RHD-RTDETR:A Cross-Gradient Lightweight Fatigue Driving Detection Algorithm

Qi Liu, and Yang Xu*

Abstract—**This paper introduces RHD-RTDETR, a lightweight fatigue driving detection algorithm designed to address the computational complexity and limited robustness of existing RTDETR algorithms in driving environments. By combining cross-stage gradient optimization with dynamic feature reconstruction, the approach achieves model compression and optimized feature representation, significantly lowering computational demands without compromising detection accuracy. The encoder incorporates the Hilo attention mechanism and an intra-scale feature interaction module, which refine the processing of both high- and low-frequency features, thus enhancing the model's ability to capture relevant patterns in complex scenarios. The parallel use of dilated and large-kernel convolutions further strengthens the network's ability to detect sparse patterns and long-range pixel dependencies. Additionally, structural reparameterization is used to reduce inference overhead effectively. Experimental results show that RHD-RTDETR enhances average fatigue driving detection accuracy by 0.6% while reducing parameter count and computational load by 39.2% and 37%, respectively, demonstrating superior performance with strong anti-interference capability and stability.**

Index Terms—**Dynamic Convolution RepConv; Hilo attention; Lightweight; Fatigue detection; RTDETR**

I. Introduction

WITH the rapid growth of the global economy, the increasing number of vehicles and drivers has led to a significant rise in traffic incidents worldwide. This trend poses considerable safety risks to drivers, passengers, and other road users. Fatigue driving is widely recognized as a major factor contributing to severe accidents [1], as it impairs key abilities such as reaction time, decision-making, and focus—critical elements for responding to sudden road events. As a result, developing effective fatigue detection systems is crucial for ensuring road safety [2]. Recent advancements in computer vision and deep learning [3] have notably improved fatigue detection algorithms based on facial feature analysis. Traditional methods, such as Histogram of Oriented Gradients (HOG) [4] and Deformable Parts Models (DPM) [5], suffer from limited generalization across different environments and target variations, which diminishes their detection performance. In contrast, Convolutional Neural Network (CNN)-based models, starting with AlexNet [6], can be categorized based on the detection approach. Two-stage methods, such as R-CNN [7], Fast R-CNN [8], Faster R-CNN [9], and Mask R-CNN [10], offer high accuracy but are hindered by computationally demanding region proposal steps. Single-stage approaches, including the YOLO series [11] and SSD [12], provide faster inference but suffer from excessive redundant bounding boxes, which can negatively impact accuracy. In 2020, Facebook introduced DETR [13], a Transformer-based model that simplifies the detection pipeline by removing the need for anchor boxes and Non-Maximum Suppression (NMS). However, DETR is limited by slow inference speeds and high computational costs. To address these issues, Zhao et al. (2023) introduced RT-DETR [14], the first real-time end-to-end Transformer detection model, which incorporates intra-scale feature interaction (AIFI) and cross-scale feature fusion (CCFF) to process multi-scale features efficiently. RT-DETR outperforms comparable YOLO models in both speed and precision [15], bypassing the need for post-processing and avoiding errors induced by NMS. To address the challenge of fatigue detection in drivers, we present RHD-RTDETR, a lightweight model built upon the RT-DETR framework. By optimizing key components, our model enhances feature extraction and integration while significantly reducing computational complexity. Compared to state-of-the-art approaches, RHD-RTDETR achieves similar detection accuracy with fewer parameters and lower computational requirements, offering a practical solution to improve driver safety.

Manuscript received May 1, 2025; revised July 4, 2025. This research was funded by the project of the Liaoning Provincial Department of Education (Project Number: LJKZ0310).

Qi Liu is a Postgraduate of University of Science and Technology Liaoning, Anshan, Liaoning, China (e-mail: laukei0305@163.com).

Yang Xu* is a Professor of University of Science and Technology Liaoning, Anshan, Liaoning, China (corresponding author to provide phone: +086-138-8978-5726; e-mail: xuyang_1981@aliyun.com).

II. Related Work

A. RTDETR Model

RT-DETR is a state-of-the-art real-time, end-to-end object detection framework that exceeds YOLO models in terms of computational efficiency, performance balance, and training convergence under comparable evaluation conditions [16, 17]. The architecture consists of three main components: a backbone network for feature extraction, a hybrid encoder for multi-scale feature integration, and a Transformer decoder equipped with an auxiliary prediction head [18].

The convolutional backbone network extracts multiscale features at spatial resolutions corresponding to 8, 16, and 32 downsampling levels. The hybrid encoder incorporates an Attention-driven Intra-scale Feature Interaction (AIFI) module, which processes high-level features, thereby significantly reducing computational complexity while enhancing inference speed and maintaining competitive accuracy. Additionally, the encoder includes a Cross-Scale Feature Fusion Module (CCFM) to integrate multiscale representations effectively [19, 20]. Using the Intersection over Union (IoU) metric, the encoder dynamically selects salient features as initial queries for the decoder, optimizing object detection prioritization. The Transformer decoder

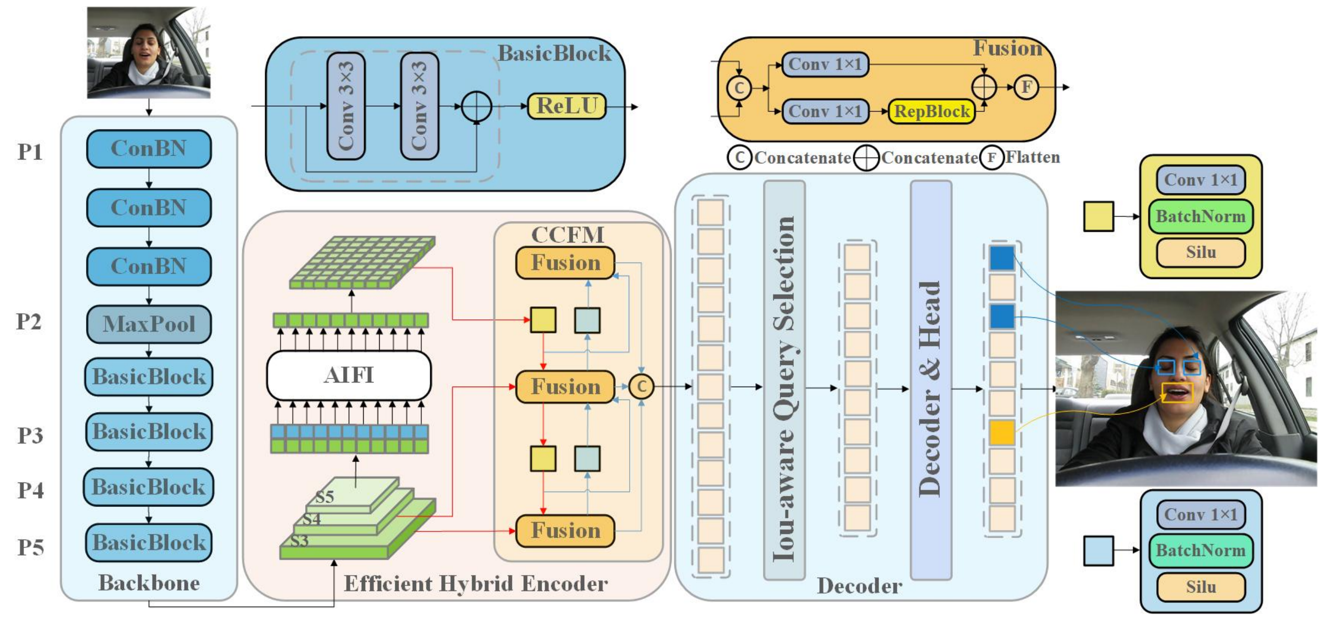


Fig. 1. RTDETR structure diagram

refines these queries iteratively through auxiliary prediction heads, yielding precise bounding boxes and confidence scores. This architecture enhances detection performance and efficiency by optimizing feature extraction and multiscale interactions. The RT-DETR framework is illustrated in Fig. 1.

B. *RHD-RTDETR*

To overcome the computational challenges associated with high parameter counts in driver fatigue detection, this paper presents RHD-RTDETR, an optimized RT-DETR-based object detection algorithm. The network architecture is shown in Fig. 2.

The specific improvements are as follows:

(1) RCCSPELAN Architecture: Introduces a cross-stage gradient optimization framework with dynamic feature reconstruction to balance computation and accuracy in fatigue detection. This design combines dual-modality processing, gradient optimization, and dynamic kernel configuration to optimize the trade-off between model efficiency, inference speed, and detection accuracy.

(2) HiLo-Enhanced Transformer Encoder: Integrates the HiLo attention mechanism into a single-scale encoder layer with a dedicated feature interaction module. By splitting the multi-head self-attention (MSA) layer into high- and low-frequency branches, this modification reduces computational overhead and enhances the extraction of facial features.

(3) DRBC3 Module Replacement: Replaces RT-DETR's RepC3 block with DRBC3 [21] to improve feature representation. The module uses multi-branch convolutions and pooling during training, and structural reparameterization consolidates the branches into standard convolutional layers during inference, maintaining performance while reducing inference costs.

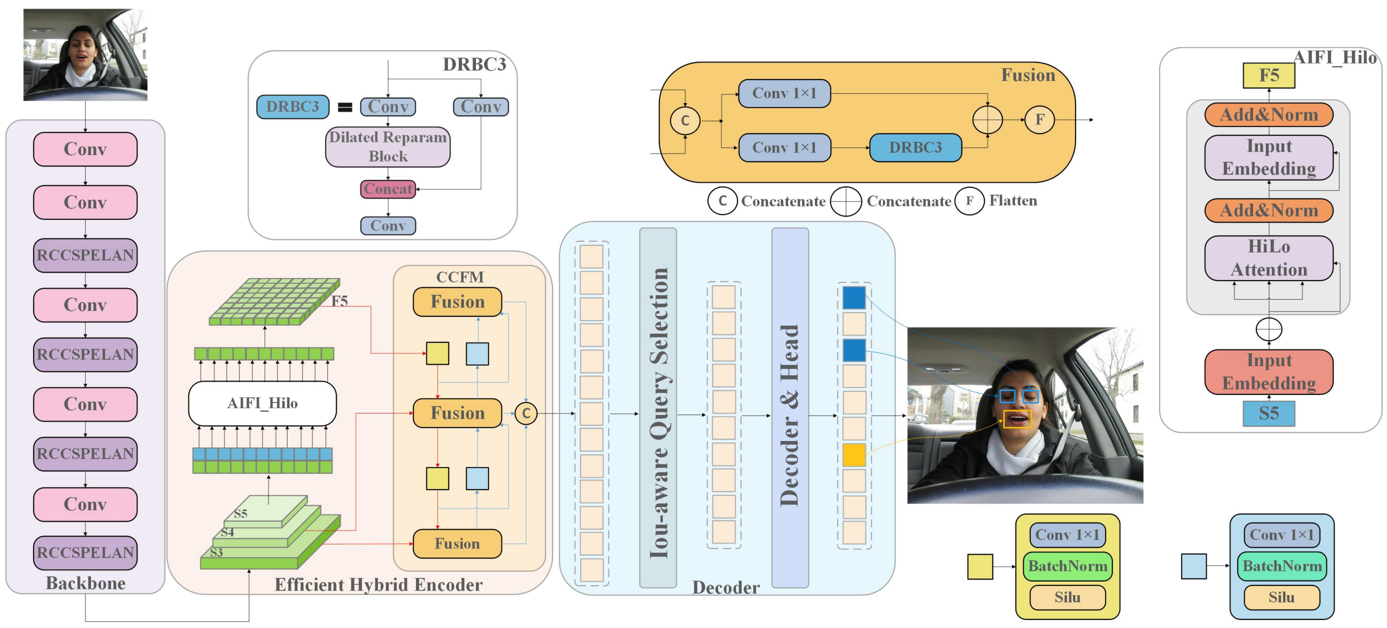


Fig. 2. RHD-RTDETR structure diagram

C. RCCSPELAN for Cross Gradient Optimization and Dynamic Feature Reconstruction

Incorporating facial detection modules into existing driver fatigue detection systems often increases model complexity, leading to higher parameter counts and computational demands. To address this, we propose the RCCSPELAN architecture, which optimizes the balance between computational efficiency and detection accuracy for fatigue monitoring. By integrating cross-stage gradient refinement and adaptive feature reconstruction, RCCSPELAN achieves a lightweight design while preserving robust feature representation. Through multi-dimensional architectural optimization, it effectively balances model compression, inference speed, and detection precision.

For gradient flow optimization, we introduce a dual-modality feature processing framework. This involves (1) a channel separation technique that divides feature maps into deep computational and shallow preservation pathways, and (2) learnable transition layers within cross-stage connections to enable dynamic calibration and nonlinear integration of diverse features. Inspired by CSPNet [22], this design enhances gradient aggregation and feature separation through pathway differentiation. To mitigate gradient vanishing in deep networks, we propose a hierarchical backpropagation strategy. By modifying the interlayer connectivity of the ELAN architecture [23], we introduce topological shortcuts that link deep computational units with shallow feature maps. This decouples the network's theoretical depth from its gradient propagation path, preserving the expressive power of deep architectures while maintaining gradient stability during backpropagation, improving convergence with fewer iterations.

At the computational unit level, we incorporate a phase-adaptive dynamic convolution kernel, RepConv [24], as a core component of the RCCSPELAN module for feature extraction and integration. During training, RepConv employs a multi-branch structure with 3x3 convolutions, 1x1 convolutions, and identity mappings to enhance feature extraction, optimize gradient flow, and mitigate performance degradation from removing traditional residual blocks. During inference, RepConv reparameterizes this multi-branch structure into a single 3x3 convolution, reducing computational overhead and memory usage. This design resolves the trade-off between representational power and computational efficiency in lightweight models, improving performance without increasing inference complexity. The RCCSPELAN architecture is depicted in Figure 3.

D. Hilo Attention

We incorporate the HiLo attention mechanism [25] into a single-scale Transformer encoder via a novel intra-scale feature interaction module, which enhances high-level feature integration and improves facial characteristic detection. Traditional multi-head self-attention (MSA) layers apply uniform attention across image patches, failing to differentiate between high- and low-frequency components, which results in significant computational overhead when processing high-resolution images. To address this, HiLo splits the MSA layer into two branches: (1) A high-frequency branch that uses local self-attention with high-resolution feature encoding (Hi-Fi), (2) A low-frequency branch

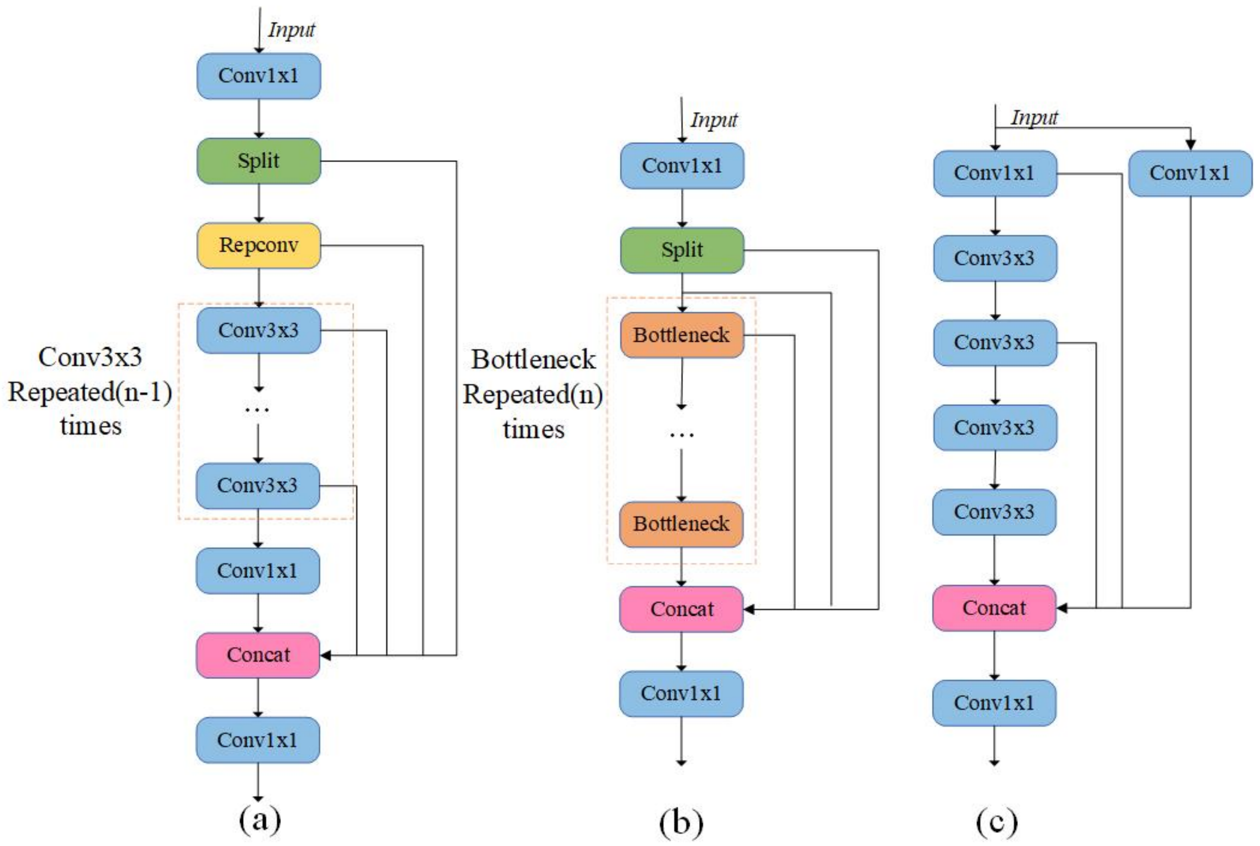


Fig. 3. RCCSPELAN, CSPNet, Structure diagram of ELAN (a) RCCSPELAN; (b) CSPNet; (c)ELAN

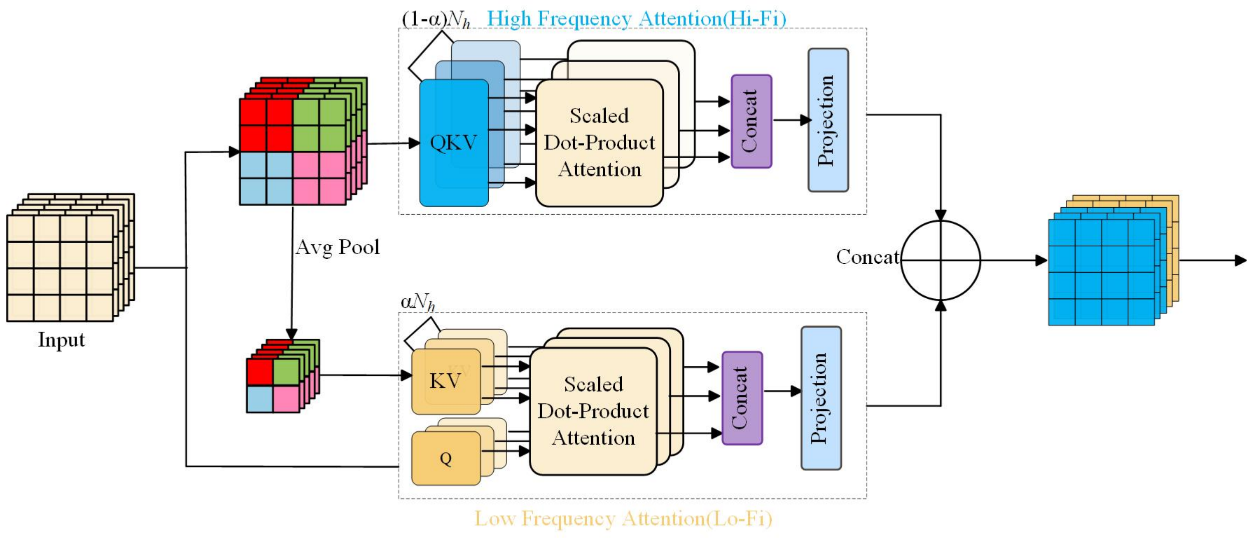

(1-α)N_h High Frequency Attention(Hi-Fi)
QKV
Scaled Dot-Product Attention
Concat
Projection
Avg Pool
Input
Concat
αN_h
KV
Q
Scaled Dot-Product Attention
Concat
Projection
Low Frequency Attention(Lo-Fi)

Fig. 4. Hilo Attention Mechanism Structure Diagram

that applies global attention to downsampled features. This dual-path design reduces computational complexity, preserves spatial details, and enhances network performance. The HiLo architecture is illustrated in Fig. 4.

N_h represents the total number of self-attention heads in the layer, and α denotes the division ratio for high-frequency or low-frequency heads.The proposed architecture replaces the standard multi-head self-attention (MSA) mechanism with the HiLo attention mechanism, which separates high- and low-frequency components in feature maps through dedicated pathways. This frequency separation improves the capture of discriminative features. The computational procedure is as follows:

$$Q = K = V = \text{Flatten}(S_5) \tag{1}$$

$$F_5 = \text{Reshape}(\text{HiloAttn}(Q, K, V)) \tag{2}$$

Among them, $HiloAttn$ represents the HiLo Attention Mechanism,S_5 represents input features, and $Reshape$Shape is used to restore features.input featuresS_5.After being transformed into a one-dimensional vector through Flatten operation Q, K, VSent to the AIFI HiLo module for further processing. The module optimizes the features using the HiLo mechanism and restores the output to its original 2D form through a reshape operation, denoted as F_5.This operation maintains the spatial resolution of feature maps, enabling subsequent fusion modules to integrate global context with localized detail.

The HiLo attention mechanism outperforms traditional approaches (self-attention, CBAM, Transformer models) by improving both representational capacity and computational efficiency. Its frequency-optimized processing facilitates more effective multi-scale feature integration. In complex scenarios, HiLo enhances the model's ability to distinguish fine-grained details from broader contexts, boosting overall representational effectiveness.

E. DRBC3 module

The RepC3 module in RT-DETR improves detection performance through convolutional layer reparameterization [26]. It uses complex multi-branch topologies during training, which are simplified into standard convolutional layers during inference, thus reducing computational overhead. We propose replacing RepC3 with the Dilated Reparameterization Block (DRB) to improve the performance of large-kernel CNNs. The DRB combines parallel dilated convolutional layers with a non-dilated large-kernel convolution, improving the detection of sparse patterns and modeling long-range pixel dependencies. Dilated convolutions expand receptive fields by skipping pixels, enhancing representational power without increasing computational cost. By varying dilation rates and kernel sizes, DRB enables flexible convolutional kernel configurations. The effective kernel size for each dilated layer is given by:

$$\text{Equivalent Kernel Size} = (k - 1) \times r + 1 \tag{3}$$

During training, the DRB module captures fine-grained local features and learns broader spatial patterns by parallelizing dilated and large-kernel convolutions. After training, structural reparameterization merges the dilated and large-kernel convolutions, minimizing computational overhead during inference. In the inference phase, dilated convolutions are converted into equivalent sparse large-kernel convolutions, removing the computational burden of dilated convolutions and enhancing inference efficiency. Specifically, when the kernel size of a dilated convolution is k and the dilation rate is r, it can be equivalently transformed into a non-dilated convolution with a larger sparse kernel. The conversion formula is as follows:

$$W' = \text{conv\_transpose2d}(W, I, \text{stride} = r) \tag{4}$$

Where W is the original kernel size, W' is the transformed kernel size, and I is the unit kernel (a 1x1 tensor).

This reparameterization process eliminates the computational overhead of dilated convolutions during inference, enabling the model to efficiently capture complex spatial patterns while maintaining a broad receptive field. It mitigates the high computational cost associated with stacking multiple small-kernel convolutions in deep

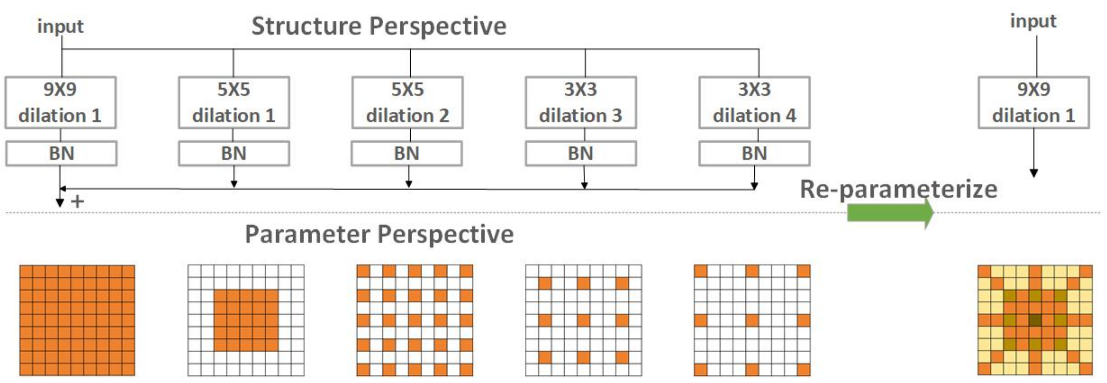


Fig. 5. DRBC3 module

neural networks, thereby improving overall efficiency. The operational principle of the DRB module is illustrated in Figure 5. In this architecture, multiple dilated convolutional layers operate in parallel, enhancing the feature extraction capacity of the non-dilated large-kernel convolution. These dilated convolutions expand the receptive field, enabling the model to capture sparse patterns without increasing computational demands. From a parametric perspective, the outputs of the dilated convolutions are transformed into an equivalent large, sparse convolutional kernel, allowing the module to be reparameterized into a single large-kernel convolution during inference. As shown in Figure 5, a 9×9 large convolutional kernel is combined with four dilated convolutional layers to strengthen feature extraction, which is then reparameterized into a single large-kernel convolution during inference, effectively reducing computational complexity and improving inference speed.

III. Experiment

A. Experimental environment

The experiments were conducted on a Windows 11 operating system with Python 3.8, CUDA 11.8, and PyTorch 2.0.0 as the development environment and deep learning framework. The GPU used was an NVIDIA GeForce RTX 3090 (24GB). ResNet18 served as the baseline model, with an input image size of 640×640, a batch size of 16, an initial learning rate of 0.01, and a total of 100 training epochs.

B. Dataset

This study utilizes two publicly available datasets: (1) the Yawning and Alertness Detection Dataset (YAWDD) [27], designed for driver fatigue monitoring, which includes facial images captured by an in-vehicle rearview mirror camera featuring drivers of different genders in natural driving conditions; and (2) the Closed Eye in Wild (CEW) dataset, developed by Nanjing University of Aeronautics and Astronautics, which focuses on eye status analysis using facial samples from diverse age groups, genders, and ethnicities in a closed-eye state. After data cleaning and feature alignment, the combined dataset comprises 9,095 high-quality facial images. During preprocessing, each image was categorized into four classes—open eyes, closed eyes, mouth open, and mouth closed—using professional annotation tools. Annotations were initially stored in XML format, then converted to TXT format, and images were saved in JPG format to ensure compatibility with standard deep learning frameworks.Examples from the dataset are shown in Figure 6.

To enhance the model's adaptability to a wide range of environments, several data augmentation techniques were applied during training. Specifically, horizontal flipping, random rotation, color jittering, and noise injection were used to augment 364 images. These augmentations simulate diverse environmental conditions, thereby improving the model's resilience to real-world variations. Horizontal flipping and random rotation increase tolerance to changes in object orientation, while color jittering and noise injection boost robustness to lighting inconsistencies and noise

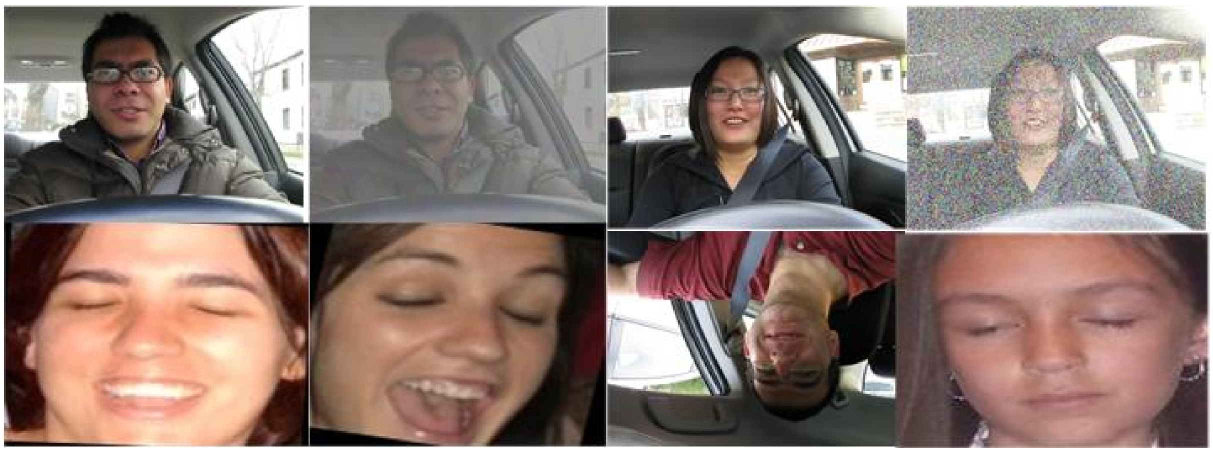


Fig. 6. Partial images in the dataset

disturbances. These strategies enable the model to effectively handle variations in input data, improving its performance across different scenarios and conditions.

C. Evaluation Metrics

The evaluation metrics used in this paper include Precision (P), Recall (R), and mean Average Precision (mAP). The formulas for these calculations are as follows:

$$P = \frac{TP}{TP + FP} \tag{5}$$

$$R = \frac{TP}{TP + FN} \tag{6}$$

$$mAP = \frac{1}{N} \sum_{i=1}^{N} AP_i \tag{7}$$

In this paper, TP refers to the number of true positives, where the target is correctly identified; FP represents false positives, where the target location is recognized but the target is misclassified; FN indicates false negatives, where the target is not detected. N represents the total number of categories, and AP_i denotes the area under the precision-recall curve for each class. Additionally, the improved model is compared with other mainstream object detection models across several metrics, including the number of parameters, computational load, and model weight size, to demonstrate its superior detection performance. The number of parameters indicates the total count of trainable parameters in the model, reflecting its complexity. While a higher parameter count generally improves representational capacity, it also increases computational and storage demands. The computational load, measured in FLOPs (floating-point operations), represents the effort required during inference, with a lower load leading to faster inference speeds. Inference time refers to the duration required to generate predictions after training, typically evaluated as the time taken to process a single input, expressed in seconds or milliseconds per frame.

D. Backbone Network Comparison Experiment

To assess the performance improvements from modifying the RTDETR backbone, we conducted comparative experiments using several widely adopted backbone networks as benchmarks. All experiments were performed under identical conditions and with the same dataset. The results are summarized in Table I.

In this experiment, we systematically evaluated different network architectures within the RTDETR framework. The baseline RTDETR model demonstrated high accuracy but incurred significant computational overhead. To address this, we conducted several experiments. Integrating EfficientViT [28] into RTDETR reduced both the parameter count and computational load but resulted in a 0.2% decrease in average precision and an increase in inference time from 8.3 ms to 9.7 ms.

Similarly, incorporating EfficientFormerV2 [29] reduced parameters and computational demands but yielded only marginal improvements in average precision and inference time. Introducing StarNet [30] and MobileNetV4 [31] also altered the parameter count and computational load, but the average precision increased by just 0.1%. In comparison to RCCSPELAN, the performance gains were minimal.

These findings show that integrating RCCSPELAN into RTDETR strikes an optimal balance between performance and computational efficiency. Although the computational load did not decrease significantly, average precision improved to 98%, and inference time was reduced to 6.6 ms. RCCSPELAN outperformed other architectures, demonstrating its superior performance in complex detection tasks. As a result, adopting RCCSPELAN as the RTDETR backbone significantly enhanced the efficiency of fatigue detection and feature extraction.

E. Ablation experiment

To assess the impact of the improved modules on the algorithm's performance, we conducted ablation experiments using the same dataset and configuration settings. We analyzed the trends of various performance metrics. The results of the experiments are presented in Table II.

The optimized RHD-RTDETR model achieves substantial reductions in parameter count, computational load, and inference time while also yielding a significant improvement in accuracy. Specifically, the stepwise integration of RCCSPELAN and DRBC3 resulted in a 30.3% and 8.7% reduction in parameters, respectively, and a 21.9% and 15.3 decrease in computational load. The average precision increased by 0.3%, and detection speed was markedly improved, lowering the model's overall complexity. The inclusion of the HiLo Attention module had minimal impact on parameter count and computational load but notably increased the average precision from 97.7% to 98.3%, reflecting enhanced performance. Replacing the RTDETR backbone with RCCSPELAN and DRBC3 reduced the parameter count from 19.88 million to 12.12 million,

TABLE I
COMPARISON EXPERIMENT RESULTS OF BACKBONE NETWORK

Model	mAP@0.5/%	Params/M	GFLOPs	Time/ms
RTDETR	97.7	19.88	57	8.3
RTDETR + EfficientViT	97.5	10.70	27.2	9.7
RTDETR + StarNet	97.8	11.21	29.7	7
RTDETR+EfficientFormerv2	97.7	11.80	29.5	8.3
RTDETR + mobilenetv4	97.8	11.31	39.5	6.4
RTDETR+ RCCSPELAN	**98**	**13.85**	**44.5**	**6.6**

TABLE II
RESULTS OF THE ABLATION EXPERIMENT

RCCSPELAN	Hilo	DRBC3	Precision/%	Recall/%	mAP@0.5/%	Params/M	GFLOPs	Time/ms
	Baseline		98.4	97.5	97.7	19.88	57	8.3
✓			98.2	97.8	98	13.85	44.5	6.6
	✓		98.5	98.3	98.3	19.84	57.1	6.8
		✓	98.5	98	98	18.15	48.3	6.9
✓	✓		99	97.5	98	13.81	44.6	6.2
✓		✓	98.5	98.6	98.6	12.12	36.2	6.3
	✓	✓	99.1	97.3	97.8	18.11	48.4	6.6
✓	✓	✓	**99**	**97.7**	**98.3**	**12.09**	**35.9**	**6**

a 39% decrease, and the computational load from 57 gigabytes to 35.8 gigabytes, a 37.2% reduction. This resulted in a 0.9% improvement in average precision, reaching a value of 98.6%. When RCCSPELAN, HiLo Attention, and DRBC3 were integrated, the RHD-RTDETR model achieved 12.09M parameters (60.8% of the original RTDETR model) and a computational load of 35.9G (63% of the baseline model). Inference time decreased by 2.3 ms, and detection accuracy increased by 0.6%, with average precision reaching 98.3%. These findings confirm that RHD-RTDETR effectively reduces model complexity while enhancing computational efficiency. The ablation study, performed under consistent experimental conditions, highlights the individual contributions of each module to overall performance. The incremental improvements in accuracy and reductions in complexity through the integration of these modules demonstrate the effectiveness of the architectural components. The analysis of parameter reduction, computational load, and detection accuracy indicates that the proposed RHD-RTDETR model strikes a well-balanced trade-off between performance and efficiency.

Despite substantial reductions in parameters and computational load, RHD-RTDETR retains impressive detection accuracy, confirming the effectiveness of the proposed optimization strategy. The ablation study results indicate that combining the RCCSPELAN, HiLo Attention, and DRBC3 modules not only improves detection accuracy but also enhances computational efficiency and reduces inference time. This optimization achieves a balanced trade-off between accuracy and speed, demonstrating that module integration can significantly enhance the performance of object detection models. Moreover, RHD-RTDETR maintains high accuracy even with reduced parameters and computational resources, further validating the effectiveness of the optimization. As shown in Figure 7, both RTDETR and RHD-RTDETR exhibit rapid accuracy improvements during the early training stages. While RTDETR shows a slight increase initially, RHD-RTDETR ultimately surpasses it in accuracy. Regarding average precision, RTDETR experiences a sharp rise before stabilizing, whereas RHD-RTDETR achieves a higher, more stable final value. As a lightweight version of RTDETR, RHD-RTDETR reduces computational costs while preserving superior accuracy and performance, demonstrating its robustness and practical value without compromising efficiency.

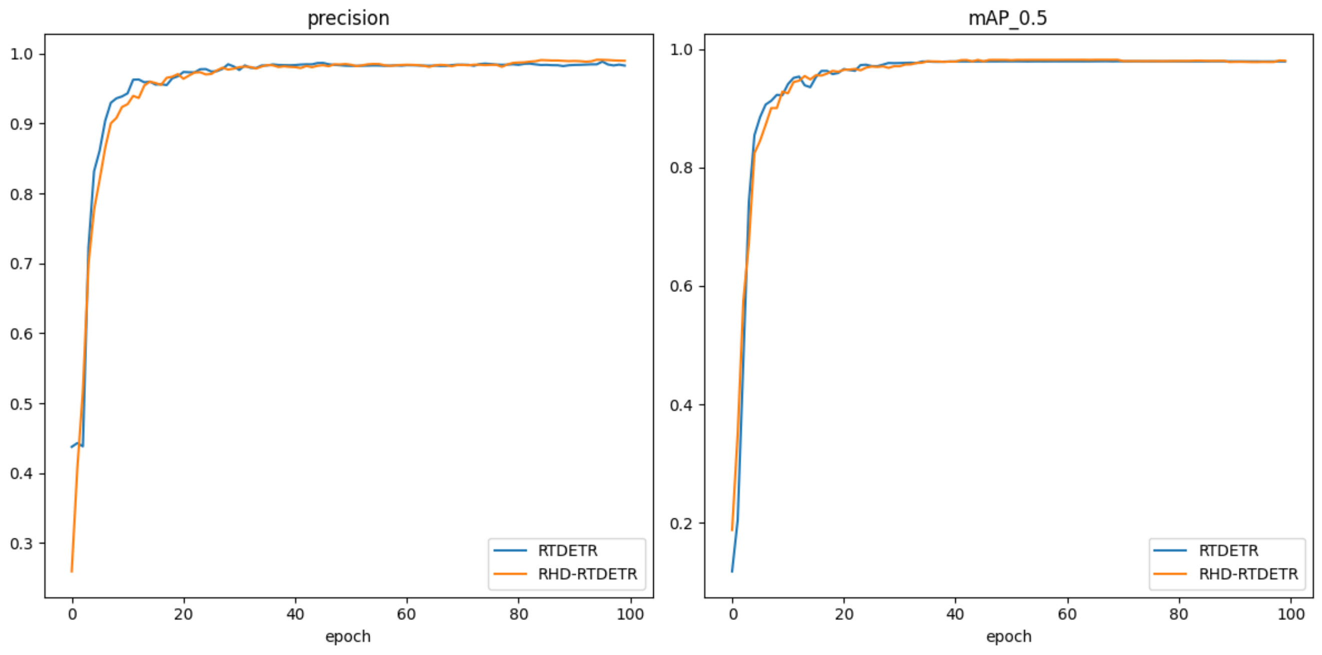


Fig. 7. Comparison diagram of RTDETR and RHD-RTDETR models

F. Generalization experiment

To assess the model's generalization capability, we tested it using a driving behavior dataset comprising 10,760 images across nine categories, including drinking water, talking on the phone, and chatting. The addition of more categories increased computational complexity; however, RHD-RTDETR effectively minimized the computational load while maintaining high accuracy, highlighting its strong generalization ability.

As shown in Table III, RHD-RTDETR retains high precision and recall even with the expanded categories. The average precision only slightly dropped to 96.9%, while computational load and inference time were well-controlled. Despite the increase in categories and samples, RHD-RTDETR's performance remained on par with the baseline RTDETR model, demonstrating its robustness and adaptability.

These results confirm that RHD-RTDETR reduces computational complexity through architectural optimizations with only a minimal loss in accuracy. The model's performance indicates that, even with a more complex dataset, RHD-RTDETR can maintain efficient and stable results in more challenging tasks, validating its ability to generalize in dynamic environments.

G. Results and Analysis

In this study, we evaluated the performance and competitiveness of the RHD-RTDETR model by comparing it against various state-of-the-art methods under identical experimental conditions. The models included in the comparison are Faster R-CNN, SSD, the RTDETR series, and the YOLO series (encompassing recent versions such as YOLOv10 and YOLOv11), all of which represent leading approaches in object detection. The experimental results are presented in Table IV.

RHD-RTDETR significantly outperforms Faster R-CNN in both detection accuracy and average precision. While Faster R-CNN is known for high detection accuracy, its large parameter size creates bottlenecks in computational efficiency and resource consumption. In contrast, RHD-RTDETR not only improves detection accuracy from 84.3% to 99% but also increases average precision from 83.5% to 98.1%. Its lightweight design enhances accuracy while reducing both parameter count and computational complexity, achieving an optimal balance between performance and resource consumption. Compared to the SSD model, RHD-RTDETR shows superior detection accuracy and precision, with improvements of 11.1% and 10.8%, respectively. Despite SSD having 26.3M parameters and a computational complexity of 62.7G, RHD-RTDETR, with only 12.08M parameters and a complexity of 35.9G, offers clear advantages in computational efficiency and resource utilization, demonstrating its high efficiency and lightweight nature.

When compared to the RTDETR series (e.g., RTDETR-R34, RTDETR-R50), RHD-RTDETR maintains an edge in both accuracy and computational efficiency. While RTDETR models perform well in detection accuracy, RHD-RTDETR achieves better performance with fewer resources, thanks to its innovative design. This advantage is especially evident in complex scenarios, where RHD-RTDETR demonstrates greater robustness and precision. With only 12.08M parameters and 35.9G of computational complexity, RHD-RTDETR significantly reduces resource consumption while improving accuracy compared to RTDETR-R50.

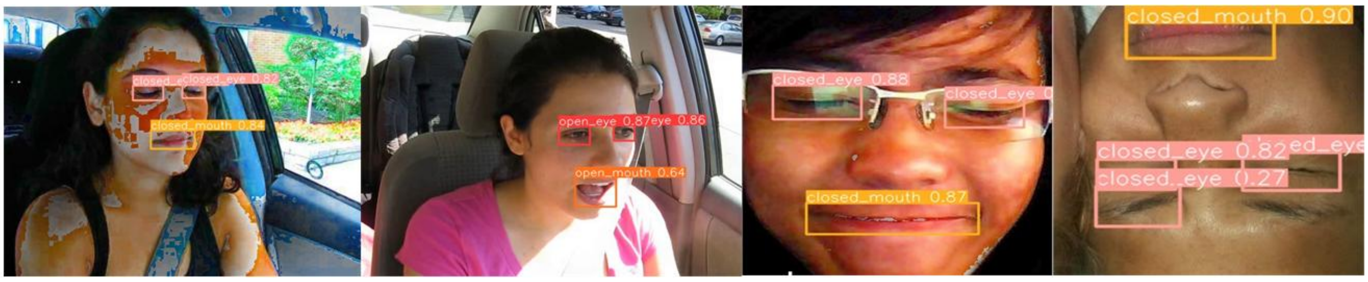
RHD-RTDETR also outperforms the YOLO series, including YOLOv5, YOLOv8, and YOLOv11. Although some YOLO models perform well in terms of detection accuracy and average precision, RHD-RTDETR excels in accuracy, particularly in comparison with YOLOv5 and

TABLE III
GENERALIZATION EXPERIMENT RESULTS

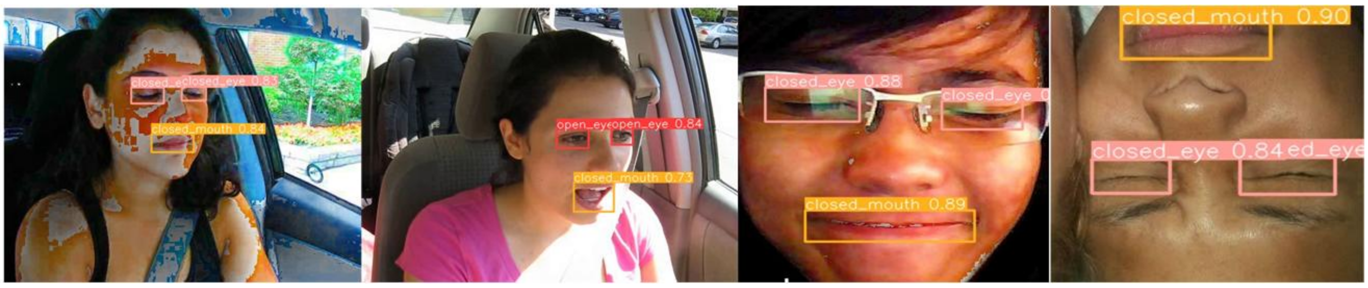
Model	**Precision/%**	**Recall/%**	**mAP@0.5/%**	**Params/M**	**GFLOPs**	**Time/ms**
RTDETR	97.7	97.8	97	19.88	57	8.2
RHD-RTDETR	97.7	97.9	96.9	12.09	36	6

TABLE IV
COMPARATIVE EXPERIMENTAL RESULTS

Model	**Precision/%**	**mAP50/%**	**Params/M**	**GFLOPs**
Faster RCNN	84.3	83.5	137.1	369.2
SSD	87.9	87.3	26.30	62.7
RTDETR-R18	98.4	97.7	19.88	57
RTDETR-R34	98.8	98.2	31.11	88.8
RTDETR-R50	98.6	97.9	41.96	129.6
yolov5	98.3	98.6	25.05	64.0
yolov8	98.7	98.4	25.84	78.7
yolov9	98.2	98.1	20.02	76.5
yolov10	97.5	98.5	16.46	63.4
yolov11	98.5	98.7	20.03	67.7
RHD-RTDETR (ours)	**99**	**98.3**	**12.08**	**35.9**



(a) RTDETR



(b) RHD-RTDETR

Fig. 8. **The visual comparison chart between RTDETR and RHD-RTDETR.**

YOLOv8. RHD-RTDETR not only offers higher accuracy and precision but also demonstrates superior computational efficiency and resource utilization. While certain YOLO models may slightly outperform RHD-RTDETR in average precision, the latter delivers better overall performance in practical applications due to its stability and lower computational demands.

As shown in Figure 8, the model performs well under both regular and low-light conditions, significantly reducing parameters while maintaining high detection accuracy across most categories. Although a slight drop in accuracy is observed in some categories, the algorithm continues to perform effectively in real-world scenarios, underscoring its practical value.

In summary, RHD-RTDETR outperforms Faster R-CNN, SSD, the RTDETR series, and the YOLO series in terms of accuracy, parameter count, and computational efficiency. By combining high accuracy with a low parameter count and reduced computational complexity, RHD-RTDETR addresses the limitations of traditional models in terms of resource consumption, offering exceptional performance, particularly in scenarios that demand stability and efficiency.

IV. Conclusions

This paper introduces the enhanced RHD-RTDETR model, which incorporates several innovations to achieve an optimal balance between computational efficiency and accuracy in edge computing environments. First, RCCSPELAN enhances the model by integrating cross-stage gradient optimization with dynamic feature reconstruction, thereby reducing computational complexity while maintaining high accuracy. This results in a lightweight model with faster inference times. Additionally, the use of dual-modal feature processing and dynamic convolution kernel design further enhances performance and real-time responsiveness. The HiLo attention mechanism optimizes the traditional Transformer encoder by splitting the MSA (Multi-Head Self-Attention) layer into high- and low-frequency branches. This reduces computational overhead while enhancing the model's ability to capture facial features and improve multi-scale feature interactions. Finally, the DRBC3 module provides a more efficient alternative to the RepC3 module in RTDETR. By improving feature representation through multi-branch convolution and pooling operations and reparameterizing these branches into standard convolution layers during inference, DRBC3 reduces unnecessary computational overhead while delivering richer feature representation and faster inference. Through these innovations, the RHD-RTDETR model significantly improves both computational efficiency and detection accuracy, showing superior performance in object detection tasks. The model demonstrates strong potential for real-world applications, and future work will focus on further optimizations to improve performance in more complex scenarios.

References

[1] A. Amodio, M. Ermidoro, D. Maggi, S. Formentin, and S. M. Savaresi, "Automatic detection of driver impairment based on pupillary light reflex," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 8, pp. 3038–3048, 2018.

[2] Q. Yihui, J. Qiong, W. Lingling, Z. Weiping, and Q. Taorong, "Transfer Learning Based Cross Subject EEG Fatigue Driving Detection," *Journal of Nanchang University (Natural Science)*, vol. 47, no. 4, 2023.

[3] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[4] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, vol. 1. Ieee, 2001, pp. I–I.

[5] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 1. Ieee, 2005, pp. 886–893.

[6] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, 2017.

[7] P. Agrawal, R. Girshick, and J. Malik, "Analyzing the performance of multilayer neural networks for object recognition," *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part VII 13*. Springer, 2014, pp. 329–344.

[8] X. Wang, A. Shrivastava, and A. Gupta, “A-fast-rcnn: Hard positive generation via adversary for object detection,” *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 2606–2615.

[9] S. Ren, K. He, R. Girshick, and J. Sun, “Faster R-CNN: Towards real-time object detection with region proposal networks,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 6, pp. 1137–1149, 2016.

[10] K. He, G. Gkioxari, P. Dollár, and R. Girshick, “Mask r-cnn,” *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 2961–2969.

[11] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection,” *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 779–788.

[12] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, “Ssd: Single shot multibox detector,” *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14*. Springer, 2016, pp. 21–37.

[13] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, “End-to-end object detection with transformers,” *European Conference on Computer Vision*. Springer, 2020, pp. 213–229.

[14] Y. Zhao, W. Lv, S. Xu, J. Wei, G. Wang, Q. Dang, Y. Liu, and J. Chen, “Detrs beat yolos on real-time object detection,” *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 16 965–16 974.

[15] Y. Chang, C. Zhang, Y. Shui, and Z. Wang, “Saliency guided RT-DETR for object detection,” *Sixteenth International Conference on Graphics and Image Processing (ICGIP 2024)*, vol. 13539. SPIE, 2025, pp. 22–29.

[16] Q. Yu, X. Ouyang, B. Su, N. Zhao, and H. You, “Vehicle Detection Algorithm in Complex Scenes Based on Improved YOLOv8,” *IAENG International Journal of Computer Science*, vol. 52, no. 4, pp. 886–893, 2025.

[17] C. Chen and B. Wu, “An Improved YOLOv8 Algorithm for Detecting Remote Sensing Target Images,” *IAENG International Journal of Applied Mathematics*, vol. 55, no. 5, pp. 1294–1303, 2025.

[18] S. Wang, H. Jiang, J. Yang, X. Ma, J. Chen, Z. Li, and X. Tang, “Lightweight tomato ripeness detection algorithm based on the improved RT-DETR,” *Frontiers in Plant Science*, vol. 15, p. 1415297, 2024.

[19] S. Wang, H. Jiang, Z. Li, J. Yang, X. Ma, J. Chen, and X. Tang, “Phsi-rtdetr: A lightweight infrared small target detection algorithm based on UAV aerial photography,” *Drones*, vol. 8, no. 6, p. 240, 2024.

[20] J. Pan, S. Song, Y. Guan, and W. Jia, “Improved Wheat Detection Based on RT-DETR Model,” *IAENG International Journal of Computer Science*, vol. 52, no. 3, pp. 705–719, 2025.

[21] X. Ding, Y. Zhang, Y. Ge *et al.*, “Unireplknet: A universal perception large-kernel convnet for audio video point cloud time-series and image recognition,” *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 5513–5524.

[22] C.-Y. Wang, H.-Y. M. Liao, Y.-H. Wu, P.-Y. Chen, J.-W. Hsieh, and I.-H. Yeh, “CSPNet: A new backbone that can enhance learning capability of CNN,” *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2020, pp. 390–391.

[23] C.-Y. Wang, I.-H. Yeh, and H.-Y. Mark Liao, “Yolov9: Learning what you want to learn using programmable gradient information,” *European Conference on Computer Vision*. Springer, 2024, pp. 1–21.

[24] M. Soudy, Y. Afify, and N. Badr, “RepConv: A novel architecture for image scene classification on Intel scenes dataset,” *International Journal of Intelligent Computing and Information Sciences*, vol. 22, no. 2, pp. 63–73, 2022.

[25] Z. Pan, J. Cai, and B. Zhuang, “Fast vision transformers with hilo attention,” *Advances in Neural Information Processing Systems*, vol. 35, pp. 14 541–14 554, 2022.

[26] X. Ding, Y. Zhang, Y. Ge, S. Zhao, L. Song, X. Yue, and Y. Shan, “Unireplknet: A universal perception large-kernel convnet for audio video point cloud time-series and image recognition,” *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 5513–5524.

[27] S. Abtahi, M. Omidyeganeh, S. Shirmohammadi, and B. Hariri, “YawDD: A yawning detection dataset,” *Proceedings of the 5th ACM Multimedia Systems Conference*, 2014, pp. 24–28.

[28] X. Liu, H. Peng, N. Zheng, Y. Yang, H. Hu, and Y. Yuan, “Efficientvit: Memory efficient vision transformer with cascaded group attention,” *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 14 420–14 430.

[29] Y. Li, J. Hu, Y. Wen, G. Evangelidis, K. Salahi, Y. Wang, S. Tulyakov, and J. Ren, “Rethinking vision transformers for mobilenet size and speed,” *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 16 889–16 900.

[30] X. Ma, X. Dai, Y. Bai, Y. Wang, and Y. Fu, “Rewrite the stars,” *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 5694–5703.

[31] D. Qin, C. Leichner, M. Delakis, M. Fornoni, S. Luo, F. Yang, W. Wang, C. Banbury, C. Ye, B. Akin *et al.*, “MobileNetV4: universal models for the mobile ecosystem,” *European Conference on Computer Vision*. Springer, 2024, pp. 78–96.