```
# Install YOLOv8 from the ultralytics library
!pip install ultralytics

# Install Roboflow to download the dataset
!pip install roboflow
```

```
Requirement already satisfied: tzdata>=2022.7 in /usr/local/lib/python3.10/dist-packages (from pandas>=1.1.4->ul
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.10/dist-packages (from request
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-packages (from requests>=2.23.0->u
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.10/dist-packages (from requests>=2.2
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.10/dist-packages (from requests>=2.2
Requirement already satisfied: filelock in /usr/local/lib/python3.10/dist-packages (from torch>=1.8.0->ultralyti
Requirement already satisfied: typing-extensions>=4.8.0 in /usr/local/lib/python3.10/dist-packages (from torch>=
Requirement already satisfied: networkx in /usr/local/lib/python3.10/dist-packages (from torch>=1.8.0->ultralyti
Requirement already satisfied: jinja2 in /usr/local/lib/python3.10/dist-packages (from torch>=1.8.0->ultralytics
Requirement already satisfied: fsspec in /usr/local/lib/python3.10/dist-packages (from torch>=1.8.0->ultralytics
Requirement already satisfied: sympy==1.13.1 in /usr/local/lib/python3.10/dist-packages (from torch>=1.8.0->ultr
Requirement already satisfied: mpmath<1.4,>=1.1.0 in /usr/local/lib/python3.10/dist-packages (from sympy==1.13.1
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-packages (from python-dateutil>=2.7->m
Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.10/dist-packages (from jinja2->torch>=1
Downloading ultralytics-8.3.43-py3-none-any.whl (898 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 898.4/898.4 kB 37.1 MB/s eta 0:00:00
Downloading ultralytics_thop-2.0.12-py3-none-any.whl (26 kB)
Installing collected packages: ultralytics-thop, ultralytics
Successfully installed ultralytics-8.3.43 ultralytics-thop-2.0.12
Collecting roboflow
  Downloading roboflow-1.1.49-py3-none-any.whl.metadata (9.7 kB)
Requirement already satisfied: certifi in /usr/local/lib/python3.10/dist-packages (from roboflow) (2024.8.30)
Collecting idna==3.7 (from roboflow)
  Downloading idna-3.7-py3-none-any.whl.metadata (9.9 kB)
Requirement already satisfied: cycler in /usr/local/lib/python3.10/dist-packages (from roboflow) (0.12.1)
Requirement already satisfied: kiwisolver>=1.3.1 in /usr/local/lib/python3.10/dist-packages (from roboflow) (1.4
Requirement already satisfied: matplotlib in /usr/local/lib/python3.10/dist-packages (from roboflow) (3.8.0)
Requirement already satisfied: numpy>=1.18.5 in /usr/local/lib/python3.10/dist-packages (from roboflow) (1.26.4)
Requirement already satisfied: opencv-python-headless==4.10.0.84 in /usr/local/lib/python3.10/dist-packages (fro
Requirement already satisfied: Pillow>=7.1.2 in /usr/local/lib/python3.10/dist-packages (from roboflow) (11.0.0)
Requirement already satisfied: python-dateutil in /usr/local/lib/python3.10/dist-packages (from roboflow) (2.8.2
Collecting python-dotenv (from roboflow)
  Downloading python_dotenv-1.0.1-py3-none-any.whl.metadata (23 kB)
Requirement already satisfied: requests in /usr/local/lib/python3.10/dist-packages (from roboflow) (2.32.3)
Requirement already satisfied: six in /usr/local/lib/python3.10/dist-packages (from roboflow) (1.16.0)
Requirement already satisfied: urllib3>=1.26.6 in /usr/local/lib/python3.10/dist-packages (from roboflow) (2.2.3
Requirement already satisfied: tqdm>=4.41.0 in /usr/local/lib/python3.10/dist-packages (from roboflow) (4.66.6)
Requirement already satisfied: PyYAML>=5.3.1 in /usr/local/lib/python3.10/dist-packages (from roboflow) (6.0.2)
Requirement already satisfied: requests-toolbelt in /usr/local/lib/python3.10/dist-packages (from roboflow) (1.0
Collecting filetype (from roboflow)
  Downloading filetype-1.2.0-py2.py3-none-any.whl.metadata (6.5 kB)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib->rob
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib->ro
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib->robo
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib->rob
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.10/dist-packages (from request
Downloading roboflow-1.1.49-py3-none-any.whl (80 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 80.9/80.9 kB 8.0 MB/s eta 0:00:00
Downloading idna-3.7-py3-none-any.whl (66 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 66.8/66.8 kB 7.1 MB/s eta 0:00:00
Downloading filetype-1.2.0-py2.py3-none-any.whl (19 kB)
Downloading python_dotenv-1.0.1-py3-none-any.whl (19 kB)
Installing collected packages: filetype, python-dotenv, idna, roboflow
  Attempting uninstall: idna
    Found existing installation: idna 3.10
    Uninstalling idna-3.10:
      Successfully uninstalled idna-3.10
Successfully installed filetype-1.2.0 idna-3.7 python-dotenv-1.0.1 roboflow-1.1.49
```

```
!pip install roboflow

from roboflow import Roboflow
rf = Roboflow(api_key="cjU1zRFgpb0vH56NFSUF")
project = rf.workspace("mohamed-traore-2ekkp").project("taco-trash-annotations-in-context")
version = project.version(16)
dataset = version.download("yolov8")
```

```
Requirement already satisfied: roboflow in /usr/local/lib/python3.10/dist-packages (1.1.49)
Requirement already satisfied: certifi in /usr/local/lib/python3.10/dist-packages (from roboflow) (2024.8.30)
Requirement already satisfied: idna==3.7 in /usr/local/lib/python3.10/dist-packages (from roboflow) (3.7)
Requirement already satisfied: cycler in /usr/local/lib/python3.10/dist-packages (from roboflow) (0.12.1)
Requirement already satisfied: kiwisolver>=1.3.1 in /usr/local/lib/python3.10/dist-packages (from roboflow) (1.4.7
Requirement already satisfied: matplotlib in /usr/local/lib/python3.10/dist-packages (from roboflow) (3.8.0)
Requirement already satisfied: numpy>=1.18.5 in /usr/local/lib/python3.10/dist-packages (from roboflow) (1.26.4)
Requirement already satisfied: opencv-python-headless==4.10.0.84 in /usr/local/lib/python3.10/dist-packages (from
Requirement already satisfied: Pillow>=7.1.2 in /usr/local/lib/python3.10/dist-packages (from roboflow) (11.0.0)
Requirement already satisfied: python-dateutil in /usr/local/lib/python3.10/dist-packages (from roboflow) (2.8.2)
Requirement already satisfied: python-dotenv in /usr/local/lib/python3.10/dist-packages (from roboflow) (1.0.1)
Requirement already satisfied: requests in /usr/local/lib/python3.10/dist-packages (from roboflow) (2.32.3)
Requirement already satisfied: six in /usr/local/lib/python3.10/dist-packages (from roboflow) (1.16.0)
Requirement already satisfied: urllib3>=1.26.6 in /usr/local/lib/python3.10/dist-packages (from roboflow) (2.2.3)
Requirement already satisfied: tqdm>=4.41.0 in /usr/local/lib/python3.10/dist-packages (from roboflow) (4.66.6)
Requirement already satisfied: PyYAML>=5.3.1 in /usr/local/lib/python3.10/dist-packages (from roboflow) (6.0.2)
Requirement already satisfied: requests-toolbelt in /usr/local/lib/python3.10/dist-packages (from roboflow) (1.0.0
Requirement already satisfied: filetype in /usr/local/lib/python3.10/dist-packages (from roboflow) (1.2.0)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib->robof
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib->robo
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib->robofl
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib->robof
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.10/dist-packages (from requests-
loading Roboflow workspace...
loading Roboflow project...
Downloading Dataset Version Zip in TACO:-Trash-Annotations-in-Context-Dataset-16 to yolov8:: 100%|██████████| 3323

Extracting Dataset Version Zip to TACO:-Trash-Annotations-in-Context-Dataset-16 in yolov8:: 100%|██████████| 7204/
Creating new Ultralytics Settings v0.0.6 file ✅
View Ultralytics Settings with 'yolo settings' or at '/root/.config/Ultralytics/settings.json'
Update Settings with 'yolo settings key=value', i.e. 'yolo settings runs_dir=path/to/dir'. For help see https://dc
```

```
from ultralytics import YOLO

# Step 1: Load a pre-trained YOLOv8 model (choose 'n', 's', 'm', 'l', or 'x' for different sizes)
model = YOLO("yolov8n.pt")  # yolov8n = nano, yolov8s = small, yolov8m = medium, etc.

# Step 2: Train the model
model.train(
    data=f"{dataset.location}/data.yaml",  # Path to the downloaded dataset's data.yaml
    epochs=50,                             # Number of epochs to train
    imgsz=640,                             # Image size
    batch=16,                              # Batch size
    workers=2                              # Number of workers (adjust based on your environment)
)
```

```
                0.46844,     0.46844,     0.46844,     0.46844,     0.46844,     0.46844,     0.46844,     0.46844,
     0.46844,     0.46832,      0.4672,     0.46666,     0.46548,      0.4648,     0.46252,     0.46252,
     0.46252,     0.46154,     0.46154,     0.46154,     0.46154,     0.46015,      0.4587,
                0.45613,     0.45464,     0.45266,     0.45168,     0.45168,     0.45168,     0.45168,     0.45137,
     0.451,     0.45069,     0.45069,     0.45037,      0.4497,     0.44927,     0.44876,     0.44872,     0.44872,
     0.44872,     0.44872,     0.44816,     0.44773,     0.44763,     0.44675,
                0.44675,     0.44675,     0.44675,     0.44675,     0.44576,     0.44576,     0.44576,     0.44477,
     0.44335,     0.44181,     0.43984,     0.43984,     0.43984,     0.43984,     0.43905,     0.43886,
     0.43886,     0.43886,     0.43754,     0.43688,     0.43688,     0.43688,     0.43688,
                0.43688,     0.43543,     0.43491,     0.43491,     0.43491,     0.43491,     0.43393,     0.43393,
     0.43393,     0.43389,     0.43321,     0.43214,     0.43195,     0.43195,     0.43097,     0.43049,
     0.42999,     0.42998,     0.42998,     0.42998,     0.42998,     0.42998,     0.42792,
                0.42725,     0.42635,     0.42604,     0.42604,     0.42529,     0.42505,     0.42505,     0.42406,
     0.42406,     0.42406,     0.42386,     0.42209,     0.42209,     0.42209,     0.42209,     0.42209,
     0.42209,     0.42209,      0.4211,     0.42012,     0.42012,     0.41913,     0.41819,
                0.41815,     0.41716,     0.41716,     0.41716,     0.41508,      0.4142,      0.4142,      0.4142,
     0.4142,     0.41382,     0.41342,     0.41321,     0.41321,     0.41321,     0.41321,     0.41223,     0.41223,
     0.41223,     0.41223,     0.41147,     0.41124,     0.41124,     0.41124,
                0.41124,     0.41124,     0.41026,     0.41026,     0.41026,     0.41026,     0.41026,     0.40828,
     0.40828,     0.40828,     0.40828,     0.40828,     0.40828,     0.40828,     0.40828,     0.40828,
     0.40828,     0.40828,     0.40828,     0.40828,     0.40828,     0.40828,     0.40828,
                0.40828,     0.40828,     0.40828,     0.40828,     0.40828,     0.40828,     0.40828,     0.40828,
     0.40828,     0.40828,     0.40828,     0.40828,     0.40761,      0.4073,     0.40633,     0.40631,
     0.40594,     0.40557,     0.40533,     0.40533,     0.40533,     0.40533,     0.40533,
                0.40485,     0.40435,     0.40377,     0.40335,     0.40335,     0.40335,     0.40305,     0.40255,
     0.40138,     0.40138,     0.40039,     0.40039,     0.40039,     0.40025,     0.39975,     0.39941,
     0.39941,     0.39941,     0.39941,     0.39941,     0.39906,     0.39855,     0.39743,
                0.39645,     0.39645,     0.39637,     0.39546,     0.39546,     0.39546,     0.39546,     0.39546,
     0.39519,     0.39461,     0.39448,     0.39448,     0.39448,     0.39448,     0.39448,     0.39448,
```

```python
# Evaluate the model on the validation set
results = model.val()

# Print validation results
print(results)
```

```
                 0.19348,     0.19252,     0.18988,      0.1882,     0.18701,     0.18646,     0.18588,     0.18535,
                 0.17025,     0.16864,     0.16715,     0.16623,     0.16543,     0.16473,     0.16144,     0.15951,
                   0.132,     0.13024,      0.1267,     0.12538,     0.12368,     0.12296,     0.12261,     0.12215,
                0.093545,    0.092404,    0.089731,    0.089136,    0.087212,    0.085403,    0.081909,     0.07969,
                0.057012,    0.056582,    0.056037,    0.054405,    0.052656,    0.051647,    0.049197,    0.048423,
                0.023175,    0.022785,      0.0212,    0.019665,    0.017519,    0.014659,    0.014314,    0.013969,
                0.001969,   0.0018674,   0.0017658,   0.0016641,   0.0015625,   0.0014609,   0.0013593,   0.0012576,
                      0,           0,           0,           0,           0,           0,           0,           0,
     fitness: 0.3652012656780426
     keys: ['metrics/precision(B)', 'metrics/recall(B)', 'metrics/mAP50(B)', 'metrics/mAP50-95(B)']
     maps: array([    0.35097])
     names: {0: 'trash'}
     plot: True
     results_dict: {'metrics/precision(B)': 0.7611350285025342, 'metrics/recall(B)': 0.39644970414201186, 'metrics/mA
     save_dir: PosixPath('runs/detect/train2')
     speed: {'preprocess': 0.31293036546994213, 'inference': 4.748849964460801, 'loss': 0.001631452885758518, 'postpr
     task: 'detect'
```

```
# Use the trained model to make predictions on test images
model.predict(
    source=f"{dataset.location}/test/images",  # Path to test images
    save=True,                                # Save prediction images with bounding boxes
    conf=0.5                                  # Confidence threshold
)
```

```
        ...,

        [[121, 119, 119],
         [122, 120, 120],
         [114, 114, 114],
         ....
```

```
# Save the trained model to Google Drive
!cp runs/detect/train/weights/best.pt /content/drive/MyDrive/best.pt
```

```
---------------------------------------------------------------------------
NotImplementedError                       Traceback (most recent call last)
<ipython-input-7-817df71dd55e> in <cell line: 2>()
      1 # Save the trained model to Google Drive
----> 2 get_ipython().system('cp runs/detect/train/weights/best.pt /content/drive/MyDrive/best.pt')

                                   2 frames
/usr/local/lib/python3.10/dist-packages/google/colab/_system_commands.py in _run_command(cmd,
clear_streamed_output)
    166     locale_encoding = locale.getpreferredencoding()
    167     if locale_encoding != _ENCODING:
--> 168       raise NotImplementedError(
    169           'A UTF-8 locale is required. Got {}'.format(locale_encoding)
    170       )

NotImplementedError: A UTF-8 locale is required. Got ANSI_X3.4-1968
```

Next steps: Explain error

```
# Export the model in various formats, e.g., ONNX, CoreML, TensorRT
model.export(format="onnx", dynamic=True)
```

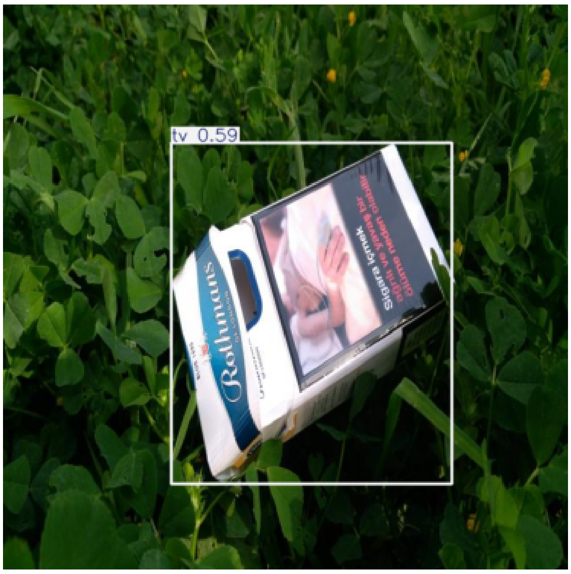
```
import cv2
from ultralytics import YOLO
from PIL import Image
import matplotlib.pyplot as plt
```

```
model = YOLO("yolov8n.pt")  # Replace with the path to your trained YOLOv8 model
```

```
import cv2
from ultralytics import YOLO
import matplotlib.pyplot as plt

# Step 1: Set the path to the test image
image_path = "/content/TACO:-Trash-Annotations-in-Context-Dataset-16/test/images/000004_jpg.rf.f220eaab0f36226385a34eb07

# Step 2: Load YOLOv8 model
model = YOLO("yolov8n.pt")  # Ensure that this is the path to your trained YOLOv8 model

# Step 3: Load the test image using OpenCV
img = cv2.imread(image_path)

# Step 4: Make predictions on the test image
results = model.predict(img, conf=0.5)  # Adjust confidence threshold as needed

# Step 5: Annotate the image with bounding boxes and category labels
annotated_img = results[0].plot()

# Step 6: Convert to RGB format for display using matplotlib
annotated_img_rgb = cv2.cvtColor(annotated_img, cv2.COLOR_BGR2RGB)

# Step 7: Display the image with bounding boxes and category labels
plt.imshow(annotated_img_rgb)
plt.axis('off')  # Hide axis for cleaner view
plt.show()
```

```
# Optional: Save the annotated image
cv2.imwrite("annotated_test_image.jpg", annotated_img)
```

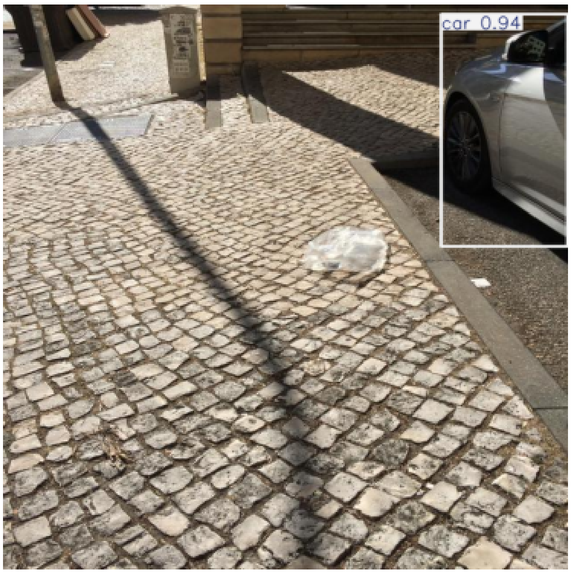
```
0: 640x640 1 tv, 17.5ms
Speed: 2.2ms preprocess, 17.5ms inference, 1.7ms postprocess per image at shape (1, 3, 640, 640)
```



```
True
```

```
import cv2
from ultralytics import YOLO
import matplotlib.pyplot as plt

# Step 1: Set the path to the test image
image_path = "/content/TACO:-Trash-Annotations-in-Context-Dataset-16/test/images/000006_jpg.rf.65db7183891046d795545fe

# Step 2: Load YOLOv8 model
model = YOLO("yolov8n.pt")  # Ensure that this is the path to your trained YOLOv8 model

# Step 3: Load the test image using OpenCV
img = cv2.imread(image_path)

# Step 4: Make predictions on the test image
results = model.predict(img, conf=0.5)  # Adjust confidence threshold as needed

# Step 5: Annotate the image with bounding boxes and category labels
annotated_img = results[0].plot()

# Step 6: Convert to RGB format for display using matplotlib
annotated_img_rgb = cv2.cvtColor(annotated_img, cv2.COLOR_BGR2RGB)

# Step 7: Display the image with bounding boxes and category labels
plt.imshow(annotated_img_rgb)
plt.axis('off')  # Hide axis for cleaner view
plt.show()

# Optional: Save the annotated image
cv2.imwrite("annotated_test_image.jpg", annotated_img)
```

```
0: 640x640 1 car, 9.5ms
Speed: 1.7ms preprocess, 9.5ms inference, 1.3ms postprocess per image at shape (1, 3, 640, 640)
```



```
True
```

```
import cv2
from ultralytics import YOLO
import matplotlib.pyplot as plt

# Step 1: Set the path to the test image
image_path = "/content/TACO:-Trash-Annotations-in-Context-Dataset-16/test/images/000074_JPG.rf.1394c939daf3a1345ce9447

# Step 2: Load YOLOv8 model
model = YOLO("yolov8n.pt")  # Ensure that this is the path to your trained YOLOv8 model

# Step 3: Load the test image using OpenCV
img = cv2.imread(image_path)

# Step 4: Make predictions on the test image
results = model.predict(img, conf=0.5)  # Adjust confidence threshold as needed

# Step 5: Annotate the image with bounding boxes and category labels
annotated_img = results[0].plot()

# Step 6: Convert to RGB format for display using matplotlib
annotated_img_rgb = cv2.cvtColor(annotated_img, cv2.COLOR_BGR2RGB)

# Step 7: Display the image with bounding boxes and category labels
plt.imshow(annotated_img_rgb)
plt.axis('off')  # Hide axis for cleaner view
plt.show()

# Optional: Save the annotated image
cv2.imwrite("annotated_test_image.jpg", annotated_img)
```

```
0: 640x640 (no detections), 13.0ms
Speed: 2.1ms preprocess, 13.0ms inference, 0.9ms postprocess per image at shape (1, 3, 640, 640)
```

```
True
```

```
# # You can upload a test image manually in Colab using the file upload widget:
# from google.colab import files
# uploaded = files.upload()

# # Load the test image (the uploaded image's file name will be the key)
```