American Journal of Traffic and Transportation Engineering
2026, Vol. 11, No. 5, pp. 110–116
https://doi.org/10.11648/j.ajtte.20261105.13

Research Article

Comparative Benchmarking of YOLOv8 Variants for Traffic Sign Detection

Nnanna Ekedebe*

Department of Information Technology, Federal University of Technology, Owerri, Nigeria

Abstract

Road traffic accidents remain a leading cause of injury and death worldwide, and driver failure to correctly identify traffic signs is a significant contributing factor. Many devastating and deadly road traffic incidents are directly linked to poor sign recognition, particularly in Nigerian and developing-region road conditions, where faded signage, poor lighting, and occlusion are common conditions poorly served by existing standardized-signage-focused research. Most existing work is either classification-only on pre-cropped signs, such as NaijaTrafficNet, or detection-based but not comparatively benchmarked for resource-constrained deployment relevant to this context. This study aims to determine which YOLOv8 variant offers the best accuracy-speed-size tradeoff for real-time detection under resource-constrained conditions relevant to developing-region deployment. This study addresses this gap by comparatively benchmarking three YOLOv8 variants — nano, small, and medium on an 11-class filtered traffic sign detection dataset, comprising 800 training, 223 validation, and 79 held-out test images, evaluated on precision, recall, mAP@0.5, mAP@0.5:0.95, FPS, and model size on a held-out test set. All variants performed closely within a narrow accuracy band, with YOLOv8n matching or exceeding the larger variants in accuracy while offering the smallest footprint and highest inference speed, showing that higher model capacity does not improve generalization at this dataset scale. YOLOv8n achieved the highest test-set mAP@0.5 of 0.897 at only 3.0M parameters and 6.2MB, versus YOLOv8m's 0.886 at 25.8M parameters and 52.0MB — despite YOLOv8m ranking highest on validation, underscoring the importance of held-out test evaluation. YOLOv8n is therefore recommended as the most viable variant for real-time, resource-constrained deployment on dashcams or roadside cameras in developing-region contexts, pending real-world Nigerian and embedded-hardware validation.

Keywords

Benchmarking, Traffic Sign, Resource-Constrained, Real-time Detection, Driver Safety

1. Introduction

Over the years, there has been a move from manual driver compliance monitoring of traffic signs to more automatic sign detection using computer vision, which has proven to be a scalable alternative. The field has steadily progressed from hand-crafted feature-based methods, such as color/shape thresholding and SVM, to deep learning, YOLO-based approaches. This study sets out to evaluate YOLOv8 variants for traffic sign detection in the Nigerian road deployment context.

Road traffic accidents have continued to be one of the leading causes of injury and death worldwide, with drivers' inability to correctly identify traffic signs a big contributing factor.

*Correspondence: Nnanna Ekedebe (nnanaekedebe@gmail.com)

Received: 13 July 2026; **Accepted:** 24 July 2026; **Published:** 18 September 2026

The traditional method of traffic sign recognition relies on manually engineered visuals, which can be hindered by complex backgrounds, sensitive lighting conditions, occlusion, and weathering. These methods perform poorly in real-time, dynamic road conditions and fail to generalize well across multiple conditions [1, 2].

The increased number of vandalized, faded, or inconsistently maintained signage, combined with variable lighting and high visual clutter from advertising/billboards competing with signage in Nigeria, poses a great challenge to sign-based recognition. Existing literature largely assumes standardized signage (Europe/China), which constitutes a poor fit, making it more evident that there is a need for lightweight, real-time, deployable detection [3, 4].

Deep learning offers a solution to this issue, with its capability for fast and accurate object detection directly from road scene images rather than from pre-cropped, isolated sign classification. The system can be deployed on in-vehicle dash cams, ADAS-lite systems, or roadside monitoring cameras. YOLOv8 is particularly suited for this job because of its single-stage architecture, and its multiple variants nano, small, and medium enable an accuracy-versus-speed-versus-size tradeoff, especially in this resource-constrained deployment context [5].

Existing literature largely targets traffic sign classification of pre-cropped signs or standardized benchmarks (GTSRB, TT100K) reflecting European/Chinese conventions, rather than full road scenes, especially in Nigerian or developing-region conditions. NaijaTrafficNet developed a custom CNN for Nigerian sign classification only not detection, and not using a YOLO-based approach [3]. This study aims to address the fact that a YOLOv8 variant comparison for resource-constrained, real-time detection remains largely unexplored.

This study sets out to evaluate YOLOv8 variants nano, small, and medium on a public traffic sign dataset using identical metrics, namely mAP@0.5, mAP@0.5:0.95, precision, recall, FPS, and model size, to demonstrate the deployment viability for real-time traffic signage detection in resource-constrained Nigerian road use.

The subsequent sections follow this structure: Section 2 presents the related work, Section 3 details the methodology, Section 4 presents the results and discussion, and finally Section 5 the conclusion.

2. Related Work

Research on traffic sign recognition has been steadily developing from classical computer vision methods to deep learning, then to YOLO-family detector approaches in particular. This section reviews the most relevant prior work across all these stages of research and identifies where this study fits in.

Wali et al. developed a system using colour segmentation, shape matching, and an SVM classifier, tested on a Malaysian traffic sign dataset [1]. It performed well and showed strong accuracy under controlled, varied weather and lighting test conditions. The limitation with this work is that classical systems like this are sensitive and subject to real-world clutter, occlusion, and unconstrained scene variability, which pushes for the need to shift towards deep learning.

Two notable studies include YOLO-BS, that is a small object detection layer added to YOLOv8 and evaluated on TT100K, and also YOLOv8-CE, that is Coordinate Attention plus EIoU added to YOLOv8 and tested on Jetson Nano for embedded real-time detection [6, 7]. For the YOLO-BS, they modified YOLOv8's architecture by inserting a dedicated small-object detection layer, while YOLOv8-CE modified YOLOv8 by incorporating a Coordinate Attention mechanism and EIoU loss function. Both were built on the standard YOLOv8 base architecture. YOLO-BS raised mAP@0.5 from 81.8% to 87.3%, precision from 81.7% to 86.3%, and recall from 73.8% to 79.2% [6]. YOLOv8-CE achieved 86.1% mAP@0.5, with improvements of 1.5%, 3.5%, and 2.8% in precision, recall, and mAP@0.5 respectively over the baseline YOLOv8 [7].

The shared limitation of the two was that they modify a single YOLOv8 variant to reach the highest accuracy rather than comparatively benchmarking the variants of YOLOv8 for deployment tradeoff, and also do not address Nigerian or developing-country road condition contexts. Similarly, work on Indian road conditions highlighted that YOLO models trained on standard benchmarks underperform whenever they are deployed outside their original geographic context, further pushing the need for region-specific detection research [4].

This trend of modifying a single YOLOv8 variant to maximize accuracy, rather than benchmarking variant-size tradeoffs, extends across several other recent studies. Xing et al. proposed FEBG-YOLOv8s, integrating FasterNet, an efficient multi-scale attention mechanism, and a bidirectional feature pyramid network into YOLOv8s, achieving improved accuracy alongside a smaller parameter count on the TT100K and CCTSDB benchmarks [8]. Similarly, Ma et al. and Wang et al. each proposed distinct YOLOv8-based architectural modifications targeting improved small-sign detection accuracy [9, 10]. Ji et al. proposed a YOLOv8n-based model incorporating attention and dynamic convolution modules specifically to improve detection under adverse weather conditions [11], while Du et al. proposed TSD-YOLO, a YOLOv8-based small-sign detector evaluated on both TT100K and CCTSDB [12]. As with YOLO-BS and YOLOv8-CE, none of these studies benchmark YOLOv8's variant sizes against one another for deployment tradeoff, reinforcing the gap this study addresses. Separately, a comparative study using a Roboflow-derived dataset structurally similar to the one used here found that YOLOv8 outperformed YOLOv5 and YOLOv7 in traffic sign detection accuracy, though the comparison did not extend to YOLOv8's own variants or to resource-constrained deployment tradeoffs [13].

NaijaTrafficNet (AJERD, Nov 2025) worked on a custom CNN classification of pre-cropped sign images; the upside was that it was centred on a Nigerian context [3]. The limitation with this work was that it did not account for full-scene detection. There was also a broader absence of an open Nigerian traffic sign image dataset in the literature.

A lightweight object detection survey covering edge-device backbone and tradeoff review [14], and the YOLOv8–YOLOv11 benchmarking study which included Nano through XLarge variants tested on an embedded 1:10 scale autonomous vehicle [5], both confirm that variant-size tradeoffs in terms of speed, model size, and accuracy are central to edge-deployment design decisions. The YOLOv8–YOLOv11 benchmarking study is a direct methodological precedent for this study, establishing that comparing YOLOv8 Nano, Small, and Medium variants specifically, rather than evaluating just one, is a valid and necessary approach for deployment-aware research [5]. Also, lighting-condition variability has been separately benchmarked using YOLOv8 for day and night detection performance further underscoring lighting as a distinct, quantifiable failure mode relevant to this study's context [2].

Existing studies in this field are mainly classification-only, and while some are detection-based, they are not within a Nigerian context or lack a resource-constrained comparative benchmark. That is where this study comes in, with its comparative YOLOv8n/s/m benchmarking for full-scene detection, framed around resource-constrained Nigerian deployment.

3. Methodology

The dataset used for this study is "traffic-sign-detection-yolov8" by huy (CC BY 4.0), a publicly available dataset obtained from Roboflow Universe [15]. The dataset's original classes numbered 17, with 6 dropped for insufficient instances. The dropped 6 were bend right, no right turn, no waiting, t_hump, turn left, and turn right. The final classes numbered 11: hump, no entry, no left turn, no overtaking, no stopping, no u turn, parking, roadwork, roundabout, speed limit 40, and stop. The training split observed was 800 train, 223 validation, and 79 test images, with 79–103 instances per class in training, except for "no left turn," which had only 17 training instances.

The dataset was filtered by removing the dropped class labels from the annotation files, with the remaining IDs remapped to contiguous 0–10. Images left with zero valid boxes after filtering were then excluded entirely, and a new data.yaml was generated to reflect the 11-class schema. It is important to note that the dataset is detection-only; therefore, a handful of leftover segmentation polygons in the raw data were discarded.

Three YOLOv8 variants YOLOv8n, YOLOv8s, and YOLOv8m, were compared, each chosen to span the accuracy–speed–size tradeoff space relevant to embedded and resource-constrained deployment. The models were pretrained on COCO and fine-tuned on the 11-class traffic sign dataset. The parameter count and GFLOPs per variant were: YOLOv8n with 3.0M parameters, 8.1 GFLOPs; YOLOv8s with 11.1M parameters, 28.5 GFLOPs; YOLOv8m with 25.8M parameters, 78.7 GFLOPs.

The framework used was Ultralytics YOLOv8 (v8.4.83) with PyTorch 2.11, on a single Tesla T4 GPU on Google Colab. Images were resized to 640×640, with a batch size of 16, epochs up to 50, and early stopping with a patience of 15. The optimizer used was AdamW, with auto-selected lr0 ≈ 0.000667 and momentum 0.9 (Ultralytics optimizer=auto). The augmentation pipeline used was Ultralytics' default mosaic (disabled in the final 10 epochs), HSV jitter, horizontal flip (p=0.5), translate/scale, plus light Albumentations (Blur, MedianBlur, ToGray, CLAHE, each p=0.01). The seed was fixed at 42 for reproducibility across all three runs.

The metrics used for evaluation were Precision, Recall, mAP@0.5, and mAP@0.5:0.95, which are standard COCO-style detection metrics. It is important to note that validation-split metrics were used during training for checkpoint selection (best.pt) and early stopping; final reported metrics were computed on the held-out test set the 79 images never seen during training or model selection, in order to avoid validation-set bias. Inference speed was measured on a fixed sample of 50 test images, single-image inference, on the same GPU (T4) for fair variant comparison. It is important to note that this is GPU benchmarking only, not embedded-hardware benchmarking; real edge-device latency, such as on a Raspberry Pi or Jetson Nano, is left to future work.

4. Results and Discussion

4.1. Overall Comparative Results

Table 1 presents the summary of all three variants nano, small, and medium with their precision, recall, mAP@0.5, mAP@0.5:0.95, FPS, and model size.

Table 1. *Presents the Comparative Performance of YOLOv8n, YOLOv8s, and YOLOv8m on the Held-Out Test Set.*

Model	Parameters	Size	Precision	Recall	mAP@0.5	mAP@0.5:0.95	FPS (ms)
YOLOv8n	3.0M	6.2 MB	0.855	0.884	0.897	0.802	32.5
YOLOv8s	11.1M	22.5 MB	0.87	0.895	0.895	0.802	29.63
YOLOv8m	25.8M	52.0 MB	0.848	0.868	0.886	0.784	30.81

It was observed overall that the three variants performed within a narrow band, with mAP50 ranging only from 0.886 to 0.897, and mAP50-95 essentially tied at 0.78–0.80, where small and nano were identical at 0.802 and medium was slightly lower at 0.784. It was also observed that precision and recall traded off slightly differently per model, but no variant dominated across all metrics.

4.2. Accuracy vs. Model Size

The core tradeoff findings are as follows: YOLOv8n achieved the highest mAP50 of 0.897 despite having only 3.0M parameters, which is 8.5× fewer than YOLOv8m's 25.8M parameters. YOLOv8m, being the most computationally expensive and largest variant at 78.7 GFLOPs versus nano's 8.1, did not yield a corresponding accuracy gain on the test set. In fact, it was the weakest of the three on test data, with mAP50 of 0.886 and mAP50-95 of 0.784. This all points to the fact that, although the dataset had 800 training images, additional model capacity does not directly translate to better generalization, most likely due to limited training data relative to model capacity, and possible overfitting toward validation-distribution characteristics during checkpoint selection.

4.3. The Validation vs. Test Discrepancy

It was noticed that during training, YOLOv8m appeared to be the clear best performer among all variants on the validation split, with mAP50 of 0.939 followed by YOLOv8s at 0.924, while YOLOv8n appeared to be the weakest on validation, with mAP50 of 0.900. Surprisingly, on the held-out test set, the ranking was reversed, as nano (0.897) and small (0.895) outperformed medium (0.886). This is most likely because validation-set metrics drove checkpoint selection (best.pt) and early stopping, so models may have been selected for characteristics specific to the validation distribution rather than true generalization. This is important to note as a caution against relying on validation metrics alone for deployment decisions, which is why this study evaluated on the test set as the "true" comparison. Figure 1 clearly shows this ranking reversal between the validation and test splits.

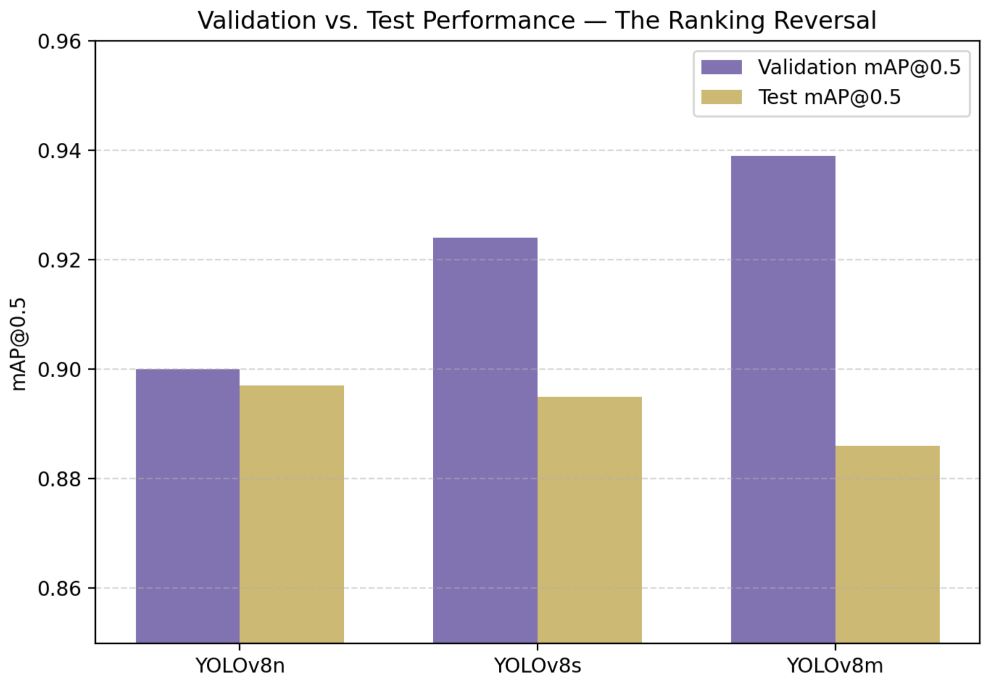

Figure 1. *Presents the Validation vs. Test performance.*

4.4. Deployment Implications

As it relates to deployment implications, YOLOv8n delivered equal-or-better accuracy at approximately 12% of the file size of YOLOv8m *6.2MB for nano versus 52.0MB for medium* and also the highest frames per second, at 32.50 versus 30.81. This is paramount for real-time, resource-constrained deployment on dash cams, roadside cameras, and similar systems in a Nigerian context. The smaller variant did not just perform well — it is the better choice, since it offers no accuracy penalty while minimizing storage and computational footprint, pointing to the fact that bigger is not always better for this task and dataset scale. Figure 2 shows this tradeoff, positioning YOLOv8n in the favorable top-left quadrant of high accuracy and low file size.

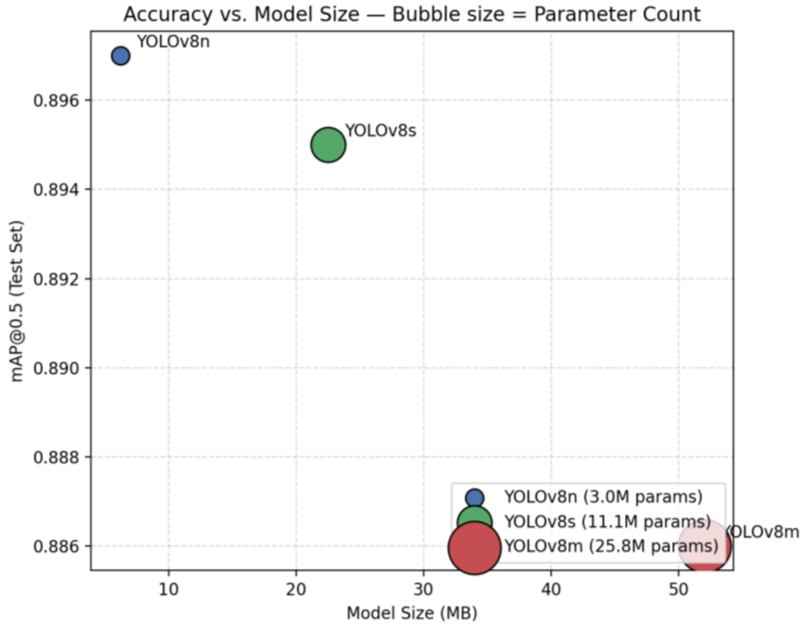

Figure 2. *Presents the Accuracy vs. Model Size Across YOLOv8 Variants.*

4.5. Frame Per-Second Behavior

It was observed that frames per second did not increase monotonically with model size; rather, YOLOv8m was slightly faster than YOLOv8s, with FPS of 30.81 versus 29.63. This is most likely due to GPU-level noise and variance from single-image inference batching rather than a genuine architectural trend. This does not indicate that the medium variant is inherently faster than small it only points to expected variance at small sample sizes (50 images) and single-GPU benchmarking. It is also important to reiterate that GPU FPS is not directly transferable to edge hardware, as the latency of such devices depends heavily on whether they are ARM-based or otherwise low-power, which is why embedded testing is left to future work.

4.6. Per-Class Performance Analysis

In terms of per-class analysis, it was observed that most classes performed strongly and consistently across all three variants, with mAP50 often at 0.9+, including no overtaking, roadwork, roundabout, speed limit 40, and stop. "No left turn" was the clear outlier, with mAP50 ranging from 0.38 for medium to 0.68 for nano across variants, making it by far the weakest class in every model. This is attributable to severe class imbalance, with only 17 training instances and 4 test instances, versus 79–103 training instances for the other classes. It is worth noting that although the small number of instances limits statistical confidence, YOLOv8n's smaller parameter count may give it a higher capacity to generalize with less overfitting risk, or a reduced tendency to simply memorize majority classes a pattern further supported by its comparatively stronger performance on this weak class relative to the medium variant.

4.7. Discussion and Limitations

With these findings, this study concludes that YOLOv8n is the best choice for the proposed deployment context and recommends it accordingly, while acknowledging that the dataset's small size and single-domain, non-Nigerian origin are factors that should be validated with real-world Nigerian road imagery in future work.

These findings depart from prior YOLOv8-based traffic sign work in a notable way. YOLO-BS and YOLOv8-CE both pursued higher accuracy by modifying YOLOv8's architecture and adding a small-object detection layer and Coordinate Attention with EIoU loss, respectively and reported gains in mAP@0.5 of roughly 5–6 percentage points and 1.5 percentage points over their baselines [6, 7]. This study's results suggest that, at least at this dataset scale, the variant-selection decision may matter more than architectural modification: YOLOv8n's mAP@0.5 of 0.897 was reached without any architectural changes, simply by using the smallest baseline variant, and outperformed the larger, unmodified YOLOv8m (0.886) on held-out test data. This raises the question of whether architectural additions like YOLO-BS or YOLOv8-CE would yield similar gains if applied to the nano variant rather than larger baselines, a question this study does not resolve but flags as a natural extension.

This study's validation-to-test ranking reversal (Section 4.3) also has a direct parallel in the Indian road-conditions study, which found that YOLO models trained and validated on standard benchmarks underperformed once deployed outside their original geographic and distributional context [4]. While the mechanism differs, that study concerned cross-domain generalization, whereas this study's reversal occurred within a single domain, driven instead by validation-based checkpoint selection, both findings converge on the same caution: validation-set performance is an unreliable proxy for real-world or held-out deployment accuracy, reinforcing why this study explicitly prioritized test-set evaluation over validation metrics.

In terms of resource-constrained deployment, the direct methodological precedent for this study, the YOLOv8–YOLOv11 benchmarking work on embedded 1:10 scale autonomous vehicles, similarly found that smaller variants offered a favorable accuracy-to-footprint tradeoff on constrained hardware [5]. This study's results are consistent with that pattern, though it extends the comparison to a Nigerian-relevant deployment context and a full accuracy/speed/size metric suite rather than embedded latency alone. Similarly, the lighting-condition benchmarking study using YOLOv8 for day/night detection isolated lighting as a distinct, quantifiable failure mode [2], a factor this study's dataset does not explicitly stratify by, and which remains an open limitation given the lighting variability characteristic of Nigerian road conditions described in Section 1.

Finally, relative to NaijaTrafficNet, the only prior Nigerian-context traffic sign work, this study extends the problem from classification of pre-cropped signs to full-scene detection [3]. The two studies are not directly numerically comparable, since NaijaTrafficNet reports classification accuracy on isolated crops rather than detection metrics on full road scenes. Nonetheless, this study's results indicate that a lightweight detection approach is viable at a scale and computational footprint suitable for the same deployment context NaijaTrafficNet targeted, suggesting a possible complementary or successor pipeline: detection followed by classification, or a unified detection approach as benchmarked here.

5. Conclusion

For future work, it is recommended that a Nigeria-specific traffic sign image dataset be curated, since currently available datasets are rarely Nigeria-specific. YOLOv8n specifically should be field-tested under real Nigerian road conditions, such as faded signage, occlusion, variable lighting, and visual clutter from billboards. For deployment testing contexts, future work should target hardware such as Raspberry Pi, Jetson Nano, or similar ARM-based and low-power devices, especially since this study's FPS benchmarking was GPU-only and cannot be assumed to transfer directly to embedded hardware deployment.

For future data curation work, efforts should be made toward specifically targeting underrepresented classes such as "no left turn," through additional annotation, targeted data collection, or synthetic augmentation techniques. Beyond standalone detection benchmarking, the detector can be integrated into a full pipeline, combining detection with tracking, driver alert systems, or ADAS-lite integration.

Further work can extend this comparative benchmarking approach to newer YOLO architectures such as YOLOv9, v10, and v11, or to YOLOv8-CE/YOLO-BS style modifications, to examine whether architectural improvements close the small-dataset generalization gap observed here. Taken together, these directions chart a path from validating this study's findings in the field to scaling the underlying approach toward more capable and deployment-ready traffic sign detection systems for Nigerian and similarly resource-constrained road environments.

Abbreviations

YOLO	You Only Look Once
mAP	Mean Average Precision
FPS	Frames Per Second
CNN	Convolutional Neural Network
GPU	Graphics Processing Unit
ADAS	Advanced Driver Assistance Systems

Acknowledgments

I acknowledge the support provided by Federal University of Technology Owerri (FUTO), through the Information Communication Technology (ICT) research center.

Author Contributions

Nnanna Ekedebe: Conceptualization, Data curation, Formal Analysis, Investigation, Methodology, Project Administration, Resources, Software, Supervision, Validation, Visualization, Writing – original draft, Writing – review & editing

Conflicts of Interest

The author declares no conflicts of interest.

References

[1] S. B. Wali, M. A. Hannan, A. Hussain, and S. A. Samad, "An automatic traffic sign detection and recognition system based on colour segmentation, shape matching, and SVM," *Mathematical Problems in Engineering*, vol. 2015, Art. no. 250461, 2015, https://doi.org/10.1155/2015/250461

[2] Z. N. Aldoski and C. Koren, "Traffic Sign Detection and Quality Assessment Using YOLOv8 in Daytime and Nighttime Conditions," *Sensors*, vol. 25, no. 4, art. 1027, Feb. 2025, https://doi.org/10.3390/s25041027

[3] D. Adjeh, I. Elesemoyo, E. Ukpuho, G. Lucky, and F. Adiotomre, "NaijaTrafficNet: A custom CNN for robust sign classification under Nigerian road conditions," *ABUAD Journal of Engineering Research and Development (AJERD)*, vol. 8, no. 3, pp. 212–220, Nov. 2025, https://doi.org/10.53982/ajerd.2025.0803.19-j

[4] H. S. G. Yaamini, S. K. J, M. N, and A. Kumar G, "Lane and Traffic Sign Detection for Autonomous Vehicles: Addressing Challenges on Indian Road Conditions," *MethodsX*, vol. 14, art. 103178, Jan. 2025, https://doi.org/10.1016/j.mex.2025.103178

[5] R. Reveles-Martínez and H. A. Gamboa-Rosales, "Benchmarking YOLOv8 to YOLOv11 architectures for real-time traffic sign recognition in embedded 1:10 scale autonomous vehicles," *Technologies*, vol. 13, no. 11, art. 531, 2025, https://doi.org/10.3390/technologies13110531

[6] H. Zhang, M. Liang, and Y. Wang, "YOLO-BS: A traffic sign detection algorithm based on YOLOv8," *Scientific Reports*, vol. 15, art. 7558, 2025, https://doi.org/10.1038/s41598-025-88184-0

[7] Y. Luo, Y. Ci, H. Zhang, and L. Wu, "A YOLOv8-CE-based real-time traffic sign detection and identification method for autonomous vehicles," *Digital Transportation and Safety*, vol. 3, no. 3, pp. 82–91, 2024, https://doi.org/10.48130/dts-0024-0009

[8] C. Xing, H. Sun, and J. Yang, "A Lightweight Traffic Sign Detection Model Based on Improved YOLOv8s for Edge Deployment in Autonomous Driving Systems Under Complex Environments," *World Electric Vehicle Journal*, vol. 16, no. 8, art. 478, 2025, https://doi.org/10.3390/wevj16080478

[9] D. Ma, X. Lyu, and Q. Zhu, "Algorithm improvement for traffic sign detection based on YOLOv8," *Optoelectronics Letters*, vol. 22, pp. 92–97, 2026, https://doi.org/10.1007/s11801-026-4136-y

[10] G. Wang, P. Jin, Z. Qi, et al., "Traffic sign detection method based on improved YOLOv8," *Scientific Reports*, vol. 15, art. 19385, 2025, https://doi.org/10.1038/s41598-025-03792-0

[11] B. Ji, J. Xu, Y. Liu, P. Fan, and M. Wang, "Improved YOLOv8 for small traffic sign detection under complex environmental conditions," *Franklin Open*, vol. 8, art. 100167, 2024, https://doi.org/10.1016/j.fraope.2024.100167

[12] S. Du, W. Pan, N. Li, S. Dai, B. Xu, H. Liu, C. Xu, and X. Li, "TSD-YOLO: Small traffic sign detection based on improved YOLO v8," *IET Image Processing*, vol. 18, pp. 2884–2898, 2024, https://doi.org/10.1049/ipr2.13141

[13] Mareeswari V., Vijayan R., Shajith Nisthar, and Rahul Bala Krishnan, "Traffic Sign Detection and Recognition Using Yolo Models," *International Journal of Information Technology and Computer Science (IJITCS)*, vol. 17, no. 3, pp. 13–25, 2025, https://doi.org/10.5815/ijitcs.2025.03.02

[14] P. Mittal, "A comprehensive survey of deep learning-based lightweight object detection models for edge devices," *Artificial Intelligence Review*, vol. 57, no. 9, art. 242, 2024, https://doi.org/10.1007/s10462-024-10877-1

[15] huy, "traffic-sign-detection-yolov8 Dataset," *Roboflow Universe*, 2024. [Online]. Available: https://universe.roboflow.com/huy-guqyn/traffic-sign-detection-yolov8-awuus-xgxej