The International Journal of Computational Science, Information Technology and Control Engineering (IJCSITCE) Vol.12, N0.2, July 2025

# REAL-TIME MOBILE APP TRAFFIC SIGN RECOGNITION WITH YOLOV10 AND CNN FOR DRIVING EDUCATION

Earl Peter J. Gangoso [1], Rolando John R. Aca-ac [2], and Patrick Zane G. Sarabia [3]

College of Computer Studies, Engineering, and Architecture, La Salle University – Ozamiz, Philippines

***ABSTRACT***

*This study presents a novel Traffic Sign Recognition system for Android devices, employing Convolutional Neural Networks (CNNs) and the YOLOv10 architecture for real-time detection and classification of Philippine traffic signs. The application improves road safety by providing auditory and visual cues for traffic sign compliance, especially in the context of driving education. The system integrates TensorFlow Lite (TFLite) to optimize performance for resource-constrained mobile platforms. The study encompasses data collection, annotation, preprocessing, model development, hyperparameter tuning, model training, model evaluation, and application development. The detection model achieved high accuracy with a mean Average Precision (mAP) of 0.823 and 99.66% accuracy for the classification model. The developed mobile app also demonstrated effective real-time recognition capabilities with a recognition inference time of 200-300ms. Challenges such as low-light performance are identified, with recommendations for future enhancements in data balancing, nighttime functionality, and multilingual feedback. This scalable, cost-effective system bridges the accessibility gap in advanced driver assistance technologies, offering the potential for wider regional adaptation.*

***KEYWORDS***

*Driving Education, Computer Vision, Machine Learning, Deep Learning with Convolutional Neural Networks, Mobile App Development*

## 1. INTRODUCTION

### 1.1. Background

With the rapidly urbanizing society, road network complexity and density become higher, increasing risks and challenges in road safety management. Traffic signs then help communicate essential warnings, prohibitions, and regulations. When properly acknowledged and observed, these signs help maintain road safety and orderly, systematic traffic flow. Yet the ability to reliably recognize and interpret these signs remains a challenge, especially for new drivers.[1] Existing advanced driver assistance systems (ADAS) often incorporate traffic sign recognition functionality in high-end vehicles, but these remain prohibitively expensive for many motorists. As a result, there is a growing demand for accessible, lower-cost solutions that run on widely available devices, such as smartphones.[2]

Studies have shown that computer vision models built on deep learning architectures outperform conventional machine learning methods in extracting meaningful features from image data.[3][4] Past research applied convolutional neural networks (CNNs) and single-shot object detectors

(e.g., YOLO variants) to identify traffic signs with promising results, particularly in daylight conditions.[5] Despite these advances, practical deployments often require robust model optimization, especially for hardware with limited resources. Researchers have explored various techniques for compressing or quantizing CNNs and object detection models, making them feasible for on-device computation without relying on powerful external servers.[2][6]

In the Philippines, there is a lack of studies focused on training traffic sign recognition models specifically for local signage designs. This discrepancy can degrade recognition accuracy when the shapes, colors, or text on signs deviate from internationally common styles. Additionally, local drivers may face unique road conditions such as partial occlusions by parked vehicles or strong sunlight glare that further complicate effective sign detection.

### 1.2. Research Objectives

Hence, the study's objective was to develop an efficient real-time traffic sign recognition (TSR) on Android devices, customized for Philippine traffic signs mandated by the Land Transportation Office (LTO) for use in driving education for both driving instructors and students with the help of deep learning techniques.

The mobile app then provided both visual and auditory cues for recognized signs, delivering an assistance tool that remains accessible to ordinary drivers and does not demand specialized in-vehicle systems.

By harnessing Convolutional Neural Networks (CNNs) with TensorFlow and the YOLOv10 architecture, this research sought to contribute to an adaptable and effective solution that can be updated or transferred to other contexts with minimal overhead. The proposed solution is designed to be practical, scalable, and cost-effective for driving education. Significantly, this study aspires to efficiently and effectively contribute to road safety efforts in the Philippines.

## 2. METHODS

### 2.1. Data Collection

A combined total of 48,316 images (20,000 for detection and 28,316 for classification) was gathered from public datasets from Kaggle and Roboflow Universe, and through manual data collection through smartphone-captured images, plus virtual captures via Google Maps Street Views. The dataset was compiled to reflect a diverse range of lighting conditions, distances, and angles, focusing on 15 traffic sign classes mandated by the Philippine Land Transportation Office (LTO), namely, Stop, No Entry, Curve Left, Curve Right, Speed Limit (30km/h), Speed Limit (50km/h), Speed Limit (60km/h), Speed Limit (80km/h), "Slippery When Wet, Traffic Lights Ahead, Keep Right, Keep Left, No Left Turn, No Right Turn, and No U-Turn.

### 2.2. Data Annotation

Two distinct datasets were prepared. For detection, bounding boxes were drawn around traffic signs using the Roboflow platform. These carefully verified annotations ensured accurate sign localization for the YOLOv10 model. For classification, each cropped sign image was stored in a folder corresponding to its label, creating a structured set of subfolders for all traffic sign classes. This approach allowed the classification model to map each input image to one of the 15 traffic sign types.

## 2.3. Data Preprocessing

Preprocessing was performed to optimize input consistency and reduce computational overhead. The detection dataset images were resized to 640×640 pixels, with transformations like grayscale conversion, blur, and mosaic augmentation applied. For classification, images were cropped more tightly around the sign region and then resized to 50×50 pixels. For normalization, each image's pixel values were scaled to a range of between 0 and 1. After normalization, the training set is shuffled to ensure random distribution and further split into 80% for training and 20% for validation.

## 2.4. Model Development

Two models were built to handle complementary tasks. The detection model, based on YOLOv10, identified the positions of traffic signs within an image by predicting bounding boxes in real time. A separate CNN model, implemented in TensorFlow, classified each detected sign into one of the 15 designated traffic sign classes. This modular design, using distinct detection and classification components, permitted targeted enhancements and updates to either model without affecting the other.

Optimal hyperparameter settings were sought to boost model accuracy and generalization. For the detection model, the Ultralytics YOLO framework handled automatic hyperparameter tuning, balancing learning rate, momentum, and augmentation thresholds. For the CNN classification model, Keras Tuner's Bayesian Optimization explored different filter sizes, dense units, dropout rates, and learning rates. Early stopping was employed, ensuring training halted when no further improvements were detected on the validation set.

Figure 1 shows a summary of the CNN model architecture's layers including the types, inputs, outputs, and number of parameters.

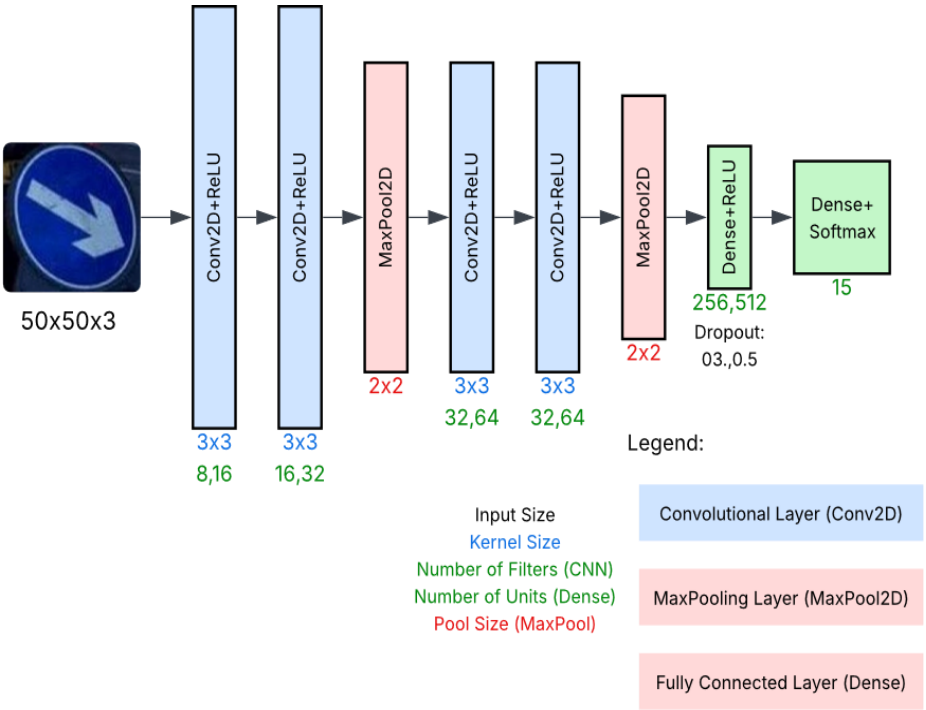


Figure 1. CNN Model Architecture

### 2.5. Model Training

Both models were trained using GPU-accelerated platforms. The YOLOv10 detection model was trained in over 300 epochs, focusing on achieving high precision and recall in bounding box predictions. The classification CNN model was trained separately in up to 30 epochs with early stopping. Batch normalization and dropout rate helped curb overfitting, while augmented data expanded the effective training set and enhanced the models' capacity to handle variations in real-world road scenarios.

### 2.6. Model Evaluation

Performance metrics were collected for both models. The detection model's effectiveness was measured through mean Average Precision (mAP) and precision-recall curves across different IoU thresholds. Meanwhile, the classification model was assessed using the following metrics: accuracy, precision, recall, and F1 scores. Additionally, a visualization through the use of a confusion matrix is used to identify any patterns of misclassifications. Once reliable, robust evaluation performance were achieved, the models were then deemed suitable for mobile deployment.

### 2.7. Mobile App Development

Both trained models were converted into TensorFlow Lite (TFLite) format for efficient on-device inference. Model quantization techniques reduce memory footprint and latency without substantially compromising accuracy. An Android application was developed using Kotlin, where the TFLite models are integrated to process the camera feed in real time. The detection component localized signs, and the classification component labeled each sign accordingly. Audio and visual feedback were then delivered to users, enabling immediate comprehension of recognized traffic signs under typical daytime conditions.

To facilitate in-vehicle use, a standard smartphone mount was used to secure the device below the driver's line of sight, adhering to the applicable hands-free driving laws in the Philippines concerning legal restrictions on mobile device usage while driving.[7][8] Once the TSR app is started and running, real-time camera feeds are processed without requiring user interaction. Audio cues announce recognized traffic signs to reduce the necessity of looking at the screen while driving, and large on-screen traffic sign icons permit quick, at-a-glance updates.

## 3. RESULTS

### 3.1. Model Development and Evaluation Results

Figure 2 illustrates the training logs that indicated progressive reductions in box loss, classification loss, and distribution-focal loss, converging around epoch 250. A slight increase in validation loss towards the last epochs possibly stemmed from the platform's mosaic augmentation closure, but overall metrics remained stable.

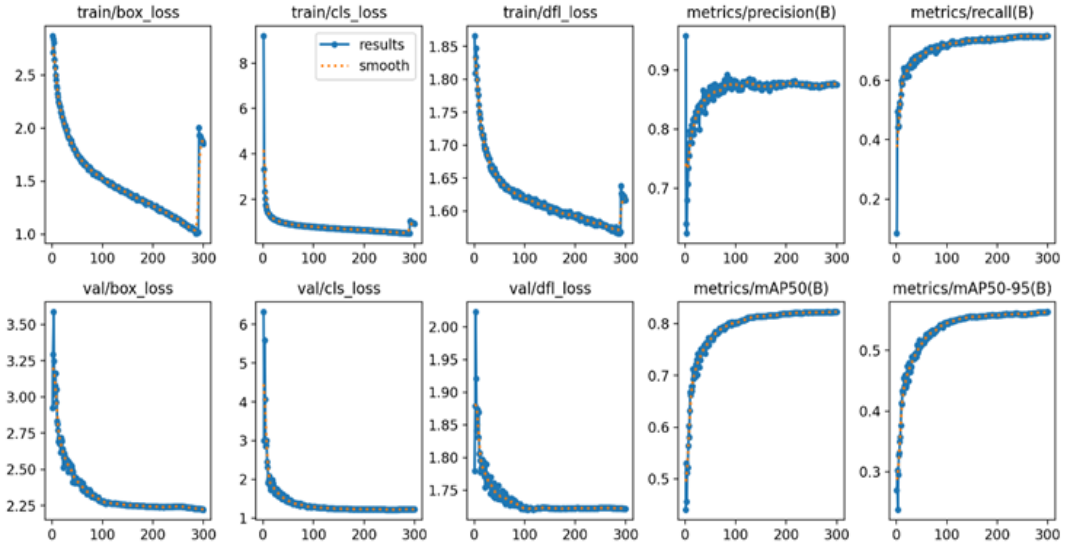


Figure 2. Training Graphs (Detection Model)

Figure 3 illustrates the training accuracy steadily increases and converges near 100%, demonstrating the model's effective learning of data patterns. Validation accuracy follows a similar trend, stabilizing near 100%, with minor fluctuations in early epochs.

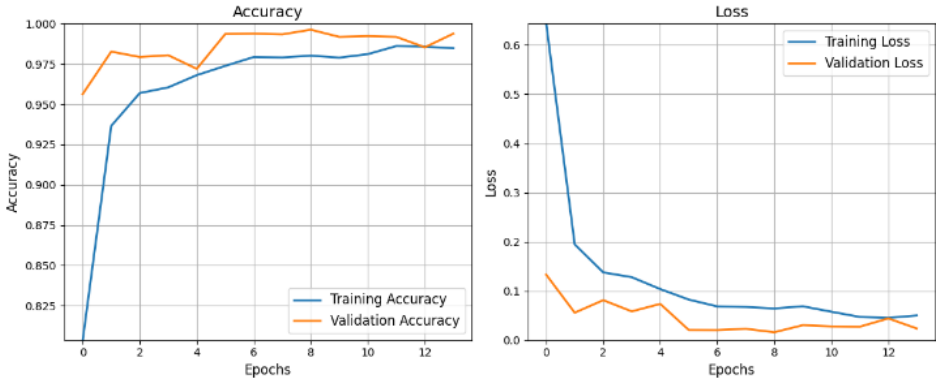


Figure 3. Training and Validation Curves (Classification Model)

Table 1 summarizes the evaluation of the classification model. As shown, its accuracy achieved 99.66% on the test set. Precision, recall, and F1 scores surpassed 0.98 for nearly all classes.

Table 1. Classification Report (Classification Model)

| **Metric** | **Precision** | **Recall** | **F1-Score** | **Support** |
|---|---|---|---|---|
| Accuracy | | | 0.99 | 5604 |
| Macro Avg | 0.9854 | 0.9859 | 0.99 | 5604 |
| Weighted Avg | 0.9966 | 0.9904 | 0.99 | 5604 |

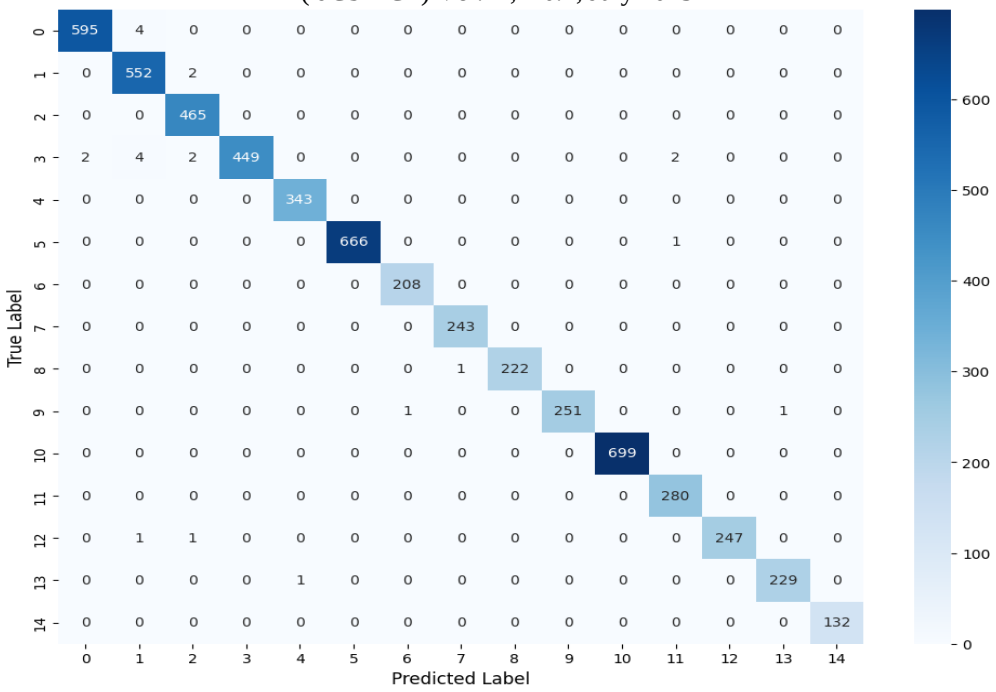


Figure 4. Confusion Matrix Results (Classification Model)

Figure 4 alternatively presents the confusion matrix of the evaluation results which showed diagonal dominance. The extremely high classification accuracy suggests reliable learning of distinguishing features even for visually similar signs.

## 3.2. Mobile Application Development and Testing Results

The app was primarily tested on a Samsung Galaxy A51 with Samsung Exynos 9 Octa 9611 processor, 8GB of memory, and a 48 MP primary back camera. The app was also tested on four other mobile phones with similar specifications. On real-time recognition, end-to-end detection and classification on a mid-range Android phone took about 200–300 ms per frame, supporting near-real-time usage.

Figure 5 shows a screenshot of the app recognizing a traffic sign.


Figure 5. Recognizing a Traffic Sign

The app's interface shows the following: 1) The preview video which covers the majority of the screen. If any supported traffic signs are identified, a bounding box and an overlay text are also displayed. Audio cues are also played simultaneously. 2) The list of traffic signs recognized on the left side of the screen. 3) The time in milliseconds it took to perform inference for debugging purposes (which can be turned off). 4) An "information" button on the bottom left that leads to the user manual.

Traffic signs were reliably detected between 30 and 50 meters away in daylight. Beyond that range, recognition declined because the sign's image resolution in the frame became insufficient. In usual conditions, no false positives appeared in scenarios where no traffic signs were present. Figure 6 shows various screenshots recognizing a traffic sign within approximately 30-50 meters.


Figure 6. Recognizing Traffic Signs at Varying Distances

The app was also tested on challenging conditions. For low-light or night-time scenarios, recognition dropped markedly due to inadequate training data under dark conditions. Reflective sign surfaces occasionally helped but were inconsistent.

Figure 7 shows that the app may not consistently recognize traffic signs in low-light or dark conditions.

Figure 7. Recognizing Traffic Signs at Varying Lighting Conditions

For partial occlusion of signs, up to roughly 25–40 percent obstruction could still be handled, but heavier coverage resulted in detection failure or misclassification. For skewed angles, moderate angles were generally recognized, but severe tilting compromised detection confidence as shown in.

Figure 8 shows a few of these test cases.

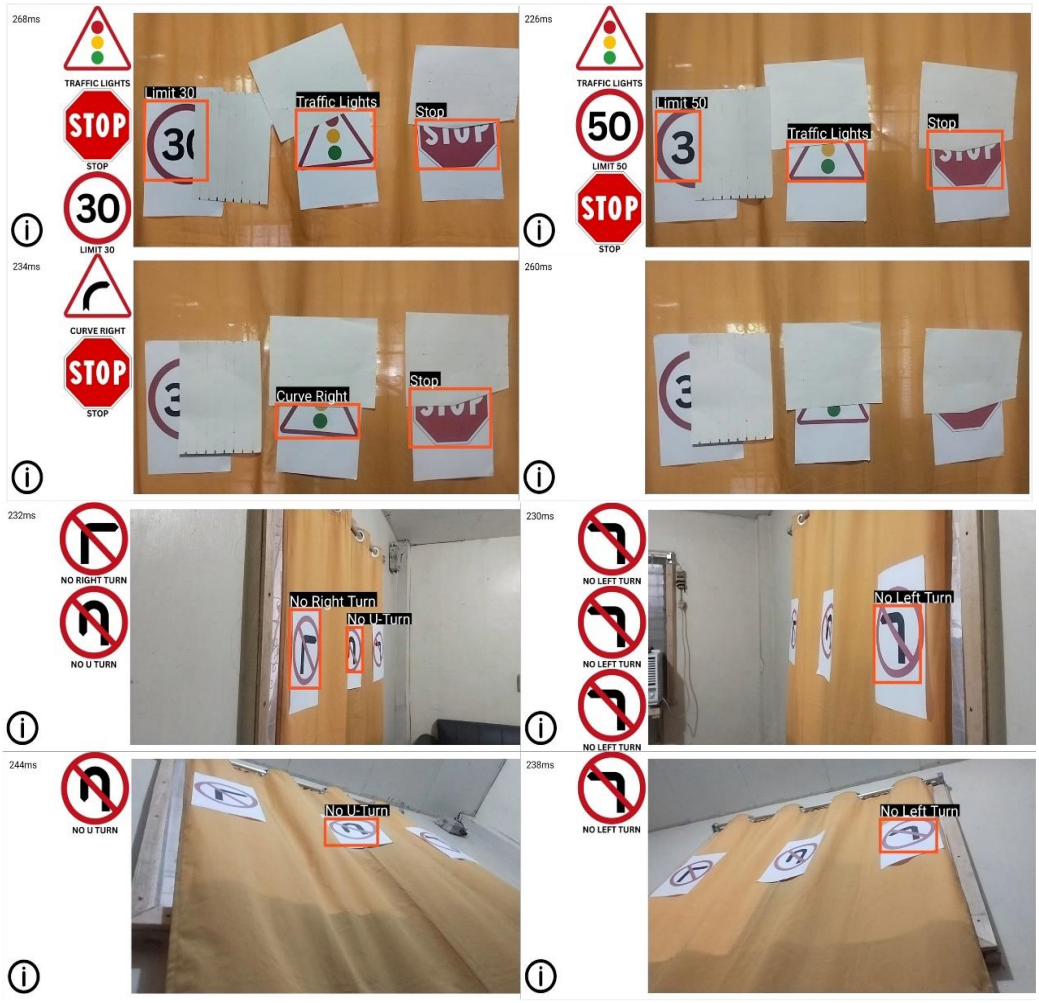


Figure 8. Recognizing Obstructed and Skewed Traffic Signs

## 3.3. Software Evaluation Results

The evaluation scores suggest that the app is highly effective in assisting new drivers with traffic sign recognition and understanding. Table 2 summarizes the user evaluation scores.

Table 2. User Evaluation Scores

| **Question** | **Average** |
|---|---|
| Effectiveness of the Visual and Audio Alerts in Identifying Road Signs | 4.25 |
| Improvement in Understanding of the Traffic Signs | 4.5 |
| Boost in Confidence in Identifying Traffic Signs Correctly | 4.125 |
| Willingness for Continued or Daily Use | 3.75 |

| | |
|---|---|
| Endorsement and Support for Integration in Driving Education | 4.375 |

The app's visual and audio alerts appear to be highly effective, aiding users in recognizing road signs correctly. However, a slightly lower score in its effectiveness suggests that while helpful, some improvements in clarity or responsiveness could further enhance recognition.

Users also expressed that the app successfully helped them in learning traffic signs, suggesting that the app is not just a recognition tool but also contributes to the effectiveness in driving education. The same users also felt more confident in identifying road signs. Finally, users expressed their approval and support for integrating the app into formal driving lessons.

However, it is important to note that while many users expressed willingness to use the app, users may not use this app beyond driving school. Often, experienced drivers may feel they won't need additional guidance once familiar with signs.

## 4. DISCUSSION

The study's findings confirm that an on-device Traffic Sign Recognition system for Philippine traffic signs can operate effectively on commonly available mobile devices, combining CNN-based classification with YOLO-based detection. Prior studies likewise showed that single-stage object detectors achieve a favorable balance between speed and accuracy.[10][11]

Achieving inference times of 200–300 ms on standard Android hardware was facilitated by carefully curating and augmenting the datasets, tuning hyperparameters to balance accuracy and speed, and applying TensorFlow Lite (TFLite) quantization for efficient mobile inference. These strategies collectively addressed the main objective of providing near-real-time TSR functionality without relying on specialized, high-end in-vehicle systems. Despite employing integer quantization to reduce model size for mobile inference, the results showed little loss in detection or classification performance, underscoring the practicality of real-time inference on smartphones as compared with similar studies.[2][5]

In line with the current knowledge of embedded computer vision [9], the success of high-accuracy, low-latency detection, and classification highlights the significance of employing deep learning optimizations tailored to mobile devices.[2][12][13] At the same time, the system's weaker performance in nighttime environments or under severe obstruction indicates specific areas needing further work. For low-light scenarios, future studies could gather more real-world low-light or nighttime images and augment existing data by randomly decreasing brightness or exposure to simulate challenging lighting conditions. Similarly, the model struggled with partially obstructed or occluded signs, suggesting the need to collect additional real-world images where traffic signs are partially blocked. In addition, artificially introducing blocks of pixels over traffic signs during data augmentation would help the model learn to detect partially visible traffic signs. Such domain-specific data enhancement strategies would likely improve the reliability of traffic sign detection and classification in real-world driving.

Investigating specialized CNN variants such as MobileNet or SqueezeNet could reduce the model size further [14][15][16], enhancing compatibility with older or lower-powered phones.

From a global perspective, the approach used here, assembling locally relevant traffic sign data, augmenting it comprehensively, deploying an efficient detection-classification pipeline, and optimizing for on-device inference, presents a repeatable blueprint for extending traffic sign recognition to various countries and contexts.

Overall, the study advances driving education by demonstrating an accessible solution highly effective for learning and reinforcing traffic sign recognition. By focusing on the Philippine context where many vehicles are not equipped with factory-installed traffic sign recognition, the work underscores the potential of deep learning–based recognition apps on everyday smartphones. From a global perspective, the pipeline presents a repeatable blueprint for extending traffic sign recognition to various regions and contexts, provided that traffic sign data and environmental conditions are accounted for. In effect, this project closes a crucial gap between theoretical machine learning approaches and practical, cost-effective implementations that foster beneficial driving educational aid and safer driving practices.

# 5. CONCLUSIONS AND RECOMMENDATIONS

## 5.1. Conclusions

The study successfully developed and evaluated an end-to-end Traffic Sign Recognition system that is compatible with most Android devices, employing both YOLOv10 and CNN for real-time detection and accurate classification of 15 types of Philippine traffic signs. The models collectively achieved strong performance in daytime scenarios, evidenced by a detection mean Average Precision (mAP) of 0.823 and classification accuracy of 99.66%, all while running in real-time at 200–300 ms inference time. These outcomes suggest that a low-latency, high-accuracy TSR pipeline is achievable on consumer-grade hardware.

Nonetheless, performance limitations were observed in low-light scenarios and for signs that were partially obstructed or highly skewed. Addressing these challenges will require more diverse datasets (e.g., images captured at night), targeted data augmentation, and potentially lighter architectures that optimize speed on lower-powered devices. Ultimately, this work demonstrates how deep learning can be leveraged for road safety, offering a scalable alternative to expensive in-vehicle ADAS systems and reinforcing the viability of mobile-based solutions for practical traffic management.

Functional tests confirmed that the integrated mobile app correctly displayed bounding boxes, labeled recognized signs, and immediately provided an audio cue. Users expressed that the app is highly effective for learning and reinforcing traffic sign recognition, with particularly strong support for integration into driving lessons.

## 5.2. Recommendations

Based on the evaluation and testing results of the study, future developments could prioritize optimizing model performance for faster prediction by exploring alternatives like MobileNet SSD or SqueezeNet SSD. While YOLO offers high accuracy and real-time detection capabilities, these models are known for faster detection speeds and may be more suitable for systems where performance and low latency are key. Enhancing the system’s ability to detect traffic signs in real time can make it more effective in dynamic environments.

Additionally, the data distribution imbalance can be addressed by increasing the number of images for underrepresented traffic sign classes across various conditions. Expanding the dataset will ensure more balanced model learning and improve accuracy across all traffic sign classes. Further research could also directly focus on improving the system’s performance under challenging conditions, such as recognizing traffic signs that are excessively obstructed or skewed. Further research could also explore extending the system to function in nighttime or

adverse conditions, which could improve its reliability in diverse environments, especially by enhancing low-light recognition accuracy.
Furthermore, the integration of multilingual audio feedback could make the system more accessible in different regions. Finally, the TSR system's performance should be tested further across a more varied range of Android devices with different hardware configurations to ensure that it maintains consistent performance and scalability.

## ACKNOWLEDGMENTS

The researchers would like to express their appreciation to the following: Firstly, to the Almighty God who gave us strength and wisdom and made the success of this study possible. Secondly, to the researcher's family for their steadfast support that has enabled us to persevere towards the completion of this study. And finally, to the community of La Salle University – Ozamiz, whose camaraderie, encouragement, and shared experiences created a supportive environment throughout this journey. Their collaboration and assistance during critical moments were invaluable in overcoming challenges and fostering mutual growth.

## FUNDING

The study received partial funding from La Salle University – Ozamiz.

## REFERENCES

[1] Robielos, R. a. C., & Lin, C. J. (2022). Traffic Sign Comprehension among Filipino Drivers and Nondrivers in Metro Manila. Applied Sciences, 12(16), 8337. https://doi.org/10.3390/app12168337
[2] Widad, R. (2024). Implementation of machine learning in Android Applications. Theseus. https://www.theseus.fi/handle/10024/859085
[3] Alzubaidi, L., Zhang, J., Humaidi, A. J., Al-Dujaili, A. Q., Duan, Y., Al-Shamma, O., Santamaría, J., Fadhel, M. A., Al-Amidie, M., & Farhan, L. (2021). Review of deep learning: concepts, CNN architectures, challenges, applications, future directions. Journal of Big Data, 8(1). https://doi.org/10.1186/s40537-021-00444-8
[4] Taye, M. (2023). Theoretical understanding of convolutional neural network: concepts, architectures, applications, future directions. Multidisciplinary Digital Publishing Institute. https://doi.org/10.3390/computation11030052
[5] Kim, C., Park, J., Park, Y., Jung, W., & Lim, Y. (2023). Deep Learning-Based Real-Time Traffic Sign Recognition System for urban environments. Multidisciplinary Digital Publishing Institute. https://doi.org/10.3390/infrastructures8020020
[6] Alvi, F. (2024, February 7). Research areas in Computer Vision: Trends and challenges. OpenCV. https://opencv.org/blog/research-areas-in-computer-vision/
[7] Land Transportation Office (LTO) Philippines. (2024). RA 10913: Anti-Distracted Driving Act Philippines. LTO Portal PH. https://ltoportal.ph/anti-distracted-driving-act/
[8] Congress of the Philippines. (2016). Republic Act No. 10913: Anti-Distracted Driving Act. Official Gazette of the Republic of the Philippines. https://legacy.senate.gov.ph/republic_acts/ra%2010913.pdf
[9] Iandola, F. N., Han, S., Moskewicz, M. W., Ashraf, K., Dally, W. J., & Keutzer, K. (2016, February 24). SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5MB model size. arXiv.org. DOI: https://doi.org/10.48550/arXiv.1602.07360
[10] Gorospe, J., Mulero, R., Arbelaitz, O., Muguerza, J., & Antón, M. Á. (2021). A generalization performance study using deep learning networks in embedded systems. Sensors, 21(4), 1031. DOI: https://doi.org/10.3390/s21041031
[11] Wu, J., Leng, C., Wang, Y., Hu, Q., Cheng, J. (2016). Quantized convolutional neural networks for mobile devices. IEEE Conference Publication | IEEE Xplore. DOI: https://doi.org/10.1109/CVPR.2016.521

[12] Diwan, T., Anirudh, G., & Tembhurne, J. V. (2022). Object detection using YOLO: challenges, architectural successors, datasets and applications. Multimedia Tools and Applications, 82(6), 9243–9275. https://doi.org/10.1007/s11042-022-13644-y

[13] Luo, C., He, X., Zhan, J., Wang, L., Gao, W., & Dai, J. (2020). Comparison and benchmarking of AI models and frameworks on mobile devices. arXiv.org. DOI: https://doi.org/10.48550/arXiv.2005.05085

[14] Lin, Y., Tu, C., Kurosawa, L., Liu, J., Wang, Y., & Roy, D. (2024). Applications of Computer Vision in Transportation Systems: A Systematic Literature Review. Social and Human Sciences Web of Conferences. https://doi.org/10.1051/shsconf/202419401004

[15] Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., & Adam, H. (2017). MobileNets: efficient convolutional neural networks for mobile vision applications. arXiv.org. DOI: https://doi.org/10.48550/arXiv.1704.04861

[16] Bhowmik, D., & Appiah, K. (2018). Embedded Vision Systems: A Review of the Literature. Lecture Notes in Computer Science, 204–216. DOI: https://doi.org/10.1007/978-3-319-78890-6_17

## AUTHORS

**Rolando John R. Aca-ac** is a Bachelor of Science in Computer Science student from La Salle University – Ozamiz. As a student, he has been active in engaging in research, community extensions, and workshops. He has earned Data Analyst, Data Science, AI Engineering, and Data Engineering professional certificates from IBM through Coursera. He also passed the PhilNITS Information Technology Passport (IP) examination.

**Patrick Zane G. Sarabia** is a Bachelor of Science in Computer Science student from La Salle University – Ozamiz. He focuses on improving his technical skills through engagement in research and community extension. He also passed the PhilNITS Information Technology Passport (IP) examination.

**Earl Peter J. Gangoso** is a research adviser and a university instructor at La Salle University – Ozamiz. His primary field of study is data science, machine learning, and web development.