Object detection using DOTA dataset and training using YOLOV8

ANN NIBANA S L

1 Dataset Preparation and Conversion

- Downloaded **DOTA-v1.0** dataset and extracted image and labelTxt.
- Selected **40 images** with aeroplane using a Python script.
- Used **dotadevkit** to convert the dataset to **COCO format** → generated DOTA_1.0.json.
- Split the dataset:
  - 30 images to images/train
  - 10 images to images/validation
- Created corresponding **COCO-style JSON** files: instances_train.json, instances_val.json

2. Folder Structure and Configuration

Organized dataset as:

mini_train/

├── images/

| ├── train/

| └── val/

├── labels/

| ├── train/

| └── val/

├── annotations/

| ├── instances_train.json

| └── instances_val.json

Created a data.yaml file

3. YOLOv8 Setup, & Training

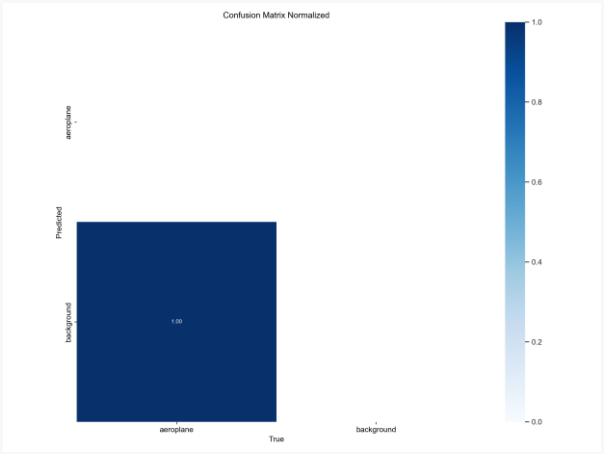
- Installed **Python 3.9** and YOLOv8 using pip install ultralytics.
- Resolved:
  - CUDA compatibility (GPU verified)
- Fixed class ID issues in label .txt files (converted all class IDs to 0).
- Resized all training/validation images to **640×640**.

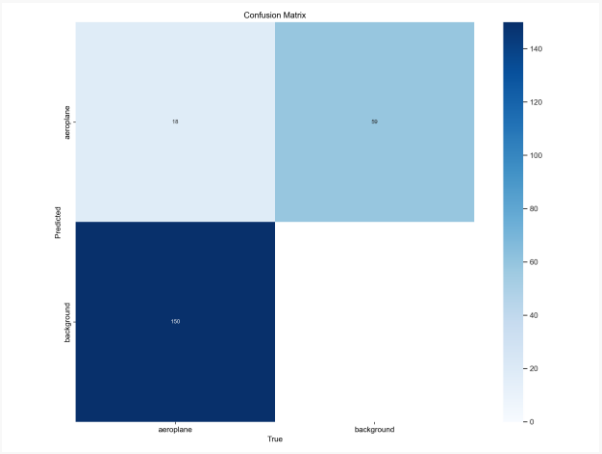
4. Training Models

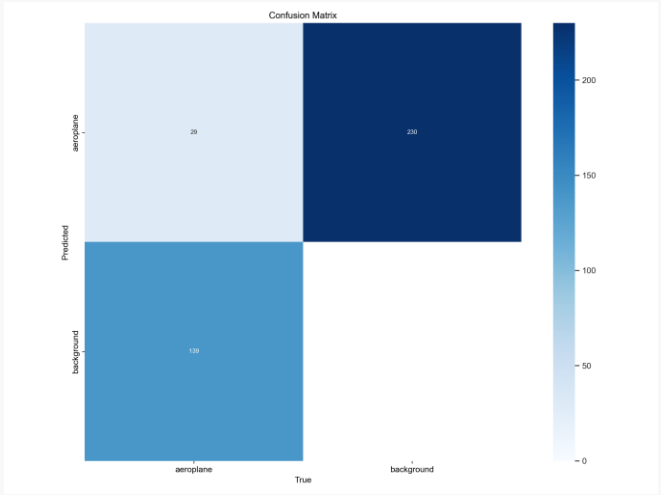
Epoch10(none)

Epoch30(11)

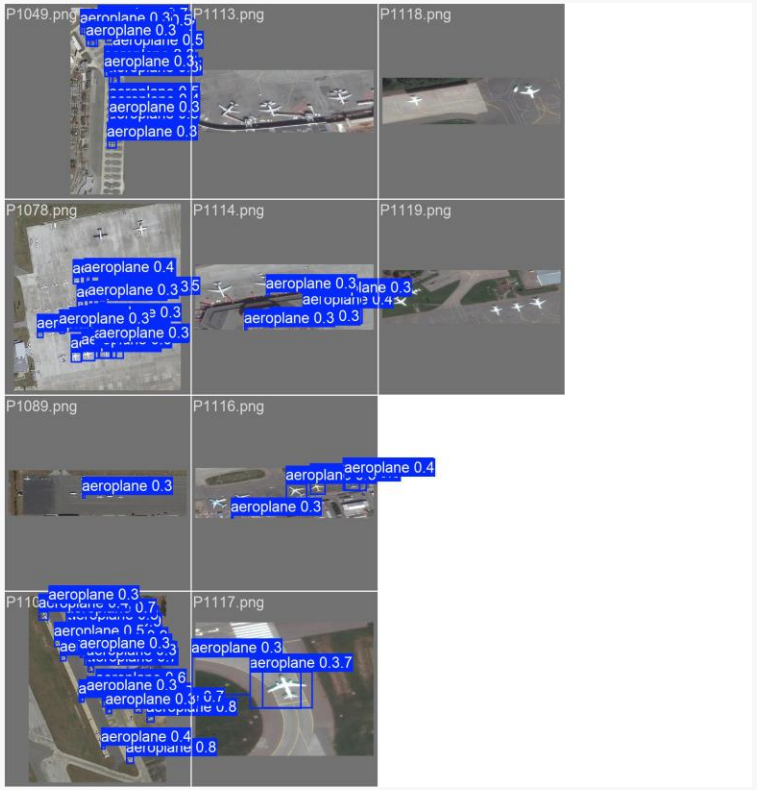
Epoch50(29)

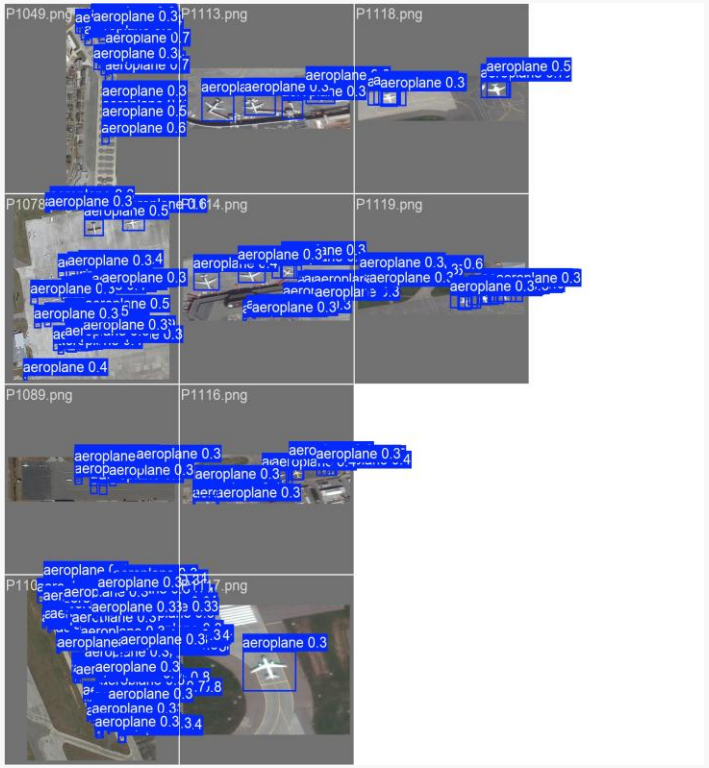







P1049.png
P1113.png
P1118.png
P1078.png
P1114.png
P1119.png
P1089.png
P1116.png
P1109.png
P1117.png


P1049.png
P1113.png
P1118.png
P1078.png
P1114.png
P1119.png
P1089.png
P1116.png
P1117.png


P1049.png
P1113.png
P1118.png
P1078.png
P1114.png
P1119.png
P1089.png
P1116.png
P1117.png

Verification and Display in JupyterLab

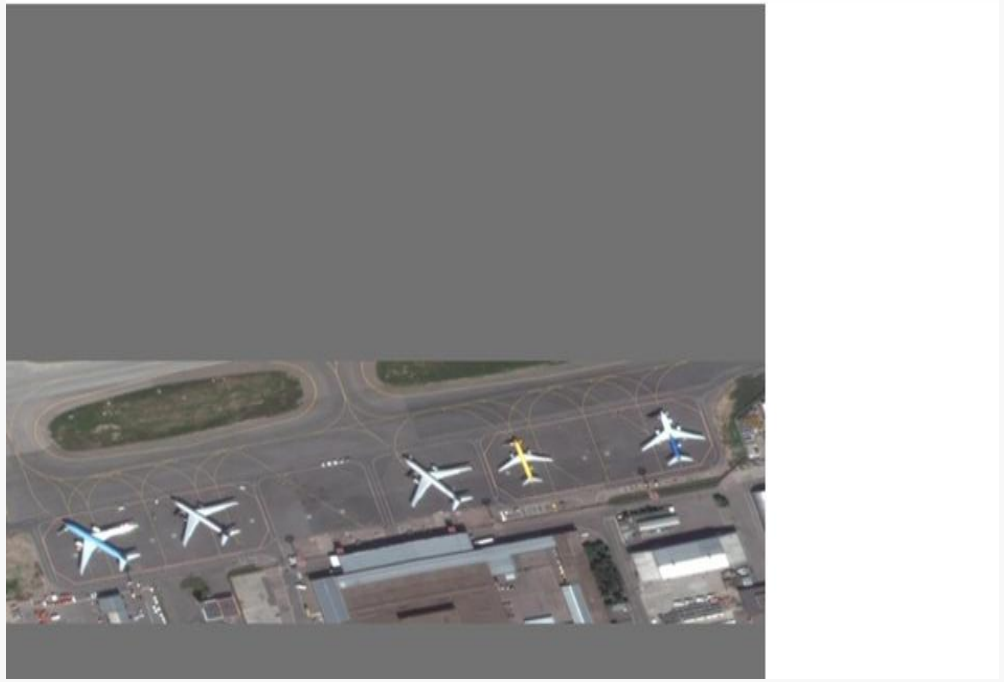
In this JupyterLab environment, a notebook named demo.ipynb is used to visually inspect an input image before making predictions.

The script utilizes IPython's display and Image modules to define the image path (P1116.png) and render it inline within the notebook interface.

Loads and displays a sample input image using the IPython.display module.

Renders aerial imagery inline for visual confirmation before detection.

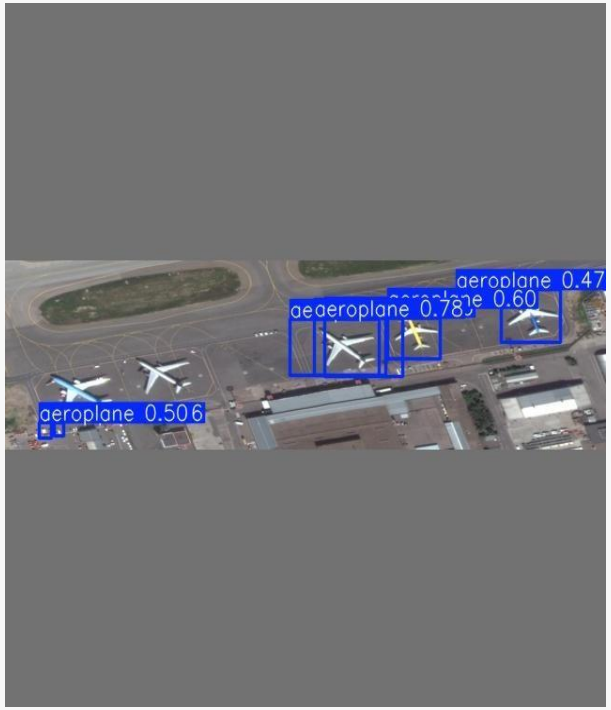
```
from IPython.display import display, Image

# --- Define path to input image ---
image_path = r"C:\Users\annni\Documents\1_Spring_2025\Projectspring\drone_yolo_demo\yolodata\input_images\P1116.png"

# --- Display input image ---
display(Image(filename=image_path))
```

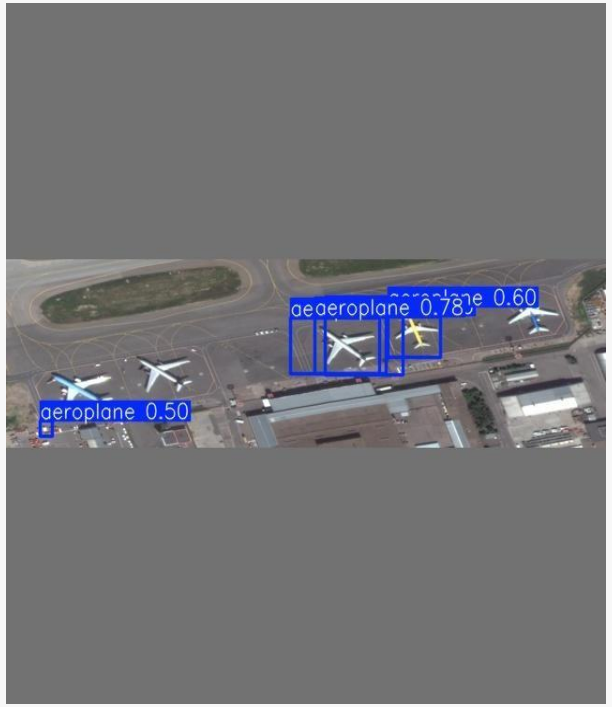


Confidence Threshold Variations

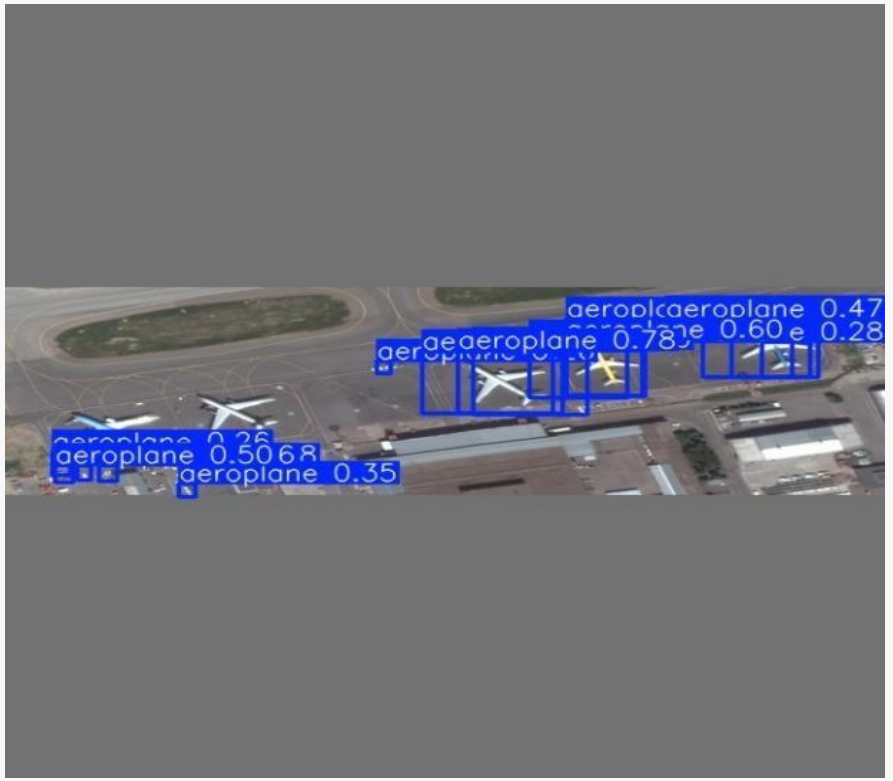
Tested thresholds: 0.25, 0.4, 0.5 on the same input image.
Observation: Lowering the threshold increases sensitivity (more detections) but decreases precision.

0.5: Detects only highly confident objects – fewer boxes, higher precision.
0.4: Balanced detection – includes moderately confident predictions with acceptable accuracy.





0.25: Captures even low-confidence predictions – more boxes, but also more false positives.



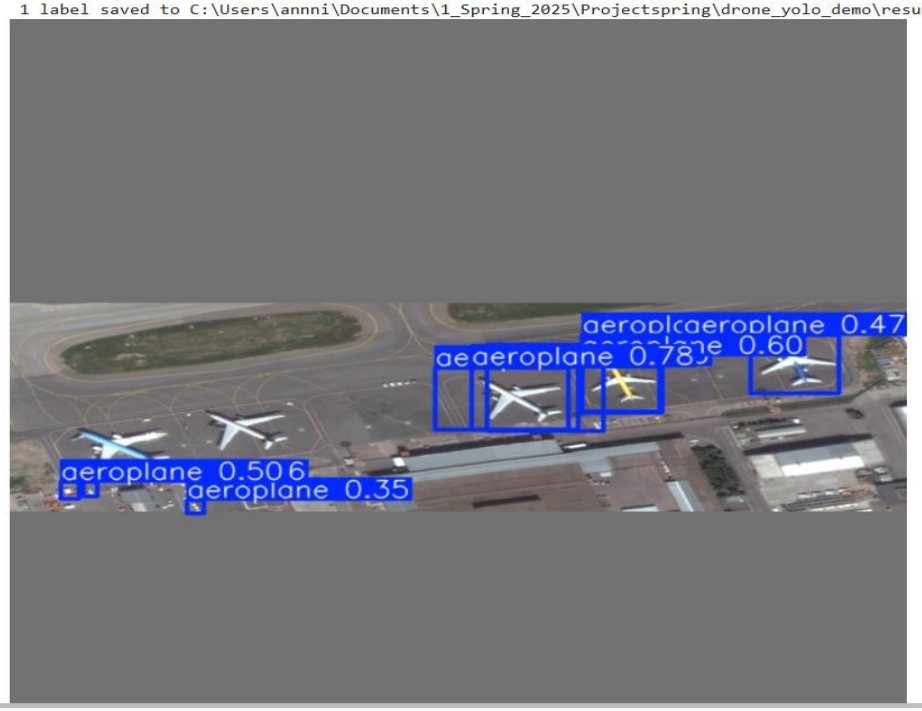
Detection results

Cell 2 performs object detection using a YOLOv8 model within JupyterLab:

- Loads trained YOLOv8 model (best.pt).
- Runs predictions with a confidence threshold of 0.45.
- Displays annotated images inline for visual analysis.

```
[30]: import os
import shutil
from ultralytics import YOLO
from IPython.display import display, Image

# --- Define paths ---
model_path = r"C:\Users\annni\Documents\1_Spring_2025\Projectspring\drone_yolo_demo\weights\best.pt"
output_dir = r"C:\Users\annni\Documents\1_Spring_2025\Projectspring\drone_yolo_demo\results"
predict_name = "predict_output2"
predict_folder = os.path.join(output_dir, predict_name)

# --- Remove old output folder if it exists ---
if os.path.exists(predict_folder):
    shutil.rmtree(predict_folder)

# --- Load YOLOv8 model ---
model = YOLO(model_path)

# --- Run prediction ---
results = model.predict(
    source=image_path,
    save=True,
    save_txt=True,
    project=output_dir,
    name=predict_name,
    imgsz=640,
    conf=0.6
)

# --- Display prediction image ---
predicted_path = os.path.join(results[0].save_dir, os.path.splitext(os.path.basename(image_path))[0] + ".jpg")
display(Image(filename=str(predicted_path)))
```

```
Speed: 9.8ms preprocess, 1196.9ms inference, 4.1ms postprocess per image at shape (1, 3, 640, 640)
Results saved to C:\Users\annni\Documents\1_Spring_2025\Projectspring\drone_yolo_demo\results\predict_output2
1 label saved to C:\Users\annni\Documents\1_Spring_2025\Projectspring\drone_yolo_demo\results\predict_output2\labels
```



Conclusion:

This project demonstrates the application of YOLOv8 for aerial object detection using a subset of the DOTA-v1.0 dataset.

- The workflow involved dataset preparation, COCO format conversion, training the model for 50 epochs, and evaluating detection accuracy.
- Predictions were visualized through annotated outputs and confusion matrices, providing insights into model performance.
- An interactive JupyterLab environment was developed for streamlined testing, visualization, and reproducibility of results.