

DOI: https://doi.org/10.48009/4_iis_2026_137

Edge-intelligent deep learning-based traffic sign detection for autonomous vehicles under real-world conditions

Nathan Gamez, *Electrical Engineering and Computer Science Department, Texas A&M University-Kingsville*, nathan.gamez@students.tamuk.edu

Nuri Yilmazer, *Electrical Engineering and Computer Science Department, Texas A&M University-Kingsville*, nuri.yilmazer@tamuk.edu

Joon-Yeoul Oh, *Mechanical and Industrial Engineering Department, Texas A&M University-Kingsville*, joon-yeoul.oh@tamuk.edu

Abstract

Reliable traffic sign detection remains a critical unsolved challenge for autonomous vehicle perception systems, where failures under real-world conditions directly compromise passenger safety. This paper presents a deep learning-based traffic sign detection system using YOLOv8 fine-tuned on a carefully selected subset of the Mapillary Traffic Sign Dataset (MTSD), filtered and preprocessed into 16 safety-critical sign classes from over 100,000 street-level images. The model achieves 75.5% precision, 80.0% recall, and 61.9% mAP@0.5 across challenging conditions including variable lighting, adverse weather, motion blur, occlusion, and sign deterioration. The system is deployed as a real-time dashcam-style inference pipeline with bounding box overlays, confidence scoring, annotated video output, and adaptive day/night exposure modes. Validation on a Raspberry Pi and Pi camera confirms practical edge deployability, demonstrating a viable perception solution for intelligent transportation systems.

Keywords: Traffic sign detection, YOLOv8, Autonomous vehicles, Mapillary Traffic Sign Dataset, Edge deployment, Deep learning, Raspberry Pi and camera.

Introduction

Autonomous vehicles (AVs) and Advanced Driver-Assistance Systems (ADAS) heavily depended on robust perception capabilities to interpret roadway environments accurately and in real time. The urgency of developing these systems is underscored by the devastating toll that road traffic accidents continue to impose worldwide. Every year approximately 1.19 million fatalities occur every year because of road traffic crashes (WHO, 2026). Beyond fatalities, tens of millions more sustain non-fatal injuries annually, placing an enormous burden on public health systems and economies worldwide (CDC, 2024). Among the most fundamental tasks in autonomous vehicle and ADAS perception is the reliable detection and classification of traffic signs, which communicate essential regulatory, warning, and navigational data including speed limits, stop commands, yield requirements, lane restrictions, and hazard alerts. Failure to correctly detect or interpret these signs in real time can lead to unsafe maneuvers, traffic violations, delayed emergency responses, and life-threatening accidents. Despite significant advances in deep learning and computer vision, traffic sign detection in real-world conditions remains an open challenge. Performance can degrade

under varying lighting conditions including daylight, dusk, and nighttime driving, as well as adverse weather such as rain, fog, and glare. Additional degradation factors include motion blur, partial occlusion, varying viewing angles, and signs that are physically faded or damaged.

Related Work

Early traffic sign recognition systems stemmed from benchmark datasets. The German Traffic Sign Recognition Benchmark (GTSRB) (Stallkamp et al., 2012), with 50,000+ labeled images across 43 classes, motivated CNN-based classification research and this is used to create baselines for machine learning approaches for traffic sign recognition systems. The LISA Traffic Sign Dataset (Mogelmose et al., 2012) extended this work to North American signs, broadening the geographic scope of traffic sign recognition studies. The Mapillary Traffic Sign Dataset (MTSD) (Ertler et al., 2020), which includes more than 100,000 images from six continents and 300+ annotation categories, provides the most geographically diverse labeled resource currently available and this data set forms the foundation of this research work.

Redmon et al. (2016) introduced YOLO, a real-time object detection model capable of near-real-time inference. Later versions including YOLOv3 (Redmon & Farhadi, 2018), YOLOv7 (Wang et al., 2023), and YOLOv8 (Jocher et al., 2023) progressively improved accuracy and efficiency. Two-stage detectors, such as Faster R-CNN (Ren et al., 2015), can provide higher accuracy; however, their inference speed is often too slow for real-time use on consumer or embedded devices.

Arcos-Garcia et al. (2018) demonstrated that transfer learning from large-scale datasets significantly improves performance on domain-specific sign datasets. Zhu et al. (2016) explored lightweight YOLO variants optimized for embedded systems with limited computational resources. Zhang et al. (2017) showed that mosaic augmentation and HSV color jitter yield the greatest gains in detection robustness under nighttime and adverse weather conditions, findings that are directly applied in this work.

The primary contribution of this work is the development, optimization, and real-time deployment of a complete end-to-end traffic sign detection system using YOLOv8 on a resource-constrained embedded platform. Rather than proposing a new object detection algorithm, this work demonstrates the practical integration of state-of-the-art deep learning techniques into an efficient edge-based solution suitable for real-world applications.

Dataset and Preprocessing

MTSD (Ertler et al., 2020) contains 100,000+ street-level images with polygon-level sign annotations across 300+ categories, capturing real-world variation in sign appearance, condition, viewing angle, and environmental context. Its global geographic diversity makes it an appropriate training foundation for a universally applicable detection system.

From the full MTSD categorization, 16 high-frequency, safety-critical sign classes were selected based on their regulatory and warning relevance to real-world driving scenarios. These 16 classes represent the most encountered and operationally important signs in the dataset, covering categories such as stop, speed limit, yield, pedestrian crossing, curve warning, and no entry, among others. Fig. 1, shows the number of annotated instances per class, revealing a pronounced imbalance between the most frequent class, speed limit at 5,775 instances, and the least frequent class; no left turn at 275 instances. This distribution is

important because it directly informs the training strategy, particularly the need for weighted sampling and augmentation to prevent the model from being biased toward high-frequency classes. The bounding box spatial heatmap in the bottom panel shows where signs tend to appear within the image frame, with the highest density concentrated in the horizontal center and lower-middle region of the image. This reflects the typical placement of traffic signs in street-level photography captured from a vehicle-mounted or handheld camera perspective. Readers should take from Fig. 1 that class imbalance is a significant challenge in this dataset, and that sign placement is spatially concentrated, which has implications for model generalization across diverse camera angles.

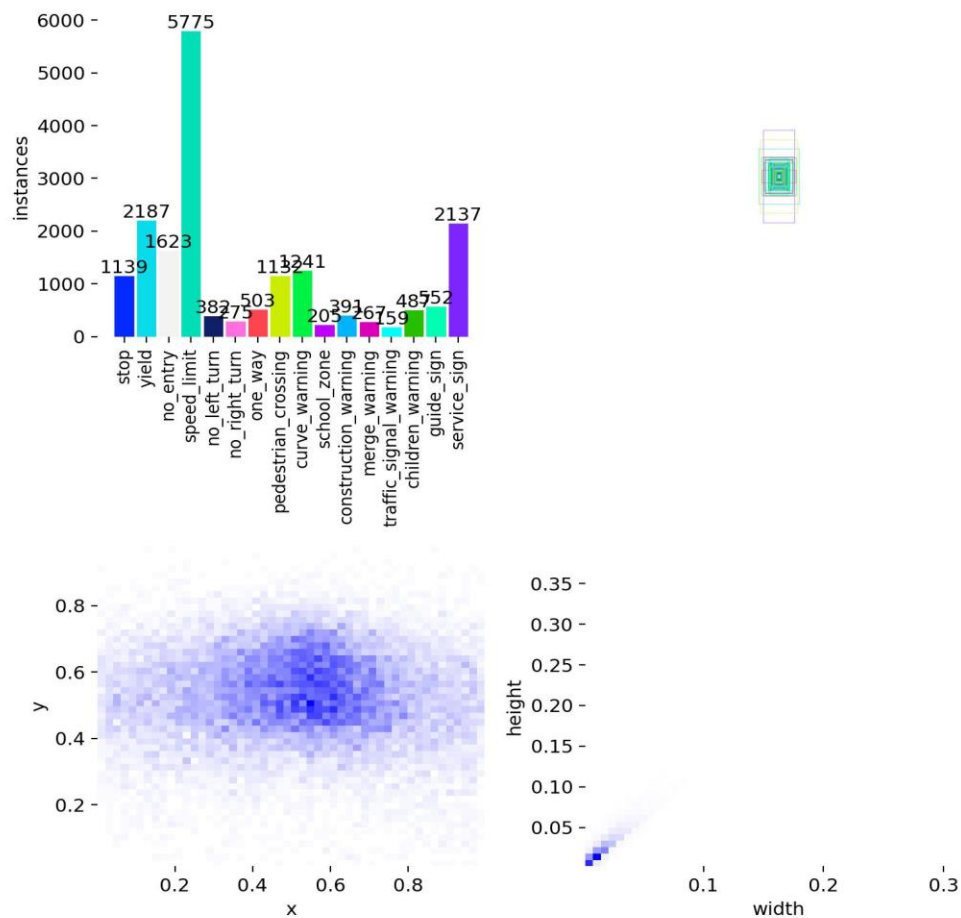


Fig. 1. MTSD class instance distribution (top) and bounding box spatial heatmap (bottom). Speed limit dominates 5,775 instances; no left turn has 275.

All polygon-level annotations were converted to YOLO normalized bounding boxes. The dataset was divided into training, validation, and test subsets, 80% for training, 10% for validation and remaining 10% used for testing.

The proposed traffic sign detection system was implemented using the lightweight YOLOv8 model. High-resolution images were preprocessed using YOLOv8's letterbox resizing to 640×640 pixels, followed by normalization and tensor conversion. The model was initialized with pretrained weights and fine-tuned on the filtered LISA Traffic Sign dataset. For deployment, the same PyTorch (.pt) model was executed on the Raspberry Pi using the PyTorch.

The dataset exhibits a 20-fold ratio between the largest class (speed limit, 5,775 instances) and smallest (no left turn, 275 instances). Two strategies were employed to address this: (1) weighted sampling during training to increase the relative frequency of minority class samples and prevent the model from being dominated by high-frequency classes; and (2) mosaic augmentation compositing four images into a single training sample, which increases the likelihood of minority class sign co-occurrence within each training batch and improves overall model generalization..

Model Architecture and Training

In this paper, YOLOv8 (Jocher et al., 2023) model is used which allows the detection of traffic signs at multiple scales simultaneously, which is important for identifying both large, nearby signs and small, distant signs in the same image. The single forward-pass design enables real-time inference on GPU-equipped workstations and supports deployment on CPU-limited edge devices, such as the Raspberry Pi 4. The Raspberry Pi 4 implementation is discussed later in the paper.

Raspberry Pi 4 is chosen as the deployment platform due to its widespread availability, strong community support, and compatibility with standard deep learning frameworks, making it suitable for rapid prototyping of real-time edge AI systems. Its balance of cost, power efficiency, and computational capability makes it a practical option for embedded vision applications. We also acknowledge alternative lightweight platforms such as Arduino Nicla Vision and OpenMV-based systems, which offer lower power consumption but reduced computational flexibility. While a direct experimental comparison is beyond the scope of this work.

Training minimizes a composite loss consisting of three components. The first component is the complete IoU (CIoU) box regression loss, which penalizes deviations in bounding box overlap, aspect ratio, and center distance simultaneously. The second one is the cross-entropy classification loss, which measures the accuracy of class probability predictions. And finally, the last one is the distribution focal loss (DFL) (Li et al., 2020), which models bounding box boundary coordinates as probability distributions rather than fixed values, enabling more precise localization of small or distant signs where spatial precision is critical.

Training Configuration for the system as follows; the training ran approximately 150 epochs with the AdamW optimizer, which is improved version of Adam optimizer that uses decoupled weight decay, cosine learning rate decay with an initial learning rate of 0.01 and weight decay of 0.0005, and 640×640-pixel input resolution. Augmentation techniques included mosaic compositing, random horizontal flip excluding directional signs to avoid label inversion, and HSV color jitter to simulate varying lighting conditions. Fig. 2, presents the training and validation of loss curves and metric curves across all approximately 150 training epochs. This figure is included to demonstrate that the model trained stably without signs of overfitting or training instability. The top panel shows the three loss components, CIoU, classification, and DFL, for both the training and validation sets, all of which converge smoothly over the training period. The bottom panel tracks precision, recall, and mAP@0.5 over epochs. Readers should observe that mAP@0.5 grows consistently from 0.28 in early epochs to 0.619 by epoch 150, while precision plateaus near 75% after epoch 100. As can be seen in figure 2, the training and validation curves confirms that the model shows strong generalization ability.

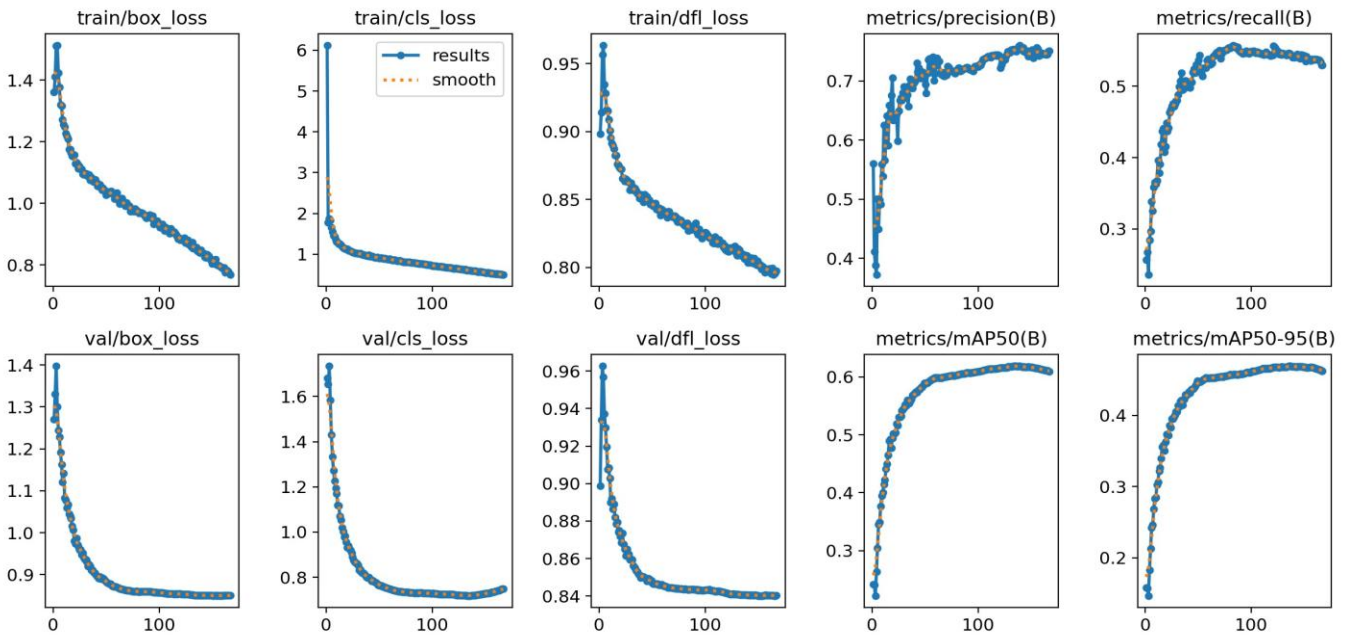


Fig. 2. Training/validation loss curves and metric curves (~150 epochs). All losses converge cleanly; $mAP@0.5$ grows from 0.28 to 0.619 with no overfitting detected.

The deployed system captures dashcam-style video frames via OpenCV, performs per-frame YOLOv8 inference renders color-coded bounding box overlays with class labels and confidence scores, and writes annotated video to disk. An adaptive day/night exposure mode computes per-frame luminance and dynamically adjusts camera exposure and gain parameters when luminance falls below a configurable threshold, significantly reducing false negatives from underexposed nighttime frames. Fig. 3, presents the overall system architecture, illustrating the complete data flow from the MTSD dataset through preprocessing and YOLOv8 training to the real-time inference pipeline and final annotated output. This diagram is included to give readers a high-level understanding of how each component of the system connects to the next. The flow begins at the left with the raw MTSD dataset, passes through the class filtering and YOLO format conversion stage, enters the YOLOv8 training pipeline, and terminates at the right with the real-time inference output deployed on the Raspberry Pi 4.

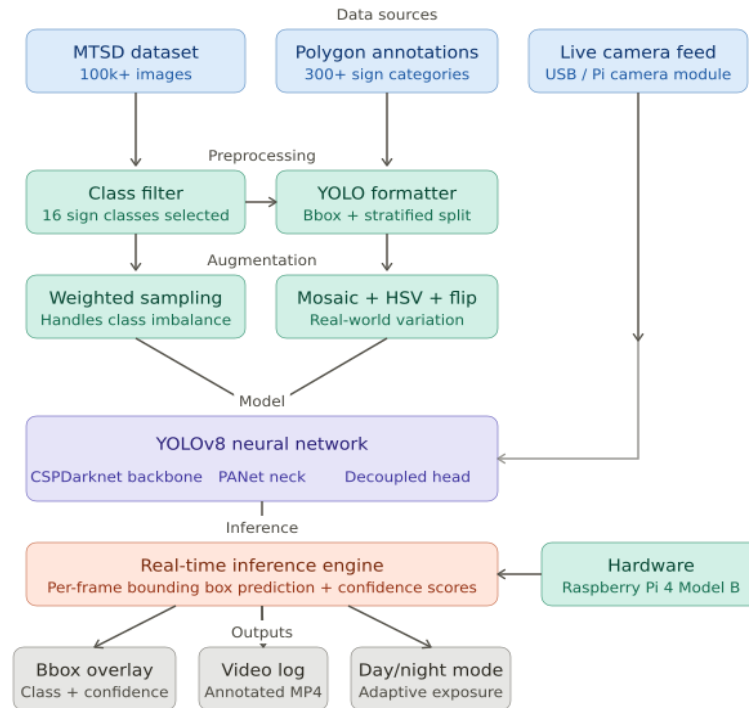


Fig. 3. Overall system architecture. data flow from MTSD dataset through preprocessing, YOLOv8 training, and real-time inference pipeline to annotated output. Hardware deployment target is a Raspberry Pi 4.

The YOLOv8 model was evaluated on the test data set of approximately 1,200 images drawn from the MTSD set. Table I presents the overall detection metrics across all 16 sign classes and serves as the primary quantitative summary of model performance. Table I is included to give readers a concise overview of how well the model performs at the system level before examining individual sign categories. The table reports five key metrics: precision, recall, mAP@0.5, F1 score, and mAP@0.5 :0.95, each measured at the confidence threshold at which it is optimized. Precision of 75.5% at a confidence threshold of 0.989 indicates that when the model makes a high-confidence detection, it is correct approximately three out of four times. A recall of 80.0% at the lowest threshold indicates that the model successfully identifies 80% of all true signs present in the test images. The mAP@0.5 of 61.9% represents the mean detection accuracy across all 16 classes at an IoU overlap threshold of 0.5, providing a balanced measure of both localization and classification quality. The F1 score of 0.62 at confidence threshold 0.312 represents the optimal operating point for balanced precision and recall in deployment. Readers should take from Table I that while the model performs well for high-frequency, visually distinctive sign classes, overall system-level performance reflects the difficulty of detecting all 16 classes equally under real-world conditions. Confidence thresholds indicate operating points at which each metric was measured.

TABLE I. OVERALL DETECTION METRICS (YOLOV8 ON MTSD TEST SET)

Metric	Value	Threshold	Epochs
Precision	75.5%	0.989	~150
Recall	80.0%	0.000	~150

mAP@0.5	61.9%	All	~150
F1 Score	0.62	0.312	~150
mAP@0.5:0.95	~45%	All	~150

All training loss components converged cleanly as shown in Fig. 2. Precision plateaued near 75% after epoch 100, while mAP@0.5 grew steadily from 0.28 to 0.619 across the full training run.

Table II below presents the per-class AP@0.5 score for all 16 sign categories and provides a detailed breakdown of model performance at the individual class level. This table is important because it reveals which sign types of the model detects reliably and which ones remain challenging, information that is directly relevant for assessing deployment safety in real-world ADAS applications. Model performance was categorized into three tiers: strong for F1 scores above 0.75, moderate for scores between 0.50 and 0.75, and weak for scores below 0.50.

TABLE II. PERFORMANCE ON MTSD TEST SET

Sign Class	AP@0.5	Performance
Stop	0.807	Strong
Curve Warning	0.804	Strong
Speed Limit	0.768	Strong
Yield	0.752	Strong
Pedestrian Crossing	0.752	Strong
Children Warning	0.661	Moderate
No Entry	0.659	Moderate
Traffic Signal Warning	0.637	Moderate
School Zone	0.624	Moderate
Merge Warning	0.599	Moderate
Construction Warning	0.548	Moderate
One Way	0.533	Moderate
No Right Turn	0.518	Moderate
Service Sign	0.469	Weak
Guide Sign	0.443	Weak
No Left Turn	0.323	Weak

The table organizes classes into three performance tiers: Strong, defined as AP@0.5 above 0.75; Moderate, defined as AP@0.5 between 0.50 and 0.75; and Weak, defined as AP@0.5 below 0.50. One should observe that the five strongest classes, stop at 0.807, curve warning at 0.804, speed limit at 0.768, yield at 0.752, and pedestrian crossing at 0.752, are all visually distinctive signs with relatively high training instance counts or highly recognizable shapes and colors. In contrast, the three weakest classes, service sign at 0.469, guide sign at 0.443, and no left turn at 0.323, are affected by limited training data and high visual similarity to other sign types. The takeaway for readers is that class-level performance varies significantly across the 16 categories, and that the model should be used with awareness of these per-class limitations, particularly

for safety-critical applications where missed detections of weak-performing sign types could have serious consequences.

Classes with the highest training instance consistently achieve AP@0.5 above 0.75. Curve warning achieves an AP of 0.804 despite moderate training data of 1,241 instances, due to its visually distinctive diamond shape and yellow coloration, which the model can reliably identify even under variable lighting. The weakest classes, service sign, guide sign, and no left turn, suffer from limited training data and high background false negative rates of 57 to 61 percent, as shown in Figures 5 and 6.

Fig. 4, presents the Precision-Recall curve for all 16 sign classes and is included to show how each class performs across the full range of confidence thresholds, not just at a single operating point. Each colored line in the figure represents one sign class, and the thick blue line represents the mean across all classes. The area under each curve corresponds to the AP@0.5 score for that class reported in Table II. A curve that remains high across a wide recall range indicates a class that the model detects reliably without sacrificing precision. Readers should note that stop, curve warning, and speed limit maintain high precision across a wide recall range, indicating strong and consistent detection performance for these classes. Classes with curves that drop steeply at low recall values, such as no left turn and guide sign, indicate classes where the model struggles to find true positive detections without also introducing a high number of false positives. The overall mAP@0.5 of 0.619 is represented by the area under the mean blue curve.

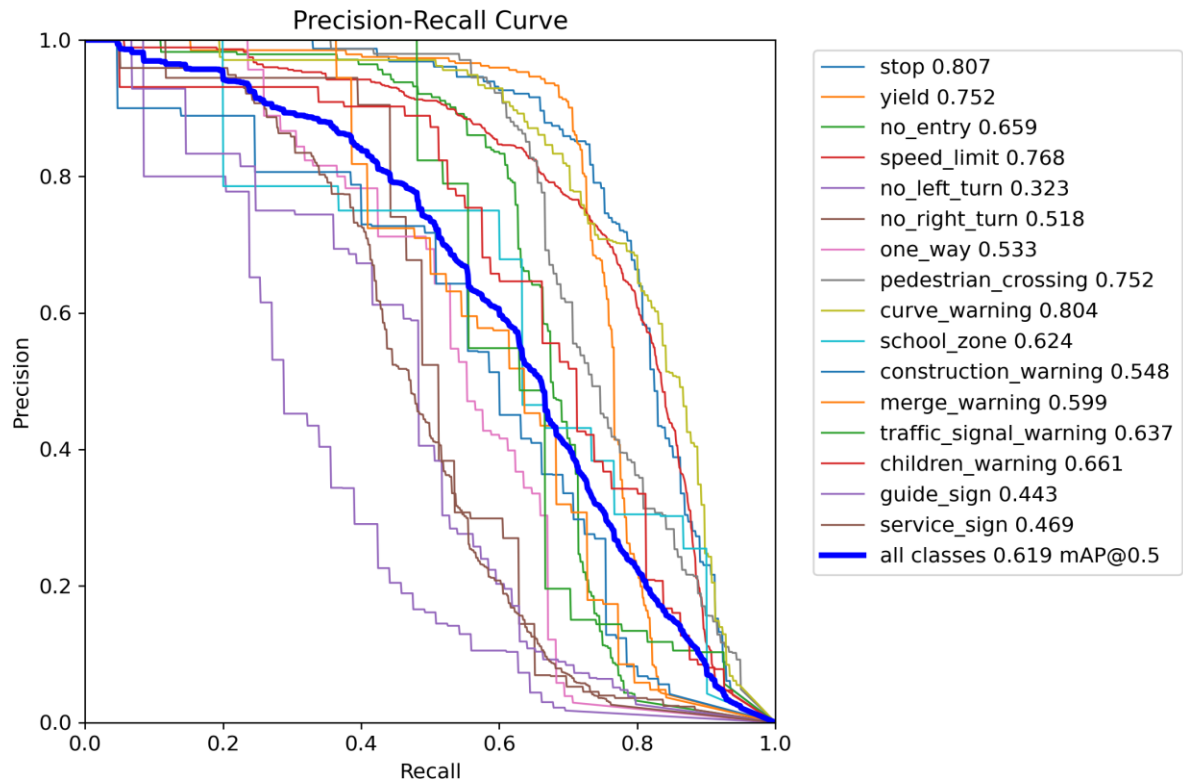


Fig. 4. Precision-Recall curve (16 classes). $mAP@0.5 = 0.619$. The thick blue line represents the MEAN across all classes.

Stop at AP 0.807, curve warning at AP 0.804, and speed limit at AP 0.768 to maintain high precision across a wide recall range. The construction and merge warning classes show elevated confusion with children's

warning due to visual similarity between yellow diamond warning signs that differ only in their interior pictograms, as confirmed by Fig. 5.

Fig. 5, presents the normalized confusion matrix. This matrix shows the proportion of true signs from each class that were correctly classified versus misclassified as other classes or missed entirely as background. This figure is included to reveal patterns of confusion between visually similar sign classes and to identify which classes suffer from the highest rates of missed detection. The diagonal entries represent correct classifications, and off-diagonal entries represent misclassifications. One should pay attention to the background column on the right side of the matrix, which represents signs that the model failed to detect at all. Classes such as no left turn, guide sign, and service sign show the highest background false negative rates of 57 to 61 percent, confirming that these classes are the most difficult for the model to detect. The confusion between construction warning, merge warning, and children's warning is also visible in the off-diagonal entries, reflecting the challenge of distinguishing yellow diamond signs that share the same shape and color but differ only in their interior symbol.

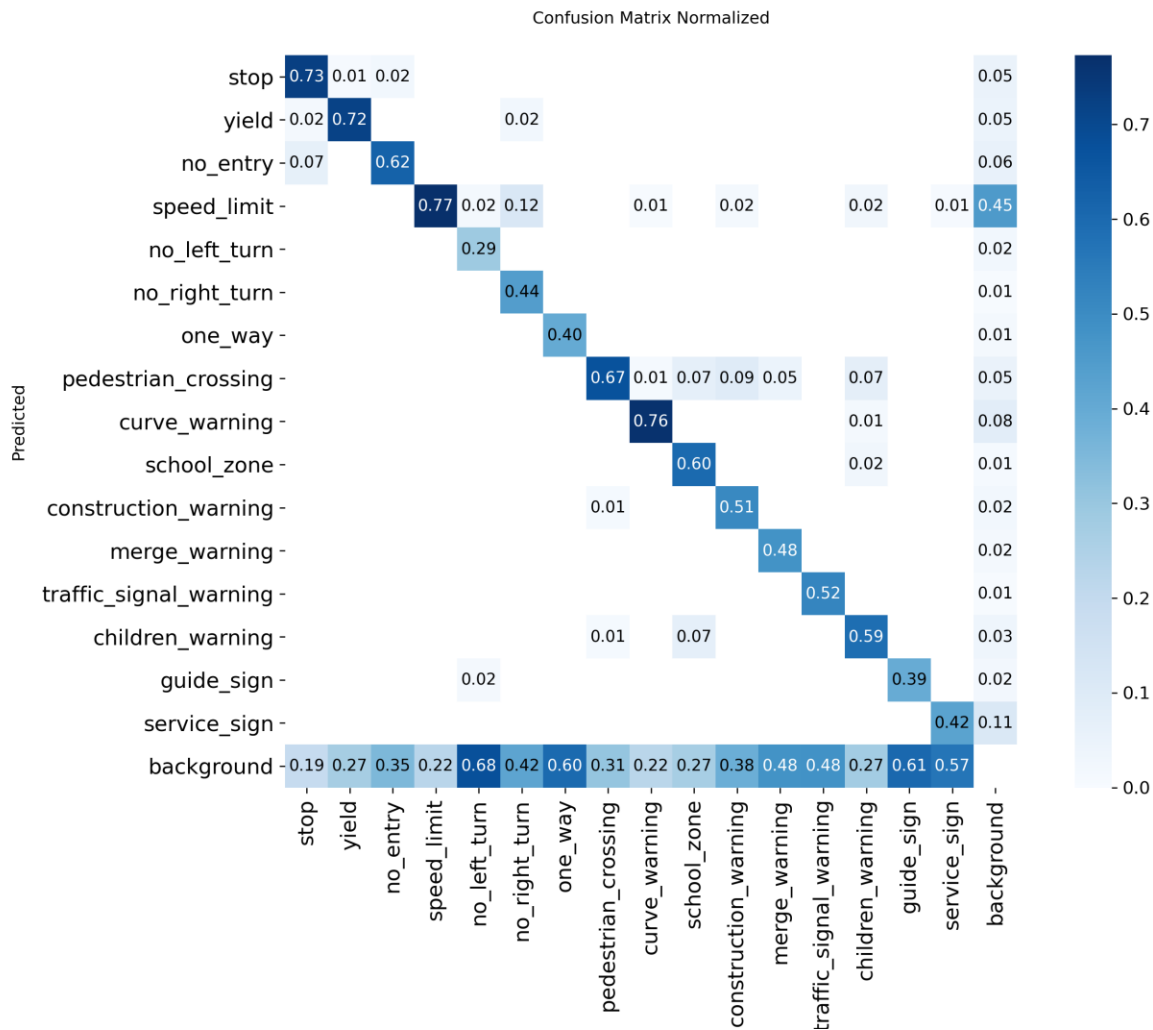


Fig. 5. Normalized confusion matrix. Strong diagonal for stop (0.73), speed limit (0.77), and curve warning (0.76). The background column shows the highest false negative rates.

Fig. 6, presents the raw count confusion matrix, which shows the same classification results as Figure 5 but expressed in absolute counts rather than proportions. This figure is included alongside the normalized matrix to give an idea of the scale of the test set and the absolute number of true positive, false positive, and false negative detections per class. Speed limit accounts for the highest absolute true positive count of 729, reflecting its dominance in the training data drawn from MTSD (Ertler et al., 2020). The background row at the bottom of the matrix shows the total number of false negatives per class, reinforcing the pattern observed in Figure 5 that low-frequency classes have the highest absolute missed detection counts.

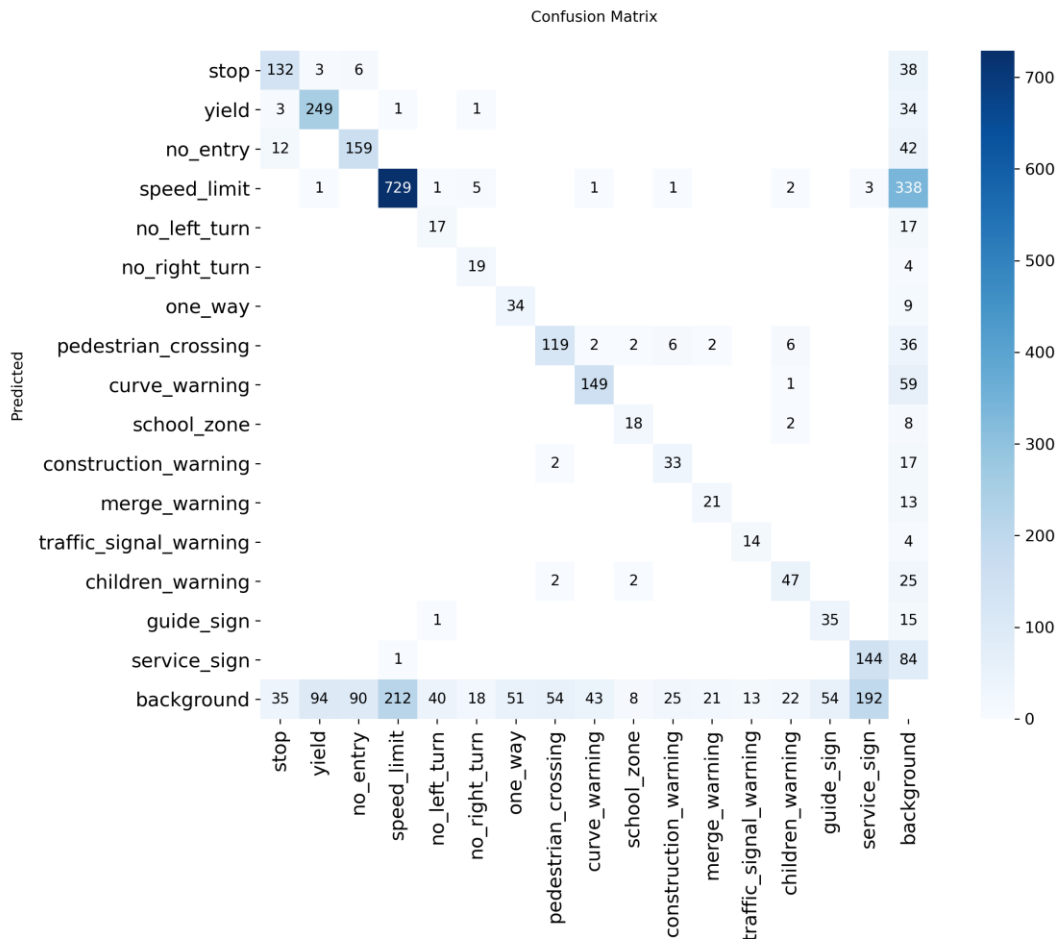


Fig. 6. Raw count confusion matrix. Speed limit accounts for the highest absolute true positive count (729). The background row reflects false negative totals per class.

Fig. 7 presents the F1-Confidence curve for all 16 classes and is included to help identify the optimal confidence threshold for deployment. The F1 score is a performance metric that combines precision and recall into one value. So, plotting it against confidence threshold shows how the trade-off between these two metrics changes as the detection threshold is adjusted. Each colored line represents one class, and the thick blue line represents the mean across all classes. Readers should observe that the peak mean F1 score of 0.62 occurs at a confidence threshold of 0.312, which is therefore the recommended operating threshold for balanced deployment. At thresholds below 0.312, recall increases, but precision decreases as more false positives are accepted. At thresholds above 0.312, precision increases, but recall decreases as more true signs are missed. For safety-critical ADAS applications where missing a sign is more dangerous than a

false alarm, a lower threshold of 0.25 to 0.30 is advisable, accepting a minor increase in false positives in exchange for higher recall.

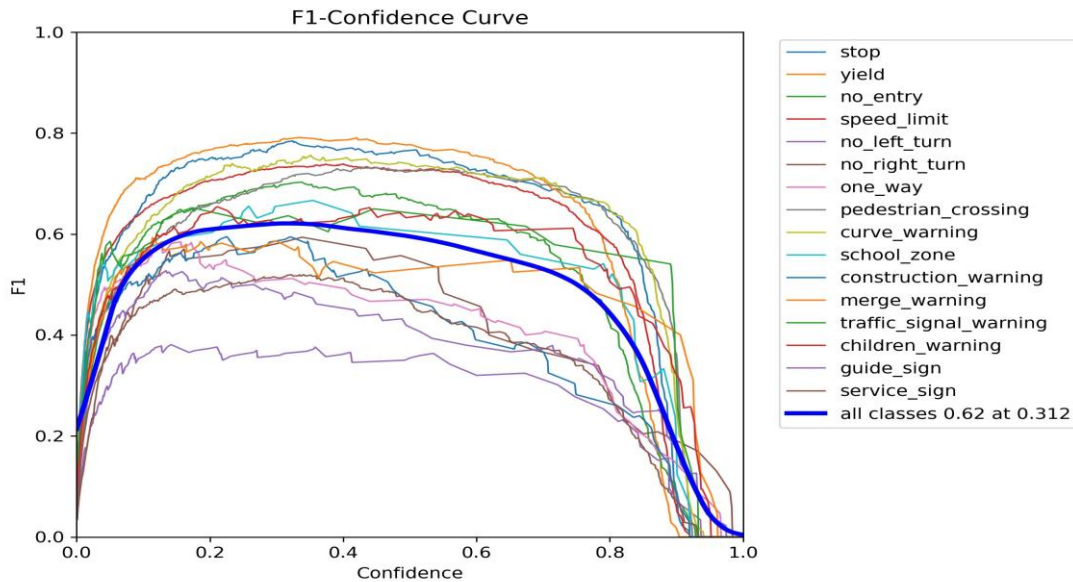


Fig. 7. F1-Confidence curve. Peak F1 = 0.62 at confidence threshold 0.312 across all 16 classes.

Raspberry Pi-Based Edge Implementation

Fig. 8 below, provides the diagram of the embedded inference system that we implemented on the Raspberry Pi 4 at the hardware and software level. The diagram shows the complete processing flow beginning with the camera input from the Raspberry Pi camera module, passing through the OpenCV frame capture and preprocessing stage where frames are resized to 320×320 pixels, and then executing through the sequential stages of the YOLOv8 inference pipeline for feature extraction, multi-scale aggregation, and the decoupled detection head for bounding box regression and class prediction.

The real-time inference system was physically deployed on a Raspberry Pi 4 paired with a Pi camera module, casing printed with a 3D printer, as shown in Fig. 9. The camera was connected through the CSI ribbon interface and used for live image acquisition. This hardware setup served as the edge platform for executing the traffic sign detection program, allowing the captured frames to be processed directly on device and in real time through the YOLOv8 inference pipeline. The compact and power efficient nature of the Raspberry Pi 4 and camera module makes the system practical for embedded intelligent transportation and ADAS-oriented applications. Fig. 10 shows representative detection of outputs from the deployed system under nighttime and daytime driving conditions. Stop signs were detected at confidence levels of 0.85–0.89 in both conditions. Speed limit signs were detected with confidence of 0.85 in daytime, consistent with Table II. On the Raspberry Pi 4 with reduced resolution (320×320), throughput was 4–5 FPS, adequate for low-speed scenarios. Fig. 11 presents four additional detection outputs across varying sign types and environmental conditions, demonstrating the breadth of the system’s real-world detection capability.

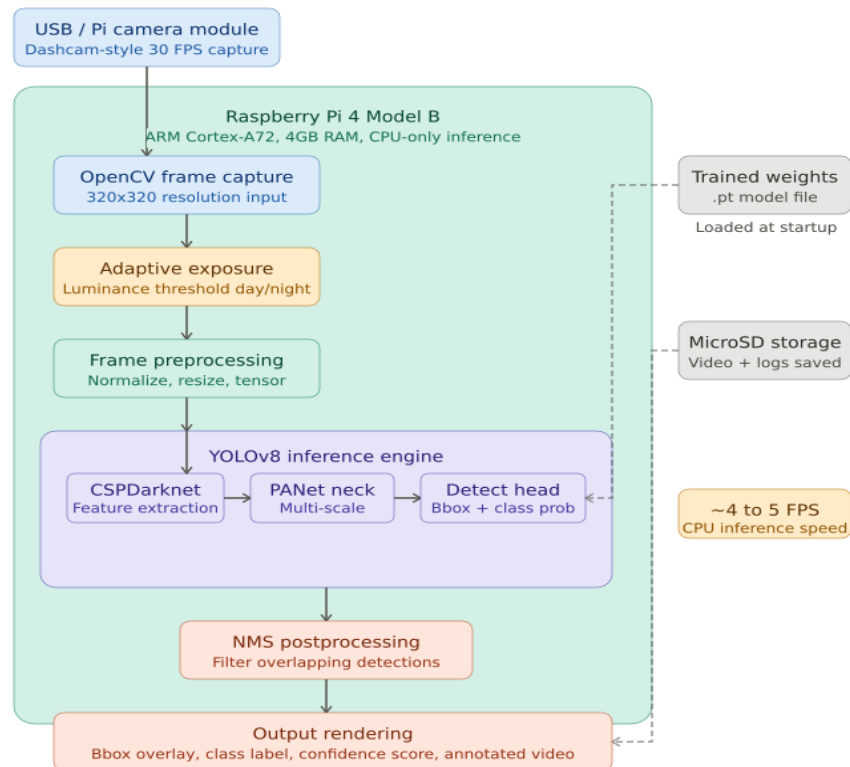


Fig. 8. Raspberry Pi 4 embedded detection system. Diagram shows camera input, OpenCV frame capture, preprocessing, YOLOv8 inference pipeline, neural network layers, and output components including annotated video and adaptive exposure control.



Fig. 9. Shows Raspberry Pi 4 and Pi camera module used to run the real-time traffic sign detection system.

The implemented Raspberry Pi and Pi camera system was mounted on the dashboard of the car to capture real-time road images during driving. The system was tested under both daylight and nighttime conditions to evaluate its performance in different lighting environments. Daytime testing was used to assess detection accuracy under clear visibility, while nighttime testing helped examine the system's robustness under low-light conditions. These experiments provided a practical evaluation of the system's ability to operate in real-world driving scenarios.



Fig. 10. Real-world detection outputs. Top: nighttime stop sign (conf: 0.89). Bottom: daytime pedestrian crossing (conf: 0.72). System running at ~4.8 FPS on Raspberry Pi 4.



Fig. 11. Additional real-world detection outputs. Top Left: Stop Sign at night (conf: 0.87). Top Right: Speed limit 60, highway daytime (conf: 0.45). Bottom Left: curve warning, nighttime (conf: 0.21, due to foggy weather and motion blur). Bottom Right: curve warning and speed limit 15 (conf: 0.31 and 0.19, due to sign distance, motion blur, and foggy weather).

Fig. 12 presents the Precision-Confidence and Recall-Confidence curves independently. At high confidence thresholds (conf = 0.989), all classes reach precision of 1.0. At the lowest threshold, all classes achieve recall = 0.80. These curves guide threshold selection for deployment scenarios prioritizing either precision (false alarm reduction) or recall (missed detection reduction).

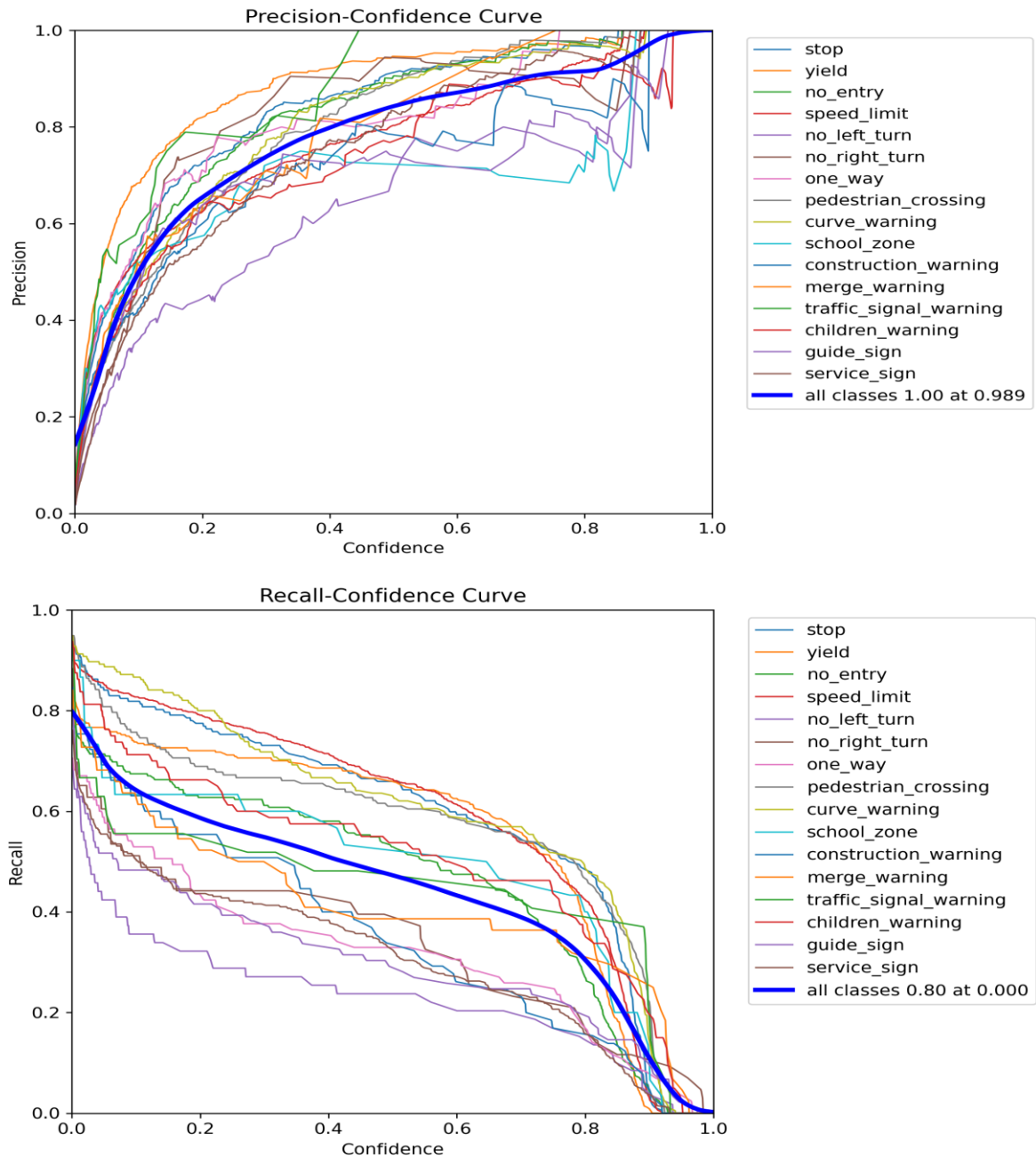


Fig. 12. Precision-Confidence (top) and Recall-Confidence (bottom) curves for all 16 classes. The thick blue line is the mean across classes.

The model was trained exclusively on MTSD annotations, potentially limiting generalization to regional sign designs not represented in the dataset. While the United States follows the MUTCD as the national

standard for traffic sign design, significant variation exists in practice. Individual states, counties, and municipalities frequently deploy non-standard signs that differ in color, shape, font, iconography, and reflectivity from federally recommended designs. Older signs and temporary construction signage add further visual diversity. Class imbalance for no left turn, merge warning, and guide sign classes could not be fully remediated by weighted sampling and augmentation alone, suggesting a need for additional data collection targeting regional and non-standard U.S. sign variants.

Edge-based traffic sign detection systems must consider real-world deployment constraints such as reliability, latency, and robustness under varying conditions. While the proposed system demonstrates feasibility on low-cost embedded hardware, it is intended as a proof-of-concept rather than a safety-certified ADAS solution. Broader socio-technical issues, including accountability and risk in safety-critical environments, highlight the need for further validation before real-world deployment.

Recent object detection research includes newer YOLO versions and transformer-based models such as RT-DETR, which improve accuracy and generalization in complex scenes. These approaches provide stronger feature representations compared to earlier CNN-based methods. Future work may explore integrating these recent models to further improve performance on embedded edge platforms.

As a future work, we plan to conduct a comparative evaluation of the proposed YOLOv8-based system with other state-of-the-art object detection models, such as YOLOv7, SSD, Faster R-CNN, and transformer-based approaches. This comparison will assess detection accuracy, inference speed, and computational efficiency on embedded edge platforms to better understand the trade-offs among different architectures.

Conclusion

This paper presents a traffic sign detection system built on YOLOv8 and trained on a filtered subset of the MTSD dataset encompassing 16 traffic signs. The system achieved a mAP@0.5 of 0.619 and a peak F1 score of 0.62 at a confidence threshold of 0.312, with strong per-class performance for stop (AP = 0.807), curve warning (AP = 0.804), and speed limit (AP = 0.768). Real-world deployment on a Raspberry Pi 4 confirmed practical inference speeds suitable for dashcam-style applications, with the adaptive day/night exposure mode significantly improving nighttime detection reliability. Class imbalance remains the primary limitation. Future work should target additional data collection for underrepresented classes, integration of temporal tracking (e.g., SORT or ByteTrack) to reduce per-frame flickering, and expansion to regional and non-standard U.S. sign variants. Overall, the results demonstrate that a carefully curated dataset and a lightweight detection architecture can yield a robust, deployable traffic sign recognition system under real-world driving conditions.

References

- Arcos-Garcia, A., Alvarez-Garcia, J. A., & Soria-Morillo, L. M. (2018). Deep neural network for traffic sign recognition systems. *Neural Networks*, 99, 158–165.
- CDC. (2024, May 17). *Global Road Safety*. Transportation Safety. <https://www.cdc.gov/transportation-safety/global/index.html>

- Ertler, C., Mislej, J., Olber, T., Porber, L., Šegvić, S., Pečenović, Z., Kuang, D., & Čaklović, L. (2020). The Mapillary Traffic Sign Dataset for Detection and Classification on a Global Scale. In *Proc. ECCV*.
- Jocher, G., Chaurasia, A., & Qiu, J. (2023). *Ultralytics YOLOv8*. GitHub.
<https://github.com/ultralytics/ultralytics>
- Li, X., Wang, W., Wu, L., Chen, S., Hu, X., Li, J., Tang, J., & Yang, J. (2020). Generalized Focal Loss: Learning Qualified and Distributed Bounding Boxes for Dense Object Detection. In *Advances in NeurIPS*.
- Mogelmose, A., Trivedi, M. M., & Moeslund, T. B. (2012). Vision-based traffic sign detection and analysis for intelligent driver assistance systems. *IEEE Transactions on Intelligent Transportation Systems*, 13(4), 1484–1497.
- Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You Only Look Once: Unified, Real-Time Object Detection. In *Proc. IEEE CVPR*.
- Redmon, J., & Farhadi, A. (2018). YOLOv3: An Incremental Improvement. *arXiv preprint arXiv:1804.02767*.
- Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. In *Advances in NIPS* (Vol. 28).
- Stallkamp, J., Schlipsing, M., Salmen, J., & Igel, C. (2012). Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition. *Neural Networks*, 32, 323–332.
- World Health Organization. (2023). *Road traffic injuries*. World Health Organization.
<https://www.who.int/news-room/fact-sheets/detail/road-traffic-injuries>
- Wang, C.-Y., Bochkovskiy, A., & Liao, H.-Y. M. (2023). YOLOv7: Trainable Bag-of-Freebies Sets New State-of-the-Art for Real-Time Object Detectors. In *Proc. IEEE CVPR* (pp. 7464–7475).
- Zhang, J., Huang, M., Jin, X., & Li, X. (2017). A Real-Time Chinese Traffic Sign Detection Algorithm Based on Modified YOLOv2. *Algorithms*, 10(4), 127.
- Zhu, Z., Liang, D., Zhang, S., Huang, X., Li, B., & Hu, S. (2016). Traffic-Sign Detection and Classification in the Wild. In *Proc. IEEE CVPR* (pp. 2110–2118).