Traffic Sign Detection in Vietnam

Phan Van Hoang [1]

Autonomous vehicles are increasingly being adopted worldwide, and the demand for accurate and rapid traffic sign recognition and detection is increasing. This study was conducted to compare the performance between two advanced deep learning architectures: YOLOv8 (You Only Look Once version 8) and Faster R-CNN (Region-based Convolutional Neural Network) in detecting and classifying Vietnamese traffic signs. Both are evaluated based on multiple important criteria such as mAP@50, mAP@0.5:0.95, FPS, etc. The paper uses the Vietnamese traffic signs detection and recognition dataset [1] (rotation, flip, brightness, blur, rain, ...) resulted in 10,170 images. Results show that YOLOv8 achieved a mean Average Precision (mAP) of 92.68%, Precision of 95.83%, and superior 45 FPS, making it suitable for real-world deployment and real-time applications.

Keywords: Traffic sign recognition, YOLOv8, Faster R-CNN, Deep Learning, ADAS, Autonomous vehicles, Computer Vision.

1. Introduction.

In the rapidly developing era of artificial intelligence (AI), applying AI to transportation has become a global trend. Traffic sign detection is a core fundamental problem in computer vision and plays a crucial role in supporting autonomous driving. In Vietnam, with its complex transportation system and diverse types of traffic signs, research and development of effective traffic sign detection solutions has become more necessary than ever.

According to statistics from the National Traffic Safety Committee, "*Vietnam in the first 6 months of 2024 had 12,321 traffic accidents nationwide, killing 5,255 people and injuring 9,599 people*" [2], with a significant proportion due to non-compliance with traffic signs. Automatic traffic sign recognition systems not only help reduce human errors but also improve traffic safety efficiency.

However, Vietnam's specific weather and terrain conditions are also major obstacles. Vietnam has rainy weather conditions and high humidity causing water accumulation on signs or cameras, reducing visibility and obscuring information. Mountainous terrain with various slopes and surfaces distorts traffic sign shapes, making detection and classification more difficult. This makes selecting appropriate model architecture more important in the Vietnamese context.

Real-world applications of this research:

- **Autonomous vehicles:** Providing traffic sign information to automatic or semi-automatic control systems, ensuring traffic safety.
- **Traffic monitoring systems:** Automatically detecting and processing traffic sign violations.
- **Mobile applications:** Supporting drivers by indicating speed limits for current road sections or applicable prohibition signs, command signs, and guidance signs.

- **ADAS systems:** Warning drivers when violating speed limits, driving in wrong lanes, or ignoring important signs.

This study applies two state-of-the-art deep learning architectures to solve object detection problems:

YOLOv8 (You Only Look Once version 8): Belongs to the single-stage detector group, processing detection and classification in a single forward pass. YOLOv8 was chosen for its high real-time performance, optimized architecture, and superior performance compared to previous YOLO versions.

Faster R-CNN: Representative of two-stage detectors with Region Proposal Network (RPN) as a special subnet to extract object regions and Faster R-CNN detector to accurately determine traffic sign types and locations. This model was selected due to its high accuracy in both object localization and classification.

This study aims to provide a comprehensive and objective comparison between the two methods, thereby offering specific recommendations for selecting appropriate algorithms in practical applications in Vietnam.

2. Related Work.

Traffic sign detection using deep learning has undergone significant advancements, with numerous studies focusing on improving detection accuracy, optimizing real-time processing speed, and addressing challenges in complex environments. In recent years, the topic has attracted considerable research interest, with notable contributions from Gaihua Wang [3], Njayou Youssouf [4], R.K Megalingam [5], R.K Megalingam [6], Xiang Gao [7]. Two major families of architectures have been extensively explored: single-stage detectors (e.g., the YOLO series) and two-stage detectors (e.g., Faster R-CNN), each offering distinct advantages in balancing speed and accuracy.

Recently, Gaihua Wang et al. [3] proposed comprehensive improvements to YOLOv8 to address the challenge of detecting small objects in complex environments. Using the TT100K dataset, the study revealed that the original YOLOv8 model still suffers from missed detections and false positives under occlusions and densely distributed objects. Three main enhancements were introduced: the AFP (Attention Feature Pyramid) module with ASF framework to improve the Neck component, a lightweight LW_C2f module to reduce model parameters, and the Wise-IoU loss function as a replacement for CIoU to optimize bounding box regression. These improvements are particularly crucial when handling small-scale traffic signs - a common challenge in real-world traffic scenarios.

In addition, several studies have conducted comparative evaluations of different model architectures. Njayou Youssouf et al. [4] compared the performance of Faster R-CNN and YOLOv4 on the GTSRB dataset. The results showed that YOLOv4 outperformed Faster R-CNN with a mAP of 59.88% at 35 FPS, while Faster R-CNN achieved only 43.26% mAP at 6 FPS. This difference highlights the advantage of single-stage detectors in achieving a better speed-accuracy trade-off. However, both methods showed limited mAP, especially Faster R-CNN, indicating that further enhancements are necessary for practical deployment.

Acknowledging these limitations, Xiang Gao et al. [7] integrated the Feature Pyramid Network (FPN) into Faster R-CNN to enhance multi-scale feature extraction. Their research addressed real-world challenges such as weather effects, poor lighting conditions, long-distance detection, and distinguishing visually similar signs. The results demonstrated significantly improved performance in detecting small objects and under low-light conditions, contributing meaningfully to real-world deployment efforts.

Despite these advancements, several critical research gaps remain. First, most existing studies rely on international datasets (e.g., TT100K, GTSRB), while Vietnamese traffic signs possess unique characteristics in terms of design and layout. Second, Vietnam's distinct traffic conditions—including high density, complex occlusions, and tropical weather - have not been thoroughly investigated. Finally, the optimization of speed-accuracy trade-offs for real-time applications in Vietnam's context remains an open challenge. This study aims to address these gaps by comparing the performance of YOLOv8 and Faster R-CNN on a Vietnamese traffic sign dataset and evaluating their capabilities in handling the unique challenges of domestic traffic environments.

3. Problem Statement.

3.1. Problem.

Vietnamese Traffic Sign Detection is an object detection problem in computer vision, belonging to supervised learning. The goal is to automatically detect locations and classify traffic signs in images or videos from car cameras, phone cameras, or traffic sign photos.

Input consists of RGB images of various resolutions in formats such as PNG and JPG.

Output is rectangular bounding boxes around traffic signs (x, y, width, height) and class labels of traffic signs.

3.2. Algorithms.

3.2.1. YOLOv8 (You Only Look Once version 8).

YOLOv8 belongs to the single-stage object detection group with supervised learning approach, developed by Ultralytics as an improvement from YOLOv5. The YOLOv8 model architecture is shown in *Figure 3.1.*

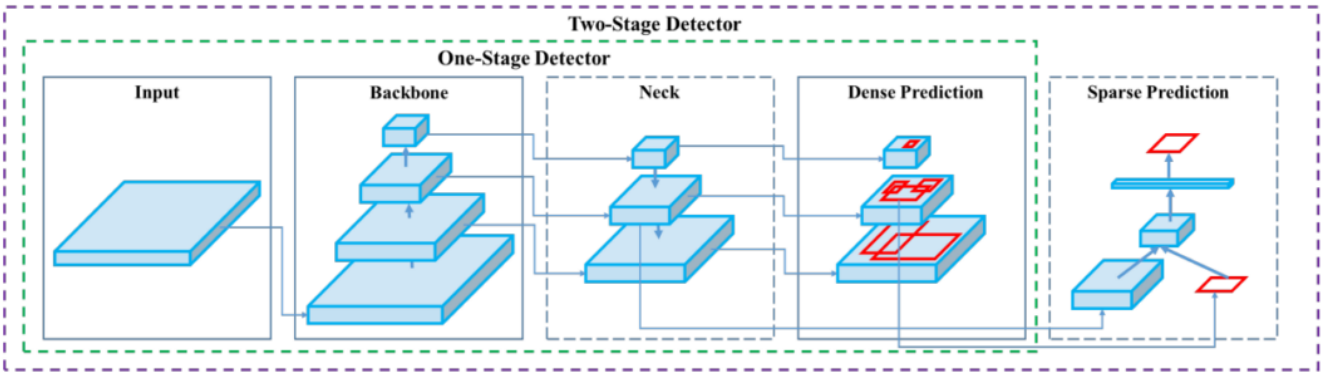

Figure 3.1: YOLOv8 Architecture.

Component Details:

1. ***Backbone (New CSPDarknet backbone):*** It can represent features at multiple scales, capturing information from different levels of abstraction. Hierarchical representation of the input, which forms the foundation for subsequent processing.
2. ***Neck (PANet-style neck):*** Acts as a bridge between Backbone and Head. Merges feature maps obtained from different stages of the backbone to ensure the network can detect objects of various sizes. Reduces spatial resolution and dimensionality of resources, facilitating computation, which increases speed while potentially reducing model quality.
3. ***Head (Decoupled Head):*** The head is the final part of the network and is responsible for generating the outputs, such as bounding boxes and confidence scores for object detection. This layer helps produces bounding boxes associated with potential objects in the image, then assigns confidence scores to each bounding box to indicate the likelihood of object presence, and sorts the detected objects within the bounding boxes according to their respective categories.

3.2.2. Faster R-CNN.

Faster R-CNN belongs to the **two-stage object detector** group with supervised learning approach, combining Region Proposal Network (RPN) and Fast R-CNN [9]. The architecture is shown in *Figure 3.2*.

1. ***Backbone (ResNet-50/101):*** Uses famous CNNs like ResNet50, VGG16, or MobileNet to extract features (feature maps). Output is a feature map smaller than the original image but with deeper dimensions (channel size), very meaningful for region detection and object classification.
2. ***Region Proposal Network (RPN):*** Functions as a sliding filter over predicted regions and assigns scores indicating the probability of object appearance in the input image.

   RPN introduces the concept of ***anchor boxes***: predetermined bounding boxes with specific sizes and aspect ratios. They are generated at each position on the feature map. Anchor boxes have diverse sizes and typically have 9 anchors (3 scales × 3 aspect ratios 0.5, 1, 2) at each position, covering potential areas where objects might appear.

   From these anchors, RPN can simultaneously predict multiple regions and calculate object existence probability in each area, adjusting bounding boxes to better fit objects.

3. ***ROI Pooling:***
   ROI Pooling (The Region of Interest pooling layer) extracts fixed-size features from proposals. Specifically, ROI Pooling divides each Region Proposal into fixed grids, then performs max-pooling for each cell, generating fixed-size feature maps for each Proposal.

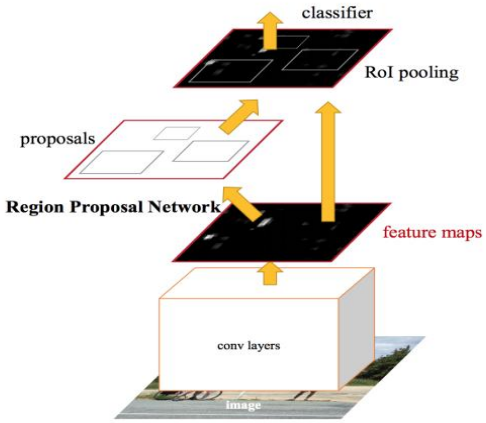

Figure 3.2: Faster R-CNN Architecture

4. ***Classification and Bounding Box:***
   The final part of Faster R-CNN consists of two parallel fully connected layers:
   - **classification head** that predicts the class of the object in each region proposal
   - **Bounding box regression head**: Further refines the coordinates of the detected object.

   These heads operate on fixed-size feature maps generated by RoI Pooling. Through the bounding box regression head, the network obtains refined coordinates for each class, allowing it to accurately predict and adjust bounding boxes as needed.

4. Experiments.

4.1. Data.

4.1.1. Data Description.

The dataset used is Vietnamese traffic signs detection and recognition [1], containing 1,170 labeled images across 29 different traffic sign types. Bounding boxes are formatted according to YOLO standard. The training set contains 900 images (77%), test set 90 images, and validation set 180 images. Labels in the dataset include:

0: one way prohibition, 1: no parking, 2: no stopping and parking, 3: no turn left, 4: no turn right, 5: no u turn, 6: no u and left turn, 7: no u and right turn, 8: no motorbike entry/turning, 9: no car entry/turning, 10: no truck entry/turning, 11: other prohibition, 12: indication, 13: direction, 14: speed limit, 15: weight limit, 16: height limit, 17: pedestrian crossing, 18: intersection danger, 19: road danger, 20: pedestrian danger, 21: construction danger, 22: slow warning, 23: other warning, 24:

vehicle permission lane, 25: vehicle and speed permission lane, 26: overpass route, 27: no more prohibition, 28: other.

4.1.2. Data Distribution.

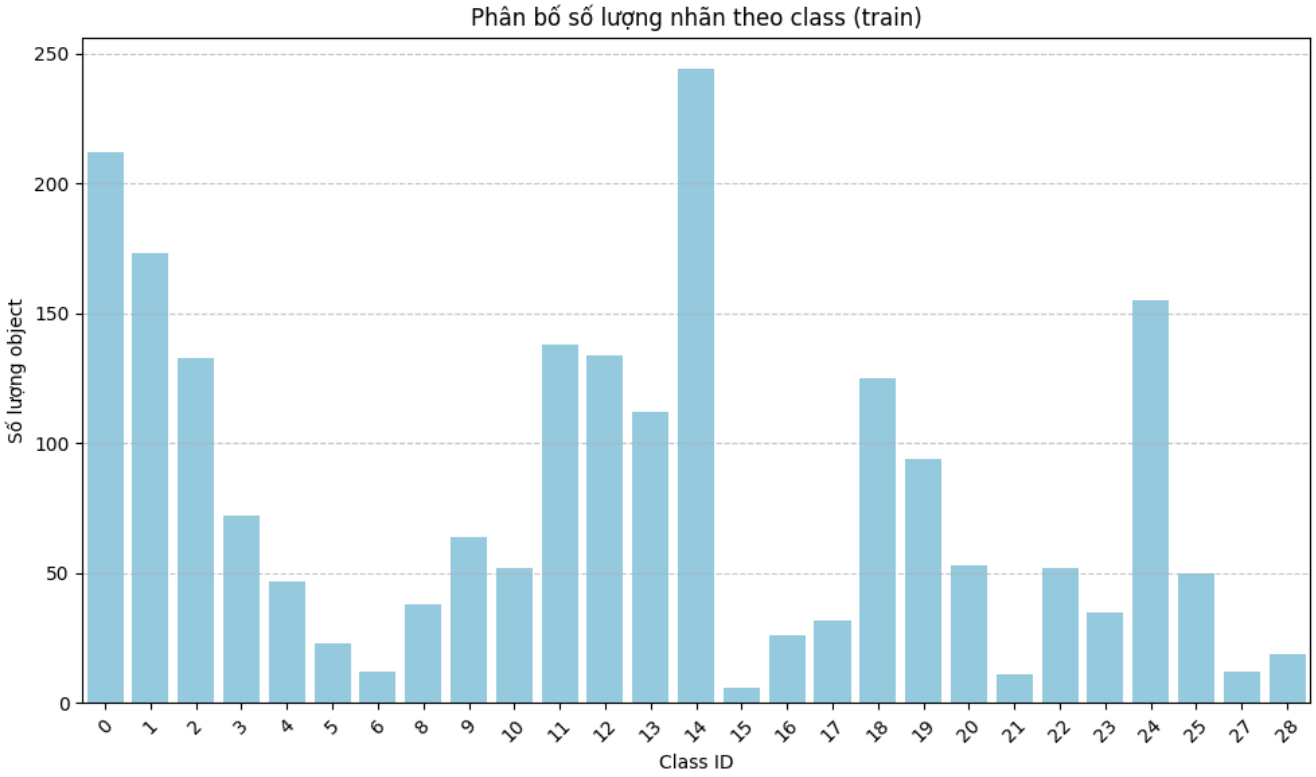

Figure 4.1: Distribution chart of label quantities by class in training set

The label distribution chart by class *Figure 4.1* shows the dataset is imbalanced with significant differences between classes:

- 0 and class 14 have the highest number of instances (212 and 224 respectively).
- Classes like Class 15 (6 instances), Class 21 (11 instances), Class 6 (12 instances), Class 27 (12 instances), Class 28 (19 instances) have very few samples.

This imbalance significantly affects prediction. The model tends to favor dominant classes while ignoring minority ones, as correctly predicting large classes has a greater impact on the loss function.

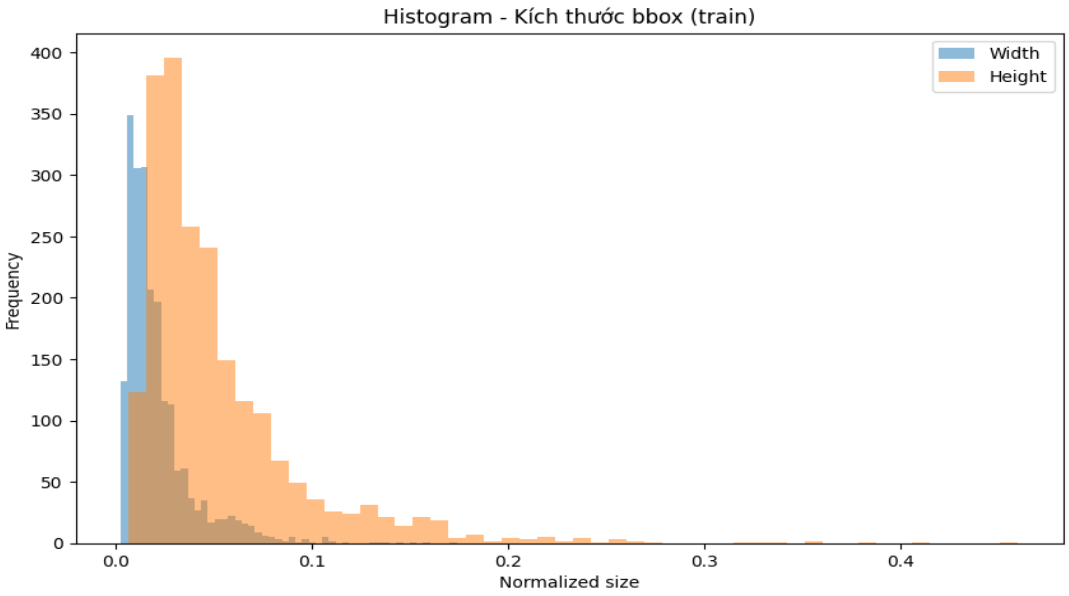

Figure 4.2: Bounding box size distribution chart

X-axis (Normalized size): width (blue) and height (orange) values of bbox (after dividing by original image width/height, normalized to [0, 1]) according to YOLO format**.**

Y-axis (Frequency): number of bboxes with width or height falling into that size range.

The bounding box size distribution chart *Figure 4.2* shows:

- Distribution heavily skewed toward small bounding boxes for both width and height, heavily skewed toward 0-0.1. Highest in the 0-0.05 range, meaning bounding boxes are typically 5% of the total image width/height.
- Height (orange) distribution is more spread out, width (blue) concentrated more on the left side, showing more images have height greater than width. This reflects that traffic signs have more vertical rectangular shapes.

Small bounding boxes can make traffic sign detection during model training difficult and inaccurate, requiring techniques like anchor boxes or increasing input image size to detect small signs.

4.1.3. Data Preprocessing.

Data augmentation techniques were uniformly applied to both models. Each original image is augmented 10 times.

Technique	Parameters	Purpose
Horizontal Flip	p=0.5 (50% probability)	Horizontal image flipping
Random Brightness Contrast	p=0.5 (50% probability)	Randomly change image brightness and contrast
Motion Blur	p=0.3 (30% probability)	Apply motion blur effect to image
Rotate	limit=15, p=0.5 (rotate ±15°, 50% probability)	Rotate image within -15 to +15 degrees

Affine	scale=(0.8, 1.2), translate_percent=(0.1, 0.1), p=0.5 (scale 80-120%, translate ±10%, 50% probability)	**scale:** randomly scale image 80% to 120%. **translate_percent:** translate image along X/Y axis up to 10%.
Resize	height=416, width=416	Resize image to standard value

Table 4.1: Data augmentation technique parameters

4.2. Results.

4.2.1. Experimental Setup.

Parameter	YOLOv8	Faster R-CNN
Batch Size	32	4
Learning Rate	0.01	0.001
Epochs	150	15
Optimizer	SGD	SGD
Weight Decay	0.0005	0.0005
Warmup Epochs	3	3

Table 4.2: Hyperparameters during model training

4.2.2. Evaluation Metrics.

Precision và Recall:

$$\text{Precision} = \frac{TP}{TP + FP} \; ; \; \text{Recall} = \frac{TP}{TP + FN}$$

IoU (Intersection over Union): Used to determine if a prediction is correct or incorrect. True Positive if IoU between predicted box and ground truth box exceeds threshold (usually 0.5 in AP@0.5).

$$\text{IoU} = \frac{Overlap\ area}{Union\ area}$$

AP (Average Precision): Area under the Precision-Recall (P-R Curve) calculated separately for each class.

$$AP = \sum_{n} (R_n - R_{n-1}) P_n$$

Where: R_n is recall at point n, P_n is precision at point n

mAP (mean AP): Average AP across all classes. [10]

$$mAP = \frac{1}{N} \sum_{i=1}^{N} AP_i$$

mAP@0.5 is average AP of each class with IoU threshold = 0.5

mAP@0.5:0.95 calculates AP across multiple IoU thresholds 0.50, 0.55, 0.60, ..., 0.95 (11 points) then averages all.

4.2.3. Overall Results.

Method	mAP@0.5	mAP@0.5:0.95	Recall	Precision
Faster R-CNN	0.9025	0.6561	0.6904	0.9025
YOLOv8	0.9268	0.7728	0.8302	0.9583

Table 4.3: Performance comparison between models.

Method	Model Size (MB)	Parameters (M)	Inference Time (s)	FPS
Faster R-CNN	158.60	41.44	0.0491	20.37
YOLOv8	49.62	25.86	0.0219	45.75

Table 4.4: Speed and Efficiency comparison table.

Method	mAP50 (%)	Precision (%)	Recall (%)	Parameters (M)	Size (MB)
D-DETR	76.3 [3]	73.2	69.5	20.6	41.7
YOLOv3	75.5	74.9	68.3	8.9	18.7
YOLOv7	73.6	73.1	68.6	37.4	75.2
Faster-RCNN	69.4	69.3	63.6	27.6	108.6
Faster R-CNN (ours)	90.25	90.25	69.04	41.44	158.60
Our YOLOv8 (ours)	92.68	95.83	83.02	25.86	49.62

Table 4.5: Performance comparison with other papers.

4.2.4. Predictions and Reality of Our Two Models.

4.2.5. Detection Analysis.

Regarding prediction capability, YOLOv8 outperforms in all metrics: mAP@0.5 is 0.9268 vs 0.9025 for Faster R-CNN. YOLOv8's mAP@0.5:0.95 is significantly higher (0.7728 vs 0.6561), reflecting better quality across multiple IoU thresholds. Both Recall and Precision of YOLOv8 are higher, showing YOLOv8 detects more objects while making fewer mistakes. Faster R-CNN has good precision (0.9025) but significantly lower recall (0.6904), indicating it misses more traffic signs. The low recall of Faster R-CNN may be due to its two-stage architecture being more conservative in making predictions, leading to missing edge cases.

Regarding size, YOLOv8 is much more compact (3 times smaller than Faster-RCNN), with fewer parameters showing easier deployment. YOLOv8 is also twice as fast with ~45.75 FPS vs ~20.37 for Faster R-CNN, and lower inference time (0.0219s vs 0.0491s), making YOLOv8 more suitable for real-time applications.

When predicting with real data, both have some incorrect predictions. However, YOLOv8 detects more traffic signs including small ones, with higher accuracy. Faster R-CNN maintains high prediction rates but easily misses or mislabels between similar signs.

5. Conclusion.

This study conducted a comprehensive comparative evaluation between YOLOv8 and Faster R-CNN on Vietnamese traffic sign detection problems. Through experiments, the study provided insights into performance and applicability of each method. The research proves YOLOv8 is the superior method for this problem, with mAP@0.5 of 0.9268 and real-time processing speed of 45.75 FPS, meeting Vietnam's traffic requirements.

Although YOLOv8 shows superior overall performance, analyzing advantages and disadvantages of both architectures is necessary to understand optimal application contexts for each method.

YOLOv8 excels with high real-time processing speed, ideal for applications requiring quick response like autonomous vehicles and Advanced Driver Assistance Systems (ADAS). Its single-stage architecture ensures optimal balance between accuracy and speed, plus compact size makes it well-suited for deployment on mobile and edge devices. However, YOLOv8 may struggle with extremely small or severely obscured traffic signs, and in some complex cases, localization accuracy may not match two-stage detectors

Conversely, Faster R-CNN, with its two-stage architecture, excels in providing higher localization accuracy and better handling of objects with different sizes through Region Proposal Network (RPN). However, Faster R-CNN's core disadvantage is slow processing speed, making it unsuitable for real-time applications. Large model size also requires higher computational resources, and lower Recall may lead to missing some important traffic signs.

To improve traffic sign detection system effectiveness, several future research and development directions can be applied: focus on handling data imbalance to improve prediction performance for classes with few samples; experiment with upgrading YOLO algorithms to higher versions like YOLOv11; apply attention mechanisms to improve detection of small signs; apply ensemble methods

to combine advantages of different models; collect more traffic signs for larger training data, combine damaged images to avoid overfitting; deploy models in real applications to become truly practical applications.

This study proves that artificial intelligence (AI), especially YOLOv8, has great potential for effective application in solving traffic sign detection problems in Vietnam. The results obtained are not only scientifically significant but also open tremendous practical application potential in improving traffic safety, aiming to minimize traffic accidents and build safe, efficient transportation environments for people.

References

[1] D. Nguyen, "Vietnamese traffic signs detection and recognition," 2023. [Online]. Available: https://www.kaggle.com/datasets/jaydenguyenx/vietnamese-traffic-signs-detection-and-recognition.

[2] Báo Nhân Dân and Lê Quốc Minh, "Tai nạn giao thông trên toàn quốc 6 tháng đầu năm 2024," [Online]. Available: https://nhandan.vn/infographic-tai-nan-giao-thong-tren-toan-quoc-6-thang-dau-nam-2024-post817294.html.

[3] Gaihua Wang, Peng Jin, Zhiwei Qi and Xiaohuan Li, "Traffic sign detection method based on improved YOLOv8," 2025.

[4] Njayou Youssouf, "Traffic sign classification using CNN and detection using faster-RCNN and YOLOv4"," 2022.

[5] Rajesh Kannan Megalingam, Kondareddy Thanigundala, Sreevatsava Reddy Musani, Hemanth Nidamanuru and Lokesh Gadde, "Indian traffic sign detection and recognition using deep learning," 2022.

[6] Rajesh Kannan Megalingam, Kondareddy Thanigundala, Sreevatsava Reddy Musani, Hemanth Nidamanuru and Lokesh Gadde, "CTM-YOLOv8n: A Lightweight Pedestrian Traffic-Sign Detection and Recognition Model with Advanced Optimization," 2024.

[7] Xiang Gao, Long Chen, Kuan Wang, Xiaoxia Xiong, Hai Wang and Yicheng Li, "Improved Traffic Sign Detection Algorithm Based on Faster R-CNN," 2022.

[8] Akshit Mehra, "Understanding YOLOv8 Architecture, Applications & Features," 12 Sep 2024. [Online]. Available: https://www.labellerr.com/blog/understanding-yolov8-architecture-applications-features/.

[9] Gaudenz Boesch, "The Fundamental Guide to Faster R-CNN," 5 October 2024. [Online]. Available: https://viso.ai/deep-learning/faster-r-cnn-2/.

[10] Deval Shah, "Mean Average Precision (mAP) Explained: Everything You Need to Know," 7 mar 2022. [Online]. Available: https://www.v7labs.com/blog/mean-average-precision#what-is-mean-average-precision-(map).