1. PascalVOC Format to YOLOV8 Format

1.1 PascalVOC and YOLOv8 Format

PascalVOC format provides annotations in XML files. The XML file for an image contains annotations specifying the bounding boxes and class labels of objects present within the image.

YOLOv8 model expects annotations of the objects present within the image in text files, where each line represents an object annotation in a specific format. The text files should have annotations in the following format: *object_class_id x_center y_center width height*. *x_center, y_center, width* and *height* should be normalized relative to the image dimension.

1.2 Conversion Process

The conversion algorithm (**pascalVOC_to_yolo.py**) performs the following steps:

Step 1: Read and parse the contents of each XML file. Extract all the relevant information such as object class labels, bounding box coordinates, image dimensions.

Step 2: Calculate *x_center*, *y_center, width* and *height* and normalize them relative to the image dimensions. If the boxes are in pixels, for normalization, *x_center* and *width* needs to be divided by image width, and *y_center* and *height* by image height.

Step 3: Save the class id (number specifying the class) along with the normalized coordinates in a text file. Class numbers are zero-indexed (starts with 0) for YOLOv8 model.

2. Person Detection

2.1 Dataset

The above mentioned conversion algorithm (**pascalVOC_to_yolo.py**) was used to generate the relevant text (.txt) files which were used along with corresponding images for training.

2.2 Training

YOLOv8.s pre-trained model was used for person detection. This model was fine-tuned over 350 image samples for 50 epochs.

The area under curve (AUC-PR) is shown in Fig. 1. The AUC value is very high indicating the model is able to effectively identify persons in the image**.**

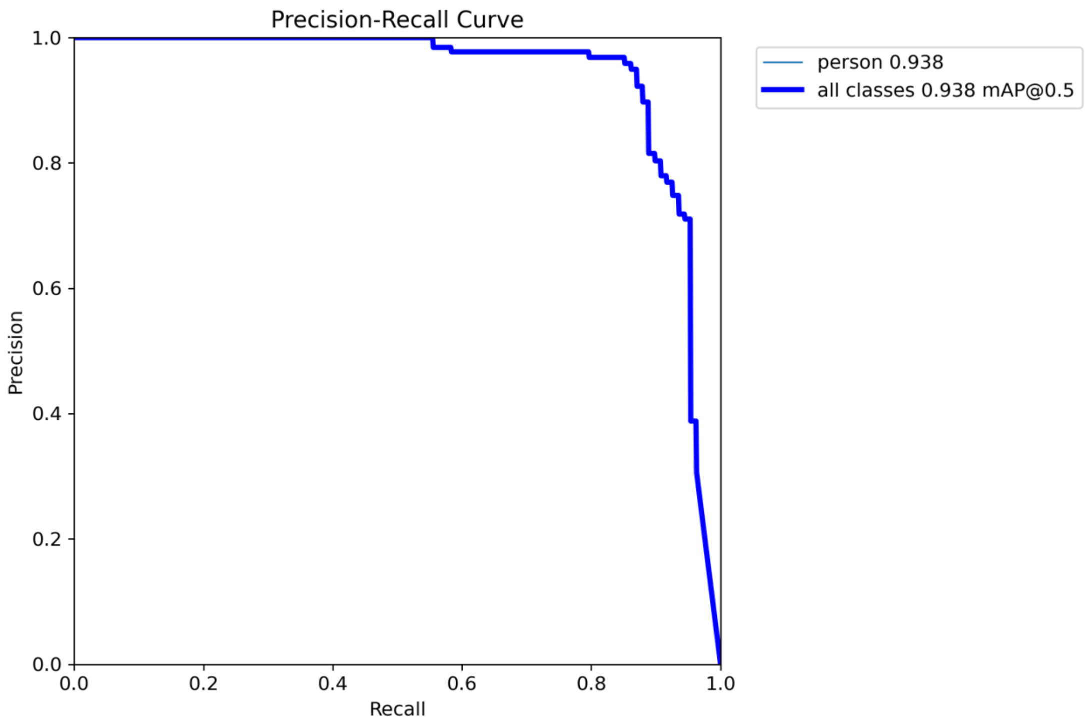

Fig. 1: AUC-(PR) for person detection model.

2.3 Validation Results

Model predictions on the validation dataset samples are given in Fig. 2.

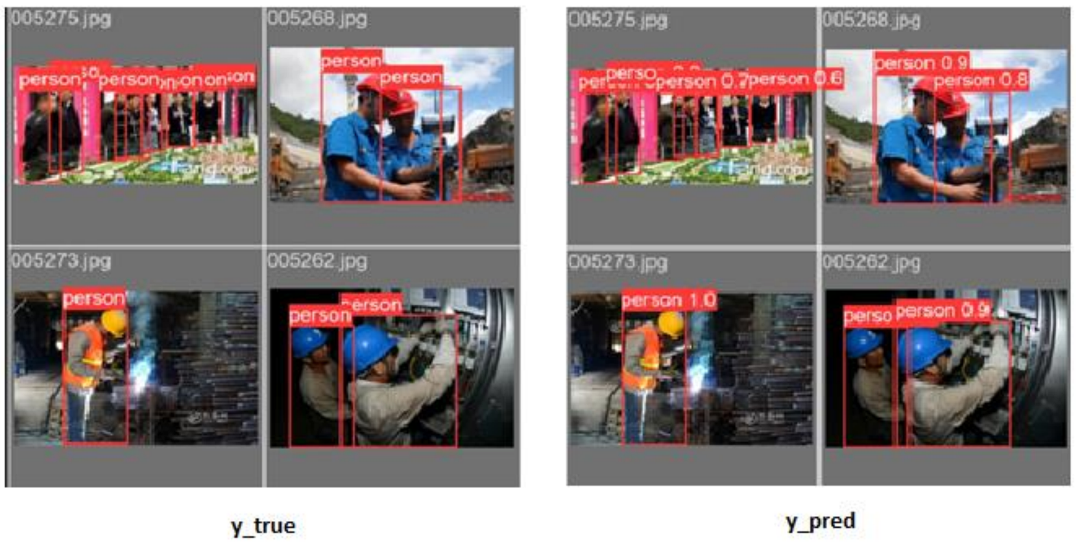

Fig. 2: Expected bounding boxes v/s predicted bounding boxes of person.

3. PPE Detection

3.1 Dataset preparation

Following steps need to be performed on each XML file (containing annotations for all objects within the image) and the corresponding image in order to prepare the dataset:

Step 1: Parse the contents of the XML file and extract the bounding box coordinates of all the objects present in the image.

Step 2: Iterate over the object-coordinate pair list and check if the object is a “person”. If it is, crop its bounding box and save it. This cropped image will be the input image during the training process.

Step 3: Iterate again over the object-coordinate list and search for objects (ppe-items) that fall inside the person’s bounding box.

Step 4: Adjust the coordinates of objects that fall within the cropped image (person’s bounding box) relative to the cropped image dimensions to accurately represent the position of objects within the cropped image.

Step 5: Save the coordinates of the objects inside the person’s bounding box in an XML file.

Step 6: Convert the XML file to a YOLOv8 format text file. This YOLOv8 file will be used during training.

The code to prepare the dataset is given in **ppe_detection/dataset.py**

3.2 Class Distribution Analysis

Class imbalances can adversely affect the performance of machine learning models, leading to decreased accuracy, especially for classes with fewer instances. Class distribution analysis was conducted to gain insights into the distribution of different objects within the dataset. Objects here refer to various ppe items such as hard-hat, gloves, mask, glasses, boots, vest, ppe-suit, ear-protector and safety-harness.

The class distribution analysis was carried out on the train data (XML files) of ppe-detection model. The results are pictorially represented in the Fig. 3.

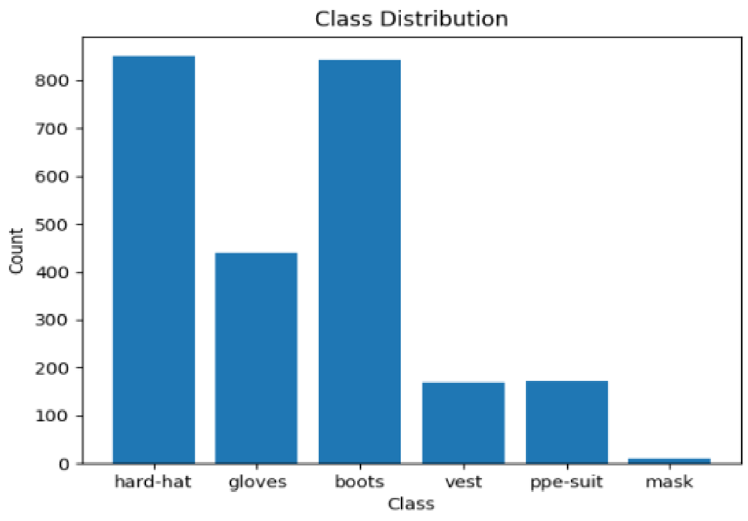

Fig. 3: Bar chart showing class distribution in the training data for ppe-detection.

It was found that the classes **“glasses”, “ear-protector” and “safety-harness” had no samples** at all. The **“mask” class had very few samples and thus wasn’t included in training data**. The remaining classes data were used for training**.**

3.3 Training

YOLOv8.s pre-trained model was used for ppe detection. This model was fine-tuned over 1000 image samples for 50 epochs.

The area under the curve (AUC-PR) as shown in Fig. 4 provides a summary of the model's overall performance across all classes, with higher values indicating better performance**.**

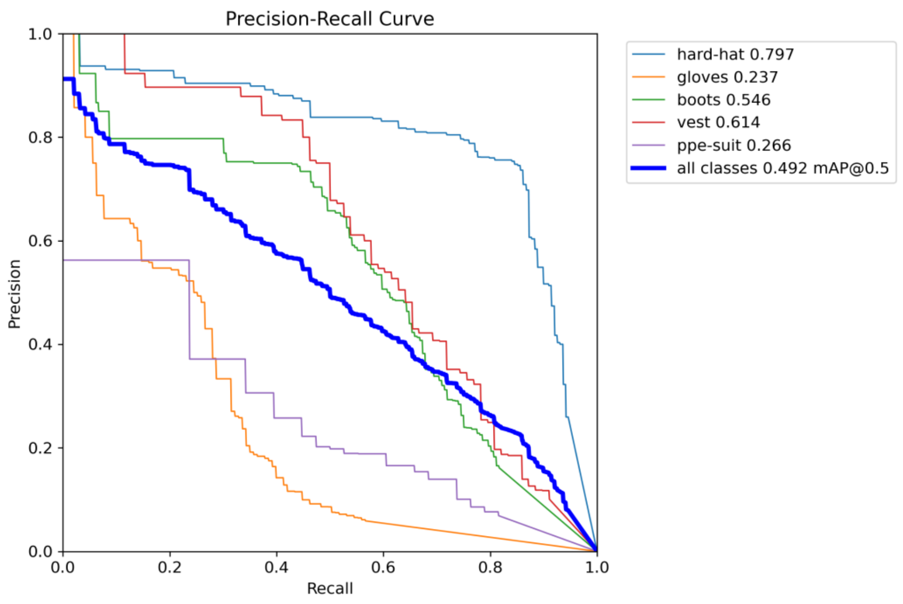

Fig. 4: AUC-(PR) for ppe-detection model.

3.4 Validation Results

Model predictions on the validation dataset samples are given in Fig. 5.

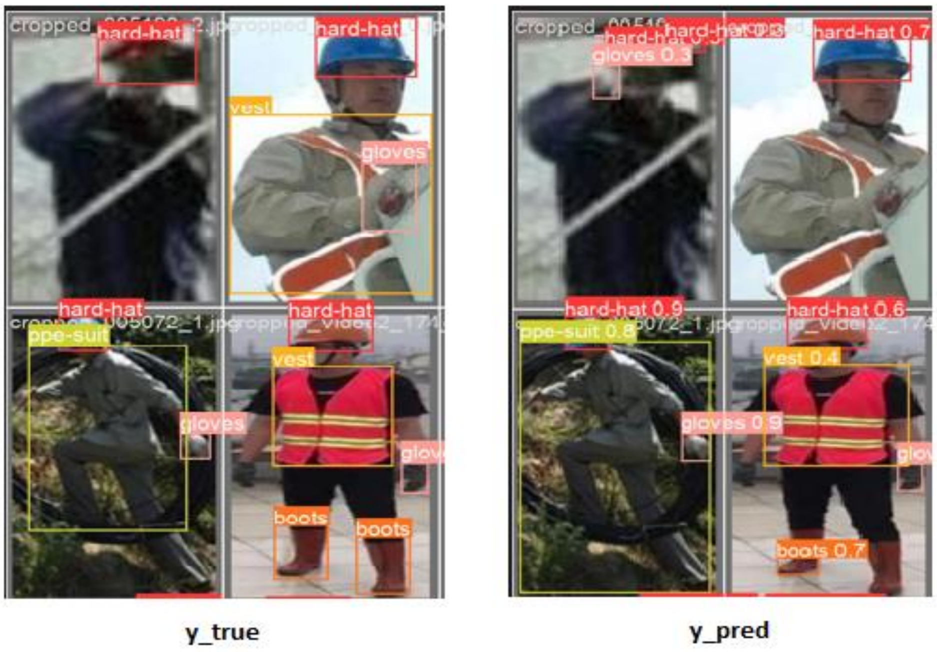

Fig. 5: Expected bounding boxes v/s predicted bounding boxes of ppe.

4. Learning

Few learning from this assignment were:

1. Learned to work with XML files.
2. Gained practical experience working with YOLOv8 models for object detection task.
3. Utilizing Python’s *argparse* library.