OCR Custom Model Training Project Documentation

# Overview

This project focuses on building a robust OCR pipeline using custom training models for both text detection and text recognition tasks. The detection model is based on YOLOv9-gelan-e, and the recognition model is based on PaddleOCR (PPOCR). The entire process includes large image preprocessing, data augmentation, label transformation, and model training.

# Step-by-Step Project Execution

## 1. Initial Image Preprocessing

### Objective:

To handle large images (e.g., 8000×9000 pixels) that are not compatible with YOLO input directly.

### Process:

- Initially, images were cropped into 2048×2048 patches for compatibility with the YOLOv8 model.  
- Due to a licensing issue with YOLOv8 (non-GPL compatible), we shifted to YOLOv9-gelan-e, which is under the GPL-3.0 license.  
- YOLOv9-gelan-e is larger model which fails to properly fit into the GPU of 15GB, So the large patches of 2048×2048 cropped into images of 1792×1792 dimensions.

### Automation Script:

- A robust Python script was written in the notebook Preparing\_Dataset.ipynb.  
- It:  
 - Skips already cropped 1792×1792 images.  
 - Precisely crops larger patches (2048×2048) into 1792×1792 slices.  
 - Ensures compatibility for detection.

## 2. Dataset Verification and Label Preprocessing

Location: Preparing\_Dataset.ipynb

### Script 1: Dataset Verification

- Verifies the dataset structure and label consistency for both:  
 - Detection dataset (bounding boxes).  
 - Recognition dataset (text images with transcription).

### Script 2: Label Transformation with 90-Degree Rotation

- Applies a 90° clockwise rotation to both images and their corresponding labels.  
- Updates bounding box coordinates accordingly.  
- This increases model robustness to various text orientations.

### Script 2.1: Optional Augmentation (Coloring, Blurring, etc.)

- This optional script applies additional augmentations such as color distortions, Gaussian blurring, and noise injection.

- These augmentations can improve the robustness and generalization of the detection model.

- It is particularly useful when training a base model from scratch.

- However, when fine-tuning an already well-trained model, it may not be necessary to use this script.

### Script 3: Preparing Recognition Dataset

- Uses label.json and the original images folder.  
- Generates a dataset compatible with PaddleOCR recognition model, including:  
 - Cropped text regions (image crops of individual words/text lines).  
 - Corresponding labels in the correct format.

Final dataset structure:

recognition\_dataset/  
 ├── train\_images/  
 ├── valid\_images/  
 ├── train.txt  
 └── valid.txt

further described the exact data structure format below.

## 3. Object Detection Model Training

Location: train\_yolov9\_object\_detection\_on\_custom\_dataset.ipynb

### Model Used:

- YOLOv9-gelan-e from the official YOLOv9 repository.

### Dataset Format for YOLO:

detection\_dataset/

├── train/

│ ├── images/

│ └── labels/

├── val/

│ ├── images/

│ └── labels/

└── data.yaml

**data.yaml**

*names:*

*- Text*

*nc: 1*

*train: ../dataset/train/images*

*val: ../dataset/valid/images*

### Training Configuration:

- Batch size: 1, image size: 1792, and hyperparameters configured in YOLOv9 training script.  
- Model checkpoints saved during training.

## 4. Text Recognition Model Training

Location: Train\_PPOCR\_Text\_Recognition\_Model.ipynb

### Model Used:

- PPOCR (PaddleOCR) from PaddlePaddle's official GitHub repository.

### Dataset Format:

- Each image in train\_images/ corresponds to a line in train.txt:  
 train\_images/img\_1.jpg Hello  
 train\_images/img\_2.jpg World

### Training Flow:

- Customized configuration YAML for PaddleOCR to load dataset, model architecture, and optimizer.  
- Finetuning on custom cropped word images.  
- Evaluation done on validation subset during training.

# Additional Details

## Label Structure for Detection

- YOLO format:  
 class\_id x\_center y\_center width height  
- All values are normalized between 0 and 1 with respect to the image width and height.  
- Rotated labels are recalculated for proper detection alignment.

## Label Structure for Recognition

- txt format for initial labels like:  
   
 "img\_1.jpg": text: “Example”

“img\_2.jpg”, text: “Example2”  
   
- Processed into PaddleOCR-compatible format.

# Conclusion

This project successfully implements a custom OCR pipeline leveraging YOLOv9 for detection and PaddleOCR for recognition. Key innovations include:  
- Smart image cropping.  
- Rotation-based label augmentation.  
- Full compatibility with open-source tools under GPL-3.0.  
- End-to-end dataset generation and training pipeline.