# Data Collection and Preprocessing Phase

| | |
|---|---|
| Date | 28 Sep 2024 |
| Team ID | 739648 |
| Project Title | Guardian Eye:Yolo-Based Smart Helmet Detection System For Enhanced Safety In Real-Time |
| Maximum Marks | 6 Marks |

**Preprocessing Template**

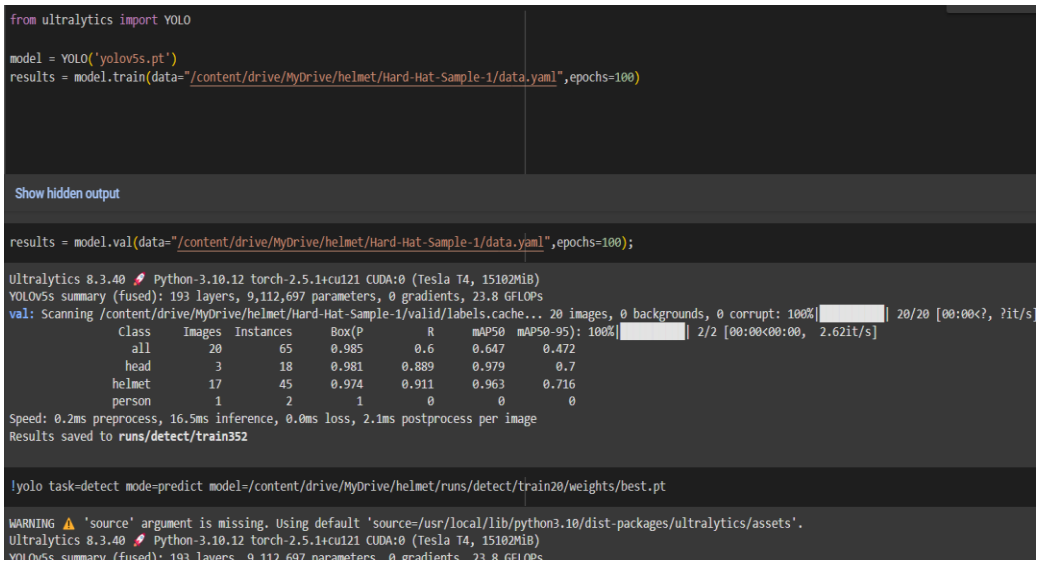
The preprocessing template for a smart helmet detection system involves capturing real-time video data, followed by image enhancement techniques such as noise reduction and contrast adjustment for clearer visibility. The next step is to apply object detection algorithms to localize and identify helmets in the scene. Finally, region-based processing helps isolate areas of interest for efficient and accurate real-time detection..

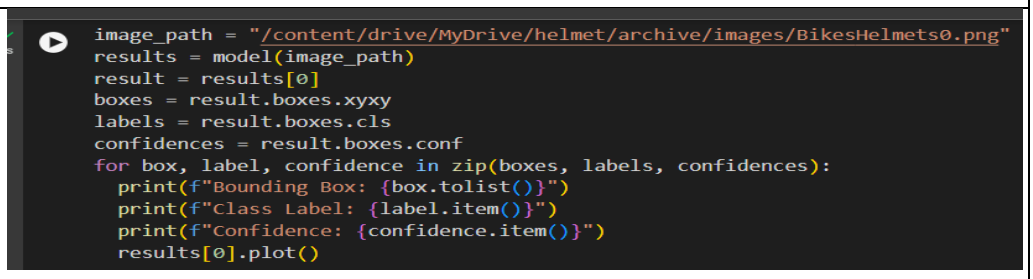
| **Section** | **Description** |
|---|---|
| Data Overview | Give an overview of the data, which you're going to use in your project. |
| Resizing | Resize images to a specified target size. |
| Normalization | Normalize pixel values to a specific range. |
| Data Augmentation | Apply augmentation techniques such as flipping, rotation, shifting, zooming, or shearing. |
| Denoising | Apply denoising filters to reduce noise in the images. |

| | |
|---|---|
| Edge Detection | Apply edge detection algorithms to highlight prominent edges in the images. |
| Color Space Conversion | Convert images from one color space to another. |

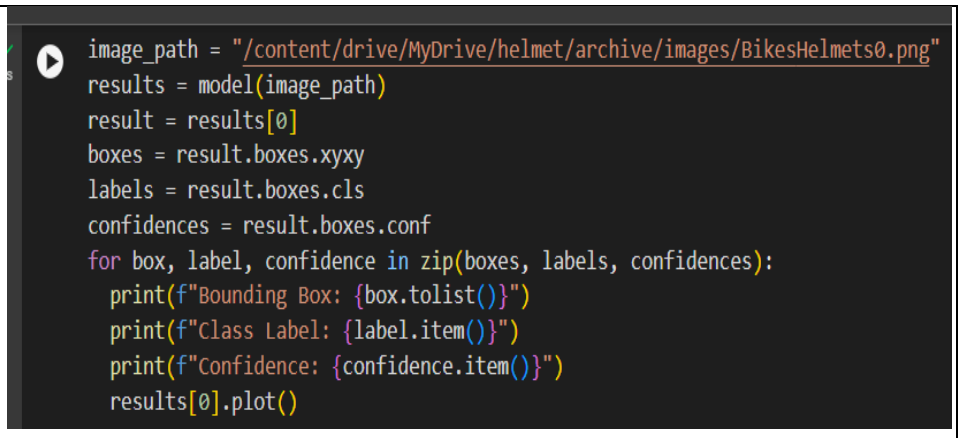
| | |
|---|---|
| Image Cropping | Crop images to focus on the regions containing objects of interest. |
| Batch Normalization | Apply batch normalization to the input of each layer in the neural network. |

**Data Preprocessing Code Screenshots**

| | |
|---|---|
| Loading Data |  |
| Resizing | ---------- |

| | |
|---|---|
| Normalization | ---------- |
| Data Augmentation | ---------- |
| Denoising | ---------- |
| Edge Detection | ---------- |
| Color Space Conversion |  |

```
image_path = "/content/drive/MyDrive/helmet/archive/images/BikesHelmets0.png"
results = model(image_path)
result = results[0]
boxes = result.boxes.xyxy
labels = result.boxes.cls
confidences = result.boxes.conf
for box, label, confidence in zip(boxes, labels, confidences):
  print(f"Bounding Box: {box.tolist()}")
  print(f"Class Label: {label.item()}")
  print(f"Confidence: {confidence.item()}")
  results[0].plot()
```

| | | |
|---|---|---|
| Image Cropping | |  |

```
image_path = "/content/drive/MyDrive/helmet/archive/images/BikesHelmets0.png"
results = model(image_path)
result = results[0]
boxes = result.boxes.xyxy
labels = result.boxes.cls
confidences = result.boxes.conf
for box, label, confidence in zip(boxes, labels, confidences):
  print(f"Bounding Box: {box.tolist()}")
  print(f"Class Label: {label.item()}")
  print(f"Confidence: {confidence.item()}")
  results[0].plot()
```