epochs=20 imgsz=640 plots=True save=True
```

یولو را فراخوانی میکنیم و برای دیتکشن میگداریم(میتواند کلسیفیکیشن یا سگمنتیشن هم بگیرد) و نوع بررسی را بروی ترین کردن میگداریم(میتواند تست یا ولیدیشن باشد که در آینده آن را هم مشاهده میکنیم) و سایز تصاویر را بروی 640 میگداریم چون در پراپرتی تصاویر این عرض و ارتفاع را دارند و سیو کردن تشکیل نمدار را هم فعال میکنیم تا در فایل run خود این موارد را هو ذخیره کند برا 20 بار ترین را انجام میدهیم. باید توجه داشت که فایل yaml را برای این دیتا ست باید بسازیم که به فرم زیر می باشد و باید این فایل را برای مسیر دیتا به یولو بدهیم.

```
train: /content/...
val: /content/..
test:/content/...

nc: 1
names: ["Drone"]
# roboflow:
#   workspace: hdetection
#   project: fetal-head-detection
#   version: 3
#   license: CC BY 4.0
detection/dataset/3
```

برای تحلیل ترین از کانفیوژن ماتریکس و ریزالت استفاده میکنیم که در انتها توضیح داده شده به چه فرمی.

```
%cd {HOME}
Image(filename=f'{HOME}/runs/detect/train11/confusion_matrix.png',
width=600)
```

```
%cd {HOME}
Image(filename=f'{HOME}/runs/detect/train11/results.png', width=600)
```

نمونه ای از پردیکشن مدل را بررسی میکنیم و بعد از ان بسراغ داده های تست و ولیدیشن میرویم

```
%cd {HOME}
Image(filename=f'{HOME}/runs/detect/train11/val_batch0_pred.jpg',
width=600)
```

دوباره مدل یولو را فراخوانی میکنیم و اینبار برای ولیدیشن و پردیکشن

```
%cd {HOME}
```

```
!yolo task=detect mode=val
model={HOME}/runs/detect/train11/weights/best.pt data=/content/data.yaml
```

```
%cd {HOME}
!yolo task=detect mode=predict
model={HOME}/runs/detect/train11/weights/best.pt
source=/content/datasets/test save=True
```

سپس با استفاده از کتابخانه ی glob فایل های jpeg موجود در مسیر مشخص شده را پیدا میکنیم و نمایش میدهیم تا ببینیم با چه دقتی محاسبه انجام شده است.

نکته ی حایز اهمیت این است که بعد از ترین شدن داده باید فایل های best.pt از قسمت weight را برداریم تا بتوانیم به کمک ان ها مدل را در واقعیت صحت سنجی کنیم

پس از ران شدن با امتحان کردن تصاویر متفاوت با کد تست که بفرم زیر میباشد میتوانیم مدل ها را با یکدیگر مقایسه کنیم و با کنار دیتای result و confusion matrix گذاشتن میتوانیم بفهمیم که مدل نیاز به ترین بیشتر دارد یا خیر مخصوصا با بررسی maP و loss که ایا به فرم خطی در امده یا شیب رو هنوز حفظ کرده میتوانیم بفهمیم که نیاز به ترین شدن بیشتر دارد مدل یا باید دیتاست تغییر کند یا فاکتور دیگری.

```
import glob
from IPython.display import Image, display

for image_path in glob.glob(f'{HOME}/runs/detect/predict3/*.jpg')[:3]:
      display(Image(filename=image_path, width=600))
      print("\n")
```

```
from ultralytics import YOLO
import os
HOME = os.getcwd()
print(HOME)
%cd {HOME}
model = YOLO('/content/best (2).pt')
import glob
from IPython.display import Image, display
import cv2
from PIL import Image
%cd {HOME}
```

```
# Open an image
img = Image.open('/content/images.jpeg')
# new_size = (640, 640)
# resized_img = img.resize(new_size)
# # Convert the resized image to a NumPy array
# resized_array = np.array(resized_img)

# # Convert RGB to grayscale
# gray_image = cv2.cvtColor(resized_array, cv2.COLOR_BGR2GRAY)


results = model.predict('/content/drone.jpg', save=True, conf=0.2)
cv2_imshow(results)

for result in results:
    boxes = result.boxes  # Boxes object for bbox outputs
    masks = result.masks  # Masks object for segmentation masks outputs
    keypoints = result.keypoints  # Keypoints object for pose outputs
    probs = result.probs  # Probs object for classification outputs
print('number of Drone in Image',len(result.boxes))
```

همچنین با بررسی طول ارایه boxes میتوانیم تعداد پهباد های پیدا شده در تصویر را تشخیص دهیم و با قرار دادن conf (که همون ترش هولد) روی 0.2 میتوانیم نتایج دقیق تی بگیریم.

مدلی که در اینجا استفاده کردیم توسط دیتای دیگر برای 20 بار ترین شده است که دقت خوبی داشته و در این مرحله تاش بر بالا بردن دقت این مدل بوسیله ترین کردن بروی epoch ان را داریم و با مقایسه انجام همین کار بروی مدل yolov8n.pt(بدلیل کمبود وقت از فرمت نانو بجای لارج یا مدیوم استفاده شده است) میتوانیم مشاهده کنیم در عکس هایی که بروی ان تست شده که این مدل دقت بهتری داشته است و با افزایش تعداد دفعات ترین به 40 بار مدل دقت کمتری را از خود نشان می دهد که نشان از اور فیت شدن مدل دارد(تحلیل من) پس بهترین مدل های ترین شده از 4 مدلی که وجود دارد

1. مدلی که بررسی کردیم
2. 2.مدلی که ابتدا با روبوفلو بررسی شد

نکته: علت انتخاب دیتا ست جدید این است که دیتاست قبلی بیشتر از تصاویر پهباد در هوا استفاده کرده بود و با دادن تصاویری از پهباد در نمای جلو ومستقیم تلاش کردیم که مدل را بهینه کنیم(هرچند چون درک عمیقی از نحوه تغغیر دادن وزن ها نداشتیم با بنچ مارک کردن ترین همین دیتاست بروی یولو(فرمت نانو) مقایسه را انجام دادیم و نتیجه گیری کردیم.

.