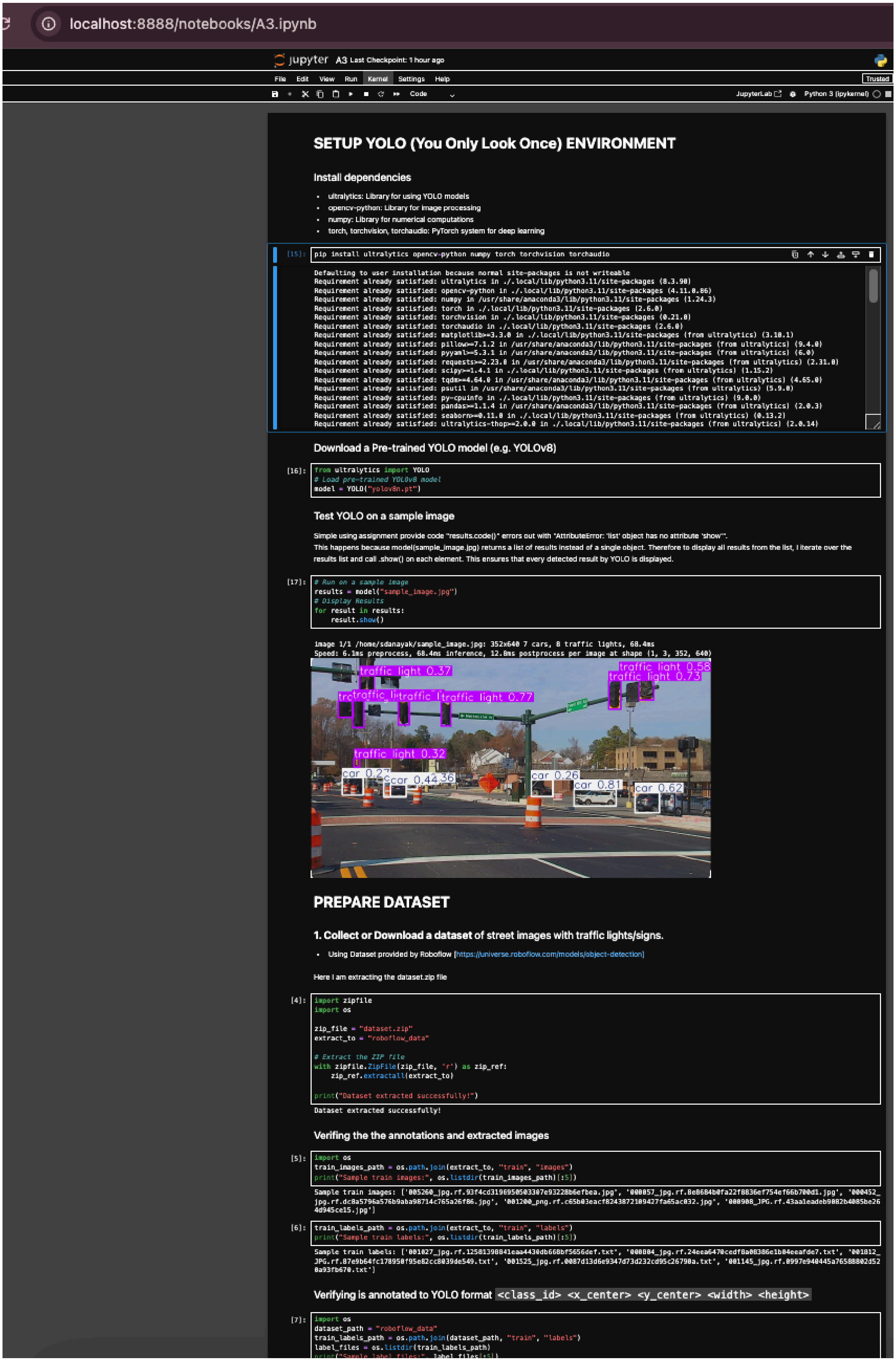


SETUP YOLO (You Only Look Once) ENVIRONMENT

Install dependencies

- ultralytics: Library for using YOLO models
- opencv-python: Library for image processing
- numpy: Library for numerical computations
- torch, torchvision, torchaudio: PyTorch system for deep learning

[15]:
```
pip install ultralytics opencv-python numpy torch torchvision torchaudio
```

```
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: ultralytics in ./.local/lib/python3.11/site-packages (8.3.90)
Requirement already satisfied: opencv-python in ./.local/lib/python3.11/site-packages (4.11.0.86)
Requirement already satisfied: numpy in /usr/share/anaconda3/lib/python3.11/site-packages (1.24.3)
Requirement already satisfied: torch in ./.local/lib/python3.11/site-packages (2.6.0)
Requirement already satisfied: torchvision in ./.local/lib/python3.11/site-packages (0.21.0)
Requirement already satisfied: torchaudio in ./.local/lib/python3.11/site-packages (2.6.0)
Requirement already satisfied: matplotlib>=3.3.0 in ./.local/lib/python3.11/site-packages (from ultralytics) (3.10.1)
Requirement already satisfied: pillow>=7.1.2 in /usr/share/anaconda3/lib/python3.11/site-packages (from ultralytics) (9.4.0)
Requirement already satisfied: pyyaml>=5.3.1 in /usr/share/anaconda3/lib/python3.11/site-packages (from ultralytics) (6.0)
Requirement already satisfied: requests>=2.23.0 in /usr/share/anaconda3/lib/python3.11/site-packages (from ultralytics) (2.31.0)
Requirement already satisfied: scipy>=1.4.1 in ./.local/lib/python3.11/site-packages (from ultralytics) (1.15.2)
Requirement already satisfied: tqdm>=4.64.0 in /usr/share/anaconda3/lib/python3.11/site-packages (from ultralytics) (4.65.0)
Requirement already satisfied: psutil in /usr/share/anaconda3/lib/python3.11/site-packages (from ultralytics) (5.9.0)
Requirement already satisfied: py-cpuinfo in ./.local/lib/python3.11/site-packages (from ultralytics) (9.0.0)
Requirement already satisfied: pandas>=1.1.4 in /usr/share/anaconda3/lib/python3.11/site-packages (from ultralytics) (2.0.3)
Requirement already satisfied: seaborn>=0.11.0 in ./.local/lib/python3.11/site-packages (from ultralytics) (0.13.2)
Requirement already satisfied: ultralytics-thop>=2.0.0 in ./.local/lib/python3.11/site-packages (from ultralytics) (2.0.14)
```

Download a Pre-trained YOLO model (e.g. YOLOv8)

[16]:
```
from ultralytics import YOLO
# Load pre-trained YOLOv8 model
model = YOLO("yolov8n.pt")
```

Test YOLO on a sample image

Simple using assignment provide code "results.code()" errors out with "AttributeError: 'list' object has no attribute 'show'".

This happens because model(sample_image.jpg) returns a list of results instead of a single object. Therefore to display all results from the list, I iterate over the results list and call .show() on each element. This ensures that every detected result by YOLO is displayed.

[17]:
```
# Run on a sample image
results = model("sample_image.jpg")
# Display Results
for result in results:
    result.show()
```

```
image 1/1 /home/sdanayak/sample_image.jpg: 352x640 7 cars, 8 traffic lights, 68.4ms
Speed: 6.1ms preprocess, 68.4ms inference, 12.8ms postprocess per image at shape (1, 3, 352, 640)
```

PREPARE DATASET

1. Collect or Download a dataset of street images with traffic lights/signs.

- Using Dataset provided by Roboflow [https://universe.roboflow.com/models/object-detection]

Here I am extracting the dataset.zip file

[4]:
```
import zipfile
import os

zip_file = "dataset.zip"
extract_to = "roboflow_data"

# Extract the ZIP file
with zipfile.ZipFile(zip_file, 'r') as zip_ref:
    zip_ref.extractall(extract_to)

print("Dataset extracted successfully!")
```

```
Dataset extracted successfully!
```

Verifing the the annotations and extracted images

[5]:
```
import os
train_images_path = os.path.join(extract_to, "train", "images")
print("Sample train images:", os.listdir(train_images_path)[:5])
```

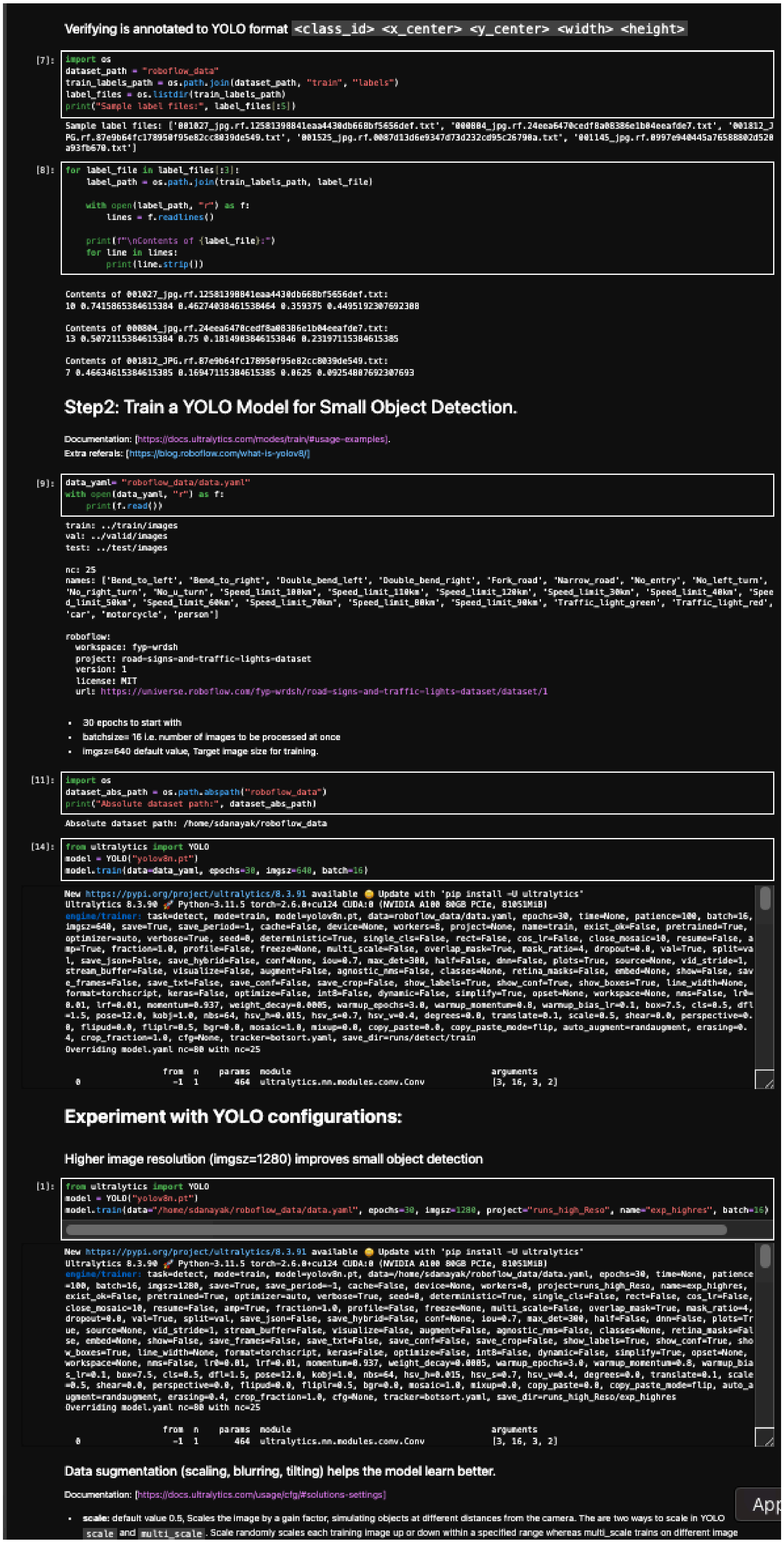
```
Sample train images: ['005260_jpg.rf.93f4cd3196950503307e93228b6efbea.jpg', '000857_jpg.rf.8e8684b0fa22f8836ef754ef66b700d1.jpg', '000452_
jpg.rf.dc8a5796a576b9aba98714c765a26f86.jpg', '001200_png.rf.c65b03eacf8243872109427fa65ac832.jpg', '000908_JPG.rf.43aa1eadeb9882b4085be26
4d945ce15.jpg']
```

[6]:
```
train_labels_path = os.path.join(extract_to, "train", "labels")
print("Sample train labels:", os.listdir(train_labels_path)[:5])
```

```
Sample train labels: ['001027_jpg.rf.12581398841eaa4430db668bf5656def.txt', '000884_jpg.rf.24eea6470cedf8a08386e1b84eeafde7.txt', '001812_
JPG.rf.87e9b64fc178958f95e82cc8039de549.txt', '001525_jpg.rf.0087d13d6e9347d73d232cd95c26798a.txt', '001145_jpg.rf.0997e940445a76588802d52
8a93fb670.txt']
```

Verifying is annotated to YOLO format `<class_id> <x_center> <y_center> <width> <height>`

[7]:
```
import os
dataset_path = "roboflow_data"
train_labels_path = os.path.join(dataset_path, "train", "labels")
label_files = os.listdir(train_labels_path)
print("Sample label files:", label_files[:5])
```



Verifying is annotated to YOLO format `<class_id> <x_center> <y_center> <width> <height>`

[7]:
```
import os
dataset_path = "roboflow_data"
train_labels_path = os.path.join(dataset_path, "train", "labels")
label_files = os.listdir(train_labels_path)
print("Sample label files:", label_files[:5])
```

```
Sample label files: ['001027_jpg.rf.12581398841eaa4430db668bf5656def.txt', '000804_jpg.rf.24eea6470cedf8a08386e1b04eeafde7.txt', '001812_J
PG.rf.87e9b64fc178950f95e82cc8039de549.txt', '001525_jpg.rf.0087d13d6e9347d73d232cd95c26790a.txt', '001145_jpg.rf.0997e940445a76588802d528
a93fb670.txt']
```

[8]:
```
for label_file in label_files[:3]:
    label_path = os.path.join(train_labels_path, label_file)

    with open(label_path, "r") as f:
        lines = f.readlines()

    print(f"\nContents of {label_file}:")
    for line in lines:
        print(line.strip())
```

```
Contents of 001027_jpg.rf.12581398841eaa4430db668bf5656def.txt:
10 0.7415865384615384 0.46274038461538464 0.359375 0.4495192307692308

Contents of 000804_jpg.rf.24eea6470cedf8a08386e1b04eeafde7.txt:
13 0.5072115384615384 0.75 0.1814903846153846 0.23197115384615385

Contents of 001812_JPG.rf.87e9b64fc178950f95e82cc8039de549.txt:
7 0.46634615384615385 0.16947115384615385 0.0625 0.09254807692307693
```

Step2: Train a YOLO Model for Small Object Detection.

Documentation: [https://docs.ultralytics.com/modes/train/#usage-examples].

Extra referals: [https://blog.roboflow.com/what-is-yolov8/]

[9]:
```
data_yaml= "roboflow_data/data.yaml"
with open(data_yaml, "r") as f:
    print(f.read())
```

```
train: ../train/images
val: ../valid/images
test: ../test/images

nc: 25
names: ['Bend_to_left', 'Bend_to_right', 'Double_bend_left', 'Double_bend_right', 'Fork_road', 'Narrow_road', 'No_entry', 'No_left_turn',
'No_right_turn', 'No_u_turn', 'Speed_limit_100km', 'Speed_limit_110km', 'Speed_limit_120km', 'Speed_limit_30km', 'Speed_limit_40km', 'Spee
d_limit_50km', 'Speed_limit_60km', 'Speed_limit_70km', 'Speed_limit_80km', 'Speed_limit_90km', 'Traffic_light_green', 'Traffic_light_red',
'car', 'motorcycle', 'person']

roboflow:
  workspace: fyp-wrdsh
  project: road-signs-and-traffic-lights-dataset
  version: 1
  license: MIT
  url: https://universe.roboflow.com/fyp-wrdsh/road-signs-and-traffic-lights-dataset/dataset/1
```

- 30 epochs to start with
- batchsize= 16 i.e. number of images to be processed at once
- imgsz=640 default value, Target image size for training.

[11]:
```
import os
dataset_abs_path = os.path.abspath("roboflow_data")
print("Absolute dataset path:", dataset_abs_path)
```

```
Absolute dataset path: /home/sdanayak/roboflow_data
```

[14]:
```
from ultralytics import YOLO
model = YOLO("yolov8n.pt")
model.train(data=data_yaml, epochs=30, imgsz=640, batch=16)
```

```
New https://pypi.org/project/ultralytics/8.3.91 available 😃 Update with 'pip install -U ultralytics'
Ultralytics 8.3.90 🚀 Python-3.11.5 torch-2.6.0+cu124 CUDA:0 (NVIDIA A100 80GB PCIe, 81051MiB)
engine/trainer: task=detect, mode=train, model=yolov8n.pt, data=roboflow_data/data.yaml, epochs=30, time=None, patience=100, batch=16,
imgsz=640, save=True, save_period=-1, cache=False, device=None, workers=8, project=None, name=train, exist_ok=False, pretrained=True,
optimizer=auto, verbose=True, seed=0, deterministic=True, single_cls=False, rect=False, cos_lr=False, close_mosaic=10, resume=False, a
mp=True, fraction=1.0, profile=False, freeze=None, multi_scale=False, overlap_mask=True, mask_ratio=4, dropout=0.0, val=True, split=va
l, save_json=False, save_hybrid=False, conf=None, iou=0.7, max_det=300, half=False, dnn=False, plots=True, source=None, vid_stride=1,
stream_buffer=False, visualize=False, augment=False, agnostic_nms=False, classes=None, retina_masks=False, embed=None, show=False, sav
e_frames=False, save_txt=False, save_conf=False, save_crop=False, show_labels=True, show_conf=True, show_boxes=True, line_width=None,
format=torchscript, keras=False, optimize=False, int8=False, dynamic=False, simplify=True, opset=None, workspace=None, nms=False, lr0=
0.01, lrf=0.01, momentum=0.937, weight_decay=0.0005, warmup_epochs=3.0, warmup_momentum=0.8, warmup_bias_lr=0.1, box=7.5, cls=0.5, dfl
=1.5, pose=12.0, kobj=1.0, nbs=64, hsv_h=0.015, hsv_s=0.7, hsv_v=0.4, degrees=0.0, translate=0.1, scale=0.5, shear=0.0, perspective=0.
0, flipud=0.0, fliplr=0.5, bgr=0.0, mosaic=1.0, mixup=0.0, copy_paste=0.0, copy_paste_mode=flip, auto_augment=randaugment, erasing=0.
4, crop_fraction=1.0, cfg=None, tracker=botsort.yaml, save_dir=runs/detect/train
Overriding model.yaml nc=80 with nc=25

                   from  n    params  module                                       arguments
  0                  -1  1       464  ultralytics.nn.modules.conv.Conv             [3, 16, 3, 2]
```

Experiment with YOLO configurations:

Higher image resolution (imgsz=1280) improves small object detection

[1]:
```
from ultralytics import YOLO
model = YOLO("yolov8n.pt")
model.train(data="/home/sdanayak/roboflow_data/data.yaml", epochs=30, imgsz=1280, project="runs_high_Reso", name="exp_highres", batch=16)
```

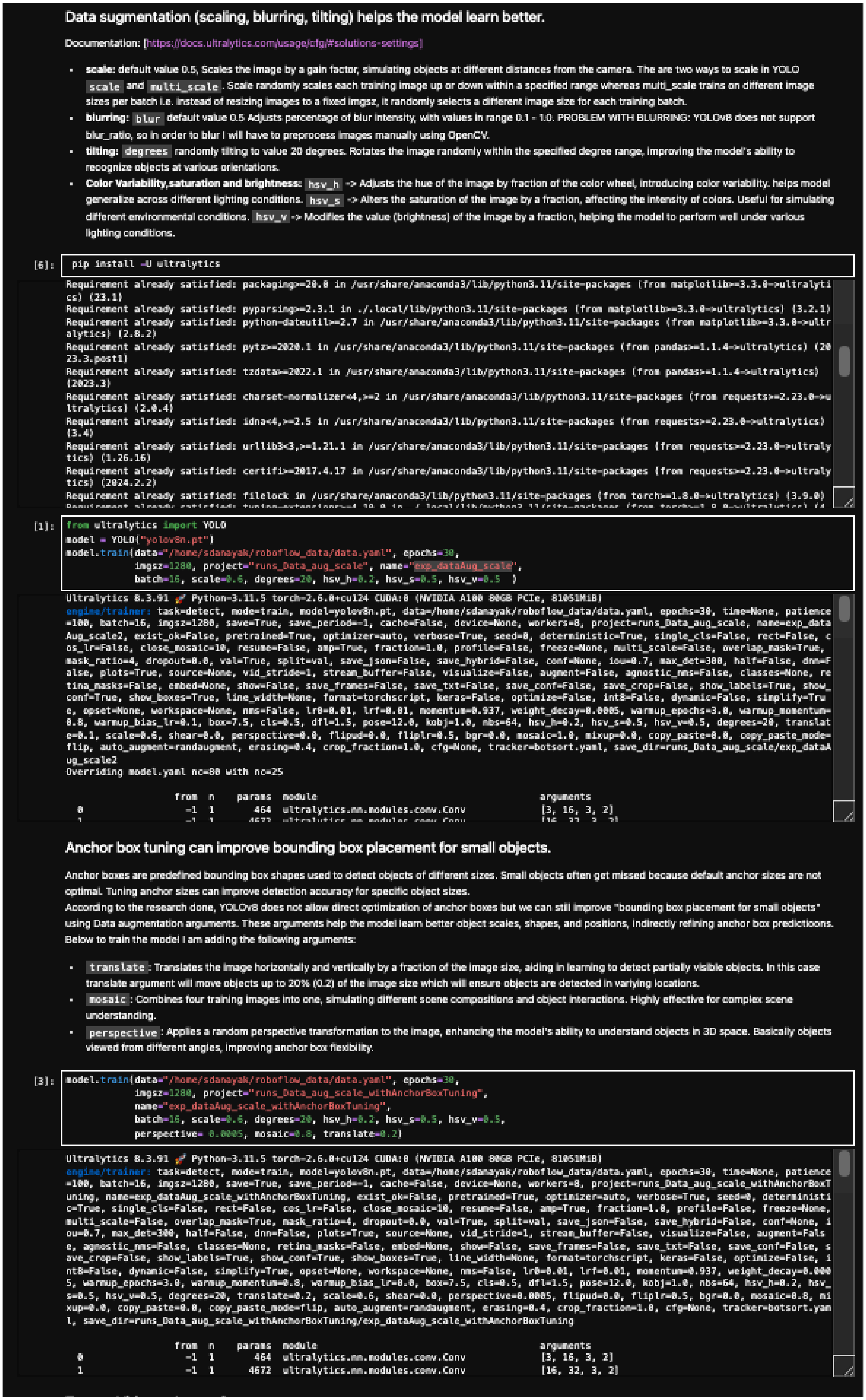
```
New https://pypi.org/project/ultralytics/8.3.91 available 😃 Update with 'pip install -U ultralytics'
Ultralytics 8.3.90 🚀 Python-3.11.5 torch-2.6.0+cu124 CUDA:0 (NVIDIA A100 80GB PCIe, 81051MiB)
engine/trainer: task=detect, mode=train, model=yolov8n.pt, data=/home/sdanayak/roboflow_data/data.yaml, epochs=30, time=None, patience
=100, batch=16, imgsz=1280, save=True, save_period=-1, cache=False, device=None, workers=8, project=runs_high_Reso, name=exp_highres,
exist_ok=False, pretrained=True, optimizer=auto, verbose=True, seed=0, deterministic=True, single_cls=False, rect=False, cos_lr=False,
close_mosaic=10, resume=False, amp=True, fraction=1.0, profile=False, freeze=None, multi_scale=False, overlap_mask=True, mask_ratio=4,
dropout=0.0, val=True, split=val, save_json=False, save_hybrid=False, conf=None, iou=0.7, max_det=300, half=False, dnn=False, plots=Tr
ue, source=None, vid_stride=1, stream_buffer=False, visualize=False, augment=False, agnostic_nms=False, classes=None, retina_masks=Fal
se, embed=None, show=False, save_frames=False, save_txt=False, save_conf=False, save_crop=False, show_labels=True, show_conf=True, sho
w_boxes=True, line_width=None, format=torchscript, keras=False, optimize=False, int8=False, dynamic=False, simplify=True, opset=None,
workspace=None, nms=False, lr0=0.01, lrf=0.01, momentum=0.937, weight_decay=0.0005, warmup_epochs=3.0, warmup_momentum=0.8, warmup_bia
s_lr=0.1, box=7.5, cls=0.5, dfl=1.5, pose=12.0, kobj=1.0, nbs=64, hsv_h=0.015, hsv_s=0.7, hsv_v=0.4, degrees=0.0, translate=0.1, scale
=0.5, shear=0.0, perspective=0.0, flipud=0.0, fliplr=0.5, bgr=0.0, mosaic=1.0, mixup=0.0, copy_paste=0.0, copy_paste_mode=flip, auto_a
ugment=randaugment, erasing=0.4, crop_fraction=1.0, cfg=None, tracker=botsort.yaml, save_dir=runs_high_Reso/exp_highres
Overriding model.yaml nc=80 with nc=25

                   from  n    params  module                                       arguments
  0                  -1  1       464  ultralytics.nn.modules.conv.Conv             [3, 16, 3, 2]
```

Data sugmentation (scaling, blurring, tilting) helps the model learn better.

Documentation: [https://docs.ultralytics.com/usage/cfg/#solutions-settings]

- **scale:** default value 0.5, Scales the image by a gain factor, simulating objects at different distances from the camera. The are two ways to scale in YOLO `scale` and `multi_scale`. Scale randomly scales each training image up or down within a specified range whereas multi_scale trains on different image



Data sugmentation (scaling, blurring, tilting) helps the model learn better.

Documentation: [https://docs.ultralytics.com/usage/cfg/#solutions-settings]

- **scale:** default value 0.5, Scales the image by a gain factor, simulating objects at different distances from the camera. The are two ways to scale in YOLO `scale` and `multi_scale`. Scale randomly scales each training image up or down within a specified range whereas multi_scale trains on different image sizes per batch i.e. instead of resizing images to a fixed imgsz, it randomly selects a different image size for each training batch.
- **blurring:** `blur` default value 0.5 Adjusts percentage of blur intensity, with values in range 0.1 - 1.0. PROBLEM WITH BLURRING: YOLOv8 does not support blur_ratio, so in order to blur I will have to preprocess images manually using OpenCV.
- **tilting:** `degrees` randomly tilting to value 20 degrees. Rotates the image randomly within the specified degree range, improving the model's ability to recognize objects at various orientations.
- **Color Variability,saturation and brightness:** `hsv_h` -> Adjusts the hue of the image by fraction of the color wheel, introducing color variability. helps model generalize across different lighting conditions. `hsv_s` -> Alters the saturation of the image by a fraction, affecting the intensity of colors. Useful for simulating different environmental conditions. `hsv_v` -> Modifies the value (brightness) of the image by a fraction, helping the model to perform well under various lighting conditions.

[6]:

```
pip install -U ultralytics
```

```
Requirement already satisfied: packaging>=20.0 in /usr/share/anaconda3/lib/python3.11/site-packages (from matplotlib>=3.3.0->ultralyti
cs) (23.1)
Requirement already satisfied: pyparsing>=2.3.1 in ./.local/lib/python3.11/site-packages (from matplotlib>=3.3.0->ultralytics) (3.2.1)
Requirement already satisfied: python-dateutil>=2.7 in /usr/share/anaconda3/lib/python3.11/site-packages (from matplotlib>=3.3.0->ultr
alytics) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in /usr/share/anaconda3/lib/python3.11/site-packages (from pandas>=1.1.4->ultralytics) (20
23.3.post1)
Requirement already satisfied: tzdata>=2022.1 in /usr/share/anaconda3/lib/python3.11/site-packages (from pandas>=1.1.4->ultralytics)
(2023.3)
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/share/anaconda3/lib/python3.11/site-packages (from requests>=2.23.0->u
ltralytics) (2.0.4)
Requirement already satisfied: idna<4,>=2.5 in /usr/share/anaconda3/lib/python3.11/site-packages (from requests>=2.23.0->ultralytics)
(3.4)
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/share/anaconda3/lib/python3.11/site-packages (from requests>=2.23.0->ultraly
tics) (1.26.16)
Requirement already satisfied: certifi>=2017.4.17 in /usr/share/anaconda3/lib/python3.11/site-packages (from requests>=2.23.0->ultraly
tics) (2024.2.2)
Requirement already satisfied: filelock in /usr/share/anaconda3/lib/python3.11/site-packages (from torch>=1.8.0->ultralytics) (3.9.0)
Requirement already satisfied: typing-extensions>=4.10.0 in ./.local/lib/python3.11/site-packages (from torch>=1.8.0->ultralytics) (4
```

[1]:

```
from ultralytics import YOLO
model = YOLO("yolov8n.pt")
model.train(data="/home/sdanayak/roboflow_data/data.yaml", epochs=30,
            imgsz=1280, project="runs_Data_aug_scale", name="exp_dataAug_scale",
            batch=16, scale=0.6, degrees=20, hsv_h=0.2, hsv_s=0.5, hsv_v=0.5  )
```

```
Ultralytics 8.3.91 🚀 Python-3.11.5 torch-2.6.0+cu124 CUDA:0 (NVIDIA A100 80GB PCIe, 81051MiB)
engine/trainer: task=detect, mode=train, model=yolov8n.pt, data=/home/sdanayak/roboflow_data/data.yaml, epochs=30, time=None, patience
=100, batch=16, imgsz=1280, save=True, save_period=-1, cache=False, device=None, workers=8, project=runs_Data_aug_scale, name=exp_data
Aug_scale2, exist_ok=False, pretrained=True, optimizer=auto, verbose=True, seed=0, deterministic=True, single_cls=False, rect=False, c
os_lr=False, close_mosaic=10, resume=False, amp=True, fraction=1.0, profile=False, freeze=None, multi_scale=False, overlap_mask=True,
mask_ratio=4, dropout=0.0, val=True, split=val, save_json=False, save_hybrid=False, conf=None, iou=0.7, max_det=300, half=False, dnn=F
alse, plots=True, source=None, vid_stride=1, stream_buffer=False, visualize=False, augment=False, agnostic_nms=False, classes=None, re
tina_masks=False, embed=None, show=False, save_frames=False, save_txt=False, save_conf=False, save_crop=False, show_labels=True, show_
conf=True, show_boxes=True, line_width=None, format=torchscript, keras=False, optimize=False, int8=False, dynamic=False, simplify=Tru
e, opset=None, workspace=None, nms=False, lr0=0.01, lrf=0.01, momentum=0.937, weight_decay=0.0005, warmup_epochs=3.0, warmup_momentum=
0.8, warmup_bias_lr=0.1, box=7.5, cls=0.5, dfl=1.5, pose=12.0, kobj=1.0, nbs=64, hsv_h=0.2, hsv_s=0.5, hsv_v=0.5, degrees=20, translat
e=0.1, scale=0.6, shear=0.0, perspective=0.0, flipud=0.0, fliplr=0.5, bgr=0.0, mosaic=1.0, mixup=0.0, copy_paste=0.0, copy_paste_mode=
flip, auto_augment=randaugment, erasing=0.4, crop_fraction=1.0, cfg=None, tracker=botsort.yaml, save_dir=runs_Data_aug_scale/exp_dataA
ug_scale2
Overriding model.yaml nc=80 with nc=25

                   from  n    params  module                                       arguments
  0                  -1  1       464  ultralytics.nn.modules.conv.Conv             [3, 16, 3, 2]
  1                  -1  1      4672  ultralytics.nn.modules.conv.Conv             [16, 32, 3, 2]
```

Anchor box tuning can improve bounding box placement for small objects.

Anchor boxes are predefined bounding box shapes used to detect objects of different sizes. Small objects often get missed because default anchor sizes are not optimal. Tuning anchor sizes can improve detection accuracy for specific object sizes.
According to the research done, YOLOv8 does not allow direct optimization of anchor boxes but we can still improve "bounding box placement for small objects" using Data augmentation arguments. These arguments help the model learn better object scales, shapes, and positions, indirectly refining anchor box predictioons. Below to train the model I am adding the following arguments:

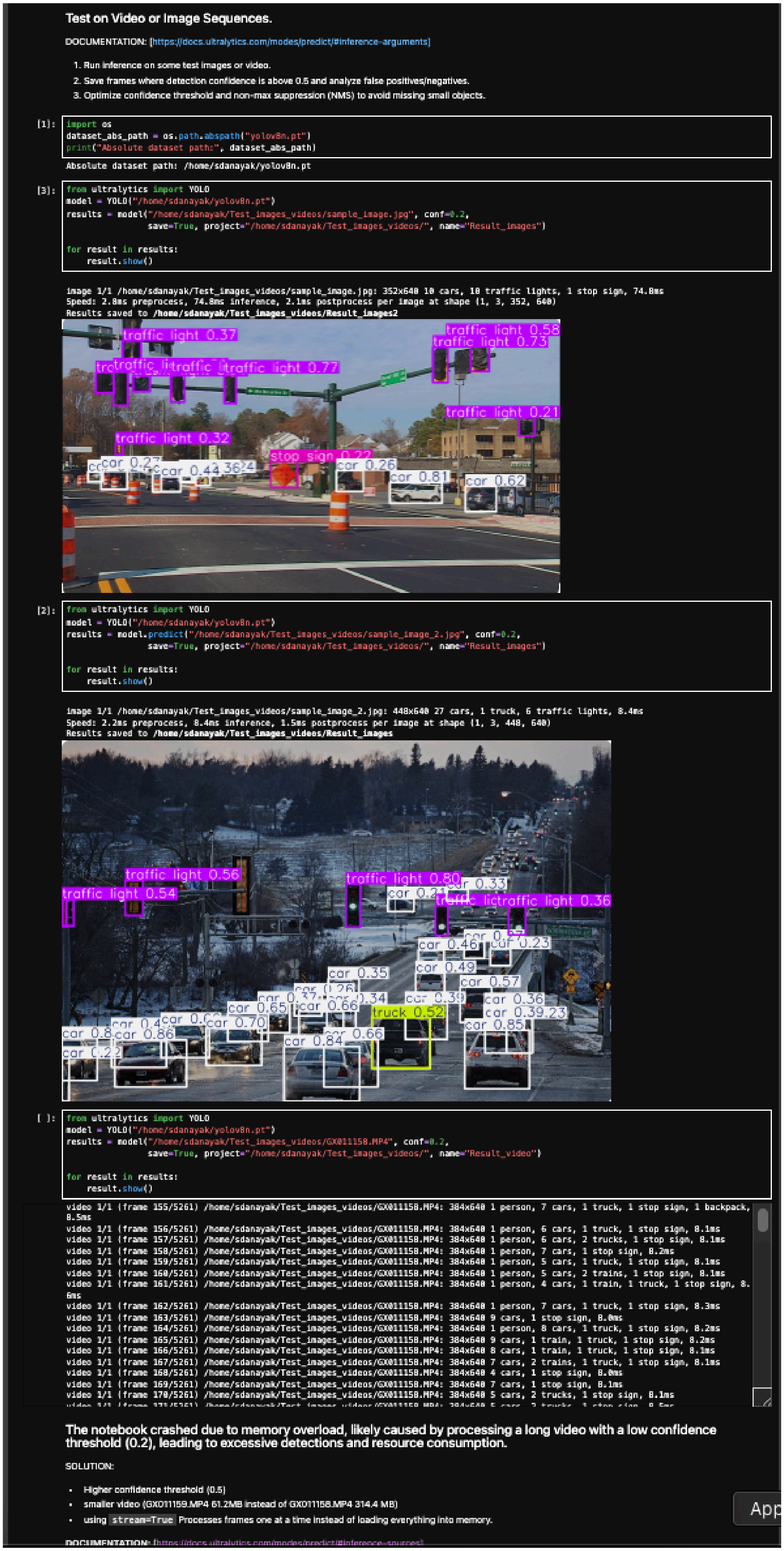
- `translate`: Translates the image horizontally and vertically by a fraction of the image size, aiding in learning to detect partially visible objects. In this case translate argument will move objects up to 20% (0.2) of the image size which will ensure objects are detected in varying locations.
- `mosaic`: Combines four training images into one, simulating different scene compositions and object interactions. Highly effective for complex scene understanding.
- `perspective`: Applies a random perspective transformation to the image, enhancing the model's ability to understand objects in 3D space. Basically objects viewed from different angles, improving anchor box flexibility.

[3]:

```
model.train(data="/home/sdanayak/roboflow_data/data.yaml", epochs=30,
            imgsz=1280, project="runs_Data_aug_scale_withAnchorBoxTuning",
            name="exp_dataAug_scale_withAnchorBoxTuning",
            batch=16, scale=0.6, degrees=20, hsv_h=0.2, hsv_s=0.5, hsv_v=0.5,
            perspective= 0.0005, mosaic=0.8, translate=0.2)
```

```
Ultralytics 8.3.91 🚀 Python-3.11.5 torch-2.6.0+cu124 CUDA:0 (NVIDIA A100 80GB PCIe, 81051MiB)
engine/trainer: task=detect, mode=train, model=yolov8n.pt, data=/home/sdanayak/roboflow_data/data.yaml, epochs=30, time=None, patience
=100, batch=16, imgsz=1280, save=True, save_period=-1, cache=False, device=None, workers=8, project=runs_Data_aug_scale_withAnchorBoxT
uning, name=exp_dataAug_scale_withAnchorBoxTuning, exist_ok=False, pretrained=True, optimizer=auto, verbose=True, seed=0, deterministi
c=True, single_cls=False, rect=False, cos_lr=False, close_mosaic=10, resume=False, amp=True, fraction=1.0, profile=False, freeze=None,
multi_scale=False, overlap_mask=True, mask_ratio=4, dropout=0.0, val=True, split=val, save_json=False, save_hybrid=False, conf=None, i
ou=0.7, max_det=300, half=False, dnn=False, plots=True, source=None, vid_stride=1, stream_buffer=False, visualize=False, augment=Fals
e, agnostic_nms=False, classes=None, retina_masks=False, embed=None, show=False, save_frames=False, save_txt=False, save_conf=False, s
ave_crop=False, show_labels=True, show_conf=True, show_boxes=True, line_width=None, format=torchscript, keras=False, optimize=False, i
nt8=False, dynamic=False, simplify=True, opset=None, workspace=None, nms=False, lr0=0.01, lrf=0.01, momentum=0.937, weight_decay=0.000
5, warmup_epochs=3.0, warmup_momentum=0.8, warmup_bias_lr=0.0, box=7.5, cls=0.5, dfl=1.5, pose=12.0, kobj=1.0, nbs=64, hsv_h=0.2, hsv_
s=0.5, hsv_v=0.5, degrees=20, translate=0.2, scale=0.6, shear=0.0, perspective=0.0005, flipud=0.0, fliplr=0.5, bgr=0.0, mosaic=0.8, mi
xup=0.0, copy_paste=0.0, copy_paste_mode=flip, auto_augment=randaugment, erasing=0.4, crop_fraction=1.0, cfg=None, tracker=botsort.yam
l, save_dir=runs_Data_aug_scale_withAnchorBoxTuning/exp_dataAug_scale_withAnchorBoxTuning

                   from  n    params  module                                       arguments
  0                  -1  1       464  ultralytics.nn.modules.conv.Conv             [3, 16, 3, 2]
  1                  -1  1      4672  ultralytics.nn.modules.conv.Conv             [16, 32, 3, 2]
```



Test on Video or Image Sequences.

DOCUMENTATION: [https://docs.ultralytics.com/modes/predict/#inference-arguments]

1. Run inference on some test images or video.
2. Save frames where detection confidence is above 0.5 and analyze false positives/negatives.
3. Optimize confidence threshold and non-max suppression (NMS) to avoid missing small objects.

[1]:

```
import os
dataset_abs_path = os.path.abspath("yolov8n.pt")
print("Absolute dataset path:", dataset_abs_path)
```

```
Absolute dataset path: /home/sdanayak/yolov8n.pt
```

[3]:

```
from ultralytics import YOLO
model = YOLO("/home/sdanayak/yolov8n.pt")
results = model("/home/sdanayak/Test_images_videos/sample_image.jpg", conf=0.2,
                save=True, project="/home/sdanayak/Test_images_videos/", name="Result_images")

for result in results:
    result.show()
```

```
image 1/1 /home/sdanayak/Test_images_videos/sample_image.jpg: 352x640 10 cars, 10 traffic lights, 1 stop sign, 74.8ms
Speed: 2.8ms preprocess, 74.8ms inference, 2.1ms postprocess per image at shape (1, 3, 352, 640)
Results saved to /home/sdanayak/Test_images_videos/Result_images2
```

[2]:

```
from ultralytics import YOLO
model = YOLO("/home/sdanayak/yolov8n.pt")
results = model.predict("/home/sdanayak/Test_images_videos/sample_image_2.jpg", conf=0.2,
                save=True, project="/home/sdanayak/Test_images_videos/", name="Result_images")

for result in results:
    result.show()
```

```
image 1/1 /home/sdanayak/Test_images_videos/sample_image_2.jpg: 448x640 27 cars, 1 truck, 6 traffic lights, 8.4ms
Speed: 2.2ms preprocess, 8.4ms inference, 1.5ms postprocess per image at shape (1, 3, 448, 640)
Results saved to /home/sdanayak/Test_images_videos/Result_images
```

[ ]:

```
from ultralytics import YOLO
model = YOLO("/home/sdanayak/yolov8n.pt")
results = model("/home/sdanayak/Test_images_videos/GX011158.MP4", conf=0.2,
                save=True, project="/home/sdanayak/Test_images_videos/", name="Result_video")

for result in results:
    result.show()
```

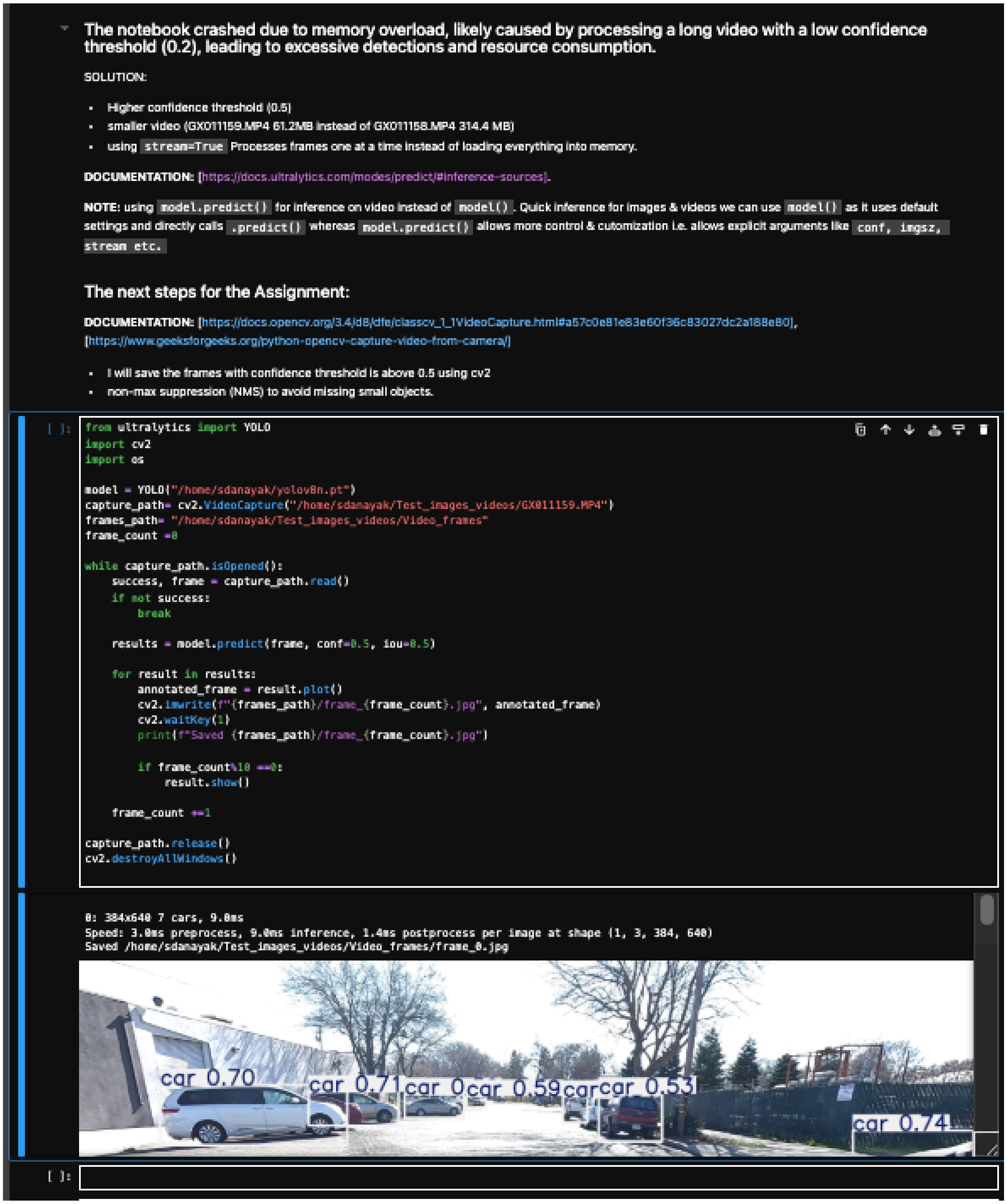
```
video 1/1 (frame 155/5261) /home/sdanayak/Test_images_videos/GX011158.MP4: 384x640 1 person, 7 cars, 1 truck, 1 stop sign, 1 backpack,
8.5ms
video 1/1 (frame 156/5261) /home/sdanayak/Test_images_videos/GX011158.MP4: 384x640 1 person, 6 cars, 1 truck, 1 stop sign, 8.1ms
video 1/1 (frame 157/5261) /home/sdanayak/Test_images_videos/GX011158.MP4: 384x640 1 person, 6 cars, 2 trucks, 1 stop sign, 8.1ms
video 1/1 (frame 158/5261) /home/sdanayak/Test_images_videos/GX011158.MP4: 384x640 1 person, 7 cars, 1 stop sign, 8.2ms
video 1/1 (frame 159/5261) /home/sdanayak/Test_images_videos/GX011158.MP4: 384x640 1 person, 5 cars, 1 truck, 1 stop sign, 8.1ms
video 1/1 (frame 160/5261) /home/sdanayak/Test_images_videos/GX011158.MP4: 384x640 1 person, 5 cars, 2 trains, 1 stop sign, 8.1ms
video 1/1 (frame 161/5261) /home/sdanayak/Test_images_videos/GX011158.MP4: 384x640 1 person, 4 cars, 1 train, 1 truck, 1 stop sign, 8.
6ms
video 1/1 (frame 162/5261) /home/sdanayak/Test_images_videos/GX011158.MP4: 384x640 1 person, 7 cars, 1 truck, 1 stop sign, 8.3ms
video 1/1 (frame 163/5261) /home/sdanayak/Test_images_videos/GX011158.MP4: 384x640 9 cars, 1 stop sign, 8.0ms
video 1/1 (frame 164/5261) /home/sdanayak/Test_images_videos/GX011158.MP4: 384x640 1 person, 8 cars, 1 truck, 1 stop sign, 8.2ms
video 1/1 (frame 165/5261) /home/sdanayak/Test_images_videos/GX011158.MP4: 384x640 9 cars, 1 train, 1 truck, 1 stop sign, 8.2ms
video 1/1 (frame 166/5261) /home/sdanayak/Test_images_videos/GX011158.MP4: 384x640 8 cars, 1 train, 1 truck, 1 stop sign, 8.1ms
video 1/1 (frame 167/5261) /home/sdanayak/Test_images_videos/GX011158.MP4: 384x640 7 cars, 2 trains, 1 truck, 1 stop sign, 8.1ms
video 1/1 (frame 168/5261) /home/sdanayak/Test_images_videos/GX011158.MP4: 384x640 4 cars, 1 stop sign, 8.0ms
video 1/1 (frame 169/5261) /home/sdanayak/Test_images_videos/GX011158.MP4: 384x640 7 cars, 1 stop sign, 8.1ms
video 1/1 (frame 170/5261) /home/sdanayak/Test_images_videos/GX011158.MP4: 384x640 5 cars, 2 trucks, 1 stop sign, 8.1ms
video 1/1 (frame 171/5261) /home/sdanayak/Test_images_videos/GX011158.MP4: 384x640 5 cars, 2 trucks, 1 stop sign, 8.5ms
```

The notebook crashed due to memory overload, likely caused by processing a long video with a low confidence threshold (0.2), leading to excessive detections and resource consumption.

SOLUTION:

- Higher confidence threshold (0.5)
- smaller video (GX011159.MP4 61.2MB instead of GX011158.MP4 314.4 MB)
- using `stream=True` Processes frames one at a time instead of loading everything into memory.

DOCUMENTATION: [https://docs.ultralytics.com/modes/predict/#inference-sources]



The notebook crashed due to memory overload, likely caused by processing a long video with a low confidence threshold (0.2), leading to excessive detections and resource consumption.

SOLUTION:

- Higher confidence threshold (0.5)
- smaller video (GX011159.MP4 61.2MB instead of GX011158.MP4 314.4 MB)
- using `stream=True` Processes frames one at a time instead of loading everything into memory.

DOCUMENTATION: [https://docs.ultralytics.com/modes/predict/#inference-sources].

NOTE: using `model.predict()` for inference on video instead of `model()`. Quick inference for images & videos we can use `model()` as it uses default settings and directly calls `.predict()` whereas `model.predict()` allows more control & cutomization i.e. allows explicit arguments like `conf, imgsz, stream etc.`

The next steps for the Assignment:

DOCUMENTATION: [https://docs.opencv.org/3.4/d8/dfe/classcv_1_1VideoCapture.html#a57c0e81e83e60f36c83027dc2a188e80], [https://www.geeksforgeeks.org/python-opencv-capture-video-from-camera/]

- I will save the frames with confidence threshold is above 0.5 using cv2
- non-max suppression (NMS) to avoid missing small objects.

[ ]:

```
from ultralytics import YOLO
import cv2
import os

model = YOLO("/home/sdanayak/yolov8n.pt")
capture_path= cv2.VideoCapture("/home/sdanayak/Test_images_videos/GX011159.MP4")
frames_path= "/home/sdanayak/Test_images_videos/Video_frames"
frame_count =0

while capture_path.isOpened():
    success, frame = capture_path.read()
    if not success:
        break

    results = model.predict(frame, conf=0.5, iou=0.5)

    for result in results:
        annotated_frame = result.plot()
        cv2.imwrite(f"{frames_path}/frame_{frame_count}.jpg", annotated_frame)
        cv2.waitKey(1)
        print(f"Saved {frames_path}/frame_{frame_count}.jpg")

        if frame_count%10 ==0:
            result.show()

    frame_count +=1

capture_path.release()
cv2.destroyAllWindows()
```

```
0: 384x640 7 cars, 9.0ms
Speed: 3.0ms preprocess, 9.0ms inference, 1.4ms postprocess per image at shape (1, 3, 384, 640)
Saved /home/sdanayak/Test_images_videos/Video_frames/frame_0.jpg
```

[ ]: