Traffic Sign Detection and Classification using YOLO and CNN

JAYPAL

July 3, 2025

Abstract

This project focuses on the detection and classification of traffic signs using a combination of YOLO (You Only Look Once) for object detection and a Convolutional Neural Network (CNN) for classification. The aim is to build an end-to-end automated system that can identify and classify traffic signs from images, providing a foundation for applications in self-driving vehicles and intelligent transportation systems.

1 Introduction

The detection and classification of traffic signs is crucial for road safety and autonomous driving systems. In this project, we employed YOLOv8 for real-time detection of traffic signs and a CNN for accurate classification.

2 Libraries and Tools Used

- Python
- Google Colab
- PyTorch
- Ultralytics YOLOv8
- Flask (for web deployment)
- LaTeX (for documentation)

3 Dataset

We used the German Traffic Sign Recognition Benchmark (GTSRB) dataset which contains over 50,000 images of 43 traffic sign classes. The dataset was used for both object detection (YOLO) and classification (CNN).

- **Dataset Link:** `https://benchmark.ini.rub.de/gtsrb_dataset.html`
- Data was converted into YOLO format for detection and resized to 32×32 for CNN classification.

4 Data Preprocessing

- All images were resized to 416×416 for YOLO and 32×32 for CNN.
- YOLO labels were created in `.txt` files containing normalized bounding box coordinates and class labels.
- CNN images were normalized and augmented using PyTorch's `transforms` module.

5 YOLO Detection Model

- YOLOv8s model was trained on a smaller subset (500 images) to reduce training time.
- Achieved mAP@0.5: **99.48%**
- Precision: **90.15% to 100%**, Recall: **84.46% to 100%**

6 CNN Classification Model

- A simple 2-layer CNN was trained on the GTSRB classification dataset.
- Achieved Test Accuracy: **XX%** (Fill in your result)
- Evaluation metrics included Accuracy and Confusion Matrix.

7 Integration: YOLO + CNN

The system was designed to first detect traffic signs using YOLO, crop the detected region, and then classify the cropped sign using the CNN model. This two-stage pipeline ensures both high detection accuracy and fine-grained classification.

8 Flask Web Deployment

We deployed the complete solution using Flask, allowing users to:

- Upload an image
- Detect traffic signs (YOLO)
- Classify signs (CNN)
- View predictions on a simple web interface

9 Key Learnings

- Understood object detection and classification fundamentals.
- Learned YOLO training, evaluation, and inference.
- Built and evaluated CNN models for multi-class classification.
- Integrated two models into a seamless detection-classification pipeline.
- Developed and deployed a web application using Flask.

10 Conclusion

This project successfully demonstrated a real-time Traffic Sign Detection and Classification system using deep learning techniques. Future work could include deploying the system on embedded devices or improving classification accuracy with advanced models.