Object Detection System Project 🚀

Step 1: Import Libraries and Define Paths 📂

In this initial step, I imported essential libraries such as **Pandas** for data manipulation, **OpenCV** for image processing, and **YOLO** from the **Ultralytics** library for object detection. I defined paths for my dataset and output directories, setting a strong foundation for the project.

Step 2: Create Directories and Split Data 🗂️

I established directories for training and validation datasets to ensure an organized workflow. I also split the dataset into an 80/20 ratio, where 80% was used for training and 20% for validation. This split is critical for evaluating the model's performance on unseen data.

Step 3: Define Image Preprocessing Function 🖼️

I created a preprocessing function to read and save images from the original dataset to the training and validation directories. This function ensures that the images are correctly organized and ready for model training, crucial for optimal model performance.

Step 4: Execute Image Preprocessing 🔄

I executed the preprocessing function to process images for both training and validation sets, ensuring that the images were correctly formatted and accessible for the model training phase.

Step 5: Prepare YAML File for YOLO 📝

To configure YOLO for training, I prepared a YAML file detailing the paths to training and validation images, the number of classes, and their names. This step is vital for YOLO to understand the structure of the dataset.

Step 6: Write YAML File 💾

I saved the YAML configuration file, which provides YOLO with the necessary data paths and structure, facilitating the subsequent training process.

Step 7: Train YOLO Model 📊

I loaded the YOLO model and initiated the training process over 10 epochs. During training, the model learns to identify the specified object classes, refining its parameters based on the provided dataset. This step is critical for developing a robust object detection model.

Step 8: Perform Inference and Visualize Results 🔍

Once trained, I performed inference on a random image from the dataset. I visualized the detection results by drawing bounding boxes around detected objects, enhancing the interpretability of the model's predictions. The final output image was saved and displayed, showcasing the model's capability to accurately identify and locate objects.

Results and Interpretation 🎯

The model successfully detected **7 persons** in the selected image, achieving an inference time of approximately **118ms**. This performance highlights the model's efficiency and effectiveness in real-time object detection scenarios. Such results are essential for applications in various domains, including surveillance, autonomous vehicles, and human-computer interaction.

Skills Acquired 🧠

Throughout this project, I honed my skills in **data preprocessing, model training, computer vision, YOLO architecture, image visualization**, and **results interpretation**.

Top Hashtags for LinkedIn 📈

#ObjectDetection #YOLO #ComputerVision #DeepLearning #DataScience #ImageProcessing #AI #MachineLearning #Ultralytics #Python #Pandas #OpenCV #NeuralNetworks #DataVisualization #ModelTraining #ComputerVisionProjects #AIProjects #DataPreprocessing #ResultsInterpretation #TechForGood