```
%pip install ultralytics
import ultralytics
ultralytics.checks()
```

```
Ultralytics 8.3.156 🚀 Python-3.11.13 torch-2.6.0+cu124 CPU (Intel Xeon 2.20GHz)
Setup complete ✅ (2 CPUs, 12.7 GB RAM, 41.5/107.7 GB disk)
```

```
import cv2
import random
from ultralytics import YOLO
```

```
from google.colab import files
uploaded = files.upload()
```

```
Choose Files 2213-1562...1_small.mp4
  • 2213-156227801_small.mp4(video/mp4) - 2488863 bytes, last modified: 17/6/2025 - 100% done
Saving 2213-156227801_small.mp4 to 2213-156227801_small.mp4
```

```
# Read class names
import random
with open('/content/coco.txt', 'r') as my_file:
    class_list = my_file.read().split('\n')

# Generate random detection colors
detection_colors = []
for _ in range(len(class_list)):
    r = random.randint(0, 255)
    g = random.randint(0, 255)
    b = random.randint(0, 255)  # ✅ FIXED: use randint
    detection_colors.append((b, g, r))

# Load YOLO model
from ultralytics import YOLO  # ✅ Add this line

# Load YOLO model
model = YOLO('/content/weights/yolov8n.pt')

cap = cv2.VideoCapturecap = cv2.VideoCapture("/content/2213-156227801_small.mp4")


if not cap.isOpened():
    print("cannot opening video file")
    exit()

while True:
    ret, frame = cap.read()
    if not ret:
        print('video ended or fail, exiting')
        break

    # ✅ Moved predict line inside the loop correctly
    detect_params = model.predict(source=[frame], conf=0.45, save=False)

    boxes = detect_params[0].boxes
    for i in range(len(boxes)):
        box = boxes[i]  # ✅ Fixed typo: box[i] → boxes[i]
        clsID = int(box.cls.cpu().numpy()[0])
        conf = box.conf.cpu().numpy()[0]
        bb = box.xyxy.cpu().numpy()[0]
        cv2.rectangle(frame, (int(bb[0]), int(bb[1])), (int(bb[2]), int(bb[3])), detection_colors[clsID], 2)
```

```
0: 384x640 1 train, 164.7ms
Speed: 4.9ms preprocess, 164.7ms inference, 1.7ms postprocess per image at shape (1, 3, 384, 640)

0: 384x640 1 train, 183.5ms
Speed: 5.5ms preprocess, 183.5ms inference, 1.3ms postprocess per image at shape (1, 3, 384, 640)

0: 384x640 2 trains, 146.8ms
Speed: 4.9ms preprocess, 146.8ms inference, 1.4ms postprocess per image at shape (1, 3, 384, 640)

0: 384x640 1 train, 145.1ms
Speed: 5.2ms preprocess, 145.1ms inference, 1.7ms postprocess per image at shape (1, 3, 384, 640)

0: 384x640 1 train, 173.5ms
Speed: 4.1ms preprocess, 173.5ms inference, 2.3ms postprocess per image at shape (1, 3, 384, 640)

0: 384x640 1 train, 244.2ms
Speed: 5.3ms preprocess, 244.2ms inference, 2.1ms postprocess per image at shape (1, 3, 384, 640)

0: 384x640 1 train, 236.3ms
Speed: 5.7ms preprocess, 236.3ms inference, 1.7ms postprocess per image at shape (1, 3, 384, 640)

0: 384x640 1 train, 231.4ms
Speed: 7.5ms preprocess, 231.4ms inference, 1.6ms postprocess per image at shape (1, 3, 384, 640)

0: 384x640 1 train, 228.8ms
Speed: 5.4ms preprocess, 228.8ms inference, 2.2ms postprocess per image at shape (1, 3, 384, 640)

0: 384x640 1 train, 241.7ms
Speed: 6.9ms preprocess, 241.7ms inference, 2.3ms postprocess per image at shape (1, 3, 384, 640)

0: 384x640 2 trains, 229.4ms
Speed: 9.6ms preprocess, 229.4ms inference, 1.8ms postprocess per image at shape (1, 3, 384, 640)

0: 384x640 2 trains, 231.6ms
Speed: 6.1ms preprocess, 231.6ms inference, 1.6ms postprocess per image at shape (1, 3, 384, 640)

0: 384x640 2 trains, 226.7ms
Speed: 5.9ms preprocess, 226.7ms inference, 2.0ms postprocess per image at shape (1, 3, 384, 640)

0: 384x640 1 train, 228.7ms
Speed: 7.1ms preprocess, 228.7ms inference, 2.0ms postprocess per image at shape (1, 3, 384, 640)
```

Start coding or generate with AI.