Automatic detection and classification of Brazilian traffic signs using YOLO

Carlos H. A. Tavares, *Undergrad student*, Gabriel M. O. Machado, *Undergrad student*, Ana C. C. Martins, *Undergrad student*, Yan S. Moura, *Undergrad student*, and Tarcisio F. Maciel, *Senior Member, IEEE*

***Abstract*—Poor traffic sign conditions compromise road safety and due to the huge dimensions of the road network, its manual inspection is costly and slow. This work demonstrates the use of the YOLO (You Look Only Once) algorithm for automated detection and classification of Brazilian traffic signs. An initial model was trained on a small dataset of manually analyzed images. The model was then used to process a larger set of images extracted from public videos and from the Internet. The resulting enlarged dataset was subjected to manual verification and correction of the initial detections/classifications, generating a dataset with 2,000+ images (6,000+ after data augmentation). A final model was then trained on the refined dataset, achieving significant accuracy, recall, and mAP@50 values for the 10 most well-represented traffic signs. The simple approach presented herein shows a strong potential for cost-effective, real-world applications in intelligent traffic systems and urban monitoring.**

***Index Terms*—Traffic signs, detection, classification, YOLO.**

I. Introduction

Urban densification in recent decades has contributed to the increased use of vehicles, which in turn raises the risk of traffic accidents which are frequently associated with missing road signage or its poor maintenance. According to the World Health Organization (WHO), traffic accidents cause between 20 and 50 million non-fatal injuries each year and account for approximately 3% of the Gross Domestic Product (GDP) of low-income countries, where 90% of traffic-related deaths occur [1]. In Brazil, the National Confederation of Transport (CNT) points out that failures in road infrastructure, including signage, are responsible for a significant number of accidents on federal highways, which cause fatalities and non-fatal injuries, which potential long-term consequences could often be prevented through proper vertical traffic signage, as later highlighted in [2].

For regional context, Ceará state's road network exceeds 53,000 kilometers (including municipal, state, and federal roads), demanding consistent monitoring of its maintenance conditions. Annually, Ceará sees thousands of road accidents, many fatal, often attributable to deficient signage and poor road conservation. Public entities, including the National Department of Transport Infrastructure (DNIT) and entities at various levels of the government are tasked with this monitoring and maintenance. Given its sheer scale, supervised automated monitoring becomes an essential tool for a more efficient management of the road network.

Teleinformatics Engineering Department, Federal University of Ceará, Fortaleza, Brasil, e-mail: {carlosheasta, gabrielmarinho, anacecilia.martins, yanmoura}@alu.ufc.br, maciel@ufc.br. This work was partially supported by INCT-Signals under grants INCT-25255-82587.32.41/64 (FUNCAP) and 406517/2022-3 (CNPq), FUNCAP/Universal under grant UNI-0210-00043.01.00/23, and CAPES (Finance code 001). T. F. Maciel was partially supported by CNPq under grant 312471/2021-1.

While agencies like DNIT already track a Maintenance Condition Index (ICM), critical aspects like horizontal and vertical road markings and signs require more focus. To the best of the author's knowledge, the Brazilian public sector currently lacks a dedicated artificial intelligence (AI) tool for automatically detecting and classifying vertical signage. In this paper, we propose to use the YOLO (You Look Only Once) detection/classification tool to automatically identify traffic signs on road images. This approach represents a crucial initial step for realizing supervised automated monitoring of vertical signage quality and improving the ICM. Furthermore, combining such a solution with Geographic Information Systems (GIS) and a specialized post-processing stage to identify the conservation state of vertical signs could offer a comprehensive solution with significant economic and social benefits for Ceará and other regions.

In this context, the use of AI to promote safer traffic shows promises and challenges in detection/analysis of visual elements in urban environments (e.g., traffic signs), especially under adverse lighting, weather, and occlusion conditions.

A. Contributions

In our approach, we initially created and labeled[1] [3] a small dataset with which we trained a preliminary YOLOv11 model. To expand the dataset and improve model performance, a Python script was used to extract frames from publicly available Internet videos depicting Brazilian road driving scenarios, which were compiled into a new dataset, on which the initial model was applied to produce coarse object annotations. These were then manually reviewed and corrected, producing a refined labeled dataset, which was subsequently used to retrain the model, resulting in improved accuracy and generalization.

It is worth noting that this semi-automated and iterative process can be repeated to continuously refine both the dataset and the trained models. This approach supports scalable dataset expansion with minimal manual annotation overhead after the initial phase.

As contributions of this paper, we highlight:

- Construction of a dataset of Brazilian traffic signs for AI applications.
- Presentation of a simple approach for traffic sign detection/classification with successive dataset building and model refinement.
- Showcase that well-represented classes of traffic signs can be efficiently detected and classified automatically using of-the-shelf tools.

[1]For instance, labelImg was used.

B. Related works

The idea of using AI to detect and classify traffic signs is not new; some works already detect and classify traffic signs with YOLO. In Brazil, the authors in [4] studied a methodology for automatically detecting and georeferencing traffic signs on rural roads using YOLOv8 and Convolutional Neural Networks to improve asset management and road safety. Although [4] investigated this subject and made efforts to create a Brazilian Regulatory Traffic Sign Recognition Dataset (BRTSD)[2] [5], as far as the authors are aware, there is no dataset for Brazilian traffic signs adapted for YOLO-based applications, maybe except for [6]. This claim is also partially evidenced, e.g., in [7] which has conducted research on the subject over several years. Anyway, the automatic traffic sign detection with YOLO in a global context remains a relevant research area [8], [9]. Indeed, the work in [8] has shown good results with the YOLOv5-TS model, a model optimized for traffic sign detection. The authors in [9] also proposed an optimized YOLO version – YOLO-BS, based on YOLOv8 – to improve traffic sign recognition. The model performed better than previous YOLO versions, especially for small objects, while maintaining real-time efficiency. In [10] and [11], the performance difference between YOLO v7 and v8 for traffic signal detection was analyzed, concluding that YOLOv8 performed better than YOLOv7. Both [10], [12] addressed traffic sign detection problem in challenging conditions using YOLOv8, with [12] proposing a combination of algorithms to create a lightweight solution for traffic sign detection. In [13], the TSD-YOLO is proposed, a traffic sign detection model based on YOLO architecture, aiming to improve robustness in autonomous driving scenarios. The model achieved strong detection performance and was evaluated on large-scale datasets, showing its potential for real-world deployment. In [14], the YOLO-SG model (based on YOLOv5) focused on detecting small traffic signs in complex scenes. By introducing new elements to the solution architecture, the method achieved better mAP and significantly reduced model size, showing strong performance on GTSDB and TT100K datasets [7], [9]. In the Brazilian context, a YOLO-based approach for detecting and georeferencing vertical traffic signs on rural roads was developed in [6]. The authors trained and compared YOLOv5, YOLOv7, and YOLOv8 using the "normal-brazilian-traffic-signs" dataset composed of over 3,000 labeled real and synthetic images. YOLOv8 achieved the highest mAP scores, particularly when the number of training epochs was increased. This work demonstrates the applicability of deep learning for automating road asset monitoring in less-structured environments such as rural highways. Additionally, all the previous works demonstrated the applicability of YOLO in various contexts with the objective of detecting traffic signs.

C. Paper organization

The remainder of the paper is organized as follows. Section II presents the materials and methods employed to build the dataset and train the YOLOv11 model of our work. Section III presents and discusses the results obtained from applying YOLOv11 on the images of the dataset. Finally, Section IV concludes this work and presents some perspectives.

II. Materials and methods

In this section, we discuss the methodology used in this work. We consider the traffic signs defined in the Brazilian Traffic Sign Manual (MBST), particularly for the set of Regulatory and Warning signs. Scenery images in different environmental conditions were used to compose the dataset employed for the detection and classification of traffic signs, including its stages of training, validation, and testing.

In Section II-A, we provide details on how the dataset was built. In Section II-B, we shortly revisit YOLO, its versions, and present the model selected for our studies. In Section II-C, we specify the hardware configuration employed and the training process adopted for the model.

A. Dataset

In order to solve the proposed task, a small dataset was created initially by manually collecting public images from Google Images with the help of a browser extension[3] The traffic signs contained in these images were then labeled manually[1] and the annotations in YOLO format were saved to text files paired to the images. This initial dataset was then split into 70/20/10% subsets for training, validation, and test, respectively. To cope with the reduced size of the dataset, data augmentation was used to generate up to three additional images for each training set entry considering the parameters in Table I. Then, an initial YOLOv11 model (YOLO11m) was trained using this dataset.

Table I
Data augmentation parameters.

Augmentation	Value
Crop	0% to 20%
Rotation	-15° to +15°
Brightness	-15% to +15%
Blur	0 to 2.5 pixels

After gathering a sufficient number of images (529 images leading to 1589 images after data augmentation) to allow the model to perform reasonably well, we began collecting road videos from YouTube. For this initial study, we collected a set of only six videos and wrote a Python script to extract out of each video one frame every two seconds using the FFMPEG utility [15]. We ran inference using the initial model on the frames extracted of these videos and for those in which some object was detected, the generated labeling was manually verified and eventually corrected afterwards [3]. Each traffic sign detected in an image was classified according to its MBST code and the labeled data (namely, class numbers followed by normalized bounding boxes) were saved in text files paired to each image. The images were divided again into training, validation, and test sets using a 70/20/10% split. From this process, after data augmentation, a dataset of 6,043 images was

[2]Dataset available at https://github.com/ludii-co/BRTSRD.

[3]For instance, Download All Images was used.

built featuring varying quality images sourced from different Brazilian urban roads. Nevertheless, as it will be discussed later, under-representation of some traffic sign classes is still an issue for this and others works.

B. YOLO model

Over the past few years, YOLO passed through significant structural enhancements, becoming a state-of-art tool for object detection, classification, and tracking. The authors in [16] detailed the evolution of YOLO from its version 1 (YOLOv1) until one of its latest versions, YOLO-NAS. The paper highlights the main architectural and methodological advances in YOLO. YOLOv1 revolutionized object detection by treating it as a single regression problem, using bounding boxes and class probabilities for the entire image. YOLOv2 significantly enhanced accuracy while maintaining speed by introducing anchor boxes learned via k-means clustering, incorporating batch normalization, and employing a stronger backbone. YOLOv3 further improved detection, especially for small objects, by using three different scales and an even more robust Darknet backbone. YOLOv4 represented a consolidation of numerous best practices, integrating new models in the backbone and in the neck of its architecture for improved feature aggregation, along with advanced data augmentation techniques. In particular, YOLOv4 consists of backbone, neck, and head with the first being responsible for extracting features from the image, the second being designed to collect feature maps from different stages and help in better feature extraction and fusion, and the third being used for bounding box prediction and for classifying objects within those boxes. YOLOv5 shifted to PyTorch, emphasizing usability and deployment with various model sizes and an auto-anchor mechanism for custom datasets. YOLOv6, developed for industrial applications, featured a further improved backbone and moved towards anchor-free detection for faster inference, while YOLOv7 focused on "trainable bag-of-freebies" and extended backbone to achieve state-of-the-art accuracy and speed performance. YOLOv8, further refined the architecture by defaulting to anchor-free detection, employing a decoupled head, better feature reuse, and expanding its capabilities to include instance segmentation and pose estimation. Finally, YOLO-NAS stands out by leveraging Neural Architecture Search (NAS) to automatically design highly efficient and accurate models with quantization-aware blocks, aiming for an optimal balance between performance and latency for real-time applications. Each YOLO version has consistently pushed the boundaries of real-time object detection by refining its architectural components, loss functions, training strategies, and deployment considerations. YOLOv11 [17], one of the most recent YOLO version, is a further improvement of this well-established framework for object detection, classification and tracking. It was chosen for its fast processing, high accuracy, and efficiency. It includes improvements for faster and more efficient feature extraction, a convolutional block with parallel spatial attention module for enhanced feature processing that selectively applies attention, and a refined overall architecture for improved feature aggregation. It also offers multi-task support for several machine intelligence and image processing algorithms, with full support for Graphical Processing Units (GPUs) and easy-of-usage for the final users. The generic architecture of YOLOv11 is presented in Fig. 1, showing the three typical elements of the YOLO architecture: backbone, neck, head. For more details, see [18].

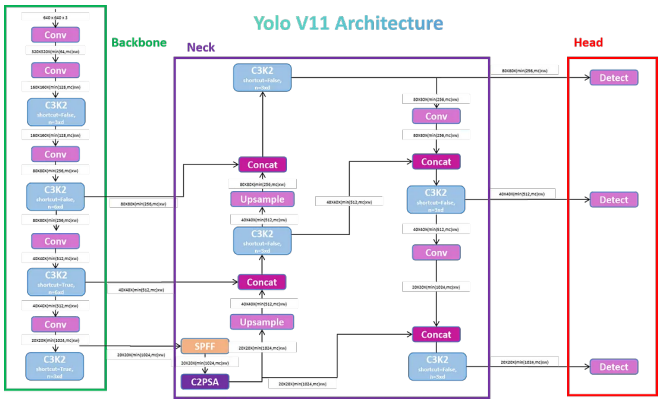

Figure 1. YOLO V11 Architecture [18].

C. Hardware setup and training process

The manual pre-processing task for the datasets mentioned in the previous sections was performed by the authors on multiple laptop and desktop computers. Training, validation and test of the corresponding YOLO models were performed on a desktop computer equipped with an AMD Ryzen 7 5800X processor (8 cores, 16 threads), 32 GB of DDR4 RAM, a 512 GB SSD, and an NVIDIA GPU GeForce RTX 3060 with 12GB of GDDR6 memory.

For all trained models, a batch size of 16 was adopted, and the input images were resized to 640×640 pixels, standard values for YOLOv11. The model was trained for 128 epochs, using the best weights from a preliminary model as a starting point. The initial learning rate was set to 0.01, with a final learning rate factor of 0.01. The optimizer was set to auto, allowing the framework to automatically select the most appropriate option for the architecture (typically SGD with momentum or Adam).

Additional default training parameters included a momentum of 0.937, a weight decay of 0.0005, and a warm-up phase of 3 epochs, with a warm-up momentum of 0.8 and a learning rate of 0.1 for biases during warm-up. No dropout was applied during training. The model was trained using 8 workers for data loading, and a fixed random seed (0) was used to ensure reproducibility. The total training time was approximately 5 hours and 17 minutes, with an average of 140 seconds per epoch. The final YOLOv11m model contained 303 layers and ±20 million parameters, occupying 39.7 MB on disk.

III. Results and Analyses

In this section, we will discuss the metrics obtained after training the full model and the measures taken to optimize it. The three metrics analyzed herein are commonly used in object detection and classification tasks.

- Precision: percentage of correct positive predictions out of all the instances the model labeled as positive. It reflects how well the model avoids generating false positive.
- Recall: The percentage of correct positives from all the true positives predictions. It is helpful to understand how well the model can detect all instance of a class.
- Mean Average Precision (mAP): The area under the precision and recall curve for every class.

Additionally, the Intersection over Union (IoU) metric is used in the performance assessment of the models. In object detection, where the goal is to accurately predict bounding boxes and compare them against the ground truth (original bounding boxes from the input images), IoU serves to evaluate the degree of overlap between predicted and corresponding true boxes. It is calculated as the area overlap between predicted and ground truth bounding boxes divided by the area of their union. In this work, we have set our IoU threshold at the default value of 70%. This means that for a predicted bounding box to be considered valid, its overlap with the true bounding box must be 70% or greater, otherwise the prediction is disregarded. In addition, a confidence limit of 0.25 was adopted during inference, which is the default value in the YOLOv11 framework. Detections with confidence scores below this value were discarded to reduce false positives.

The performance metrics presented in this work, including precision, recall, and mAP, were obtained from a single training and evaluation run, utilizing fixed data splits and a predefined random seed. Therefore, these results do not include statistical measures such as standard deviation or confidence intervals, which analysis are left for future studies.

While these point estimates provide a general indication of the model's effectiveness, we acknowledge the importance of evaluating metric variability across multiple runs. As future work, we intend to perform repeated experiments with varying random seeds or adopt k-fold cross-validation to derive more statistically robust performance indicators.

We initially evaluated the model on all classes listed in Table III and obtained the metrics shown in Table II.

Table II
METRICS FOR ALL CLASSES.

Metric	Precision	Recall	mAP
Value	61.8%	45.3%	40.7%

It can be observed that the metrics were suboptimal, primarily due to many classes being under-represented in our dataset. This situation can be easily identified in Table III which provides details on our dataset composition. The same problem occurs in other previous studies, but is often not made explicit. This issue occurs because in real-world scenarios, not all signs are equally probable to be found, making it difficult to sample them in sufficient images for effective training.

Due to this, and in order to demonstrate that well-represented classes can be efficiently detected and classified, the ten best represented classes from Table III were selected and an inference was performed only over them to assess the model performance. As it can be seen in Table IV, the performance metrics of the model for these specific traffic signs are substantially superior than those values presented in Table II and approximate those of previous studies mentioned in Section I-B.

Table III
DATASET CLASSES AND NUMBER OF INSTANCES.

Classe	Train.	Valid.	Test	Classe	Train.	Valid.	Test
R6A	859	216	107	A30C	3	1	1
R19	616	182	95	A47	3	1	0
R24A	380	117	66	A2A	3	1	0
R6C	358	91	56	R31	3	0	0
R6B	320	97	42	A24	2	3	1
R4A	154	32	15	R8A	2	1	1
R34	111	35	7	R35B	2	1	0
R32	89	31	19	A26B	2	0	3
R4B	87	25	10	R18	2	0	2
R1	71	15	6	A45	2	0	2
R5A	60	25	8	R21	2	0	0
R9	51	15	5	R17	2	0	0
R26	50	15	4	R11	2	0	0
R25D	47	19	9	A15	2	0	0
A14	37	13	4	R14	1	1	1
A30A	33	12	12	R8B	1	1	0
R15	31	4	6	A21C	1	1	0
A32B	30	11	8	R40	1	0	0
R25C	28	8	5	R39	1	0	0
R2	23	3	3	R38	1	0	0
R24B	21	2	1	R37	1	0	0
R3	19	5	4	R35A	1	0	0
A18	18	3	4	R30	1	0	0
R25B	15	11	2	R23	1	0	0
R25A	14	3	4	R22	1	0	0
R10	14	1	1	R16	1	0	0
A33A	13	2	1	R13	1	0	0
R33	13	1	3	A5B	1	0	0
A32A	12	4	3	A3B	1	0	0
R7	11	1	2	A34	1	0	0
R28	9	2	1	A20B	1	0	0
R29	8	0	0	A1B	1	0	0
A2B	6	2	2	A36	0	3	0
R36A	6	0	0	A43	0	2	0
A33B	5	2	1	A4A	0	1	0
R20	5	1	0	A3A	0	1	0
A12	5	1	0	A35	0	1	0
R27	5	0	0	A27	0	1	0
A37	4	1	1	A21E	0	1	0
R36B	4	1	0	A21D	0	1	0
R12	4	0	1	A30B	0	0	1
R5B	3	1	1	A17	0	0	1

Table IV
METRICS FOR THE TOP 10 MOST WELL-REPRESENTED CLASSES.

Metric	Precision	Recall	mAP
Value	84.3%	80.6%	85.8%

An illustration of actual and predicted traffic signs can be seen in Fig. 2. The trained model performed well on detecting traffic signs in different conditions, including variations in lighting (good and bad), distances (near and far) and image quality as can be seen in the figure.

Fig. 3 shows the Precision-Recall curve used to evaluate the model's performance in the ten best-represented classes in the dataset. The best-performing class is R19, with mAP50 equal to 96%, which, as shown in Table III, is the second best represented class. Furthermore, it shows that this class has precision and recall close to 1, indicating high model effectiveness for this category, i.e., low rates of false positives and false negatives. Conversely, class R5A has the worst performance among the ten selected traffic signs, due to its low frequency and visual similarity to other regulatory signs. In addition, signs with well-defined geometric shapes and high-contrast colors, such as R19 or R6A, tend to be better detected

Figure 2. Example of labeled (left) and prediced (right) traffic signs.

by the model due to their distinct visual patterns. On the other hand, small signs that contain a lot of text or are often partially obstructed in real conditions tend to have lower detection and classification accuracy.

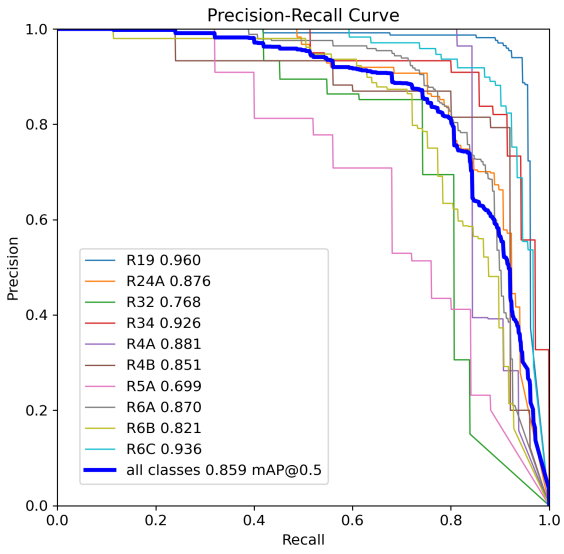

Figure 3. Precision × recall curve.

For comparison against YOLOv11, with the same training parameters, dataset and selecting the ten best represented classes, we trained a YOLOv8 model and analyzed it results. In our test, YOLOv8 achieved a precision of 58.8%, a recall rate of 32.1%, and mAP of 35.7%. Nonetheless, a deeper comparison among the performance of the different versions is left as perspective for future studies.

The analyses herein show that the representation of the classes has a great influence on the model's performance, pointing out, in the future, for an expansion of the dataset prioritizing the increase of samples of underrepresented classes, as to promote more balanced training and performance.

IV. Conclusions

This paper presents the performance of traffic sign detection and classification with an off-the-shelf YOLOv11 model. The obtained results, which are initial, show that the proposed approach presents good performance both in terms of accuracy and generalization capacity for scenes in different conditions, showing promise for future use in road monitoring and maintenance systems. As perspectives, we intend to expand the database with specific records of Brazilian traffic signs, including less common, damaged, or partially obstructed signs, improve statistical analyses and better optimize hyperparameters. In addition, strategies for optimizing inference time will be investigated aiming at applications on devices with limited computing capacity, e.g., edge computing devices. Other supervised learning techniques and adaptation to low visibility scenarios, such as nighttime or rainy conditions, are also foreseen as future research topics.

References

[1] World Health Organization, "Global status report on road safety 2018," 2018. [Online]. Available: https://www.who.int/violence_injury_prevention/road_safety_status/2018/en/

[2] Confederação Nacional do Transporte (CNT), "Acidentes Rodoviários e a Infraestrutura," 2018. [Online]. Available: https://cnt.org.br/acidentes-rodoviarios-infraestrutura

[3] Tzutalin, "labelImg: A graphical image annotation tool for creating bounding box annotations." https://pypi.org/project/labelImg/, 2015.

[4] M. Oliveira, "Metodologia para detecção e georreferenciamento de sinalização vertical de trânsito em vídeos gerados das vias de estradas rurais," in *Congresso Rio de Transportes*, 2023.

[5] R. Silva, B. Prado, L. N. Matos, F. Santo, C. Zanchettin, and P. Novais, "Construction of Brazilian Regulatory Traffic Sign Recognition Dataset," in *Lecture Notes in Computer Science*. Springer, 2021, pp. 163–172.

[6] Marcelo Franco Porto, Lucas Vinicius Ribeiro Alves, Renata Maria Abrantes Baracho Porto et al., "Metodologia para Detecção e Georreferenciamento de Placas de Sinalização em Vias Rurais," in *Congresso Rio de Transportes*, 2023.

[7] M. Flores-Calero, C. A. Astudillo, D. Guevara, J. Maza, B. S. Lita, B. Defaz, J. S. Ante, D. Zabala-Blanco, and J. M. Armingol Moreno, "Traffic Sign Detection and Recognition Using YOLO Object Detection Algorithm: A Systematic Review," *Mathematics*, vol. 12, no. 2, p. 297, Jan. 2024.

[8] J. Shen, Z. Zhang, J. Luo, and X. Zhang, "YOLOv5-TS: Detecting traffic signs in real-time," *Frontiers in Physics*, vol. 11, p. 1297828, Nov. 2023.

[9] H. Zhang, M. Liang, and Y. Wang, "YOLO-BS: a traffic sign detection algorithm based on YOLOv8," *Scientific Reports*, vol. 15, no. 1, Mar. 2025.

[10] W. Wojciechowski, L. M. Nowakowska, Z. G. Zajac, W. J. Kaczmarek, and I. Boz, "YOLOv7 versus YOLOv8: A comparative study on traffic sign detection accuracy in real-world images," in *8th Internat. Artif. Intellig. and Data Proc. Symp. (IDAP)*, 2024, pp. 1–5.

[11] A. Md. Mushfikur Rahman Shuvo, R. Dey, and M. Obaidur, "A yolo-based framework for road sign detection and recognition in the context of bangladesh," in *IEEE Internat. Conf. on Computing, Applications and Systems (COMPAS)*, 2024, pp. 1–6.

[12] B. C. e Xinwei Fan, "MSGC-YOLO: An improved lightweight traffic sign detection model under snow conditions," *Mathematics*, vol. 12, pp. 2227–7390, 2024.

[13] Ruixin Zhao, Sai Hong Tang, Jiazheng Shen, Eris Elianddy Bin Supeni, Sharafiz Abdul Rahim., "Enhancing autonomous driving safety: A robust traffic sign detection and recognition model TSD-YOLO," *Signal Processing*, vol. 225, 2024.

[14] Han, Y., Wang, F., Wang, W. et al., "YOLO-SG: Small traffic signs detection method in complex scene." *J. Supercomput.*, vol. 80, p. 2025–2046, 2024.

[15] FFmpeg team, "FFmpeg: a complete, cross-platform solution to record, convert and stream audio and video." http://ffmpeg.org/, May 2025.

[16] J. Terven, D.-M. Córdova-Esparza, and J.-A. Romero-González, "A Comprehensive Review of YOLO Architectures in Computer Vision: From YOLOv1 to YOLOv8 and YOLO-NAS," *Machine Learning and Knowledge Extraction*, vol. 5, no. 4, pp. 1680–1716, Nov. 2023.

[17] Ultralytics, "Ultralytics YOLOv11," Official Documentation and GitHub Repository, 2025. [Online]. Available: https://docs.ultralytics.com/models/yolo11/

[18] N. Rao, "YOLOv11 explained: Next-level object detection with enhanced speed and accuracy," 2025. [Online]. Available: https://medium.com/@nikhil-rao-20/