```
!pip install -U -q ultralytics roboflow matplotlib
```

```
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 89.9/89.9 kB 4.2 MB/s eta 0:00:00
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 66.8/66.8 kB 3.3 MB/s eta 0:00:00
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 49.9/49.9 MB 18.7 MB/s eta 0:00:00
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 8.7/8.7 MB 47.6 MB/s eta 0:00:00
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1.4/1.4 MB 27.3 MB/s eta 0:00:00
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 4.2/4.2 MB 42.6 MB/s eta 0:00:00
```

```
import os, yaml, shutil
from ultralytics import YOLO
from roboflow import Roboflow

rf = Roboflow(api_key="PFN4A7FKw5wg0Uw62akT")
project = rf.workspace("pram").project("ppe-detection-z3v2w")
version = project.version(2)
dataset = version.download("yolov8")
```

```
loading Roboflow workspace...
loading Roboflow project...
Downloading Dataset Version Zip in PPE-Detection-2 to yolov8:: 100%|██████████| 135846/135846 [00:02<00:00, 56773.72it/s]

Extracting Dataset Version Zip to PPE-Detection-2 in yolov8:: 100%|██████████| 6012/6012 [00:00<00:00, 7806.54it/s]
```

```
print("DATA:", os.path.join(dataset.location, "data.yaml"))
with open(os.path.join(dataset.location, "data.yaml")) as f:
    print(f.read())
```

```
DATA: /content/PPE-Detection-2/data.yaml
names:
- Hardhat
- Person
- Safety_Boots
- Safety_Gloves
- Safety_Mask
- Safety_Vest
nc: 6
roboflow:
  license: CC BY 4.0
  project: ppe-detection-z3v2w
  url: https://universe.roboflow.com/pram/ppe-detection-z3v2w/dataset/2
  version: 2
  workspace: pram
test: ../test/images
train: ../train/images
val: ../valid/images
```

Train YOLOv8

```
model = YOLO("yolov8s.pt")
data_yaml_path = os.path.join(dataset.location, "data.yaml")
results = model.train(
    data=data_yaml_path,
    imgsz=640,
    epochs=70,
    batch=16,
    device=0,
    optimizer="AdamW",
    cos_lr=True,
    patience=15
)
```

```
                   all        600       2292      0.918      0.898      0.938      0.714

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
      67/70      3.88G     0.8029     0.4471       1.11         25        640: 100% ━━━━━━━━━━━━ 132/132 3.4it/s 38.4s
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% ━━━━━━━━━━━━ 19/19 3.2it/s
                   all        600       2292      0.918      0.902       0.94      0.714

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
      68/70      3.88G     0.8036     0.4445      1.112         26        640: 100% ━━━━━━━━━━━━ 132/132 3.5it/s 38.0s
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% ━━━━━━━━━━━━ 19/19 3.7it/s
                   all        600       2292      0.916      0.903       0.94      0.715

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
      69/70      3.94G     0.8022     0.4469       1.11         20        640: 100% ━━━━━━━━━━━━ 132/132 3.4it/s 38.8s
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% ━━━━━━━━━━━━ 19/19 3.4it/s
                   all        600       2292      0.916      0.909       0.94      0.715

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
      70/70        4G      0.7983     0.4473       1.11         17        640: 100% ━━━━━━━━━━━━ 132/132 3.5it/s 38.2s
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% ━━━━━━━━━━━━ 19/19 3.5it/s
                   all        600       2292      0.918      0.904      0.941      0.715

70 epochs completed in 0.882 hours.
Optimizer stripped from /content/runs/detect/train/weights/last.pt, 22.5MB
Optimizer stripped from /content/runs/detect/train/weights/best.pt, 22.5MB

Validating /content/runs/detect/train/weights/best.pt...
Ultralytics 8.3.227 🚀 Python-3.12.12 torch-2.8.0+cu126 CUDA:0 (Tesla T4, 15095MiB)
Model summary (fused): 72 layers, 11,127,906 parameters, 0 gradients, 28.4 GFLOPs
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% ━━━━━━━━━━━━ 19/19 2.4it/s
                   all        600       2292      0.918      0.899       0.94      0.716
               Hardhat        357        383      0.963      0.966      0.985      0.858
                Person        269        323      0.958      0.911      0.973      0.831
          Safety_Boots        221        430      0.877      0.895      0.915      0.575
         Safety_Gloves        314        578      0.909      0.847      0.916      0.678
           Safety_Mask        253        285      0.898      0.861      0.899      0.572
           Safety_Vest        251        293      0.905      0.914       0.95       0.78
Speed: 0.2ms preprocess, 4.3ms inference, 0.0ms loss, 2.7ms postprocess per image
Results saved to /content/runs/detect/train
```

Evaluate

```
metrics = model.val()
metrics
```

```
       [     0.9614,      0.9614,      0.9614, ...,           0,           0,           0],
       [    0.98294,     0.98294,     0.97952, ...,           0,           0,           0]]), 'Confidence', 'Recall']]
fitness: np.float64(0.7152256699249346)
keys: ['metrics/precision(B)', 'metrics/recall(B)', 'metrics/mAP50(B)', 'metrics/mAP50-95(B)']
maps: array([    0.85766,      0.8301,     0.57265,     0.67795,     0.57307,     0.77992])
names: {0: 'Hardhat', 1: 'Person', 2: 'Safety_Boots', 3: 'Safety_Gloves', 4: 'Safety_Mask', 5: 'Safety_Vest'}
nt_per_class: array([383, 323, 430, 578, 285, 293])
nt_per_image: array([357, 269, 221, 314, 253, 251])
results_dict: {'metrics/precision(B)': 0.9185827016101996, 'metrics/recall(B)': 0.899230890914089, 'metrics/mAP50(B)':
0.9397603686405992, 'metrics/mAP50-95(B)': 0.7152256699249346, 'fitness': 0.7152256699249346}
save_dir: PosixPath('/content/runs/detect/val')
speed: {'preprocess': 1.4539456933349963, 'inference': 9.130855613332946, 'loss': 0.005382959999830442, 'postprocess':
1.1802973650007214}
stats: {'tp': [], 'conf': [], 'pred_cls': [], 'target_cls': [], 'target_img': []}
task: 'detect'
```

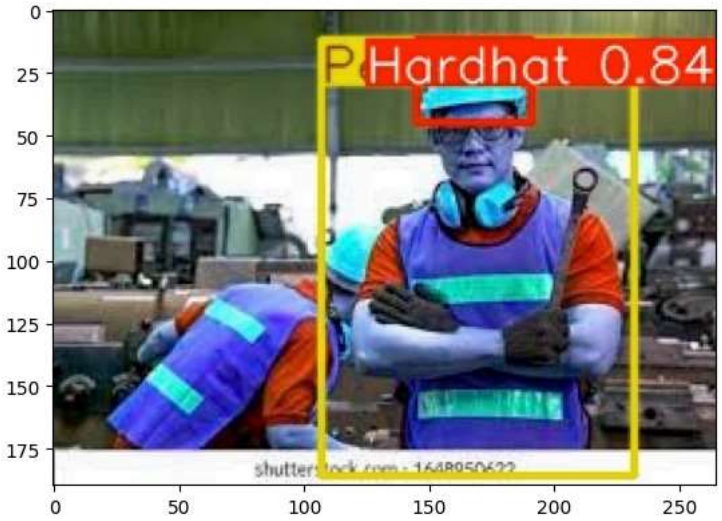
```
import cv2
import matplotlib.pyplot as plt
```

```
img_path = '/content/images.jpg'
results = model.predict(source=img_path, conf=0.5, save=True)

annotated_path = results[0].save_dir + "/" + results[0].path.split("/")[-1]
print("Annotated image saved to:", annotated_path)

img = cv2.imread(annotated_path)
plt.imshow(img)
```

```
image 1/1 /content/images.jpg: 480x640 1 Hardhat, 1 Person, 50.3ms
Speed: 2.7ms preprocess, 50.3ms inference, 1.4ms postprocess per image at shape (1, 3, 480, 640)
Results saved to /content/runs/detect/predict2
Annotated image saved to: /content/runs/detect/predict2/images.jpg
<matplotlib.image.AxesImage at 0x7c301f053710>
```



Export to ONNX + download weights

```
model.export(format="onnx", opset=12)
model.export(format="torchscript")
```

```
Ultralytics 8.3.227 🚀 Python-3.12.12 torch-2.8.0+cu126 CPU (Intel Xeon CPU @ 2.00GHz)
💡 ProTip: Export to OpenVINO format for best performance on Intel hardware. Learn more at https://docs.ultralytics.com/inte

PyTorch: starting from '/content/runs/detect/train/weights/best.pt' with input shape (1, 3, 640, 640) BCHW and output shape(
requirements: Ultralytics requirements ['onnx>=1.12.0,<=1.19.1', 'onnxslim>=0.1.71', 'onnxruntime-gpu'] not found, attemptir
Using Python 3.12.12 environment at: /usr
Resolved 14 packages in 173ms
Prepared 6 packages in 8.92s
Installed 6 packages in 404ms
 + colorama==0.4.6
 + coloredlogs==15.0.1
 + humanfriendly==10.0
 + onnx==1.19.1
 + onnxruntime-gpu==1.23.2
 + onnxslim==0.1.74

requirements: AutoUpdate success ✅ 10.1s
WARNING ⚠️ requirements: Restart runtime or rerun command for updates to take effect


ONNX: starting export with onnx 1.19.1 opset 12...
ONNX: slimming with onnxslim 0.1.74...
ONNX: export success ✅ 12.0s, saved as '/content/runs/detect/train/weights/best.onnx' (42.7 MB)

Export complete (12.7s)
Results saved to /content/runs/detect/train/weights
Predict:         yolo predict task=detect model=/content/runs/detect/train/weights/best.onnx imgsz=640
Validate:        yolo val task=detect model=/content/runs/detect/train/weights/best.onnx imgsz=640 data=/content/PPE-Detecti
Visualize:       https://netron.app
Ultralytics 8.3.227 🚀 Python-3.12.12 torch-2.8.0+cu126 CPU (Intel Xeon CPU @ 2.00GHz)

PyTorch: starting from '/content/runs/detect/train/weights/best.pt' with input shape (1, 3, 640, 640) BCHW and output shape(

TorchScript: starting export with torch 2.8.0+cu126...
TorchScript: export success ✅ 2.9s, saved as '/content/runs/detect/train/weights/best.torchscript' (42.9 MB)

Export complete (3.6s)
Results saved to /content/runs/detect/train/weights
Predict:         yolo predict task=detect model=/content/runs/detect/train/weights/best.torchscript imgsz=640
Validate:        yolo val task=detect model=/content/runs/detect/train/weights/best.torchscript imgsz=640 data=/content/PPE-
Visualize:       https://netron.app
'/content/runs/detect/train/weights/best.torchscript'
```

```
# Copy artifacts to /content/models for easy download
os.makedirs("/content/models", exist_ok=True)
latest = model.ckpt_path if hasattr(model, "ckpt_path") else "runs/detect/train/weights/best.pt"
pt_out = latest
onnx_out = "runs/detect/train/weights/best.onnx"
```

```
!cp -v {pt_out} /content/models/
!cp -v {onnx_out} /content/models/
!ls -lah /content/models
```

```
'yolov8s.pt' -> '/content/models/yolov8s.pt'
'runs/detect/train/weights/best.onnx' -> '/content/models/best.onnx'
total 65M
drwxr-xr-x 2 root root 4.0K Nov 11 06:26 .
drwxr-xr-x 1 root root 4.0K Nov 11 06:26 ..
-rw-r--r-- 1 root root  43M Nov 11 06:26 best.onnx
-rw-r--r-- 1 root root  22M Nov 11 06:26 yolov8s.pt
```

```
test_model = YOLO(pt_out)
r = test_model.predict(source=os.path.join(dataset.location, "valid", "images"), conf=0.5, save=True)
r
```

```
          [[ 24,  28,   9],
           [ 24,  28,   9],
           [ 24,  28,   9],
           ...,
           [219, 216, 212],
           [219, 216, 212],
           [219, 216, 212]],

          ...,

          [[ 27,  29,  23],
           [ 27,  29,  23],
           [ 27,  29,  23],
           ...,
           [127, 155, 172],
           [127, 155, 172],
           [127, 155, 172]],

          [[ 27,  29,  23],
           [ 27,  29,  23],
           [ 27,  29,  23],
           ...,
           [127, 155, 172],
           [127, 155, 172],
           [127, 155, 172]],

          [[ 27,  29,  23],
           [ 27,  29,  23],
           [ 27,  29,  23],
           ...,
           [127, 155, 172],
           [127, 155, 172],
           [127, 155, 172]]], dtype=uint8)
    orig_shape: (640, 640)
    path: '/content/PPE-Detection-2/valid/images/thumb2435_jpg.rf.39bcf0b0a4a54bd0aa719fecdb0e4b94.jpg'
    probs: None
    save_dir: '/content/runs/detect/predict'
    speed: {'preprocess': 1.82502999996359, 'inference': 8.160426000358711, 'postprocess': 1.4006530000187922}]
```