Real-Time Object Detection for Autonomous Vehicles

Executive Summary

This report presents the development and evaluation of a YOLOv8-based object detection system for traffic scene analysis. The model was trained on a customized COCO dataset containing various traffic-related objects like vehicles, pedestrians, and traffic lights. Through extensive data preparation, exploratory analysis, and model training, we achieved a robust detection system capable of identifying multiple object classes in traffic scenes.

Data Collection and Preparation

Dataset Overview

- **Source**: COCO format dataset (cocococo-dataset) containing traffic-related images
- **Objects**: Various traffic elements, including vehicles, pedestrians, traffic lights, and obstacles
- **Classes**: 12 distinct categories including cars, trucks, pedestrians, and various traffic light states
- **Format**: Original data in COCO JSON format, later converted to YOLO format for training

Data Cleaning Process

1. **Upload Verification**: Confirmed kaggle.json setup and verified the successful download of the dataset
2. **Identification of Non-Image Files**: Located and properly handled non-image files (primarily annotation JSON files)
3. **Deletion of Unannotated Images**: Removed images without corresponding annotations to improve dataset quality
4. **JSON File Update**: Modified the annotation file to reflect the removed images, maintaining dataset integrity

Exploratory Data Analysis (EDA)

Basic Statistics

- **Total Images**: The cleaned dataset contained annotated images ready for training
- **Total Bounding Boxes**: Multiple bounding boxes across the dataset, providing rich training examples
- **Class Distribution**: Analysis revealed class imbalance with some categories significantly more represented than others

Image Quality Analysis

- **Resolution**: Most images had standard dimensions with consistent aspect ratios
- **Blur Analysis**: Average blur scores were acceptable, indicating decent image quality
- **Noise Levels**: Noise estimation showed manageable noise levels across the dataset

Object Analysis

- **Bounding Box Dimensions**: Distribution analysis of width, height, and aspect ratios
- **Spatial Distribution**: Heat map analysis of object positions showed concentration in central areas
- **Class Co-occurrence**: Analysis of which classes appear together in images

Data Quality Metrics

- **Small Objects**: Percentage of objects with dimensions less than 32x32 pixels
- **Crowded Images**: Images with more than 10 objects identified for special attention
- **Edge Cases**: Objects located near image boundaries

Dataset Preparation for Training

Subset Creation

- Created a manageable subset of 3000 images for model development
- Preserved class distribution and annotations in the subset

Data Splitting

- **Training Set**: 75% of the subset
- **Validation Set**: 15% of the subset
- **Test Set**: 10% of the subset

YOLO Format Conversion

- Converted COCO format annotations to YOLO format:
  - COCO: [x_min, y_min, width, height]
  - YOLO: [x_center/img_width, y_center/img_height, width/img_width, height/img_height]
- Organized into the required directory structure for YOLOv8 training

Data Configuration

- Created `data.yaml` configuration file specifying:
  - Path structure
  - Number of classes (11)
  - Class names mapping

Model Training

Training Configuration

- **Base Model**: YOLOv8n (nano variant)
- **Input Size**: 640x640 pixels
- **Batch Size**: 16
- **Epochs**: 80 with early stopping (patience: 12)
- **Learning Rate**: 0.001
- **Optimizer**: SGD
- **Augmentation**: TRUE

Augmentation Strategy

Implemented extensive data augmentation to improve model robustness:

- HSV augmentation (hue, saturation, value)
- Geometric transformations (rotation, translation, scaling, shear)
- Mosaic augmentation (strength: 1.0)
- Mixup augmentation (strength: 0.2)

Training Process

- Training executed with specified hyperparameters
- Early stopping mechanism to prevent overfitting
- Progress is monitored through loss metrics and validation performance

Model Evaluation

Performance Metrics

Evaluated model performance on the test set using standard metrics:

- Precision
- Recall
- mAP (mean Average Precision)
- F1-score

Visualization

- Generated prediction visualizations on test images
- Displayed bounding boxes with class labels and confidence scores
- Visual assessment confirmed detection quality across various scenarios

Model Export

- Exported the best-performing model weights (`best.pt`)
- Created a package for easy deployment and future use
- Verified the exported model's integrity and availability

Conclusions

The developed YOLOv8-based traffic object detection system demonstrates reliable performance in identifying various traffic elements. The model successfully detects multiple object classes, including vehicles, pedestrians, and traffic lights, in different states.

Key Achievements

- Thorough data preparation and cleaning process
- Comprehensive exploratory data analysis providing insights
- Successful implementation of YOLOv8 with extensive augmentation
- Exportable model ready for deployment in traffic analysis applications

Future Improvements

- Collection of additional data for underrepresented classes
- Testing with larger YOLOv8 variants (s, m, l) for potentially higher accuracy
- Implementation of additional augmentation techniques for edge cases
- Investigation of model quantization for faster inference