Received 11 September 2025, accepted 29 April 2026, date of publication 30 June 2026
www.computingonline.net / computing@computingonline.net

Print ISSN 1727-6209
Online ISSN 2312-5381
DOI 10.47839/ijc.25.2.4662

YOLO-RTLite: A TensorRT-Optimized Lightweight Detector for Real-Time Road Object Detection

MANJOT KAUR[1], RAJNEESH RANDHAWA[2], JAGROOP KAUR[3]
[1]Research Scholar, Department of Computer Science, Punjabi University, Patiala, India
[2]Associate Professor, Department of Computer Science, Punjabi University, Patiala, India
[3]Associate Professor, Department of Computer Science and Engineering, Punjabi University, Patiala, India
[1]manjotkaur1699@gmail.com, [2] rajneesh_randhawa@pbi.ac.in, [3]jagroop_ce@pbi.ac.in

Corresponding author: Manjot Kaur (e-mail: manjotkaur1699@gmail.com).

ABSTRACT This paper presents YOLO-RTLite, an optimized lightweight object detection model based on YOLOv4-Tiny. The proposed model addresses the challenges of accurate and reliable object detection in curved and blind road conditions, where visibility is limited and timely detection is critical. YOLO-RTLite is specifically engineered for real-time identification of critical road obstacles, focusing on vehicles and animals, to reduce road accidents and improving transportation safety. The YOLO-RTLite architecture incorporates an additional convolutional layer (with Mish activation function) and three novel skip connections to reinforce multi-scale feature propagation and feature reuse. Model performance is further improved through selective tuning of filter sizes and anchor boxes which enable better adaptation to diverse object dimensions. For real-time deployment, TensorRT-based acceleration is utilized to significantly increase inference speed. Performance results demonstrate that YOLO-RTLite achieves high precision (~95%), accuracy (~99%), robust recall (~98%), and efficient inference time (0.165 s) which is a significant improvement over its baseline model (YOLOv4 tiny). These performance stats support its suitability for safety-critical intelligent transportation systems.

KEYWORDS YOLOv4, object detection, vehicle detection, animal detection, optimization

I. INTRODUCTION

Object detection is critical for enhancing road safety by enabling intelligent systems to recognize and respond to hazards such as vehicles, pedestrians, and animals in real time [1]. It is central to Advanced Driver Assistance Systems (ADAS) and autonomous vehicles to improve braking, steering, and collision avoidance by reducing human error and reaction latency [2]. Deep learning–based vision models now outperform traditional approaches (Radar, Lidar based), offering high accuracy in classifying and tracking objects across a wide range of environments [3, 4]. Convolutional neural networks, when integrated with camera modules, provide a cost-effective alternative to LiDAR and radar, making real-time object detection viable on embedded hardware [5].

Among lightweight models, YOLOv4-Tiny is widely used for this purpose, offering real-time performance with acceptable accuracy on edge platforms like Raspberry Pi and Jetson Nano [6]. On the other hand, recent YOLO variants (e.g., YOLOv8 and newer versions) introduce architectural enhancements such as transformer modules and decoupled heads. These enhancements improve detection accuracy but often increase inference latency and memory requirements [7], [8]. All these limitations create room for further research.

While YOLOv4-Tiny supports real-time detection, its lightweight design reduces its ability to capture fine details. This affects detection of small and distant objects. In curved or blind road scenarios, vehicles and animals first appear small in the frame. Detecting them early is important because it gives more time to generate alerts and allows drivers to react safely. If detection happens late, when objects are closer and larger, the available reaction time becomes very limited. In addition, weak multi-scale feature fusion and bias toward larger objects further reduce accuracy in complex road environments. To address these issues, the Yolov4-Tiny model was modified to improve feature learning and enable earlier and more reliable detection while maintaining real-time performance. YOLOv4-Tiny was selected due to its lightweight design and proven suitability for real-time detection on edge devices. In contrast, newer YOLO variants introduce more complex architectures to improve accuracy, which may increase computational requirements and affect real-time deployment.

The goal of this work is to develop a lightweight and efficient object detection model that improves detection accuracy while maintaining low computational complexity for

real-time road safety applications. To achieve this, we propose YOLO-RTLite, an optimized architecture derived from YOLOv4-Tiny. The model enhances feature extraction and multi-scale feature fusion through structural modifications and maintains efficiency. Furthermore, TensorRT-based optimization is applied to reduce inference latency. As a result, YOLO-RTLite achieves real-time inference without requiring quantization, making it suitable for deployment in resource-constrained environments.

II. RELATED WORK

YOLO architectures are widely adopted for real-time object detection due to their balance of speed and accuracy, making them suitable for accident avoidance systems. YOLOv5, YOLOv8, and YOLOv4 have each demonstrated their strengths: YOLOv5 is widely used for detecting vehicles, pedestrians, and traffic signs [3], YOLOv8 supports crowd monitoring in urban environments [9], and YOLOv4 supports real-time detection and distance estimation, which is useful for collision prevention [6]. Different YOLO models show a trade-off between speed and accuracy. YOLOv8 generally achieves higher accuracy, whereas YOLOv4-Tiny is more suitable for systems with limited computational resources. Recent variants, such as YOLO-NAS and improved YOLOv8-based models, further enhance performance, with F1 scores up to 0.74 and mAP@0.5 values reaching 0.782 [10-12]. Deep learning-based detection is widely used in autonomous driving for recognizing different road elements under varying conditions [13]. It has been applied in areas such as pothole detection [14], advanced driver assistance systems (ADAS) [15], and low-speed collision prevention [17]. Studies comparing detection models show that CNN-based approaches are more suitable for real-time applications. In contrast, R-CNN and Faster R-CNN models provide higher accuracy but are better suited for controlled or synthetic environments [14, 16]. SSD models achieve lower latency (around 30 ms per image) with reasonable accuracy (mAP 88.998%). Faster R-CNN, on the other hand, achieves higher accuracy (mAP 94.32%) but requires more computational resources [5]. These observations help in selecting an appropriate model based on system requirements such as processing power, real-time constraints, and operating conditions.

Multimodal object detection methods combine inputs from cameras, LiDAR, and radar to improve detection accuracy and robustness. Camera-based systems are often preferred because they are cost-effective and work well using deep learning-based tracking methods [2, 5]. Hybrid approaches have also shown promising results. For example, combining HOG-SVM with Faster R-CNN [15] or integrating ultrasonic sensors with CNN models [18] can improve obstacle detection and avoidance in both industrial and vehicle-based applications.

Some systems use geometric features for tasks such as lane departure warnings [19]. LiDAR-based methods, on the other hand, provide accurate 3D spatial information, which helps better understand the environment [20]. Overall, these sensor fusion techniques improve performance in low-visibility conditions and help handle challenges such as occlusion and overlapping objects. YOLOv4 follows a one-stage architecture, which supports real-time processing. Extensions such as the OSDH-Head further improve its performance [21]. The model has achieved speed of up to 60 FPS on NVIDIA Tesla K80 GPUs [22]. For embedded systems, lightweight models like SSD-MobileNet and YOLOv4-Tiny have shown good performance on devices such as Raspberry Pi and Jetson Nano [23, 24]. In addition to this, improved training strategies address challenges caused by environmental factors, including glare and low lighting [25, 26].

For pedestrian safety, combining YOLOv4 with DeepSORT enables object tracking and movement prediction [27]. Deep learning methods are also used for driver behavior monitoring and accident analysis [28-30]. Despite these advancements, real-time 3D object detection still faces issues such as occlusion, scale variation, and delays in inference [31]. In Intelligent Transportation Systems (ITS), latency is being reduced by integrating machine learning models with IoT platforms and vehicular ad-hoc networks (VANETs), enabling faster response and improved accident prevention [32].

Recent studies also highlight improvements in newer YOLO versions for resource-constrained environments. YOLOv9-Gelan and YOLOv10 have achieved mAP values of up to 93.5% and 90.8%, with inference times below 12 ms [33]. Optimized versions of YOLOv5 to YOLOv8 have demonstrated real-time performance on edge platforms such as Jetson Nano and Orin, with frame rates ranging from 5.67 FPS to 65.56 FPS depending on hardware and resolution [34]. TensorRT-optimized YOLOv5-Nano and YOLOv6-Tiny have achieved around 28.6 FPS on compact devices [35]. In addition, a lightweight Mini-YOLOv4-Tiny model showed nearly twice the inference speed of YOLOv4-Tiny while reducing parameters by 37% [36]. These results support the use of efficient and lightweight models, such as YOLO-RTLite, for real-time embedded transportation safety systems.

III. PROPOSED METHODOLOGY

This sections focuses on developing an efficient and lightweight object detection framework, named YOLO-RTLite, for real-time road safety applications. The model is derived from YOLOv4-Tiny and is specifically optimized to improve feature extraction and multi-scale representation while maintaining low computational complexity. In addition to architectural modifications, inference time is improved using TensorRT-based optimization. The overall system is designed to enable early and reliable detection of vehicles and animals in challenging scenarios such as curved or blind roads.

The proposed approach is structured into three key stages: (i) dataset preparation, (ii) architectural modification of the base model, and (iii) TensorRT-based optimization for real-time inference.

A. DATASET PREPARATION

This section describes the preparation of the custom dataset.

The dataset was created using a subset of the COCO dataset along with approximately 5000 additional images collected from Kaggle. Only images containing objects relevant to the target application were selected, covering 13 classes. The dataset is divided as follows:

Training Set: 76,672 images

Augmented Training Set: 80,878 images (after augmentation)

Validation Set: ~6,000 images

Testing Set: ~11,000 images

Object Classes

The dataset includes the following 13 classes: Person, Car, Bicycle, Motorcycle, Truck, Bus, Cat, dog, Elephant, Cow, Sheep, Horse, and SUV (Sports Utility Vehicle). Car, Bicycle, Motorcycle, and Truck classes each exceed 9,000 samples, while the animal classes help introduce environmental diversity.

Data Augmentation

To improve generalization, augmentation was applied using the imgaug library. The following transformations were used: horizontal flip (50% probability), rotation within ±20°, scaling

in the range of 0.8–1.2 along both axes, shearing up to ±10°, and translation up to ±50 pixels in both x and y directions. Brightness and contrast were also adjusted (factor 0.3).

These augmentations were applied to account for variations commonly observed in road scenes, such as changes in lighting, object orientation, and position.

Noise and lossy compression were not used in this work, as the focus was mainly on geometric transformations and basic photometric adjustments. These effects can be explored in future work to further improve model robustness.

Image Resizing: The model was trained using the Darknet framework, the original implementation of YOLOv4-Tiny. In this setup, images are resized to a fixed input size of 416 × 416. Letterbox resizing is used to preserve the aspect ratio. Padding is added where required to prevent distortion.

B. ARCHITECTURE OF YOLORT-Lite

Figure 1 shows the architecture of YOLO-RTLite. In the diagram CBL stands for Convolution + Batch Normalization + Leaky ReLU block and CSPDarknet stands for Cross stage Partial Darknet.These are basic building block in YOLO architectures. CONV is used for convolution layer in the diagram.

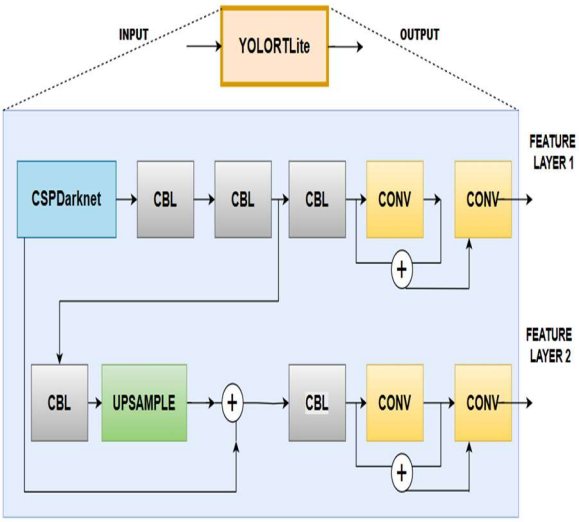

Figure 1. Architecture: YOLORTLite

The modifications in YOLO-RTLite are based on the conducted research. A convolutional layer using the Mish activation function was added to increase the network's capacity for learning fine-grained features. This benefits the detection of small or partially occluded objects [37], albeit with a slight increase in computational cost.

The number of route layers was increased from 6 to 9. Increasing the number of route layers enhances multi-scale feature reuse and improves contextual understanding, as demonstrated in feature fusion models [39]. Anchor masks were modified from mask=3,4,5 to mask= 0,1,2. The anchor masks are also adjusted from larger to smaller-scale priors (mask = 0,1,2). This improves alignment with smaller object sizes and enhances detection accuracy [38]. In addition to this, corresponding filter sizes were also updated to match the new anchor configuration.

Training Parameter Optimization

Batch & Subdivision: Increasing subdivision allows to reduce memory usage during training without lowering batch size.

Learning Rate Tuning: A higher learning rate improved convergence speed and helped with more accurate bounding box predictions.

Table 1 Training parameter optimization

Parameter	Original	Optimized
Batch/Subdivision	64 / 1	64 / 8
Learning Rate	0.00261	0.00522

C. MODEL OPTIMIZATION WITH TENSOR-RT

After training, the YOLO-RTLite model was optimized for deployment using NVIDIA TensorRT. TensorRT-based optimization introduces techniques such as layer fusion and precision calibration, significantly reducing inference latency while maintaining accuracy, making the model suitable for deployment in resource-constrained environments [40].

The process involved: Applying optimizations such as: Layer fusion, Precision calibration (e.g., FP16), Memory and kernel tuning. These optimizations significantly reduced inference latency, making the model suitable for deployment on edge devices where real-time detection is critical.

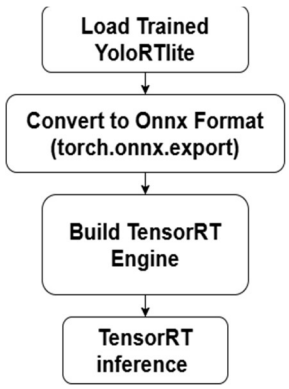

Figure 2. Tensor Rt Optimization

IV. PERFORMANCE EVALUATION METRICS

The effectiveness of the optimized YOLOv4 model was assessed using four key performance metrics: recall, precision, accuracy, and inference time.

1) Recall (Sensitivity / True Positive Rate)

Recall measures the model's ability to correctly detect objects present in an image. It is calculated as:

$$\text{Recall} = TP / (TP+FN),$$

where:

TP (True Positives) = correctly detected objects

FN (False Negatives) = missed objects

A high recall value indicates that the model successfully detects most objects, minimizing missed detections.

2) Precision (Positive Predictive Value)

Precision measures how accurate the model's detections are. It shows how many of the predicted objects are correctly classified. It is defined as:

$$\text{Precision} = TP / (TP+FP),$$

where:

TP(True Positives) = correctly detected objects

FP (False Positives) = incorrectly detected objects

Higher precision implies that the model generates less false detection, ensuring reliable object identification.

3) Accuracy

Accuracy represents the overall correctness of the model's predictions by considering both correct positive and negative detections. It is computed as:

$$\text{Accuracy} = (TP+TN) / (FP+FN+TP+TN),$$

where:

TN or True Negatives = correctly identified non-object regions

A higher accuracy score indicates the model's effectiveness in distinguishing between objects and background noise.

4) Inference Time (Computational Efficiency)

Inference time measures the total time required for the model to process an image, including: Preprocessing (image resizing and normalization), Model inference (forward pass through the network) and Post-processing (Non-Maximum Suppression and bounding box refinements).

V. EXPERIMENTAL SETUP

The model was trained and optimized using a system with the configuration details enlisted in Table 2.

Table 2. Training Environment

Component	Version / Specification
GPU	NVIDIA RTX 4090
OpenCV-Python	4.7.0.72
PyTorch	2.0.1 (CUDA-enabled)
ONNX	1.14.0
imgaug	0.4.0
numpy, pandas, tqdm	Common CV stack

VI. RESULTS

A. QUANTITATIVE RESULTS

This Section presents the performance results of YOLO-RTLite. The model shows strong performance indicating its potential for real-time road safety applications.

Table 3. Performance Results

Parameter	Precision	Accuracy	Recall	Time
value	95.11	99.20	98.23	0.0165 seconds

The model was trained for 100k iterations to avoid overfiitting and beyond this point the model's performance gains were marginal. Figure 3 shows the loss versus the trend of iterations. The training loss decreased steadily over time, indicating effective learning and convergence of the YOLOv4-Tiny model. Early epochs show rapid loss reduction, while later epochs reflect gradual fine-tuning. By the final stages, the loss stabilized at a low value, aligning with the model's high accuracy and recall.

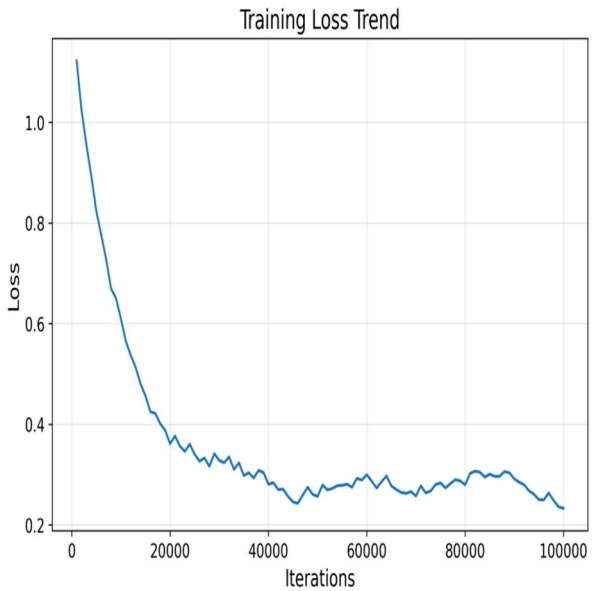

Figure 3.YOLO-RTLite Loss trend over iterations

Figure 4 shows accuracy and recall versus the trend of iterations.

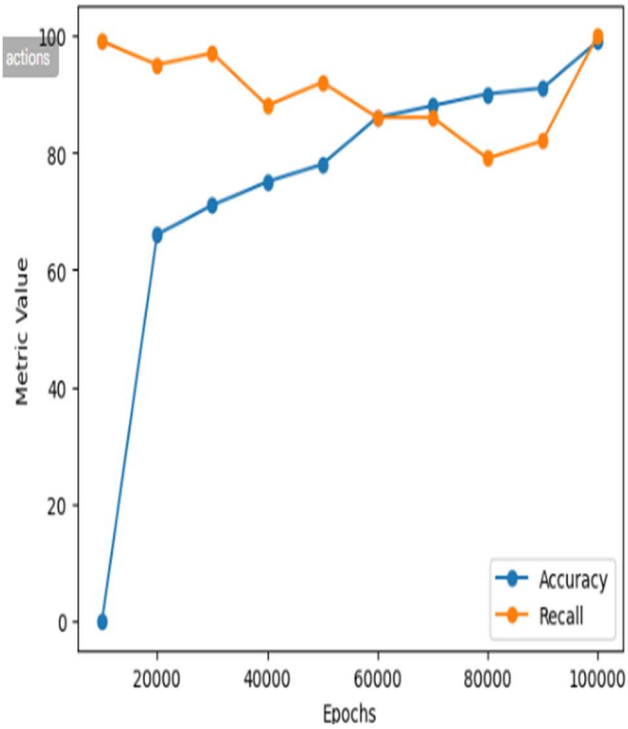

Figure 4. YOLO RTLIte : Accuracy and Recall.

B. COMPARATIVE ANALYSIS

The comparison includes detection accuracy mAP@0.5), recall, speed (inference time), and how well each model works on small devices. These results show that Yolo-RtLite is well suited for real time use on edge devices and embedded systems.

Table 4 compares YOLO-RTLite with several well-known YOLO models.

Table 4. Comparison of various YOLO variants

MODEL	mAP @0.5 (%)	Inference Time (s)	Deployment Suitability	Reference
YOLO-NAS-S	72	~0.0071	Edge / Cloud	[41]
YOLOv4-Tiny	55.4	~0.0045	Jetson Nano	[42]
YOLOv5s	~70.0	~0.006–0.007	Embedded, Jetson Nano	[43]
YOLOv7-Tiny	~56.0	~0.0045	Jetson / Embedded	[43]
YOLOv8s	84.47	–	UAV simulation	[42]
YOLOv9-GELAN-s	93.5	~0.0115	Edge (after pruning)	[42]
YOLOv11-nano	95.7	~0.0489	Embedded	[43]
YOLO-RTLite	**95.11**	**0.0165**	**Optimized for Embedded**	***This work***

B. ABLATION STUDY

The ablation study evaluates the contribution of the proposed modifications. The addition of convolutional layers improves feature representation, increasing mAP from 71.43% to 82.85% and recall from 93.96% to 95.78%. Further improvement is observed with route layer expansion, which enhances multi-scale feature fusion and leads to the final YOLO-RTLite performance (mAP 95.12%, recall 98.32%).

The addition of convolutional layers slightly increases inference time. However, TensorRT optimization reduces the latency to 0.0165 seconds. This enables efficient real-time performance.

Table 5. Ablation Study Results

Model Variant	New Conv Layers	Route layers expansion	mAP @0.5 (%)	Recall (%)	Inference Time (s)
Baseline	✗	✗	71.43	93.96	1.7254
Intermediate Stage	✓	✗	82.85	95.78	1.75
YOLO-RTLite	✓	✓	95.11	98.23	0.0165

D. VISUAL RESULTS: YOLO-RTLite

Figure 5 shows sample detection outputs of YOLO-RTLite involving vehicles and animals.

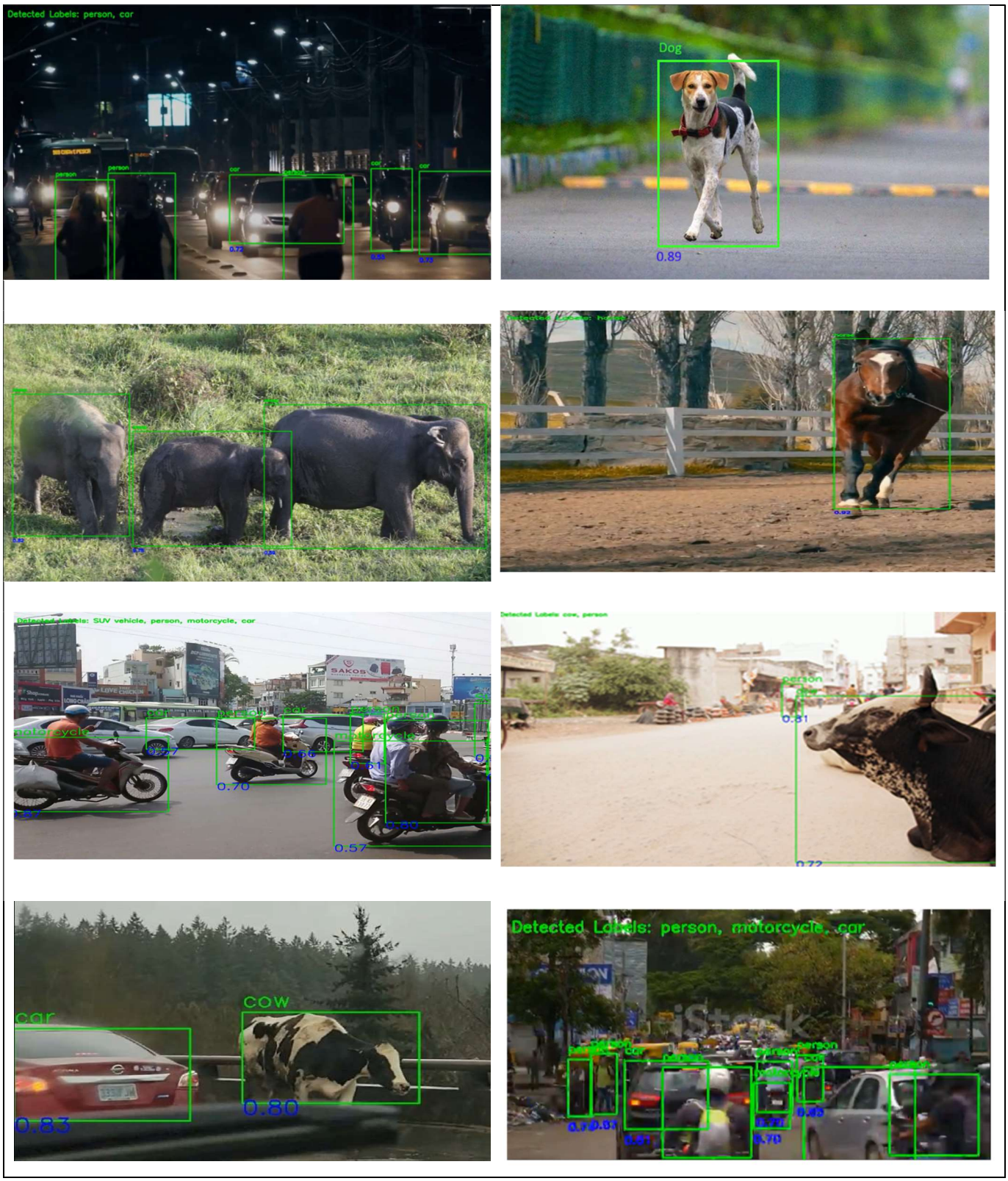

Figure 5 Detection Snapshot

VII. CONTRIBUTIONS OF THE WORK

Architecture Refinement:

YoloRT-lite, an optimised version of YOLOv4-, is developed with structural changes. It includes three additional skip connections to improve feature propagation. These changes help preserve spatial and contextual information across layers. As a result, detection accuracy improves, especially for small and occluded objects, while keeping the model lightweight.

TensorRT-Based Optimization:

The model is exported to ONNX and converted into a TensorRT engine. FP16 precision and layer fusion are applied for faster inference. This reduces the inference time to 0.0165s.

Performance Improvement:

YOLO-RTLite outperforms the baseline YOLOv4-Tiny. It achieves 99.20% accuracy, 95.11% mAP@0.5, and 98.23% recall. The inference time is reduced from 1.73 s to 0.0165 s.

Applicability:

The model is lightweight and fast. It is suitable for real-time road safety applications. It can detect vehicles, animals, and pedestrians under challenging visibility conditions.

VIII. LIMITATIONS AND FUTURE WORK

Although YOLO-RTLite achieves high accuracy and speed, several areas require further study:

Generalization: The dataset, derived from COCO and Kaggle, may not fully represent real-world diversity in lighting, weather, or camera perspectives. Broader datasets will improve domain adaptability.

Complex Scenes: The model's robustness in dense or occluded environments requires further evaluation.

Hardware Testing: Real-world validation on embedded and low-power platforms is pending; future work will include pruning and quantization for improved scalability.

IX. CONCLUSION

YOLO-RTLite combines architectural changes and TensorRT-based optimization to enhance object detection performance, particularly in terms of inference speed and Precision The optimized model achieved an inference time of 0.0165 seconds, indicating its potential suitability for real-time applications. These results suggest that the model may be useful in safety-related scenarios where fast and accurate object detection is required. Further validation in real-world conditions and broader object category support would be necessary for wider deployment.

References

[1] I. J. Daisy, A. Arun, J. Emmanual J. Antony & K. B. Muhilan, "Smart vehicle assistance and accident prevention system," *International Research Journal on Advanced Engineering Hub*, vol. 3, no. 3, pp. 412-419, 2025. https://doi.org/10.47392/IRJAEH.2025.0056.

[2] J. Negrini de Araujo, V. Palade, T. Sedighi & A. Daneshkhah, "Improving the pedestrian detection performance in the absence of rich training datasets: A UK case study," *Advances in Artificial Intelligence and Machine Learning*, vol. 2, no. 1, pp. 315-337, 2022. https://doi.org/10.54364/AAIML.2022.1121.

[3] G. Öztürk, O. Eldoğan & R. Köker, "Computer vision-based lane detection and detection of vehicle, traffic sign, pedestrian using YOLOv5," *Sakarya University Journal of Science*, vol. 28, no. 2, pp. 418-430, 2024. https://doi.org/10.16984/saufenbilder.1393307.

[4] M. Zaman, S. Saha, N. Zohrabi & S. Abdelwahed, "Deep learning approaches for vehicle and pedestrian detection in adverse weather," *Proceedings of the 2023 IEEE Transportation Electrification Conference and Expo (ITEC 2023)*, Detroit, USA, 21-23 June 2023, pp. 1-6. https://doi.org/10.1109/ITEC55900.2023.10187020.

[5] D. R. Niranjan, B. C. Vinaykarthik & Mohana, "Performance analysis of SSD and Faster RCNN multi-class object detection model for autonomous driving vehicle research using CARLA simulator," *Proc. IEEE International Conference on Electrical, Computer and Communication Technologies (ICECCT) 2021*, 2021, pp. 497. https://doi.org/10.1109/ICECCT52121.2021.9616712.

[6] R. Prabha, M. Sanjana, S. Harshavarthanan, A. Premika, and S. S. Sarveshwaar, "Collision avoidance system in autonomous vehicles using YOLOv4," *Proceedings of the IEEE International Conference on Communication and Information & Robotics Technology (ICCIRT)*, pp. 107–111, 2024. https://doi.org/10.1109/ICCIRT59484.2024.10921869.

[7] S. Arshad, M. Sualeh, D.-H. Kim, N. Dinh, and G.-W. Kim, "Clothoid: An integrated hierarchical framework for autonomous driving in a dynamic urban environment," *Sensors*, vol. 20, issue 18, article 5053, 2020. https://doi.org/10.3390/s20185053.

[8] W. Rassameepaiboon and W. Assawinchaichote, "Vehicle avoidance reaction by two-step motion flow cluster," *Proceedings of the International Computer Science and Engineering Conference (ICSEC)*, pp. 1–5, 2018. https://doi.org/10.1109/ICSEC.2018.8712678.

[9] D. Sudharson, J. Srinithi, S. Akshara, K. Abhirami, P. Sriharshitha, and K. Priyanka, "Proactive headcount and suspicious activity detection using YOLOv8," *Procedia Computer Science*, vol. 230, pp. 61-69, 2023. https://doi.org/10.1016/j.procs.2023.12.061.

[10] S. A. Dhinakar Raj, A. Singh, M. Muthanna, K. Sivasankaran, and T. S. Shekhawat, "Collision avoidance system using YOLO-based object detection and distance estimation," *Proceedings of the 3rd International Conference on Intelligent Data Communication Technologies and Internet of Things (IDCIoT)*, pp. 178–184, 2025. https://doi.org/10.1109/IDCIOT64235.2025.10915167.

[11] A. N. M. Nusari, I. Y. Ozbek and E. A. Oral, "Automatic vehicle accident detection and classification from images: A comparison of YOLOv9 and YOLO-NAS algorithms," *Proceedings of the 2024 32nd Signal Processing and Communications Applications Conference (SIU)*, pp. 1–4, 2024. https://doi.org/10.1109/SIU61531.2024.10600761.

[12] N. Choudhary, R. Sharma, D. Upadhyay, A. Verma and V. Jain, "Enhanced traffic sign recognition using advanced YOLOv8 model," *Proceedings of the 2024 4th International Conference on Intelligent Technologies (CONIT)*, Bangalore, India, 2024, pp. 1-4. https://doi.org/10.1109/CONIT61985.2024.10626450.

[13] A. Albouchi, M. A. Hajjaji and A. Mtibaa, "Deep learning-based object detection approach for autonomous vehicles," *Proceedings of the 2022 IEEE International Conference on Secure Trust and Data Management Applications (STA)*, Sousse, Tunisia, 2022, pp. 89-94. https://doi.org/10.1109/STA56120.2022.10018993.

[14] M. Jakubec, E. Lieskovská, B. Bučko and K. Zábovská, "Comparison of CNN-based models for pothole detection in real-world adverse conditions: Overview and evaluation," *Applied Sciences*, vol. 13, no. 9, article 5810, 2023. https://doi.org/10.3390/app13095810.

[15] M. Shirpour, N. Khairdoost, M. A. Bauer and S. S. Beauchemin, "Traffic object detection and recognition based on the attentional visual field of drivers," *IEEE Transactions on Intelligent Vehicles*, vol. 8, no. 1, pp. 594–604, 2023. https://doi.org/10.1109/TIV.2021.3133849.

[16] P. Goswami & A. B. M. A. Hossain, "Street object detection from synthesized and processed semantic image: A deep learning-based study," *Human-Centric Intelligent Systems*, vol. 3, pp. 487–507, 2023. https://doi.org/10.1007/s44230-023-00043-1.

[17] H. Wang, W. Niu, Y. Liu, Y. Tan & C. Xiu, "Low-speed collision prevention warning for intelligent vehicles based on digital twin and deep learning," *Proceedings of the 2024 IEEE International Conference on Digital Systems and Intelligent Systems (ICDSIS)*, Hassan, India, 2024, pp. 1-7. https://doi.org/10.1109/ICDSIS61070.2024.10594607.

[18] Y. P. Yong, S. J. Lee, Y. H. Chang, K. H. Lee, S. W. Kwon, C. S. Cho & S. W. Chung, "Object detection and distance measurement algorithm for collision avoidance of precast concrete installation during crane lifting process," *Buildings*, vol. 13, no. 10, article 2551, 2023. https://doi.org/10.3390/buildings13102551.

[19] N. Devulapalli, V. Matus, E. Eso, Z. Ghassemlooy & R. Pérez-Jiménez, "Lane-cross detection using optical camera-based road-to-vehicle communications," in *Proceedings of the 2021 17th International Symposium on Wireless Communication Systems (ISWCS)*, pp. 240–244, 2021. https://doi.org/10.1109/ISWCS49558.2021.9562196.

[20] P. V., S. Dhanya, "3DYOLO: Real-time 3D object detection in 3D point clouds for autonomous driving," *Proceedings of the International Geoscience and Remote Sensing Symposium (IGARSS)*, Ahmedabad,

India, 2021, pp. 41-44. https://doi.org/10.1109/InGARSS51564.2021.9791912.

[21] Y. Liu, Y. Shi, "VRU-YOLO: A small object detection algorithm for vulnerable road users in complex scenes," *IEEE Access*, vol. 13, pp. 19996-20015, 2025. https://doi.org/10.1109/ACCESS.2025.3534321.

[22] A.R. Pathak, A.C. Elster, "Applying transfer learning to traffic surveillance videos for accident detection," *Proceedings of the International Conference on Applied Artificial Intelligence (ICAPAI)*, Halden, Norway, 2022, pp. 1-7. https://doi.org/10.1109/ICAPAI55158.2022.9801568.

[23] J. Nine, R. Mathavan, "Traffic light and back-light recognition using deep learning and image processing with Raspberry Pi," *Engineering Science and Synthetic*, vol. 8, no. 2, 2021. https://doi.org/10.14464/ess.v8i2.490.

[24] K. Sarvajcz, L. Ari, J. Menyhrt, "AI on the road: NVIDIA Jetson Nano-powered computer vision-based system for real-time pedestrian and priority sign detection," *Applied Sciences*, vol. 14, 1440, 2024. https://doi.org/10.3390/app14041440.

[25] T. Winkle, C. Erbsmehl, K. Bengler, "Area-wide real-world test scenarios of poor visibility for safe development of automated vehicles," *European Transport Research Review*, vol. 10, p. 55, 2018. https://doi.org/10.1186/s12544-018-0304-x.

[26] N.K. Ragesh, R. Rajesh, "Pedestrian detection in automotive safety: Understanding state-of-the-art," *IEEE Access*, 2019. https://doi.org/10.1109/ACCESS.2019.2909992.

[27] A. Paiboonrattanakorn, S. Phimoltares, "Neural computing-based pedestrian detection from image sequence," *Int. Jt. Conf. Comput. Sci. Softw. Eng.*, vol. 7, pp. 47864-47890, 2019. https://doi.org/10.1109/JCSSE.2012.6261921.

[28] M.A.A. Hammoudeh, M. Alsaykhan, R. Alsalameh, N. Althwaibi, "Computer vision: A review of detecting objects in videos challenges and techniques," *Int. J. Online Eng.*, vol. 18, issue 1, 2022. https://doi.org/10.3991/ijoe.v18i01.27577.

[29] N. Yadav, U. Thakur, A. Poonia, R. Chandel, "Post-crash detection and traffic analysis," *Int. Conf. Signal Process. Integr. Netw. (SPIN)*, Noida, India, 2021, pp. 1092-1097. https://doi.org/10.1109/SPIN52536.2021.9565964.

[30] M.I.B. Ahmed, R. Zaghdoud, M.S. Ahmed, R. Sendi, S. Alsharif, J. Alabdulkarim, B.A.A. Saad, R. Alsabt, A. Rahman, G. Krishnasamy, "A real-time computer vision based approach to detection and classification of traffic incidents," *Big Data Cogn. Comput.*, vol. 7, issue 1, 22, 2023. https://doi.org/10.3390/bdcc7010022.

[31] A. Pravallika, M.F. Hashmi, A. Gupta, "Deep learning frontiers in 3D object detection: A comprehensive review for autonomous driving," *IEEE Access*, vol. 12, pp. 173936-173980, 2024. https://doi.org/10.1109/ACCESS.2024.3456893.

[32] N. Murali, D. Beulah David, "Advanced VANET technology for emergency vehicle safety in time-critical scenario," *Proceedings of the International Conference on Smart Technologies in Electrical Vehicles, Communication & Computing Control (ICONSTEM)*, Chennai, India, 2023, pp. 1-8. https://doi.org/10.1109/ICONSTEM56934.2023.10142332.

[33] R. Sapkota, Z. Meng, M. Churuvija, X. Du, Z. Ma, M. Karkee, "Comprehensive performance evaluation of YOLOv12, YOLO11, YOLOv10, YOLOv9 and YOLOv8 on detecting and counting fruitlet in complex orchard environments," 2024. https://doi.org/10.36227/techrxiv.172954111.18265256/v2.

[34] Z. E. Kaymakcı, M. Akarsu, C. Öztürk, "Multiple small-scale object detection in aerial vehicle images using standard or optimized YOLO detectors," *Proceedings of the International Symposium on Innovations in Intelligent Systems and Applications (INISTA)*, 2023, pp. 1-5. https://doi.org/10.1109/INISTA59065.2023.10310562.

[35] A. Bulut, F. Ozdemir, Y. S. Bostanci, M. Soyturk, "Performance evaluation of recent object detection models for traffic safety applications on edge," *Proceedings of the IPMV Conference*, 2023, pp. 1-6. https://doi.org/10.1145/3582177.3582178.

[36] D.-S. Bacea, F. Oniga, "Single stage architecture for improved accuracy real-time object detection on mobile devices," *Image and Vision Computing*, vol. 130, p. 104613, 2023. https://doi.org/10.1016/j.imavis.2022.104613.

[37] M. Li, C. Li, S. Wang, W. Xie, "Efficient WFFA-SSD with weighted-feature fusion block and lightweight design for small object detection," *Scientific Reports*, vol. 13, no. 1, 9883, 2023. https://doi.org/10.1038/s41598-023-36972-x.

[38] Q. Zhong, Y. Liu, L. Ma, Y. Ma, "Anchor box optimization for object detection," *Proceedings of the IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2020, pp. 1325-1333. https://doi.org/10.1109/WACV45572.2020.9093351.

[39] M. Tan, R. Pang, Q. V. Le, "EfficientDet: Scalable and efficient object detection," arXiv preprint arXiv:1911.09070, 2019. https://arxiv.org/abs/1911.09070.

[40] NVIDIA Corporation, *TensorRT Developer Guide*, NVIDIA Developer, 2019. [Online]. Available at: https://docs.nvidia.com/deeplearning/tensorrt/developer-guide.

[41] J. Terven, D.-M. Córdova-Esparza, 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, 2023. https://doi.org/10.3390/make5040083.

[42] Z.E. Kaymakcı, M. Akarsu, C. Öztürk, "Multiple small-scale object detection in aerial vehicle images using standard or optimized YOLO detectors," *Proceedings of the International Symposium on Innovations in Intelligent Systems and Applications (INISTA)*, Hammamet, Tunisia, 2023, pp. 1-5. https://doi.org/10.1109/INISTA59065.2023.10310562.

[43] A. Bulut, F. Ozdemir, Y.S. Bostanci, M. Soyturk, "Performance evaluation of recent object detection models for traffic safety applications on edge," *Proceedings of the International Conference on Image Processing and Machine Vision (IPMV)*, 2023, pp. 1-6. https://doi.org/10.1145/3582177.3582178.

***MANJOT KAUR** received her M.Tech degree in Computer Engineering and is currently pursuing a Ph.D. in Computer Science. Her research interests include computer vision, deep learning, and wireless sensor networks. Her recent work explores the integration of YOLO-based object detection models with network simulation frameworks such as OMNeT++ and MQTT for smart mobility applications.*

***Dr. RAJNEESH RANDHAWA** holds a Ph.D. in Computer Science and is currently serving as an Assistant Professor in the Department of Computer Science at Punjabi University, Patiala, Punjab, India. She has over twenty years of teaching and academic experience. Her research interests lie in the areas of Wireless Sensor Networks (WSN), computer networks, and optimization techniques.*

Dr. Randhawa has guided numerous postgraduate and research students and has contributed to several national and international publications.

***Dr. JAGROOP KAUR** is an Associate Professor in the Department of Computer Science at Punjabi University, Patiala, Punjab, India. She has extensive teaching and research experience, with primary interests in cloud computing and computer networking. Dr. Kaur has published several papers in reputed journals and has guided postgraduate and doctoral students in related fields. She has over 20 years of teaching and research experience.*