Project Report: Person and PPE Detection

1. Introduction

This project aimed to develop a robust object detection system using YOLOv8 for two key tasks: detecting persons in images and detecting personal protective equipment (PPE) on the detected persons. The project was divided into several steps, including data preprocessing, model training, inference, and evaluation. The primary goals were to accurately detect persons in images, crop those images to focus on each detected person, and then detect PPE on the cropped person images.

2. Objectives

2.1. **Person Detection:** Train a YOLOv8 model to detect persons in images.
2.2. **PPE Detection:** Train another YOLOv8 model to detect various PPE items on cropped images of detected persons.
2.3. **Data Preprocessing:**
- Convert annotations from PascalVOC format to YOLO format.
- Split the dataset into training, validation, and test sets.

2.4. **Inference:** Create a pipeline that:
- Detects persons in full images.
- Crops images around each detected person.
- Applies the PPE detection model to each cropped image.
- Maps the detected PPE back to the original full image.

2.5. **Evaluation:** Measure the performance of both models and visualize results.

3. Data Preprocessing

3.1. Dataset Structure

The dataset consisted of images and corresponding annotations in PascalVOC format. The dataset was organized as follows:

- `dataset/`
  - `images/`: Contains all images.
  - `annotations/`: Contains PascalVOC annotation files.
  - `classes.txt`: Lists the class labels.

3.2. PascalVOC to YOLO Conversion

A script named **`pascalVOC_to_yolo.py`** was developed to convert PascalVOC annotations to YOLOv8 format. This conversion was necessary because YOLOv8 requires annotations in a specific format (class ID, x_center, y_center, width, height), normalized to the image dimensions.

- **Key Steps in Conversion:**
  - Parse the PascalVOC XML files.
  - Extract bounding box coordinates and class labels.
  - Normalize the coordinates to the YOLO format.
  - Save the converted annotations in a separate directory.

3.3. New label/annotation file for person detection

A script was developed to create new annotation files for person detection.
All the annotations of PPE were deleted as this model would only be detecting the person and the annotation file contains all the annotations of persons and ppe classes,

3.4. Cropping of images for PPE Detection

A script was developed to read the images and annotation files and crop each image into separate images for every persons present in the original image and create new annotation files for the cropped images after adjusting the coordinated of the annotations as per the new cropped images and removing ‘person’ label & annotation from the file.

Also created a cropped_classes.txt file containing only the ppe annotation label classes.

3.5. Handling Class Imbalance

A script was developed to check the class imbalance and remove the classes with very low data or no data.
Here few classes were dropped having very low amounts of data.

Another script was developed to augment the data. Logic was implemented to augment the images where the annotations of low data classes are present more and less annotations of high data classes.

3.6. Dataset Splitting

The dataset was split into training, validation, and test sets:

- Training Set: 80% of the data.
- Validation Set: 20% of the training data (further split during training).
- Test Set: 20% of the data.

4. Model Training

4.1. Creating YAML files

- Two YAML files are developed **peson_detection.yaml** and **ppe_detection.yaml** containing training data information.

4.2. Person Detection Model

The first YOLOv8 model was trained to detect persons in the full images.

Training Details:

- **Base Model:** YOLOv8m (pretrained on COCO).
- **Data:** 80% dataset was used to train the model, focusing on person class.
- **Optimizer:** Adam.
- **Learning Rate:** 0.0001.
- **Epochs:** 75.
- **Image Size:** 640x640.
- **Batch Size:** 16.
- **Validation:** Enabled, with plots of training and validation metrics.

Training Command:

```
!yolo task=detect mode=train model=yolov8m.pt
data=/content/gdrive/MyDrive/object_detection/datasets/person_data.y
aml epochs=75 imgsz=640 batch=16
project=/content/gdrive/MyDrive/object_detection/datasets/training_r
esults name=person_detection val=True plots=True lr0=0.0001
verbose=True optimizer=Adam
```

4.3. PPE Detection Model

The second YOLOv8 model was trained to detect PPE items on cropped images of detected persons.

Preprocessing:

- The images were cropped based on the bounding boxes from the person detection model.
- Cropped images were saved in a separate directory, with corresponding annotations updated to reflect the cropped region.
- Class imbalance handling and data augmentation was performed.

Training Details:

- **Base Model:** YOLOv8m (pretrained on COCO).
- **Data:** Cropped images with updated annotations.
- **Classes:** Hard-hat, gloves, mask, boots, vest, PPE-suit.
- **Removed Classes :** glasses, ear-protector, safety-harness.
- **Optimizer:** Adam.
- **Learning Rate:** 0.0001.
- **Epochs:** 75.
- **Image Size:** 640x640.
- **Batch Size:** 16.
- **Validation:** Enabled, with plots of training and validation metrics.

Training Command:

```
!yolo task=detect mode=train model=yolov8m.pt
data=/content/gdrive/MyDrive/object_detection/datasets/ppe_data.yaml
epochs=75 imgsz=640 batch=16
project=/content/gdrive/MyDrive/object_detection/datasets/training_r
esults name=ppe_detection val=True plots=True lr0=0.0001
verbose=True optimizer=Adam
```

4.4. Challenges & Solutions:

- **Cropping Logic:** Ensured that bounding boxes were accurately updated or removed if outside the crop boundary.
- **Multiple Crops:** Created separate crops and annotations for each detected person in the image.
- **Corrupt Files:** Some images had corrupt or incomplete annotations, which were filtered out and logic was improved to correct these cases.

5. Inference Pipeline

The `inference.py` script was developed to perform inference using both the person detection and PPE detection models.

Steps in Inference:

- **Person Detection:** Run inference on full images to detect persons.
- **Cropping:** Crop the detected person regions.
- **PPE Detection:** Run inference on the cropped person images to detect PPE items.
- **Mapping:** Convert the PPE detections back to the original full image coordinates.

- **Output:** Save the results in output directory, drawing bounding boxes using OpenCV.

Challenges:

- **Mapping Coordinates:** Careful calculations were required to accurately map PPE detections back to the original image coordinates.
- **Handling Large Outputs:** Managed large outputs efficiently to avoid memory issues.

6. Evaluation and Metrics

6.1. Accuracy and Loss

The models were evaluated based on:

- **Precision, Recall, and mAP (mean Average Precision):** For both person and PPE detection models.
- **Training and Validation Loss:** Plotted during training to monitor overfitting and model convergence.

6.2. Visualizations

After training, the following were visualized:

- **Training and Validation Loss:** To observe the model's learning behavior.
- **Precision-Recall Curves:** To assess model performance at various confidence thresholds.

6.3. Test Set Evaluation

The models were evaluated on the test set to ensure that they generalized well to unseen data. The final metrics were reported for both person detection and PPE detection.

Challenges:

- **Class Imbalance in PPE Detection:** Despite efforts to balance the dataset, certain PPE classes were underrepresented, slightly affecting their detection accuracy.

7. Conclusion

This project successfully developed and deployed a two-stage detection system using YOLOv8. The system was capable of detecting persons in images and subsequently detecting various PPE items on the cropped images of those persons. The key challenges involved annotation preprocessing, and addressing class imbalance. The models performed well, with high accuracy on the test set, demonstrating the effectiveness of the approach.