Journal of Computing and Electronic Information Management
ISSN: 2413-1660 | Vol. 20, No. 3, 2026

YOLO-RSLW: Research on Optimization of Lightweight YOLO11 Object Detection Algorithm

Guanxun Cui and Wei Xiao

Chongqing University of Technology, Chongqing 401320, China

Abstract: To address the challenges of excessive parameters and high computational complexity in YOLO-series models under resource-constrained environments, this paper proposes an improved YOLO11 framework named YOLO-RSLW, which focuses on the synergistic optimization of lightweight design and accuracy enhancement. The proposed approach systematically reconstructs the model architecture by targeting two critical components: feature fusion and detection head design, thereby significantly reducing model complexity while improving detection performance. Specifically, a lightweight RepNCSPELAN4Lighter module is designed based on reparameterization techniques] to replace the original C3k2 modules in both the backbone and neck networks. The SPPF and C2PSA modules are removed, and an SPPELAN module is introduced to enhance feature representation capability. An efficient LiteShiftHead is proposed to optimize the detection pipeline through task-decoupled group-wise processing and lightweight convolutions. Furthermore, the WIoU loss function is adopted to replace the original loss computation, enhancing the accuracy and robustness of bounding box regression. Experimental results on the DOTAv1.5 aerial image dataset demonstrate that the improved model reduces the parameter count by 37.3% (down to 5.92M), while achieving a 1.1 percentage point improvement in both mAP50 and mAP50-95 (with mAP50 reaching 38.2%), realizing a synergistic optimization of accuracy and efficiency. Ablation studies validate the effectiveness of each proposed module, among which RepNCSPELAN4Lighter and LiteShiftHead contribute most significantly to model compression, while the WIoU loss further improves localization accuracy. This study provides a more efficient detection solution for embedded platforms and real-time vision applications.

Keywords: Object detection; YOLO11; Lightweight network; Feature fusion; Loss function optimization; DOTAv1.5 dataset.

1. Introduction

The YOLO series models have become the preferred choice for object detection in industrial applications due to their excellent balance between speed and accuracy. However, as model performance continues to improve, their parameter count and computational complexity have also increased correspondingly, limiting their deployment on mobile devices, embedded systems, or scenarios with stringent real-time requirements (e.g., unmanned aerial vehicles, autonomous driving). This issue primarily stems from the fact that performance optimization often takes precedence over efficiency in model design, leading to structural redundancy. Specifically, the multi-scale feature fusion modules in the Neck part (e.g., C3k2 in YOLO11) typically employ repetitive bottleneck structures. Although these structures enhance feature representation capability, they also introduce a large number of parameters and computations. Second, the standard detection head simultaneously handles classification and regression tasks with a relatively fixed structure, failing to optimize efficiency based on the distinct characteristics of each task. Third, commonly used IoU-based loss functions treat all samples equally during training, making it difficult to dynamically focus on hard examples, thereby limiting further improvement in regression accuracy. [1] [2 [3] [4] [5]

To address the aforementioned challenges, existing research has primarily focused on three directions: First, lightweight network design, which reduces parameters and computations directly through efficient modules (e.g., depthwise separable convolutions, channel shuffle [6], reparameterized convolutions). Second, detection head optimization, which improves detection efficiency via task decoupling [7], structural pruning [8], or dynamic routing [9]. Third, loss function improvement, which enhances bounding box regression quality by introducing dynamic weighting, distance metrics, or shape-aware terms [10]. However, most methods often face a trade-off between lightweighting and accuracy improvement: excessive compression leads to feature degradation and accuracy loss, while complex dynamic mechanisms may introduce additional overhead, contradicting the original goal of lightweight design.

In response to these issues, this study takes YOLO11 as the baseline and aims to achieve substantial lightweighting without sacrificing—or even improving—detection accuracy. We argue that the key to efficient lightweight design lies in the precise identification and replacement of redundant model components, along with targeted enhancement of critical tasks. Therefore, the core contribution of this paper is a set of synergistic improvement strategies:

(1) Lightweight backbone design: A lightweight RepNCSPELAN4Lighter module is designed to replace the C3k2 modules in the backbone. The SPPF module is replaced with the SPPELAN module to optimize the feature extraction capability of the backbone. The C2PSA attention module is removed to reduce computational complexity.

(2) Lightweight neck design: The neck structure primarily employs the RepNCSPELAN4Lighter module, which is designed based on structural reparameterization and parallelization concepts, significantly reducing parameter and computational burdens while preserving multi-scale feature fusion capability.

(3) Efficient detection head design: The LiteShiftHead is proposed to enhance detection efficiency through grouped processing of regression and classification tasks, lightweight convolutions, and channel rearrangement mechanisms.

(4) Dynamic weighting loss function: The WIoU loss is

adopted, which utilizes an adaptive gradient reweighting mechanism to enable the model to focus more on hard examples during training, thereby improving regression accuracy.

These improvements work synergistically to construct a "lighter, smarter, and more powerful" detection model.

The contributions of this work can be summarized as follows:

(1) A lightweight RepNCSPELAN4Lighter module is designed to replace the original C3k2 modules.

(2) The SPPF module is replaced with the SPPELAN module to optimize the feature extraction capability of the backbone.

(3) The C2PSA attention module is removed to reduce computational complexity.

(4) The LiteShiftHead detection head is introduced to improve detection efficiency.

(5) The WIoU loss function is adopted to enhance bounding box regression accuracy.

(6) The effectiveness of the proposed improvements is validated on the DOTAv1.5 dataset.

2. Analysis of the Original YOLO11 Architecture

The original YOLO11 architecture consists of three main components:

Backbone: A feature extraction network based on the CSP (Cross Stage Partial) structure, incorporating multiple C3k2 modules.

Neck: A multi-scale feature fusion structure adopting the FPN+PAN (Feature Pyramid Network + Path Aggregation Network) architecture.

Head: A standard detection head responsible for both classification and regression tasks.

The primary issues with the original architecture are as follows:

(1) The C3k2 modules have a relatively large parameter count and high computational complexity.

(2) The traditional SPPF (Spatial Pyramid Pooling - Fast) module suffers from limitations in feature extraction capability.

(3) Although the C2PSA attention module enhances feature extraction capability, it introduces additional computational burden.

(4) The conventional feature fusion approach fails to fully leverage multi-scale features.

(5) The standard IoU (Intersection over Union) loss function exhibits limitations in bounding box regression.

3. Proposed Improvement Scheme

3.1. Design of Lightweight RepNCSPELAN4Lighter Module

Inspired by the Programmable Gradient Information (PGI) concept in YOLOv9 and the design of its core Efficient Layer Aggregation Network (ELAN) module [11], we aim to design a lightweight module that maintains robust feature fusion capability while achieving extreme efficiency. The RepNCSPELAN series modules in YOLOv9, through their unique re-parameterization structure and multi-path layer aggregation [12], obtain richer gradient flow during training while being fused into an efficient and concise form during inference, achieving a balance between performance and speed.

Building upon this foundation, we design the RepNCSPELAN4Lighter module. It introduces key simplifications to the RepNCSPELAN4 module in YOLOv9: the original three parallel processing paths (one direct connection path and two processing branches with different depths) are streamlined into two parallel paths (one direct connection path and one core processing branch). This simplification directly eliminates an entire processing branch, substantially reducing the stacking of convolutional layers and the number of parameters, thereby significantly decreasing the modules computational complexity (FLOPs) and memory footprint, making it more suitable for deployment on edge devices.

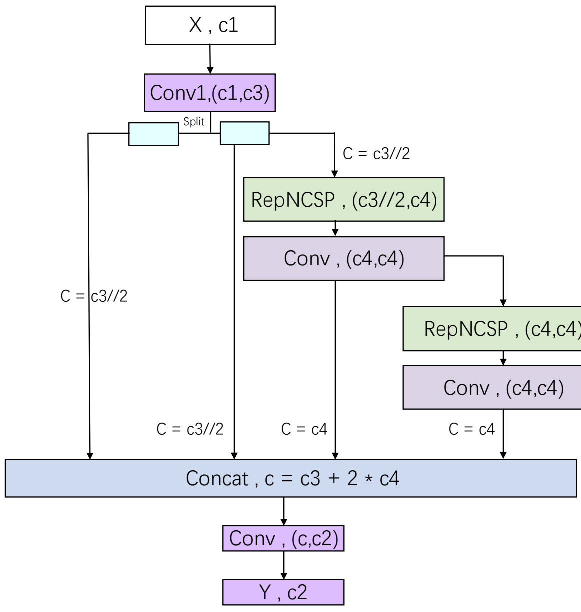


Fig.1 Flowchart of the RepNCSPELAN4 Module

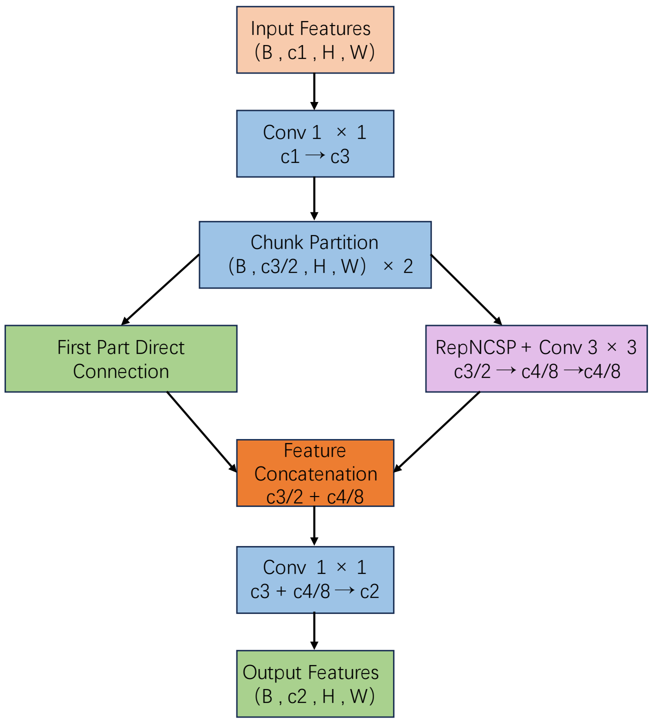


Fig.2 The workflow of the RepNCSPELAN4Lighter module

Input Splitting and Processing: The input feature map is evenly divided into two parts along the channel dimension.

Dual-Path Parallelism:

Path 1 (Direct Connection Path): This path directly participates in subsequent fusion, preserving the original feature information.

Path 2 (Feature Extraction Path): This path sequentially passes through a 1×1 convolution (for channel adjustment and fusion) and multiple stacked RepConv blocks. Each RepConv block has a multi-branch structure during training, which can be re-parameterized into a single convolution during inference. This path is responsible for in-depth feature transformation and enhancement.

Feature Fusion: The feature maps output from the two paths are concatenated along the channel dimension.

Output Projection: Finally, a 1×1 convolutional layer is applied to fuse the concatenated features and adjust the number of channels, producing the final output feature map.

The computational process of the entire module can be formally represented as:

$$\mathbf{Y} = \mathrm{Conv}_{\mathbf{1\times1}}\left(\mathrm{Concat}\left(\mathbf{X_1}, \mathrm{RepConvStack}(\mathrm{Conv}_{\mathbf{1\times1}}(\mathbf{X_2}))\right)\right)$$

Here, RepConvStack represents a sequence of re-parameterizable convolutional blocks, which constitutes the key to the modules powerful representational capacity.

Analysis of RepNCSPELAN4Lighter Module Advantages

Extreme Lightweight Design: Compared to the original RepNCSPELAN4, the streamlined single-branch structure achieves a significant reduction in both parameter count and computational complexity. The parameter efficiency is even more pronounced when compared to the original C3k2 module in YOLO11, making a core contribution to the overall lightweighting of the model.

Preservation of Efficient Gradient Flow and Feature Fusion: Despite the structural simplification, the retained dual-path design of "direct connection + processing" inherits the core concept of the ELAN architecture. The direct connection path ensures lossless transmission of original gradient information, mitigating the gradient degradation problem in deep networks. Meanwhile, the processing path provides nonlinear transformation capabilities through re-parameterizable RepConv blocks. The concatenation of features from both paths achieves efficient fusion of shallow detail information and deep semantic information.

Optimized Training-Inference Consistency: The RepConv blocks employed in this module follow the structural re-parameterization paradigm. During the training phase, their multi-branch structure facilitates learning richer feature representations. During deployment and inference phases, these branches can be equivalently fused into a standard convolutional layer, achieving "zero-overhead" performance improvement and ensuring inference speed.

Plug-and-Play Compatibility: The module design maintains standard interfaces consistent with the Neck component in the YOLO11 architecture, enabling direct replacement of existing modules such as C3k2 without adjusting the overall network connection logic. This facilitates easy integration and validation.

Through the introduction of the RepNCSPELAN4Lighter module, we substantially reduce the computational burden of the Neck component without sacrificing its core functionality of multi-scale feature fusion, thereby establishing a solid foundation for constructing efficient detection models suitable for UAV platforms.

3.2. Backbone Network Improvement

3.2.1. 1. Replacement of C3k2 Modules with RepNCSPELAN4Lighter.

The original C3k2 module employed in the backbone of YOLO11 consists of multiple Bottleneck structures connected in series. While this design ensures rich feature transmission, it also introduces significant computational and parameter overhead. To reduce the parameter count of YOLO11 for lightweighting purposes, one of the key improvements proposed in this paper is the replacement of all C3k2 modules in the backbone section of the original configuration file with the RepNCSPELAN4Lighter module. Compared to the traditional C3k2, RepNCSPELAN4Lighter offers the following advantages:

(1) Parameter Efficiency: The total parameter count is reduced by decreasing the number of repetitive convolutional layers.

(2) Computational Efficiency: A more concise structural design is adopted to reduce computational overhead.

(3) Re-parameterization: Inheriting the advantages of RepConv, branches are merged during inference to enhance speed.

(4) Channel Optimization: Computational load is further reduced by decreasing internal channel numbers (e.g., reducing the c4 parameter).

3.2.2. 2. Replacement of SPPF Module with SPPELAN.

Another significant improvement to the backbone involves replacing the SPPF module at the end of the original backbone with the SPPELAN module. The design advantages of the SPPELAN (Spatial Pyramid Pooling - ELAN) module are as follows:

(1) Structural Fusion: It combines the spatial pyramid pooling capability of SPP with the efficient connection mechanism of ELAN.

(2) Multi-scale Feature Extraction: Features from different receptive fields are extracted through multiple parallel max-pooling layers (with kernel_size=5).

(3) Efficient Connection: The original features are concatenated with features obtained after three pooling operations, enhancing feature representation capability.

(4) Parameter Optimization: Compared to SPPF, SPPELAN reduces the parameter count through its ELAN structure while maintaining the same functionality.

3.2.3. Removal of the C2PSA Attention Module

The C2PSA (Convolutional 2 Position-Sensitive Attention) module is removed from the original network structure. Although the C2PSA module can enhance feature extraction capability, its complex attention mechanism introduces the following issues:

Computational Overhead: The PSABlock contains multi-head attention mechanisms and feed-forward networks, increasing the computational burden.

Parameter Redundancy: The attention mechanism itself requires additional parameters, increasing model complexity.

Overfitting Risk: On certain datasets, excessive attention mechanisms may lead to overfitting.

Inference Latency: Attention computations are time-consuming during the inference phase.

By removing the C2PSA module, we can significantly reduce the parameter count and computational complexity, maintain the feature extraction capability of the backbone network, and improve the inference speed of the model.

3.3. Lightweight Neck Design

To address the limited computational and storage resources of UAV embedded platforms, lightweight design of the detection model is crucial. The C3k2 module employed in the original Neck section of YOLO11 consists of multiple Bottleneck structures connected in series. While this design ensures rich feature transmission, it also introduces significant computational and parameter overhead. By replacing all C3k2 modules in the Neck with the RepNCSPELAN4Lighter module, this improvement not only reduces the parameter count but also enhances the efficiency of feature fusion.

3.4. Efficient Detection Head: LiteShiftHead

3.4.1. Structural Design and Working Principle

The standard detection head handles both classification and regression tasks simultaneously with a unified structure, which may introduce redundancy. LiteShiftHead decouples these two tasks and adopts a grouped lightweight design. It primarily consists of two core modules:

REG Module: Specifically responsible for the bounding box regression task. It employs depthwise separable convolutions to reduce computational load and incorporates a coordinate attention mechanism to enhance spatial location awareness.

CLS Module: Specifically responsible for the object classification task. It utilizes partial convolutions (SPConv) to further compress computation and applies channel shuffle operations to enhance information flow between groups, thereby improving feature representation capability.

Both modules share the input features from the Neck but process them through independent lightweight convolutional paths, ultimately outputting regression and classification results. The grouped processing mechanism reduces interference between tasks, while lightweight convolutions ensure efficiency.

3.4.2. Efficiency Enhancement Mechanisms

Task Decoupling: The separation of regression and classification tasks allows for customized network layers tailored to the characteristics of each task, improving task execution efficiency.

Lightweight Convolutions: Extensive use of depthwise separable convolutions and partial convolutions substantially reduces floating point operations (FLOPs) while maintaining performance.

Enhanced Channel Interaction: Channel shuffle operations promote feature information exchange between different groups, compensating for potential representational capacity loss that may result from lightweight convolutions.

3.5. Dynamic Weighting Loss Function: WIoU

3.5.1. Definition and Motivation

The accuracy of bounding box regression significantly impacts detection performance. WIoU (Weighted IoU) builds upon the standard IoU by incorporating a dynamic weighting term based on the geometric relationship between bounding boxes, enabling the loss function to adaptively adjust attention to samples of varying difficulty. Its formula is defined as follows:

$$L_{WIoU} = \exp\left(\frac{(x_{gt} - x_{pd})^2 + (y_{gt} - y_{pd})^2}{(w_g^2 + h_g^2)}\right) \cdot (1 - IoU)$$

Where (x_{gt}, y_{gt}) and (x_{pd}, y_{pd}) represent the center coordinates of the ground-truth box and the predicted box, respectively, and (w_g, h_g) denote the width and height of the smallest enclosing rectangle covering both the ground-truth and predicted boxes. The exponential term measures the normalized distance of the center points relative to the size of this enclosing rectangle.

3.5.2. Advantages of the Dynamic Weighting Mechanism

Focusing on Hard Examples: For samples with large deviations in center point coordinates (i.e., hard examples), the exponential term increases, thereby assigning a larger loss weight to these samples and driving the model to prioritize optimizing such difficult cases.

Scale Adaptivity: The denominator $(\boldsymbol{w_g^2} + \boldsymbol{h_g^2})$ in the weighting term renders it adaptive to objects of varying scales, effectively mitigating the issue where small objects are underweighted due to their small absolute coordinate errors.

Improved Regression Accuracy: Through this dynamic re-weighting mechanism, WIoU enables the model to refine bounding box positions more precisely during the later stages of training, resulting in tighter bounding boxes.

4. Dataset and Evaluation Metrics

4.1. Dataset

This study employs the DOTAv1.5 dataset [13] and the VisDrone dataset [14] for training and validation. The DOTAv1.5 dataset is a large-scale aerial image object detection dataset containing 14,113 images with annotations for 15 object categories (e.g., airplanes, ships, storage tanks, baseball fields, etc.), comprising approximately 428,000 instances. The images have diverse sizes, and the objects exhibit characteristics such as large scale variations, arbitrary orientations, and dense distribution, making this dataset particularly suitable for validating model detection capability and generalization in complex scenarios. We follow the official data split, using the training set for model training and evaluating performance on the validation set. The VisDrone2019 dataset was collected and constructed by the AISKYEYE team from the Machine Learning and Data Mining Laboratory at Tianjin University. This dataset contains 8,599 static images captured by drones from high altitudes across various locations. Among these, the training set comprises 6,471 images, the validation set contains 548 images, and the test set includes 1,610 images [15]. The dataset encompasses 10 object categories: pedestrian, people, bicycle, car, van, truck, tricycle, awning-tricycle, bus, and motor.

4.2. Evaluation Metrics

To accurately quantify the detection performance of the proposed model on UAV-captured images, this paper adopts the widely recognized evaluation metric in multi-object detection tasks: mean Average Precision (mAP). mAP50-95 is the standard evaluation metric in the COCO dataset, representing the mean average precision computed across ten IoU (Intersection over Union) thresholds ranging from 0.5 to 0.95 with a step size of 0.05. In this paper, it is referred to simply as precision, providing a comprehensive assessment of model performance under varying IoU thresholds. mAP50 refers to the mean average precision calculated at a fixed IoU threshold of 0.5, reflecting the detection performance under a relatively lenient IoU threshold and typically used to evaluate the recall capability of the model. The computation process of mAP is as follows:

$$P = \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FP}}$$
$$R = \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}}$$
$$\mathrm{J}(P,R)_k = \int_0^1 P(R)dR$$
$$mAP = \frac{1}{n}\sum_{k=1}^{n} \mathrm{J}(P,R)_k$$

where P represents precision, and R represents recall; TP denotes true positives, FP denotes false positives, and FN denotes false negatives; J(P, R)k represents the average precision function, with k indexing the total number of categories.

According to the guidelines of the COCO dataset, this paper defines objects with dimensions smaller than 32 pixels × 32 pixels as small objects. The detection accuracy for small objects is denoted as AP_S, which is used to evaluate the detection performance for small objects. To assess the improvement of the proposed model in detecting extremely small objects, this paper defines objects with dimensions smaller than 16 pixels × 16 pixels as extremely small objects and uses AP_XS to represent the average precision for extremely small objects. Both AP_S and AP_XS are calculated using formula (12). Unlike the metric mAP50, the calculation of AP_S only considers small objects with dimensions smaller than 32 pixels × 32 pixels, while the calculation of AP_XS only considers extremely small objects with dimensions smaller than 16 pixels × 16 pixels.

In addition, the number of parameters (Params) and the floating-point operations (FLOPs) are adopted as model efficiency metrics. Params represent the number of trainable parameters in the model, measured in millions (M). The fewer the parameters, the more lightweight the model. FLOPs represent the number of floating-point operations required during model inference, measured in billions (B). The lower the computational cost, the faster the inference speed of the model.

5. Overall Network Architecture

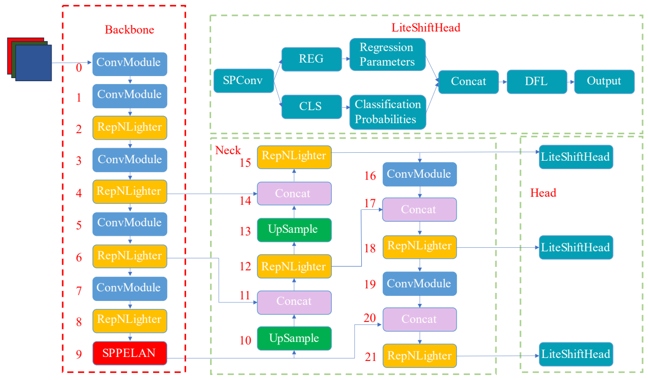


Fig.3 Overall Network Architecture of YOLO-RSLW

6. Experiments and Analysis

6.1. Environment and Parameter Settings

The experiments were conducted under standardized software and hardware configurations. The PyTorch 2.6.0 deep learning framework [16] was adopted, and the YOLOv11s model served as the baseline for performance comparison. The hardware platform was equipped with an Intel i7-13700KF processor, running on the Windows 11 operating system. An NVIDIA GeForce RTX 4090 GPU (with 24 GB of video memory) was utilized, accelerated by CUDA 12.6. Code development and experimental execution were carried out within the PyCharm 2024.1.4 integrated development environment.

During the model training phase, input images were uniformly resized to a resolution of 640×640 pixels. The base learning rate was set to 0.01, and the Stochastic Gradient Descent (SGD) algorithm[17] was employed for model optimization. The training batch size was configured to 8, with a total of 400 training epochs. Detailed hardware configurations and training hyperparameters are summarized in Table 1 and Table 2, respectively.

Tab.1 Experiential environment

Environment	Model
Operating System	Windows11
CPU	Intel®Core™i7-13700KF
GPU	NVIDIA GeForce RTX 4090 24G
Development Environment	Python 3.10
CUDA	12.6
Deep Learning Framework	Pytorch 2.6.0
Compiler	Pycharm 2024.1.4

Tab.2 Training parameters

Parameter Name	Value
batchsize	8
Initial Learning Rate (lr0)	0.01
Final Learning Rate(lr1)	0.01
workers	8
optimizer	SGD
momentum	0.937

6.2. Ablation Studies

To verify the effectiveness of each improved component, we conducted systematic ablation experiments on the DOTAv1.5 validation set. The results are presented in Table 3. Using the original YOLO11-s as the baseline, the parameter count was 9.43M, with an mAP50 of 37.1% and an mAP50-95 of 22.4%. After sequentially introducing the RepNCSPELAN4Lighter lightweight Neck, the LiteShiftHead detection head, and the WIoU loss function, the models parameter count continuously decreased, while detection accuracy steadily improved. When incorporating all three enhancements, the resulting model (Ours) achieved a parameter count of 5.91M (a reduction of 37.3%), along with an mAP50 of 38.2% (+1.1%) and an mAP50-95 of 23.6% (+1.2%). This fully demonstrates the complementarity and synergistic effects of each module in terms of lightweighting and accuracy improvement.

Tab.3 Ablation Studies on the DOTAv1.5 Validation Set

Baseline	RepNCSPELAN4Lighter	LiteShiftHead	WIoU	mAP50-95/%	Params/M	FLOPS/B
√				37.1	22.4	21.3
√	√			37.6	22.9	20.7
√		√		37.8	23.0	23.4
√			√	37.5	23.0	21.3
√	√	√		38.0	23.3	22.8
√	√		√	38.0	23.4	20.7
√	√	√	√	38.2	23.5	22.8

6.3. Comparison Experiments with Mainstream Detectors

We compared our improved model (YOLO-RSLW) with other versions of the YOLO series and several representative lightweight models on the DOTAv1.5 dataset. The results are shown in Table 4. Compared to the lightweight designs of YOLOv5s and YOLOv8s, our model achieves a significant accuracy advantage while maintaining a similar or even smaller parameter count. Compared to the baseline YOLO11, our model achieves a substantial 37.2% reduction in parameters while surpassing its accuracy. This demonstrates that our improvement scheme achieves an excellent balance between "lightweighting" and "high precision."

Tab.4 Comparison of experimental results with other state-of-the-art methods on DOTAv1.5.

Model	mAP50/ (%)	mAP50-95/ (%)	Params/M	FLOPs/B
YOLOv5-s	36.5	22.3	7.8	19.0
YOLOv6-s	36.2	21.9	15.9	43.0
YOLOv8-s	36.0	22.1	9.8	23.6
YOLOv9-s	36.7	22.6	6.3	22.8
YOLO11-s	37.1	22.4	9.4	21.6
ours	38.2	23.6	5.9	22.8

The best results are shown in bold.

6.4. Results of the Improved Model

Figure 4 and Figure 5 the variation curves of various metrics for the original YOLO11-s algorithm and the YOLO-RSLW algorithm on the DOTAv1.5 dataset as a function of the number of training epochs. It can be observed from the curves that, under identical training conditions and the same number of epochs, the detection accuracy of YOLO-RSLW is significantly enhanced, and other metrics are also markedly improved, further substantiating the effectiveness of the proposed model enhancements.

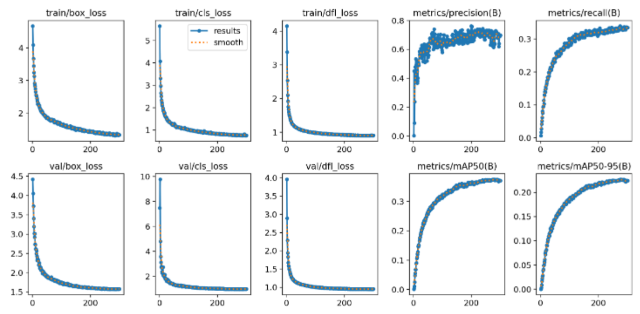


Fig. 4 Performance metrics of the YOLO11-s algorithm on the DOTAv1.5 dataset

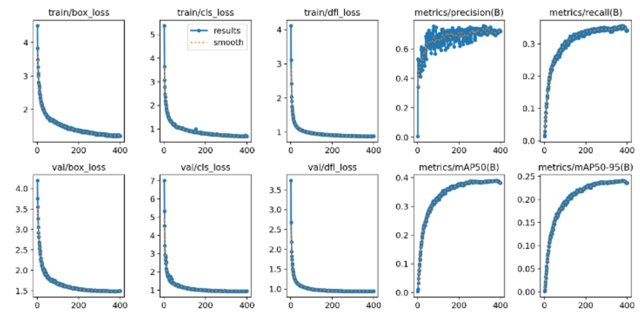
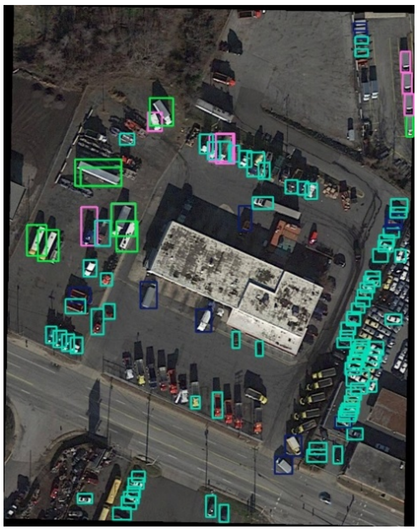


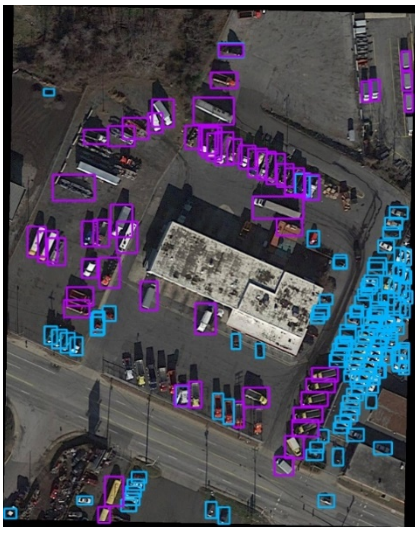
Fig. 5 Performance metrics of the YOLO-RSLW algorithm on the DOTAv1.5 dataset

6.5. Visualization Analysis

To intuitively demonstrate the effectiveness of the proposed improvements, we selected complex scenes from the DOTAv1.5 dataset containing objects of various scales, densities, and orientations for visual comparison. As shown in Figure 6, the (a) side displays the detection results of the original YOLO11, while the (b) side shows the results of our improved model. It can be observed that our model (YOLO-RSLW) performs better in the following aspects: fewer missed detections for dense small objects; more accurate bounding box localization; and higher classification confidence for large-scale objects. This visually confirms the enhanced role of the lightweight Neck and the improved detection head in feature extraction and localization/classification, as well as the contribution of the WIoU loss to improving regression accuracy.



(a) YOLO11 detection results



(b) YOLO-RSLW detection results

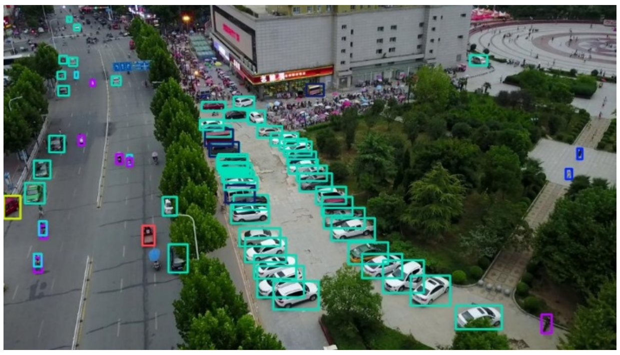
Fig. 6 The DOTAv1.5 dataset containing objects of various scales, densities, and orientations for visual comparison

6.6. Generalization Experiments

To evaluate the generalization capability of the model, generalization experiments were conducted using the UAV-captured aerial image dataset VisDrone. A comparative analysis was performed against specialized detection models developed over the past three years on this dataset. It is evident from Table 5 that YOLO-RSLW outperforms the state-of-the-art YOLO11-s model on this dataset, achieving an improvement in mAP50 of approximately 1.5%. This indicates that YOLO-RSLW not only excels in object detection within remote sensing imagery but also demonstrates strong detection capabilities in UAV-captured images. It is apparent from the aforementioned experiments that YOLO-RSLW exhibits robust generalization, making it suitable for diverse task scenarios. Validation performed by YOLO-RSLW on the VisDrone dataset, as shown in Figure 7, reveals that the YOLO-RSLW model achieves a lower false detection rate and higher prediction confidence compared to the original YOLO11-s model, while also reducing the parameter count (Params/M) by 41%. This demonstrates that the YOLO-RSLW algorithm possesses better robustness and greater lightweight efficiency, enabling its effective application for detection across multiple datasets.

Tab.5 Generalization Experiments on the VisDrone-2019 Dataset

Model	mAP50/(%)	mAP50-95/(%)	Params/M	FLOPs/B
YOLO11-s	40.0	24.0	10.0	27.4
ours	41.5	25.6	5.9	22.8


(a) YOLO11 detection results

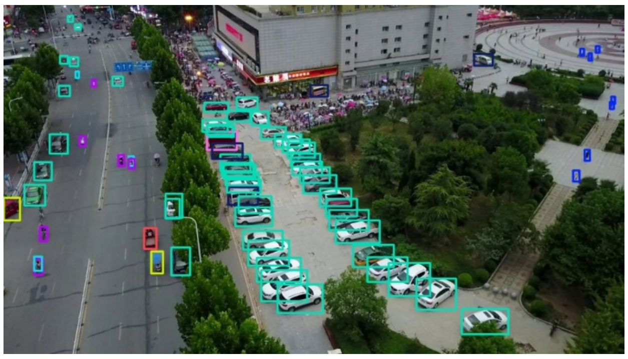

(b) YOLO-RSLW detection results

Fig. 7 Comparison of detection results on the VisDrone2019 dataset.

7. Conclusion

This paper addresses the challenges of high parameter count and computational complexity in the YOLO11 model for practical deployment by proposing a systematic lightweighting and performance enhancement scheme. Through in-depth analysis of model bottlenecks, we focused on optimizing the Neck structure and the detection head: we designed a lightweight Neck module, RepNCSPELAN4Lighter, based on the reparameterization concept; proposed an efficient decoupled detection head, LiteShiftHead; and introduced the dynamically weighted WIoU loss function to improve regression accuracy. Experiments conducted on the DOTAv1.5 aerial imagery dataset demonstrate the significant effectiveness of this scheme. Ablation studies validated the efficacy and synergy of each individual component. Comparative experiments with mainstream models showed that the improved model achieves a substantial 37.2% reduction in parameter count, while concurrently improving the core detection metrics mAP50 and mAP50-95 by 1.1% and 1.2% respectively, successfully achieving the goal of "weight reduction and efficiency enhancement." Visualization results further confirm the models more robust detection capability for multi-scale objects, particularly small and densely packed targets, in complex scenes.

References

[1] Redmon J, Divvala S, Girshick R, et al. You only look once: Unified, real-time object detection[C]. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Las Vegas, NV, USA, 2016: 779-788.

[2] Ding X, Zhang X, Ma N, et al. Repvgg: Making vgg-style convnets great again[C]. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. Nashville, TN, USA, 2021: 13733-13742.

[3] Ge Z, Liu S, Wang F, et al. Yolox: Exceeding yolo series in 2021[J]. arXiv preprint arXiv:2107.08430, 2021.

[4] Howard A G, Zhu M, Chen B, et al. Mobilenets: Efficient convolutional neural networks for mobile vision applications[J]. arXiv preprint arXiv:1704.04861, 2017.

[5] Tong Z, Chen Y, Xu Z, et al. Wise-IoU: Bounding box regression loss with dynamic focusing mechanism[J]. arXiv preprint arXiv:2301.10051, 2023.

[6] Zhang X, Zhou X, Lin M, et al. Shufflenet: An extremely efficient convolutional neural network for mobile devices[C]. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Salt Lake City, UT, USA, 2018: 6848-6856.

[7] Tian Z, Shen C, Chen H, et al. Fcos: Fully convolutional one-stage object detection[C]. Proceedings of the IEEE/CVF International Conference on Computer Vision. Seoul, Korea (South), 2019: 9627-9636.

[8] Liu Z, Li J, Shen Z, et al. Learning efficient convolutional networks through network slimming[C]. Proceedings of the IEEE International Conference on Computer Vision. Venice, Italy, 2017: 2736-2744.

[9] Yang B, Bender G, Le Q V, et al. Condconv: Conditionally parameterized convolutions for efficient inference[C]. Advances in Neural Information Processing Systems. Vancouver, Canada, 2019, 32.

[10] Zheng Z, Wang P, Liu W, et al. Distance-IoU loss: Faster and better learning for bounding box regression[C]. Proceedings of the AAAI Conference on Artificial Intelligence. New York, USA, 2020, 34(07): 12993-13000.

[11] Wang C Y, Yeh I H, Liao H Y M. YOLOv9: Learning what you want to learn using programmable gradient information[C]. European Conference on Computer Vision. Milan, Italy, 2024: 1-18.

[12] Lee J H, Kim D H, Park S J. Enhanced Swine Behavior Detection with YOLOs and a Mixed Efficient Layer Aggregation Network in Real Time[J]. Animals, 2024, 14(23): 3410.

[13] Doloriel C T C, Cajote R D. Improving the detection of small oriented objects in aerial images[C]. Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision Workshops. Waikoloa, HI, USA, 2023: 176-185.

[14] Zhu P, Wen L, Du D, et al. VisDrone2019[DS]. 2024. https://doi.org/10.57702/auwyaezl

[15] Xie J. STK-YOLO[DS]. IEEE DataPort, 2025. https://doi.org/10.21227/1rh2-kc55

[16] Paszke A, Gross S, Massa F, et al. Pytorch: An imperative style, high-performance deep learning library[C]. Advances in Neural Information Processing Systems. Vancouver, Canada, 2019, 32.

[17] Bottou L. Large-scale machine learning with stochastic gradient descent[C]. Proceedings of COMPSTAT'2010. Paris, France, 2010: 177-186.