MFWSD-YOLO: Lightweight Multi-scale Feature-fusion Wood Surface Defect Detection Algorithm

Jun Wu,[a,b] Ao Zhang,[a] Chao Deng,[a,*] and Jun Xu [c]

Wood surface defect detection confronts critical challenges including cross-scale feature extraction, excessive parametric burden, and inadequate small-target recognition. This study proposes MFWSD-YOLO, a lightweight multi-scale feature fusion detection algorithm to address these limitations. The algorithm introduces an adaptive downsampling module utilizing dual-path parallel processing to preserve spatial information, designs a shared convolution detection head enabling efficient cross-scale feature interactions, proposes a progressive feature integration block strengthening multi-scale semantic fusion, and embeds a local attention mechanism enhancing spatial modeling precision. Experimental validation demonstrates substantial enhancements, achieving mAP@0.5 and mAP@0.5:0.95 improvements of 8.90% and 5.17% respectively over baseline YOLOv12n. Concurrently, efficiency gains include 52.73% parameter reduction, 33.33% computational complexity decrease, and 50.94% model size compression, maintaining 195.6 frames per second inference capability. Cross-dataset validation substantiates robust generalization across diverse wood defect scenarios and industrial applications. These advances establish an effective computational solution for automated wood quality inspection within intelligent manufacturing environments.

DOI: 10.15376/biores.21.1.1779-1806

Keywords: Wood defect detection; Lightweight network; Multi-scale feature fusion; Object detection; Deep learning

Contact information: a: School of Physics and Electronic Information Engineering, Henan Polytechnic University, Jiaozuo 454000, Henan, China; b: Photoelectric Detection and Sensing Integrated Engineering Technology Research Center of Henan Province, Jiaozuo 454000, Henan, China; c: School of Chemistry and Chemical Engineering, Henan Polytechnic University, Jiaozuo 454000, Henan, China;
** Corresponding author: super@hpu.edu.cn*

INTRODUCTION

Forestry is an important part of modern industrial production (Wang *et al*. 2021). Wood, its core product, serves as a crucial building and decorative material. Wood defects refer to the general names of various characteristics that reduce the commodity and use value of wood (Xie and Ling 2023). During the growth process of wood, the connection points between branches and the main trunk form knots (Zhang *et al*. 2025), while growth and processing procedures also generate various defects including cracks and pith spots. These defects – either formed before or after the manufacture of wood products such as houses, bridges, and furniture – can lower the efficient usage of materials (Hubbe 2017). Wood surface defect detection methods can be broadly categorized into traditional approaches and deep learning-based techniques. In traditional timber production, defects in timber are mainly removed by manual detection (Chen *et al*. 2023). Traditional manual inspection methods, which have high production cost and low efficiency (Hu *et al*. 2020), also suffer from strong subjectivity, failing to meet industrial production requirements. With the advancement of computer vision technology, deep learning has achieved remarkable progress in object detection domains in recent years (Zhao *et al*. 2025). However, despite the promising application prospects demonstrated by deep learning methods for wood surface defect detection, challenges persist in dataset construction, data source acquisition, and model design and optimization (Long *et al*. 2025). From an algorithmic architecture perspective, deep learning-based object detection algorithms are primarily divided into two-stage and one-stage types (Jiang *et al*. 2024).

Two-stage representative algorithms include Fast R-CNN (Girshick 2015), Faster R-CNN (Ren *et al*. 2016), and Mask R-CNN (He *et al*. 2017), among others. These algorithms achieve high detection accuracy but involve complex computations, making it difficult to fully satisfy real-time requirements. One-stage representative algorithms include the YOLO series (Redmon *et al*. 2016), RT-DETR (Zhao *et al*. 2024), and TOOD (Feng *et al*. 2021), among others, with the core advantage lying in fast detection speed. In practical applications, wood surface defects exhibit large-scale span and diverse morphologies, ranging from tiny cracks to large-area missing knots across multiple scale levels, with many belonging to small-target defects. Such issues lead to detection difficulties and affect the model's accurate recognition and localization capabilities for targets (Hu *et al*. 2024). Meanwhile, industrial production lines impose higher requirements on the real-time performance and lightweight design of detection systems. Therefore, achieving model lightweight character while maintaining detection speed and accuracy represents a research topic worthy of investigation (Deng *et al*. 2025).

Research on deep learning-based wood surface defect detection has achieved certain progress. Kurdthongmee and Suwannarat (2019) focused on wood stem cross-section pith localization, achieving 76.3% accuracy, though the applicable scenarios exhibited obvious limitations. Ling and Xie (2022) proposed a ResNet-v2 wood defect detection model, leveraging the fusion of ResNet and GoogLeNet modules to achieve recognition of three defect types; however, this model only supported single defect classification and could not simultaneously detect multiple defects on the same wooden board. Li and Peng (2024) addressed the issues of missed detections and false alarms caused by small size and irregular morphology of wood surface defects by proposing an improved algorithm based on YOLOv8n. By embedding a global context module, integrating deformable large kernel attention into fast spatial pyramid pooling, and optimizing multi-scale feature fusion with a weighted bidirectional feature pyramid, mAP@0.5 increased by 3.1% and recall improved by 6.8% compared to the original baseline algorithm, effectively resolving the previously high rates of missed detections and false alarms. Chen *et al*. (2025) proposed the YOLOv8-OCHD model, introducing omni-dynamic convolution to avoid information omission, strengthening deep feature learning through a C2f_RVB structure constructed by integrating RepViTBlock while simplifying parameters. In addition, the receptive field was expanded while simplifying computation using a Haar wavelet downsampling module. The mAP@0.5 improved by 5.9%, effectively reducing mobile terminal deployment difficulty. However, there remains optimization space in balancing model lightweight design and detection accuracy.

Although the aforementioned research has achieved certain progress in balancing high accuracy and lightweight design, numerous deficiencies remain in wood surface defect detection scenarios: standard convolutional downsampling causes dead knot void boundaries and crack linear structures to degrade in deep networks; fixed convolutional kernels struggle to effectively extract crack directional information and irregular geometric morphologies of resin pockets and pith spots; small-target defects such as cracks and live knots suffer severe semantic information attenuation during dimensionality reduction; traditional decoupled heads exhibit parameter redundancy and limited inter-detection-layer information flow, resulting in insufficient detection performance in multi-scale defect mixed scenarios; feature extraction modules demonstrate weak integration capability for multi-level information of composite defects; neck networks exhibit deficiencies in spatial localization modeling for irregular defects, among others.

In summary, existing wood defect detection methods struggle to balance accuracy and lightweight design. These issues were addressed through improvements in feature extraction, multi-scale fusion, detection head design, among others, with the MFWSD-YOLO algorithm being proposed to achieve unification of high accuracy and real-time performance. The primary research contributions:

1. Introduction of the Adaptive Downsampling Module (ADown) (Wang *et al*. 2024). Proposed in the YOLOv9 model by the cited authors, this module constructed parallel channels of average pooling and max pooling, simultaneously preserving global distribution characteristics and local salient features during spatial dimensionality reduction. Through channel reorganization and cascaded fusion, it generated compressed representations with higher information density, effectively suppressing the degradation of critical features such as dead knot void boundaries and crack linear structures in deep networks.

2. Design of a Lightweight Shared Convolution and Group Normalization Detection Head (LSGNDH). This detection head innovatively adopted a cross-branch parameter sharing topology, enabling three detection layers to share a unified convolutional kernel group. Through dynamic scale calibration layers that adaptively compensated for semantic gaps across different feature levels, it reduced redundant weights while establishing explicit feature flow mechanisms between detection layers, achieving collaborative optimized representation of multi-scale targets and significantly improving detection performance in scenarios mixing defects of different sizes.

3. Proposal of a Progressive Lightweight Reparameterized Feature Integration Block (PLRFIB). This module constructed a heterogeneous dual-path feature transmission architecture, with a preservation path ensuring the structural integrity of original representations, while a processing path performed deep extraction of semantic information through structural reparameterization operators. It configured cascaded convolutional sequences to form a gradient diffusion pattern of spatial receptive fields, achieving progressive feature aggregation from microscopic textures to macroscopic structures. The module ultimately performed deep fusion of shallow geometric encoding and deep semantic encoding through cross-level feature bridging strategies, generating multi-level feature representations with strong discriminative power for composite features such as resin pocket lenticular gloss variations and missing knot internal material deficiencies.

4. Introduction of the Efficient Local Attention (ELA) mechanism (Xu *et al*. 2025). As proposed by the cited authors, this mechanism decoded spatial statistical distributions through bidirectional adaptive pooling, combined with directional encoding from one-dimensional convolution to generate spatial weight matrices, strengthening the model's representation capability for irregular defect spatial positions.

EXPERIMENTAL

MFWSD-YOLO

In wood surface defect detection, YOLOv12n served as the baseline model due to its effective integration of attention mechanisms and real-time performance. YOLOv12n embedded regional attention mechanisms into the feature extraction process through the A2C2f module, achieving collaborative optimization of receptive field expansion and computational load compression. This module, combined with the residual aggregation strategy, stabilized the training process of large-scale models through block-level residual connections and optimized feature fusion pathways. FlashAttention technology improved computational efficiency by reconstructing memory access patterns, while 7×7 separable convolution replaced traditional position encoding to reduce parameter burden.

However, this model exhibited limitations in wood surface defect detection scenarios: traditional convolutional downsampling demonstrated insufficient capability to preserve fine boundary features; the decoupled detection head structure impeded inter-layer feature flow; fixed feature processing procedures showed poor adaptability to heterogeneous defect morphologies; the neck network lacked targeted spatial position encoding mechanisms, among others. To address these deficiencies exposed in YOLOv12n for wood surface defect detection scenarios, the MFWSD-YOLO algorithm was proposed, with its structure illustrated in Fig. 1.

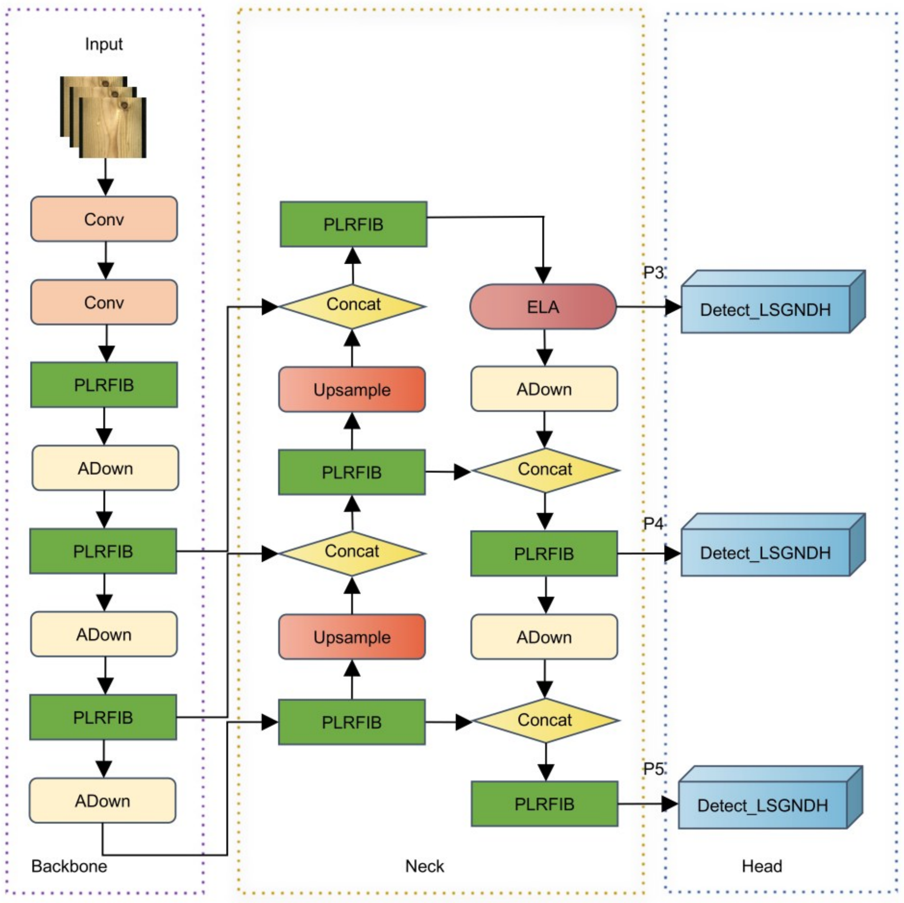

Fig. 1. Algorithm structure of MFWSD-YOLO

ADown Downsampling Module

Traditional convolutional networks face severe information loss dilemmas during layer-by-layer transmission in wood defect detection. Knot-type defect boundary details exhibit progressive weakening during dimensionality reduction, cracks are highly susceptible to missed detection due to extremely small line widths and variable directional characteristics, and resin pocket and pith spot morphological features suffer severe distortion after multi-level transformations. To address these issues, the ADown module proposed in YOLOv9 was introduced, implementing replacement of certain convolutional layers in the original network. Its structure is detailed in Fig. 2.

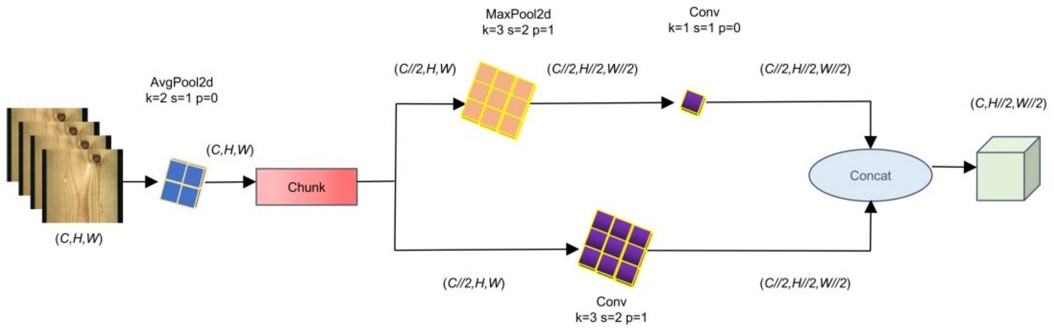

Fig. 2. ADown downsampling structure

This module first performed preliminary aggregation on the input using average, pooling AvgPool2d with kernel size k=2, stride s=1, and padding p=0, preserving contextual information while reducing dimensions. Subsequently, the pooling results were evenly divided (Chunk) along the channel direction into two paths, requiring the input channel number C to be even. The first path employed convolution with k=3, s=2, p=1 to extract local textural relationships, while the second path first used max pooling MaxPool2d with k=3, s=2, p=1 to preserve extreme value responses, followed by convolution with k=1, s=1, p=0 to adjust channels. In the figure, H denotes height, W denotes width, and // represents integer division. After processing through both paths, features of dimension (C//2, H//2, W//2) were obtained, which were finally merged through concatenation (Concat) into output of dimension (C, H//2, W//2).

The adoption of ADown brought significant improvements. The dual-path architecture enabled the convolutional branch to focus on texture detail capture while the pooling branch locked onto salient regional responses, constructing a complementary expression system after concatenation. For small-target defects such as fine cracks, this structure achieved collaborative preservation of edge sharpness and positional information, with detection accuracy improved compared to single-path convolution. When addressing other defects such as dead knots, the branch mechanism also endowed features with stronger adaptability.

LSGNDH

The original detection head of YOLOv12n adopted a decoupled head structure. Although it introduced Depthwise Separable Convolution (DWConv), the P3, P4, and P5 three-layer structure configured complete convolutional weights for classification and regression branches separately, with the accumulation across three layers resulting in persistently high total parameter count. This design exhibited dual limitations in wood defect processing: first, parameter redundancy caused model volume inflation; second, detection layers operated independently, with fine detail information from the P3 layer unable to transmit to the P5 layer, and semantic information from the P5 layer unable to feedback to the P3 layer, severely limiting adaptability to defects of different scales. This scale difference required the detection head to possess more flexible feature processing mechanisms. Based on this, a Lightweight Shared Convolution and Group Normalization Detection Head (LSGNDH) was designed, enabling three detection layers to share a single set of convolutional parameters, achieving model scale compression through parameter reuse while simultaneously introducing normalization layers to ensure training stability. The overall architecture of LSGNDH is detailed in Fig. 3.

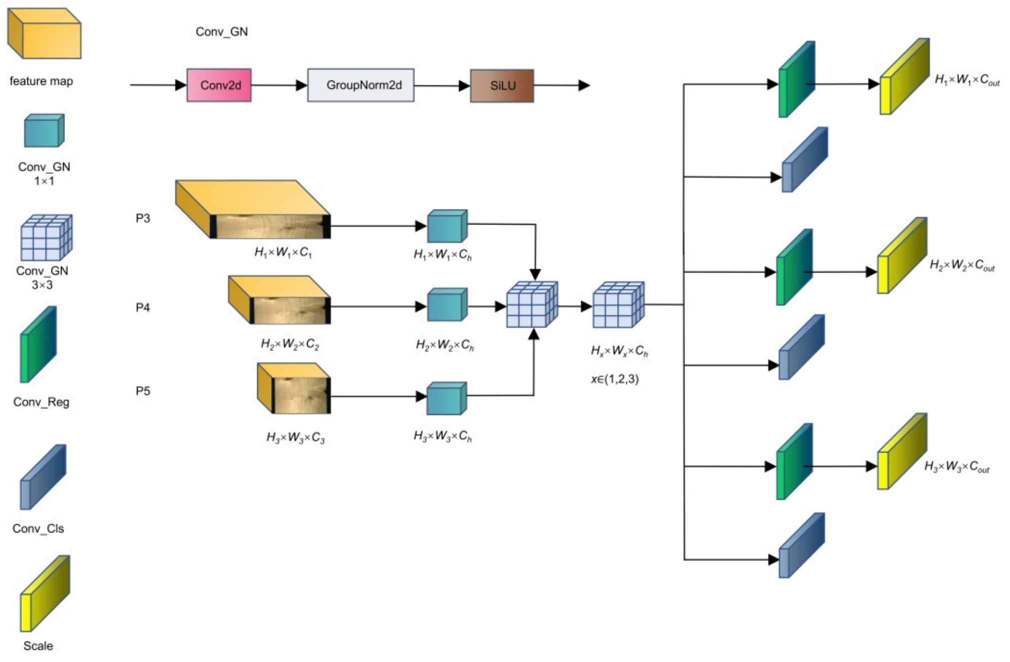
feature map
Conv_GN
Conv2d
GroupNorm2d
SiLU
Conv_GN 1×1
Conv_GN 3×3
Conv_Reg
Conv_Cls
Scale
P3
P4
P5
$H_1 \times W_1 \times C_1$
$H_2 \times W_2 \times C_2$
$H_3 \times W_3 \times C_3$
$H_1 \times W_1 \times C_h$
$H_2 \times W_2 \times C_h$
$H_3 \times W_3 \times C_h$
$H_x \times W_x \times C_h$
$x \in (1,2,3)$
$H_1 \times W_1 \times C_{out}$
$H_2 \times W_2 \times C_{out}$
$H_3 \times W_3 \times C_{out}$

Fig. 3. Lightweight shared convolution and group normalization detection head

The P3, P4, and P5 feature layers output from the neck network first flowed through independent 1×1 Conv_GN modules to complete channel adjustment, normalizing the channel count of each layer to a unified dimension. In the figure, Conv_GN represents convolutional operations fused with group normalization, consisting of three parts: standard convolutional layer Conv2d, group normalization layer GroupNorm2d, and activation function SiLU. The channel adjustment process is shown in Eq. 1,

$$\boldsymbol{F}_{i,\mathrm{adj}} = \mathrm{SiLU}\left[\mathrm{GroupNorm2d}\left[\mathrm{Conv2d}\left(\boldsymbol{F}_{i,\mathrm{in}}, C_{\mathrm{h}}\right)\right]\right], \quad i \in \{3,4,5\} \tag{1}$$

where $\boldsymbol{F}_{i,\mathrm{in}}$ denotes the input feature, $\boldsymbol{F}_{i,\mathrm{adj}}$ denotes the adjusted feature with unified channel count C_h, i represents the detection layer index, the Conv2d function represents two-dimensional convolution with kernel size 1×1, the GroupNorm2d function represents group normalization dividing feature channels into 16 groups for intra-group normalization, and SiLU serves as the activation function.

Feature layers unified in channels entered the shared convolution module, which consisted of two cascaded 3×3 Conv_GN layers. Features from three detection layers sequentially passed through this group of convolutional weights to complete transformation. The first layer performed depthwise separable convolution to extract spatial patterns, while the second layer performed pointwise convolution to fuse channel information, with all detection layers sharing a single set of parameters throughout the entire process. The feature extraction process was expressed as follows,

$$\boldsymbol{F}_{\mathrm{s}} = \mathrm{Conv}_{\mathrm{GN}}^{(2)}\left[\mathrm{Conv}_{\mathrm{GN}}^{(1)}\left(\boldsymbol{F}_{i,\mathrm{adj}}, g = C_{\mathrm{h}}\right)\right] \tag{2}$$

where $\mathrm{Conv}_{\mathrm{GN}}^{(1)}$represents the first layer of depthwise separable convolution with parameter g as the group number such that $g = C_h$ indicated depthwise convolution, $\mathrm{Conv}_{\mathrm{GN}}^{(2)}$ represents the second layer of pointwise convolution, and $\boldsymbol{F}_s$ denotes the unified feature representation.

Shared convolution output features diverged into dual parallel processing paths: the regression branch Conv_Reg handled bounding box position prediction, while the classification branch Conv_Cls discriminated defect categories. In the figure, Conv_Reg represents the regression convolutional branch, generating bounding box position offset predictions through a 1×1 convolutional layer. After regression branch output, a Scale layer was introduced. This layer compensated for response differences when processing targets at different scales with shared parameters through learnable scalar parameters s_i. The regression prediction scaling process is shown in Eq. 3,

$$\boldsymbol{B}_i = s_i \cdot \mathrm{Conv2d}\left(\boldsymbol{F}_{\mathrm{s}}, N_{\mathrm{reg}}\right) \tag{3}$$

where $\boldsymbol{B}_i$ denotes the bounding box prediction output for the i-th detection layer, s_i represents the learnable scaling parameter corresponding to the i-th layer, and N_{reg} denotes the regression branch output channel count, equal to $4 \times r_{\max}$, where $r_{\max}$ represents the maximum regression range parameter for distribution focal loss.

Conv_Cls represents the classification convolutional branch, performing category prediction through a 1×1 convolutional layer. The classification branch prediction could be expressed as follows,

$$\boldsymbol{P} = \mathrm{Conv2d}\left(\boldsymbol{F}_{\mathrm{s}}, N_{\mathrm{cls}}\right) \tag{4}$$

where $\boldsymbol{P}$denotes the category prediction output, and N_{cls}represents the classification branch output channel count, equal to the number of categories n_c, with each channel corresponding to a confidence score for one defect category. The convolutional layer weights for the regression and classification branches were completely independent, ensuring the model optimized separately for localization and classification tasks.

During inference, regression branch outputs were decoded into precise coordinates through the distribution focal loss mechanism. The decoding process is shown in Eq. 5,

$$\boldsymbol{B} = \sum_{j=0}^{r_{\max}-1} j \cdot \mathrm{softmax}\left(\boldsymbol{B}_i^{(j)}\right) \tag{5}$$

where $\boldsymbol{B}_i^{(j)}$represents the j-th channel of the i-th detection layer's regression prediction, j denotes the channel index, the softmax function normalized $r_{\max}$ channels to obtain probability distribution, and $\widehat{\boldsymbol{B}}$ represents the decoded bounding box offset.

The decoded offsets were combined with anchor points and converted to bounding box coordinates, which were concatenated with classification results before output. The final detection output could be expressed as follows,

$$\boldsymbol{Y} = \left[\text{dist2bbox}\left(\boldsymbol{B} \cdot S_{\text{d}}, \boldsymbol{A} \right), \sigma(\boldsymbol{P}) \right] \tag{6}$$

where $\boldsymbol{Y}$ represents the final detection output containing bounding box coordinates and category confidence, dist2bbox represents the core coordinate transformation function converting offset distances and anchor points $\boldsymbol{A}$ into bounding box coordinates, S_d represents the downsampling stride for each detection layer, and σ denotes the Sigmoid function.

In the LSGNDH design, the shared convolution mechanism compressed the original three sets of independent parameters into a single set, with parameter reduction amplitude being significant, which held great significance for production line systems requiring frequent model loading. The introduction of group normalization further consolidated training stability, avoiding convergence difficulties caused by gradient fluctuations during small-batch training. The introduction of the Scale layer compensated for potential scale adaptability degradation caused by parameter sharing, enabling the model to maintain flexibility when localizing defects of different sizes through learnable scaling coefficients.

PLRFIB

The difficulty in wood surface defect recognition focused on morphological diversity. Dead knots exhibited irregular circular or elliptical contours, live knots maintained close connections with surrounding xylem, cracks displayed linear or dendritic extensions, and missing knots formed cavity-like depressions. The C3k2 and A2C2f modules of YOLOv12n (structures detailed in Fig. 4) employed serially constructed feature channels using Bottleneck residual units.

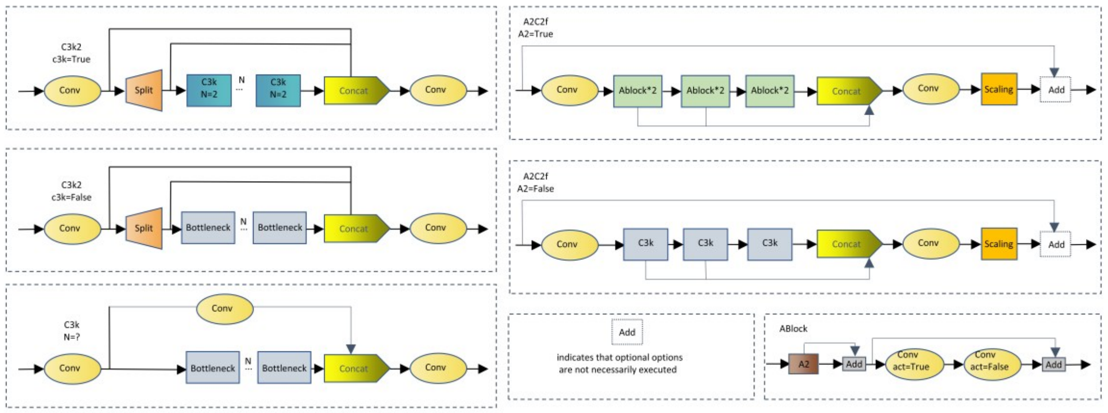

Fig. 4. C3k2 and A2C2f structure

This design limitation lay in convolutional layers processing inputs step-by-step according to preset fixed procedures, with each Bottleneck executing the same flow from compression to transformation to restoration, lacking mechanisms for dynamic adjustment based on target characteristics. Standard convolution generated feature maps containing numerous redundant encodings with similar responses; such redundancy both occupied storage space and slowed computational speed. When addressing missing knots—composite defects mixing void boundaries, peripheral textures, and missing regions with three types of information—single-path serial processing could not effectively separate and integrate different levels of discriminative bases. In view of this, the Progressive

Lightweight Reparameterized Feature Integration Block (PLRFIB) was designed, with its structure detailed in Fig. 5.

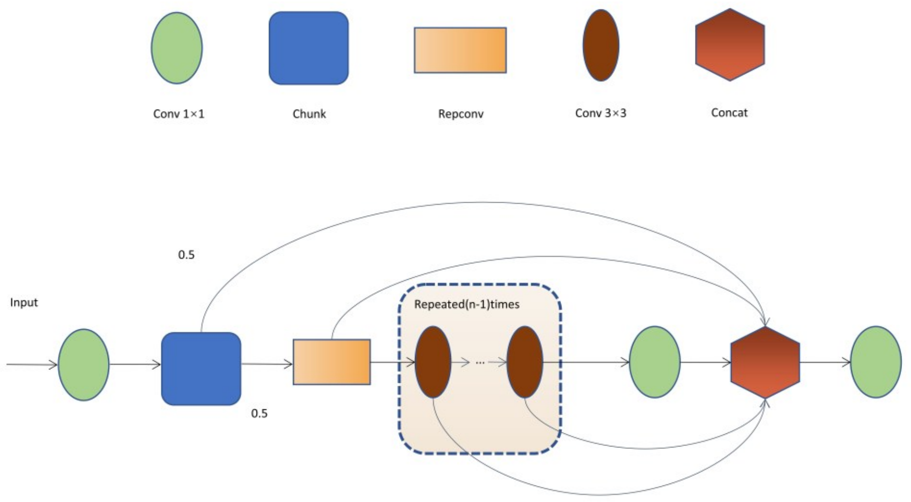

Fig. 5. PLRFIB structure

PLRFIB first employed 1×1 convolution to perform channel expansion on input feature maps, reserving sufficient space for branch processing, as follows,

$$\boldsymbol{F}_{\mathrm{e}} = \mathrm{Conv}_{1\times1}(\boldsymbol{X};\boldsymbol{W}_{\mathrm{e}}) \tag{7}$$

where $\boldsymbol{X}$ denoted the input feature, $\boldsymbol{F}_e$ represented the expanded feature, and $\boldsymbol{W}_e$ denoted convolutional kernel weights. Subsequently, through the Chunk operation, features were evenly divided along the channel dimension into two branches,

$$\{\boldsymbol{F}_{\mathrm{a}}, \boldsymbol{F}_{\mathrm{b}}\} = \mathrm{Chunk}(\boldsymbol{F}_{\mathrm{e}}, 2, 1) \tag{8}$$

where $\boldsymbol{F}_a$ and $\boldsymbol{F}_b$ represented the two branch features after chunking, respectively. The numbers 2 and 1 indicated division into 2 chunks and splitting along the channel dimension (dim=1). Branch a directly transmitted to the fusion stage to preserve shallow information, while branch b entered the RepConv module (structure shown in Fig. 6) for deep feature extraction.

RepConv adopted a three-path parallel structure during training: 3×3 convolution captured spatial neighborhood relationships, 1×1 convolution adjusted inter-channel connections, and identity mapping maintained input signals. In the figure, BN represented batch normalization layer, Train indicated training mode, Val indicated validation/ inference mode, Conv represented convolution operations, and SiLU served as the activation function. The forward propagation during the training phase was expressed as follows,

$$\boldsymbol{F}_{\mathrm{r}} = \sigma[\mathrm{BN}(\boldsymbol{F}_{\mathrm{b}}) + \mathrm{Conv}_{3\times3}(\boldsymbol{F}_{\mathrm{b}};\boldsymbol{W}_3) + \mathrm{Conv}_{1\times1}(\boldsymbol{F}_{\mathrm{b}};\boldsymbol{W}_1)] \tag{9}$$

where $\boldsymbol{F}_r$ denotes the RepConv output feature with subscript r indicating reparameterization, $\boldsymbol{W}_3$ and $\boldsymbol{W}_1$ represents 3×3 and 1×1 convolutional kernel weights respectively with subscripts 3 and 1 indicating kernel sizes, σ denotes the activation function, BN represents the batch normalization function, and the Conv operation already included BN. The three paths each collected gradients during training, enabling the network to learn more comprehensive feature representations.

During inference, the three-path convolutional kernels and batch normalization parameters were synthesized into a single equivalent convolutional kernel,

$$\boldsymbol{W}_{\mathrm{f}} = \boldsymbol{W}_3 \odot \frac{\gamma_3}{\sqrt{\boldsymbol{\sigma}_3^2+\varepsilon}} + \mathrm{Pad}_{3\times3}\left\{\boldsymbol{W}_1 \odot \frac{\gamma_1}{\sqrt{\boldsymbol{\sigma}_1^2+\varepsilon}}\right\} + \boldsymbol{I} \odot \frac{\gamma_{\mathrm{d}}}{\sqrt{\boldsymbol{\sigma}_{\mathrm{d}}^2+\varepsilon}} \tag{10}$$

where $\boldsymbol{W}_f$ denotes the fused equivalent convolutional kernel weight (subscript f indicating fusion), γ_3, γ_1, and γ_d represents the scaling parameter vectors from batch normalization layers of the three branches respectively (subscript d indicating direct connection branch), σ_3^2, σ_1^2, and σ_d^2 represents the running variances of corresponding branches, ε denotes a numerical stability constant variable, $\boldsymbol{I}$ represents the identity mapping unit convolutional kernel, $\odot$indicates element-wise multiplication, and Pad represents the padding function.

After obtaining $\boldsymbol{F}_r$, this feature transmitted along two paths in parallel: one directly participated in subsequent feature fusion, while the other entered a progressive convolutional sequence composed of multiple 3×3 convolutions connected end-to-end. Each convolutional layer in the sequence only processed local regions of upper-layer outputs, but as layer count accumulated, spatial coverage continuously expanded. This progressive expansion approach proved particularly suitable for recognizing features requiring observation of larger regions, such as crack extension directions and dispersed knot distributions, while being more parameter-efficient than directly using large convolutional kernels. The processing flow, *i.e.*, features after RepConv processing entering the progressive convolutional sequence and achieving receptive field expansion through stacking $n-1$ 3×3 convolutions, was expressed as follows,

$$\boldsymbol{F}_k = \mathrm{Conv}_{3\times3}(\boldsymbol{F}_{k-1};\boldsymbol{W}_k), k=1,2,\ldots,n-1 \tag{11}$$

where $\boldsymbol{F}_k$ denotes the k-th convolutional layer output feature, $\boldsymbol{W}_k$ represents the k-th convolutional kernel weight, with initial input being $\boldsymbol{F}_r$, and n represents the total number of convolutional layers.

After progressive sequence processing, the shallow representations preserved by branch a, mid-level representations extracted by RepConv, deep features generated by each convolutional layer, and features processed by 1×1 convolution at the sequence end were all aggregated. In Fig. 5, Concat completed feature concatenation along the channel dimension, finally adjusting channel count through 1×1 convolution,

$$\boldsymbol{Y} = \mathrm{Conv}_{1\times1}\{\mathrm{Concat}[\boldsymbol{F}_{\mathrm{a}},\boldsymbol{F}_{\mathrm{r}},\boldsymbol{F}_1,\ldots,\boldsymbol{F}_{n-1},\mathrm{Conv}_{1\times1}(\boldsymbol{F}_{n-1};\boldsymbol{W}_{\mathrm{c}})];\boldsymbol{W}_{\mathrm{o}}\} \tag{12}$$

where $\boldsymbol{Y}$represents the final output feature, $\boldsymbol{W}_c$ denotes the convolutional kernel weight at the sequence end (subscript c indicating channel adjustment), $\boldsymbol{W}_o$ represents the output convolutional kernel weight (subscript o indicating output), and Concat denotes the concatenation function.

Cross-level concatenation enabled the module to simultaneously utilize shallow details and deep semantics, thereby strengthening discriminative capability when detecting different defect types such as pith spots and resin pockets. Particularly for composite defects such as knot_with_crack, the dual-path architecture enables simultaneous extraction of crack directional features through the progressive convolutional sequence and knot morphological features through the preservation path, achieving effective feature complementarity. Through cooperation of branch diversion, reparameterized extraction, and progressive aggregation, PLRFIB both reduced redundant computation and enhanced recognition accuracy for complex defects.

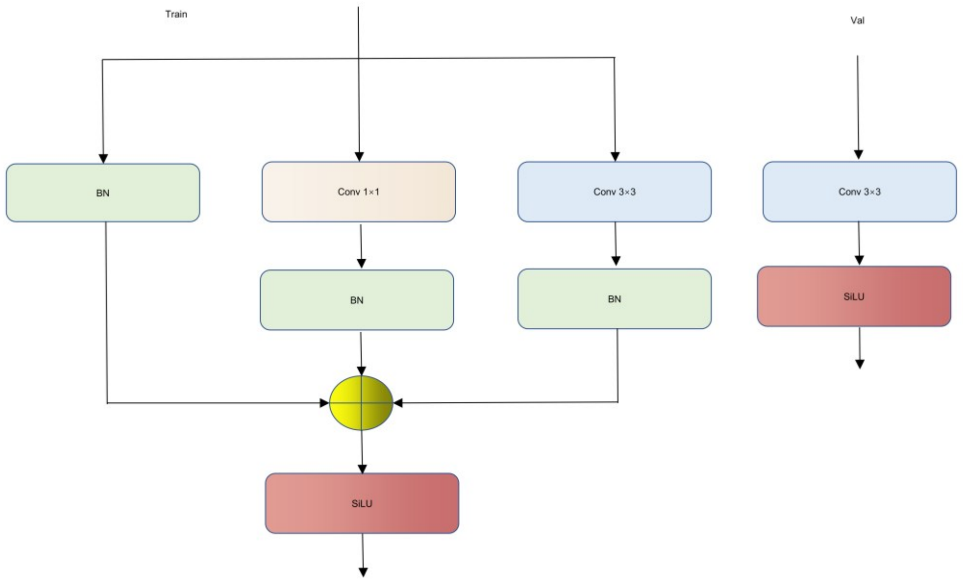

Fig. 6. RepConv structure

ELA Attention Mechanism

Wood defect position distribution directly affected the practical value of detection results. The appearance of dead knots and live knots at board edges versus center positions produced vastly different impacts on wood grade determination; crack extension directions determined the feasibility of subsequent cutting schemes; and the aggregation positions of resin pockets and pith spots related to surface treatment difficulty.

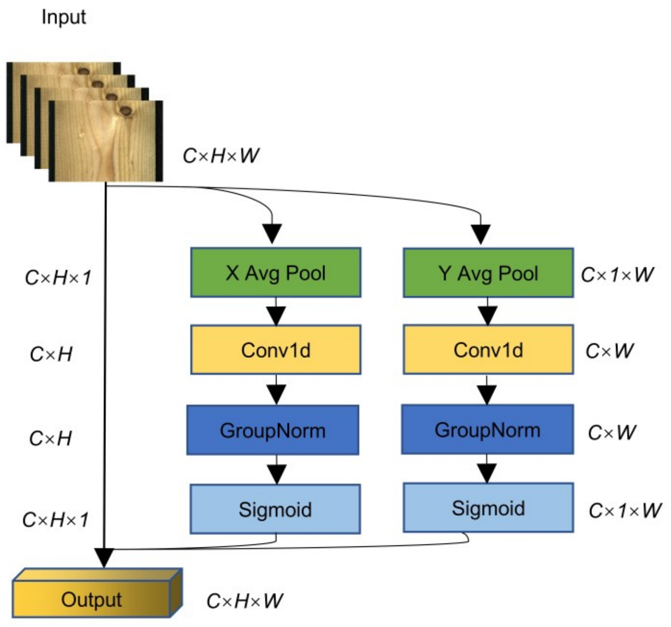

Fig. 7. ELA attention mechanism structure

Detection systems must precisely capture defect spatial coordinate information. To strengthen the network's discriminative capability for positions, ELA proposed by Xu *et al*. (2025) was introduced into the YOLOv12n neck. Its structure is detailed in Fig. 7.

Input features had dimensions $C \times H \times W$, where C denoted channel count, H represented height, and W represented width. ELA divided the input into two independent pathways for position encoding. The left branch employed XAvgPool for average pooling along the horizontal direction, compressing each row's features to obtain a $C \times H \times 1$ tensor. The right branch used YAvgPool for pooling along the vertical direction, averaging each column to obtain a $C \times 1 \times W$ tensor. The pooled sequences were respectively fed into one-dimensional convolution Conv1d with kernel size set to 7; the left side output $C \times H$ encoding, while the right side output $C \times W$ encoding. A kernel size of 7 could cover sufficient neighborhood ranges.

Subsequently, GroupNorm group normalization was used to perform standardized processing on features. This approach divided channels into groups before calculating statistics, receiving less influence from sample batch size compared to conventional normalization methods. After normalization, the Sigmoid activation function was used to generate attention weight of $C \times H \times 1$ and $C \times 1 \times W$. These two weights multiplied to obtain a complete spatial weight map, which finally performed element-wise multiplication with the original input, outputting enhanced features of $C \times H \times W$.

ELA enabled the network to explicitly clarify defect positions in feature maps, with particularly remarkable improvement effects on localization accuracy for targets such as knots with cracks exhibiting significant edge-center response differences. Compared to global attention mechanisms requiring computation of relationships among all positions, this dimension-reduction-then-convolution processing approach substantially reduced computational load.

RESULTS AND DISCUSSION

Wood Surface Defects Dataset

A subset of the large-scale wood surface defect image dataset publicly released by Kodytek *et al*. (2022) was adopted. The original dataset encompassed annotations for 10 defect categories. According to research requirements, seven common defect types with sufficient sample quantities were retained after screening for model training and evaluation: Dead_Knot, Live_Knot, knot_with_crack, Crack, resin, Marrow, and Knot_missing, totaling 3,593 valid images. To match detection network input specifications, all images were preprocessed to 640×640 pixel resolution. The dataset was randomly divided into training, validation, and test sets at an 8:1:1 ratio. During the training phase, the mosaic data augmentation strategy was enabled to enrich sample diversity and enhance the model's adaptability to complex morphologies of wood surface defects. The seven defect categories included in the dataset are detailed in Fig. 8.

The sample distribution across defect categories presented a characteristic imbalance that merits attention, as depicted in Fig. 9. Live_Knot and Dead_Knot dominated the dataset composition, contributing 45.2% and 32.5% of total samples respectively. The remaining five categories occupied considerably smaller proportions: Resin at 7.3%, Crack at 5.8%, knot_with_crack at 5.7%,

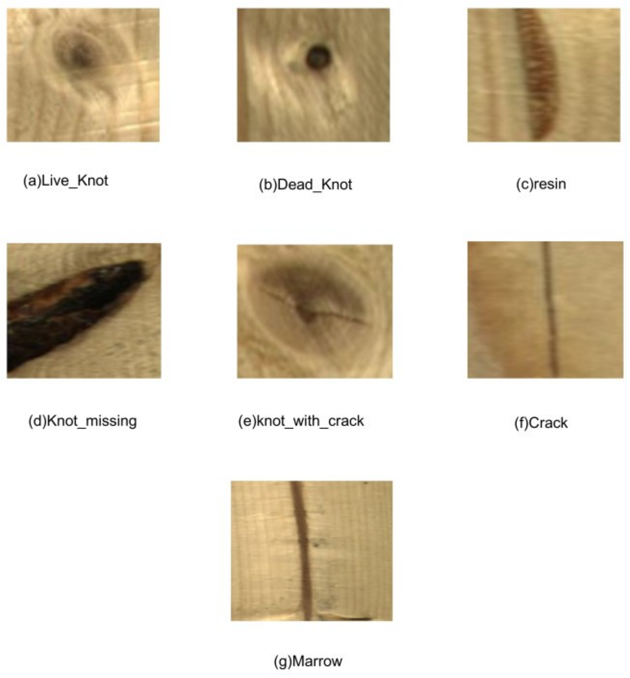

Fig. 8. Dataset defect categories

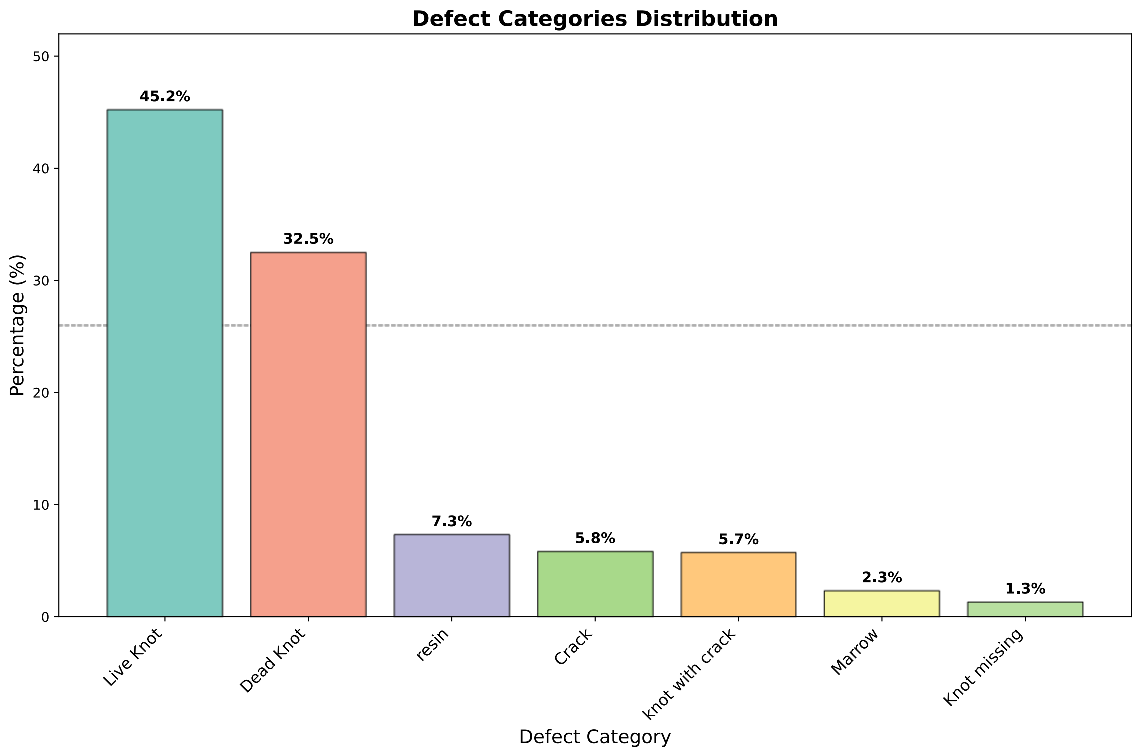

Fig. 9. Histogram of defect category distribution

Marrow at 2.3%, and Knot_missing at merely 1.3%. Such distributional disparity originates from the intrinsic biological and mechanical processes governing wood defect genesis. Knots—whether dead or live—arise inevitably at branch-trunk junctions throughout tree growth, rendering them pervasive across virtually all timber specimens.

Knot_with_crack demands the concurrent manifestation of knot presence alongside crack propagation, a compound condition triggered predominantly by differential shrinkage stresses during kiln drying or ambient seasoning. Resin accumulation depends upon species-specific secretory physiology characteristic of certain conifers, whereas Marrow defects emerge from suboptimal sawing practices that fail to exclude the pith zone. Knot_missing represents perhaps the rarest category, requiring complete knot dislodgement through either natural abscission or deliberate excision during processing—both statistically uncommon events.

The procedures used in the present work deliberately preserved the native distributional characteristics without resorting to oversampling or synthetic augmentation, since such interventions risk contaminating the training-test partition boundary and consequently yielding inflated performance metrics that misrepresent true generalization capacity. Maintaining fidelity to the original distribution not only facilitates reproducibility across independent research efforts but also ensures that the evaluation conditions approximate the authentic defect occurrence patterns encountered on industrial grading lines.

Experimental Environment and Parameter Settings

The experimental environment was constructed based on the PyTorch deep learning framework, with the operating platform being the Linux operating system. The GPU was NVIDIA GeForce RTX 4090 (24GB memory), the CPU was AMD EPYC 7K62 48-Core Processor 2.6GHz, and GPU acceleration utilized CUDA 12.1.

Key hyperparameters during the training process were set as follows: batch size was set to 16, patience for early stopping mechanism was set to 50, Stochastic Gradient Descent (SGD) was selected as the parameter optimization strategy, the number of workers was set to 4, total training epochs were determined as 200, and the initial learning rate was set to 0.01. Additionally, no pre-trained weights were introduced for initialization during the model training process.

The detailed hyperparameter configuration is presented in Table 1. The selection of these hyperparameters was primarily guided by the work of Yan *et al.* (2025), who conducted systematic experiments on wood defect detection tasks using similar parameter configurations.

Table 1. Detailed Table of Hyperparameter Configuration

Hyperparameter	Value	Justification
Batch Size	16	Optimal for RTX 4090 memory; balances gradient stability and training speed
Initial Learning Rate	0.01	Standard for SGD optimizer in YOLO models; validated through grid search [0.001, 0.01, 0.1]
Epochs	200	Sufficient for convergence; early stopping prevents overfitting
Patience	50	Allows adequate exploration while preventing unnecessary computation
Optimizer	SGD	Superior generalization compared to Adam for object detection
Momentum	0.937	Standard YOLO configuration for stable convergence
Weight Decay	0.0005	Regularization to prevent overfitting; standard practice

Evaluation Metric System

To ensure comprehensiveness and accuracy of model performance evaluation, a multi-dimensional evaluation metric system was constructed, specifically including: mean

Average Precision (mAP), Parameters (Params/M), model Size (Size/MB), Floating Point Operations (FLOPs/G), inference speed FPS (frames per second, f/s), and Inference Time. Among these, mAP@0.5 represents the average precision at IoU threshold 0.5, while mAP@0.5:0.95 represents the mean average precision across IoU thresholds from 0.5 to 0.95 (step size 0.05). Params denotes Parameters, FLOPs indicates Floating Point Operations, FPS represents Frames Per Second, and Inference Time was calculated as Inference Time = 1/FPS, representing the time required to process a single frame. MB denotes megabytes, M represents the unit million, G indicates 1 billion floating-point operations per second, f represents frames, and s represents seconds.

Ablation Experiments

To verify the effectiveness of each improved module in MFWSD-YOLO, ablation experiments were conducted on this dataset. The experiments employed identical hyperparameters and training environments, progressively integrating the four improvement points of ADown, LSGNDH, PLRFIB, and ELA through eight ablation experiments. Each experiment selected the best weights for detection, obtaining validation results as shown in Table 2.

Table 2. Ablation Experiments

Model	ADown	LSGNDH	PLRFIB	ELA	mAP @0.5	mAP@0.5 :0.95	Params/M	Size/MB	FLOPs/G	FPS(f/s)
YOLOv12n					0.6403	0.3353	2.56	5.3	6.3	92.3
YOLOv12n	√				0.6684	0.3499	2.05	4.4	5.2	92.2
YOLOv12n		√			0.6949	0.3796	2.34	4.9	5.1	119.3
YOLOv12n			√		0.7143	0.3602	1.80	3.8	5.4	186.7
YOLOv12n				√	0.6959	0.3474	2.53	5.3	5.8	104.2
YOLOv12n	√	√			0.6991	0.3670	1.89	4.0	4.4	127.0
YOLOv12n	√	√	√		0.7253	0.3805	1.18	2.6	4.1	160.5
YOLOv12n	√	√	√	√	0.7293	0.3870	1.21	2.6	4.2	195.6

The baseline YOLOv12n model achieved mAP@0.5 of 64.03% and mAP@0.5:0.95 of 33.53%. After integrating the ADown module into the baseline model, mAP@0.5 increased to 66.84%, representing a 2.81% improvement over the baseline. This module adopted a spatial rearrangement mechanism, achieving lossless information transmission through channel-dimensional feature reorganization, which proved particularly critical for preserving spatial continuity of elongated defects such as pith spots. Params decreased to 2.05 M, a reduction of 19.92%, and FLOPs decreased to 5.2G, a reduction of 17.46%.

The LSGNDH detection head elevated mAP@0.5 by 5.46% and increased mAP@0.5:0.95 by 4.43%, representing the most significant accuracy improvement among single modules. Its core lay in cross-scale gradient sharing and normalization strategies, demonstrating outstanding scale adaptability. FLOPs decreased to 5.1 G, a reduction of

19.05%, and FPS reached 119.3 f/s.

The PLRFIB module caused mAP@0.5 to climb to 71.43%, an improvement magnitude reaching 7.40%. This module constructed multi-level semantic aggregation pathways through progressive receptive field expansion, with low layers capturing local texture mutations of cracks and high layers integrating global geometric morphologies of dead knot boundaries. Params decreased to 1.80 M, a compression of 29.69%, Size was 3.8 MB, a reduction of 28.30%, and FPS reached 186.7 f/s, representing 2.02 times the original.

The ELA mechanism increased mAP@0.5 by 5.56%, strengthening feature responses in texture-dense regions through adaptive weighting within local receptive fields.

The combination of ADown and LSGNDH increased mAP@0.5 to 69.91%, an improvement of 5.88%, and mAP@0.5:0.95 increased by 3.17%, exhibiting synergistic gain effects in void boundary localization. Params decreased to 1.89 M, a reduction of 26.17%, and FLOPs decreased to 4.4 G, a reduction of 30.16%.

Three-module fusion elevated mAP@0.5 to 72.53%, an increase of 8.50%, and mAP@0.5:0.95 improved by 4.52%. Progressive receptive fields and adaptive detection heads demonstrated deep coupling on composite defects such as knots with cracks, forming multi-scale feature complementary enhancement. Params decreased to 1.18 M, a sharp reduction of 53.91%, FLOPs decreased to 4.1 G, a reduction of 34.92%, and FPS reached 160.5 f/s. Although Params and FLOPs represented optimal values among all experiments, other metrics still possessed certain improvement space.

After complete fusion of four modules, mAP@0.5 reached 72.93%, an improvement magnitude of 8.90%, and mAP@0.5:0.95 reached 38.70%, an increase of 5.17%, achieving optimal performance. Params was 1.21 M, a sharp reduction of 52.73%, Size was 2.6 MB, a reduction magnitude reaching 50.94%, FLOPs was 4.2 G, a decrease of 33.33%, and FPS reached 195.6 f/s, representing 2.12 times the original.

Comparative Experiments of Different Algorithms

To verify the detection effectiveness of MFWSD-YOLO, multiple mainstream algorithms were selected for comparative experiments, with results detailed in Table 3. To ensure experimental fairness, all experiments were conducted on identical hardware. For YOLO-series models, unified training configurations were applied, including batch size, learning rate, and optimization strategy. For non-YOLO architectures, their officially recommended configurations were adopted, as these methods originate from distinct detection paradigms with specialized loss functions and optimization schemes that would yield suboptimal convergence under arbitrarily imposed settings.

The proposed algorithm achieved an mAP@0.5 of 72.93%, representing improvements of 8.73%, 10.83%, and 7.73% compared to RT-DETR-R18, ATSS-R50, and TOOD-R50, respectively. mAP@0.5:0.95 reached 38.70%, representing improvements of 1.54%, 6.30%, and 6.00% compared to the three aforementioned algorithms, respectively.

Although RT-DETR-R18 possessed global modeling capability based on Transformer architecture, its Params of 19.90 M, FLOPs of 57.0 G, and Size of 77.0MB limited edge deployment due to computational intensity characteristics. The proposed algorithm achieved reduction magnitudes of 93.92%, 92.63%, and 96.62% in these three metrics compared to RT-DETR-R18.

Table 3. Comparative Experiments of Different Algorithms

Model	mAP@0.5	mAP@0.5:0.95	Params/M	FLOPs/G	Size/MB	FPS(f/s)
RT-DETR-R18	0.6420	0.3716	19.90	57.0	77.0	130.7
ATSS-R50	0.6210	0.3240	38.91	68.2	256.3	15.4
TOOD-R50	0.6520	0.3270	32.04	125.9	251.0	29.3
YOLOv3-tiny	0.6441	0.3419	12.13	18.9	23.3	133.4
YOLOv5n	0.6393	0.3043	2.51	7.1	5.0	103.3
YOLOv6n	0.6164	0.3073	4.23	11.8	8.3	114.0
YOLOv8n	0.6558	0.3433	3.01	8.1	6.0	114.5
YOLOv9t	0.6154	0.3278	1.97	7.6	4.4	96.8
YOLOv10n	0.5819	0.2971	2.27	6.5	5.5	105.8
YOLOv11n	0.6246	0.3373	2.58	6.3	5.2	125.9
YOLOv12n	0.6403	0.3353	2.56	6.3	5.3	92.3
YOLOv13n	0.6128	0.3211	2.45	6.1	5.2	47.6
This study	0.7293	0.3870	1.21	4.2	2.6	195.6

ATSS-R50 adopted an adaptive sample allocation strategy, but optimization only functioned during the training phase. Its Params of 38.91M, FLOPs of 68.2 G, and Size of 256.3 MB contrasted sharply with the proposed algorithm, which achieved Params reduction of 96.89%, FLOPs reduction of 93.84%, Size reduction of 98.99%, and FPS reaching 12.70 times that of ATSS-R50, demonstrating significant advantages in inference efficiency.

TOOD-R50 employed a task alignment mechanism but exhibited high model complexity and sensitivity to sample distribution. Its Params of 32.04 M, FLOPs of 125.9 G, and Size of 251.0 MB contrasted with the proposed algorithm, which achieved reductions of 96.22%, 96.66%, and 98.96% in Params, FLOPs, and Size, respectively, with FPS being 6.68 times that of TOOD-R50, significantly reducing computational overhead.

In YOLO series algorithm comparisons, the proposed algorithm's mAP@0.5 improved by 8.52%, 9.00%, 11.29%, 7.35%, 11.39%, 14.74%, 10.47%, and 11.65% compared to YOLOv3-tiny, YOLOv5n, YOLOv6n, YOLOv8n, YOLOv9t, YOLOv10n, YOLOv11n, and YOLOv13n, respectively. mAP@0.5:0.95 improved by 4.51%, 8.27%, 7.97%, 4.37%, 5.92%, 8.99%, 4.97%, and 6.59% compared to the aforementioned algorithms, respectively, while Params, FLOPs, and Size were only 1.21 M, 4.2 G, and 2.6MB, respectively, with FPS reaching 195.6 f/s, demonstrating significant lightweight advantages.

YOLOv3-tiny's Params of 12.13 M suffered from insufficient feature extraction depth. The proposed algorithm achieved a 90.02% reduction compared to it, representing breakthrough progress in parameter compression. YOLOv5n's FLOPs of 7.1G exhibited limitations in feature fusion strategy for detail preservation. The proposed algorithm reduced FLOPs by 40.85% compared to it.

YOLOv8n introduced the C2f module and decoupled head, but independent optimization branches increased network complexity. The proposed algorithm achieved a 59.80% reduction in Params compared to it. YOLOv10n adopted end-to-end design, but one-to-one label assignment also exhibited numerous deficiencies in dense scenarios. The proposed algorithm reduced FLOPs by 35.38% compared to it.

The corresponding radar chart is detailed in Fig. 10. From multi-metric distribution perspective, MFWSD-YOLO's positive metrics such as mAP@0.5, mAP@0.5:0.95, and FPS were positioned at the outer edge of the radar chart, while negative metrics such as Params and FLOPs were close to the center, visually confirming its comprehensive advantages of high accuracy, fast speed, and low resource consumption. In summary, the MFWSD-YOLO algorithm maintained high detection accuracy while possessing low computational complexity and model scale.

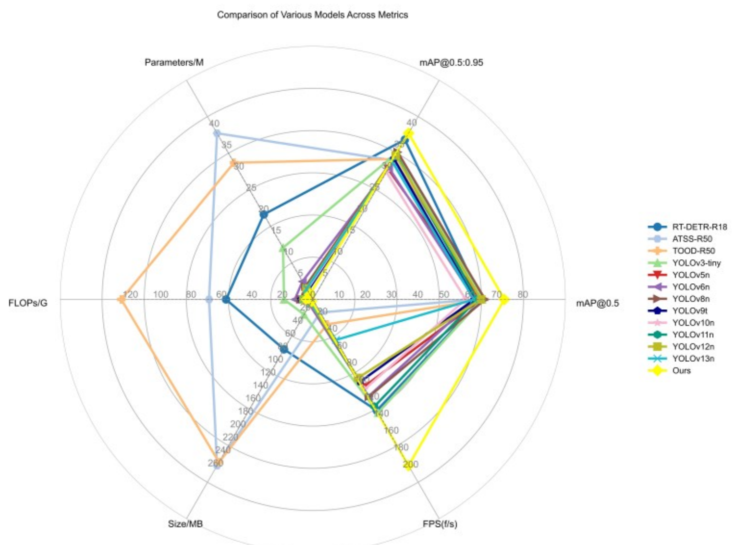

Fig. 10. Radar chart for comparison of different algorithms

Comparative Experiments with Different Downsampling Methods

To verify the superiority of ADown, five widely used downsampling operations were selected—WaveletPool, ContextGuidedDown, SPDConv, PSConv, and wConv—for comparative experiments with ADown on YOLOv12n, as shown in Table 4.

Table 4. Comparative Experiments with Different Downsampling Methods

Conv	mAP@0.5	mAP@0.5:0.95	Params/M	Size/MB	FLOPs/G	FPS(f/s)
YOLOv12n	0.6403	0.3353	2.56	5.3	6.3	92.3
WaveletPool	0.6298	0.3233	2.05	4.3	5.1	92.1
ContextGuidedDown	0.6575	0.3487	3.47	7.1	9.0	92.2
SPDConv	0.6485	0.3449	4.54	9.1	11.3	86.9
PSConv	0.6639	0.3415	2.41	5.1	6.3	76.1
wConv	0.6085	0.3089	2.54	5.3	4.6	92.1
ADown	0.6684	0.3499	2.05	4.4	5.2	92.2

Experimental results indicated that ADown achieved mAP@0.5 of 66.84% and mAP@0.5:0.95 of 34.99%, representing improvements of 2.81% and 1.46% compared to the baseline model, respectively. Compared to WaveletPool, ADown led by 3.86% in mAP@0.5, indicating that the dual-path design possessed greater advantages in capturing wood defect details.

Although ContextGuidedDown reached 34.87% in mAP@0.5:0.95, its Params reached 3.47M and FLOPs was 9.0G, with resource consumption far exceeding ADown.

SPDConv's Params and FLOPs reached 4.54 M and 11.3 G, respectively, with FPS of only 86.9 f/s, exhibiting obvious gaps in comprehensive performance. PSConv exhibited the lowest FPS. wConv exhibited disadvantages across all metrics.

Comprehensive analysis indicated that while ADown achieved the highest detection accuracy, Params was only 2.05 M, Size was 4.4 MB, FLOPs was 5.2 G, and it maintained detection speed of 92.2 f/s, fully demonstrating its balanced advantages between accuracy and efficiency.

Comparative Experiments with Different Detection Heads

To verify the effectiveness of the LSGNDH detection head, on the basis of introducing the ADown module into YOLOv12n, five mainstream detection heads—dyhead, EfficientHead, LQEHead, MultiSEAMHead, and SEAMHead—were integrated for comparative experiments. Experimental results are shown in Table 5.

Data indicated that LSGNDH's mAP@0.5 reached 69.91%, representing a 5.88% improvement compared to the baseline model YOLOv12n's 64.03%. mAP@0.5:0.95 improved from 33.53% to 36.70%, an increase of 3.17%, achieving optimal detection accuracy among all comparison methods. This benefited from the shared convolution mechanism effectively integrating multi-scale features.

dyhead elevated mAP@0.5 to 66.43% through multiple attention mechanisms, but computational complexity resulted in FPS of only 77.0 f/s, with significantly insufficient inference efficiency. Although EfficientHead's Params was only 1.79 M, mAP@0.5 was only 63.66%.

Table 5. Comparative Experiments with Different Detection Heads

Head	mAP@0.5	mAP@0.5:0.95	Params/M	Size/MB	FLOPs/G	FPS(f/s)
YOLOv12n	0.6403	0.3353	2.56	5.3	6.3	92.3
dyhead	0.6643	0.3366	2.57	5.4	6.3	77.0
EfficientHead	0.6366	0.3327	1.79	3.8	4.0	119.5
LQEHead	0.6485	0.3356	2.06	4.4	5.2	113.7
MultiSEAMHead	0.6838	0.3619	4.07	8.3	4.9	71.1
SEAMHead	0.6708	0.3469	1.97	4.2	4.6	93.4
LSGNDH	0.6991	0.3670	1.89	4.0	4.4	127.0

LQEHead's mAP@0.5 was 64.85%. Although FPS reached 113.7 f/s, Params was 2.06M and FLOPs reached 5.2 G, failing to achieve ideal balance in accuracy and computational efficiency. MultiSEAMHead's mAP@0.5 was 68.38%, but its Params reached 4.07 M and FPS was the lowest, limiting its practical deployment value.

SEAMHead's mAP@0.5 was 67.08%, Params was 1.97 M, and FPS was 93.4 f/s, demonstrating good performance in lightweight design, but accuracy remained lower than LSGNDH. In contrast, LSGNDH controlled Params at 1.89 M, representing a 26.17% reduction compared to the baseline model, compressed Size to 4.0 MB, reduced FLOPs to 4.4 G, while FPS reached 127.0 f/s.

Experimental results indicated that LSGNDH achieved simultaneous improvement in detection accuracy and inference speed under the premise of maintaining lightweight design.

Comparative Experiments with Different Attention Mechanisms

To verify the effectiveness of ELA in wood surface defect detection, on the basis of YOLOv12n fusing ADown, LSGNDH, and PLRFIB, ELA was compared with six classic attention mechanisms—CAA (Cai *et al*. 2024), AFGCAttention, CPCA (Huang *et al*. 2024), LSKA (Lau *et al*. 2024), LSKBlock, and LocalWindowAttention—as shown in Table 6.

Table 6. Comparative Experiments with Different Attention Mechanisms

Model	mAP@0.5	mAP@0.5:0.95	Params/M	Size/MB	FLOPs/G	FPS (f/s)
YOLOv12n	0.6403	0.3353	2.56	5.3	6.3	92.3
CAA	0.7046	0.3601	1.20	2.6	4.3	169.2
AFGCAttention	0.7070	0.3666	1.19	2.6	4.1	166.2
CPCA	0.7121	0.3758	1.20	2.6	4.4	160.7
LSKA	0.7221	0.3761	1.19	2.6	4.2	173.5
LSKBlock	0.6895	0.3689	1.21	2.6	4.4	163.6
LocalWindowAttention	0.7123	0.3770	1.20	2.6	4.3	164.4
ELA	0.7293	0.3870	1.21	2.6	4.2	195.6

Analysis revealed that CAA's global pooling lost crack local information; AFGCAttention's matrix operations exhibited computational redundancy when processing pith spots; CPCA faced limitations in spatial localization for resin pockets; LSKA's dilated convolution produced grid effects at dead knot boundaries; LSKBlock exhibited feature allocation conflicts when addressing complex defect features; LocalWindowAttention's window partitioning limited global modeling. ELA preserved directional and positional information through bidirectional adaptive pooling, achieving optimal balance between accuracy and speed.

Generalization Experiments

To comprehensively evaluate the generalization performance of the MFWSD-YOLO model, generalization experiment validation was conducted on three types of publicly available datasets.

First, the OULU-DET (Luo *et al*. 2025) wood surface defect dataset from the University of Oulu was adopted. This dataset contained 3,773 images encompassing six typical defect categories: dry knot, sound knot, edge knot, small knot, split, and wave.

Second, the PCB_DATASET (Ding *et al*. 2019) printed circuit board defect dataset released by Peking University was selected. The original dataset contained 693 images, including six defect types: spurious copper, open circuit, mouse bite, short circuit, missing hole, and spur. Given the small sample scale, it was expanded to 6,330 images through data augmentation techniques.

Finally, the authoritative solar panel defect dataset PVEL-AD (Su *et al*. 2022) was introduced. This dataset encompassed 12 defect types: black_core, corner, crack, finger, fragment, horizontal_dislocation, printing_error, scratch, short_circuit, star_crack, thick_line, and vertical_dislocation. From this, 3,645 high-quality images were carefully selected for experiments.

The hyperparameter settings, training environment, and dataset division strategy for the three aforementioned datasets remained consistent with ablation experiments. Experimental results are shown in Table 7.

Table 7. Generalization Experiments

Datasets	Model	mAP@0.5	mAP@0.5:0.95	FPS(f/s)	InferenceTime(s)
OULU-DET	YOLOv12n	0.8699	0.5308	107.2	0.0093
	MFWSD-YOLO	0.9584	0.6248	125.2	0.0080
PCB_DATASET	YOLOv12n	0.9696	0.6505	96.4	0.0104
	MFWSD-YOLO	0.9849	0.7474	134.1	0.0075
PVEL-AD	YOLOv12n	0.7366	0.4997	80.5	0.0124
	MFWSD-YOLO	0.8245	0.5194	122.4	0.0082

Experimental results demonstrated that on OULU-DET, mAP@0.5 improved from 86.99% to 95.84%, mAP@0.5:0.95 leaped from 53.08% to 62.48%, FPS increased to 125.2 f/s, and inference time decreased from 0.0093s to 0.0080s. Cross-domain experiments indicated that on PCB_DATASET, mAP@0.5:0.95 improved from 65.05% to 74.74%, with FPS reaching 134.1 f/s and inference time reduced to 0.0075s per frame. On PVEL-AD, mAP@0.5 increased from 73.66% to 82.45%, with FPS reaching 122.4 f/s and inference time compressed from 0.0124 s to 0.0082 s, representing a 33.9% reduction in processing latency.

These excellent results stemmed from the powerful feature expression capability constructed by the proposed method, enabling the model to exhibit outstanding generalization performance in both wood and cross-domain detection tasks.

Analysis of mAP@0.5 Improvement Effect for Various Defect Categories

As shown in Table 8, the improved model achieved gains across all defect categories.

Table 8. Comparison of mAP@0.5 Before and After for Various Defect Types

Model	Live_Knot	Dead_Knot	resin	Knot_missing	knot_with_crack	Marrow	Crack
	mAP@0.5	mAP@0.5	mAP@0.5	mAP@0.5	mAP@0.5	mAP@0.5	mAP@0.5
YOLOv12n	0.8247	0.8847	0.6656	0.6296	0.6224	0.5655	0.2898
MFWSD-YOLO	0.8319	0.8891	0.6785	0.6362	0.6358	0.8016	0.6320

Crack mAP@0.5 increased from 0.2898 to 0.6320, a gain of 34.22%, which can be attributed to PLRFIB's progressive receptive field expansion effectively capturing the extension trajectories of elongated, variably-oriented crack structures. Marrow improved by 23.61%, as the strip-shaped small-target characteristics benefited substantially from multi-scale semantic fusion. Knot_with_crack gained 1.34%, owing to LSGNDH’s cross-scale parameter sharing mechanism that enabled coordinated expression of composite features. Resin improved by 1.29%, where ELA’s spatial weight modeling enhanced localization precision for lens-shaped boundaries. Knot_missing increased by 0.66%, with Adown's dual-path architecture preserving boundary details of cavity structures. Dead_Knot and Live_Knot showed marginal improvements of 0.44% and 0.72% respectively, due to their already high baseline precision limiting further enhancement potential.

Visualization Results Analysis

To deeply demonstrate the detection performance advantages of MFWSD-YOLO, the GradCAM++ method (Chattopadhay *et al*. 2018) was adopted to obtain heatmaps, comparing the performance differences between the baseline model YOLOv12n and the improved model in detecting seven wood defect types. Visualization results are detailed in Fig. 11.

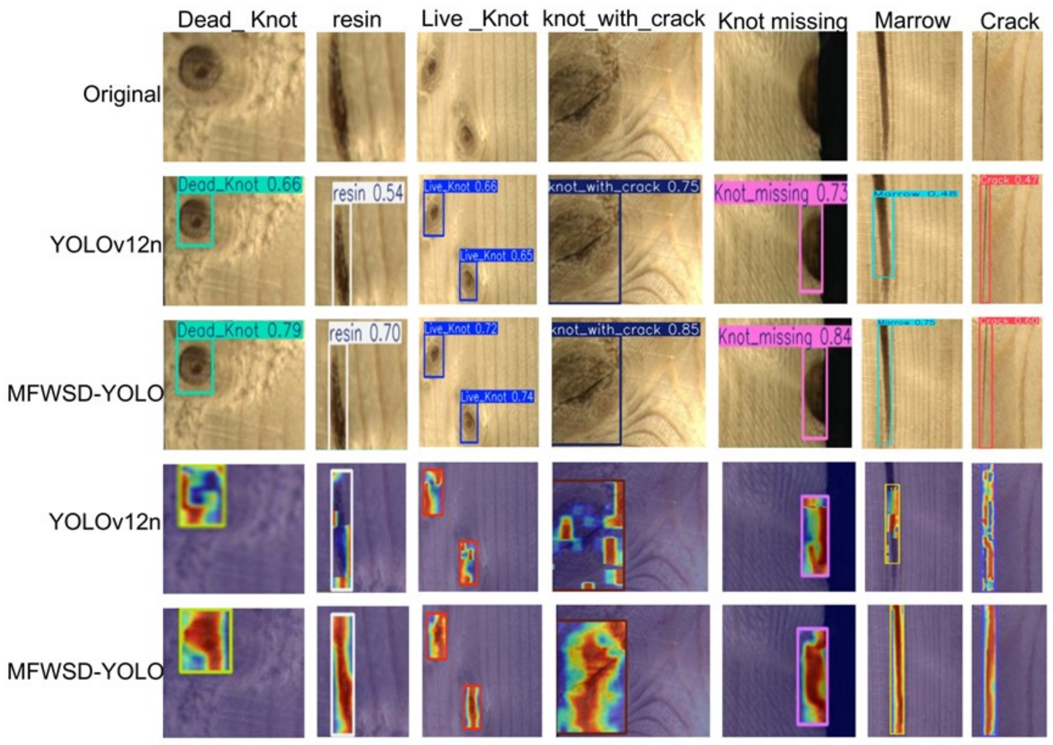

Fig. 11. Visualization results

Figure 11 presents the attention distribution of both models through GradCAM++ heatmaps. The color gradient encodes confidence levels: red denotes regions where the model exhibits strongest responses, yellow and green indicate intermediate activation intensities, whereas blue and purple correspond to areas receiving minimal attention. The quantitative comparison reveals consistent performance gains achieved by MFWSD-YOLO across all seven defect categories. Two small-scale defect types merit particular discussion. Marrow detection confidence rose from 48% to 75%, yielding a 27% improvement margin. Crack recognition similarly benefited, with confidence advancing from 47% to 60%. These gains can be attributed to the progressive receptive field expansion mechanism embedded in PLRFIB, which preserves fine-grained spatial details during feature aggregation. The composite defect category knot_with_crack warrants separate examination given its inherent detection complexity. This defect type simultaneously presents crack linear extension patterns and knot circular boundary characteristics, demanding the detector to integrate geometrically heterogeneous features within a unified representation. MFWSD-YOLO elevated the detection confidence from 75% to 85% for this category. The dual-path architecture of PLRFIB contributes to this improvement: the preservation branch retains knot boundary information while the processing branch extracts crack directional features through cascaded convolutions, enabling coherent feature fusion for such morphologically diverse targets. The heatmap visualization corroborates these quantitative findings. YOLOv12n generates fragmented activation patterns characterized by scattered red regions and extensive yellow-green transitional zones, indicating imprecise attention allocation. The model struggles to distinguish defect boundaries from surrounding wood textures, particularly when processing targets with irregular geometries. MFWSD-YOLO produces markedly different activation maps: red high-response regions form compact clusters that align closely with actual defect contours, accompanied by sharper transitions toward peripheral areas. This concentrated activation pattern demonstrates that the ELA mechanism successfully guides the network toward defect-relevant spatial locations while suppressing background interference, thereby enhancing localization precision for wood surface defects exhibiting diverse morphological characteristics.

To thoroughly evaluate the classification performance, normalized confusion matrices of YOLOv12n and MFWSD-YOLO across four datasets were constructed, as illustrated in Fig. 12.

On the large-scale wood defect dataset, the baseline model achieved merely 40% recall for Crack, with the lower-left triangular region revealing substantial false negatives where crack samples were misclassified into other categories including background. This deficiency stemmed from the inherent characteristics of cracks: their extremely fine line widths and variable extension directions caused severe feature attenuation during conventional downsampling. The ADown module addressed this issue through its dual-path architecture, where the max pooling branch preserved edge sharpness while the convolutional branch retained textural details, jointly elevating Crack recall to 57%. Notable confusion between Live_Knot and Dead_Knot appeared in the confusion matrix, attributable to their morphological similarity in circular contours and color gradients, though dead knots typically exhibit darker peripheral boundaries. The LSGNDH module mitigated this inter-class confusion through cross-scale parameter sharing, enabling semantic information from the P5 layer to guide fine-grained discrimination at the P3 layer. For the Dead_Knot category, recall improved from 70% to 78%, demonstrating enhanced boundary recognition. Regarding false positives, the upper-right triangular region indicated that background and other categories were occasionally misclassified as target defects; the ELA mechanism suppressed such false alarms by strengthening spatial position encoding, particularly effective for cavity-type defects such as Knot_missing, where recall improved from 55% to 66%.

On the OULU-DET dataset, small_knot recall improved substantially from 74% to 92%, with confusion matrix values showing significant false negative reduction for small-scale defects. The wave category also demonstrated notable improvement, with recall increasing from 85% to 95%, benefiting from the progressive feature integration of PLRFIB. The PCB_DATASET experiments validated cross-domain adaptability, where open_circuit recall rose from 94% to 98%, and mouse_bite recall improved from 91% to 95%.

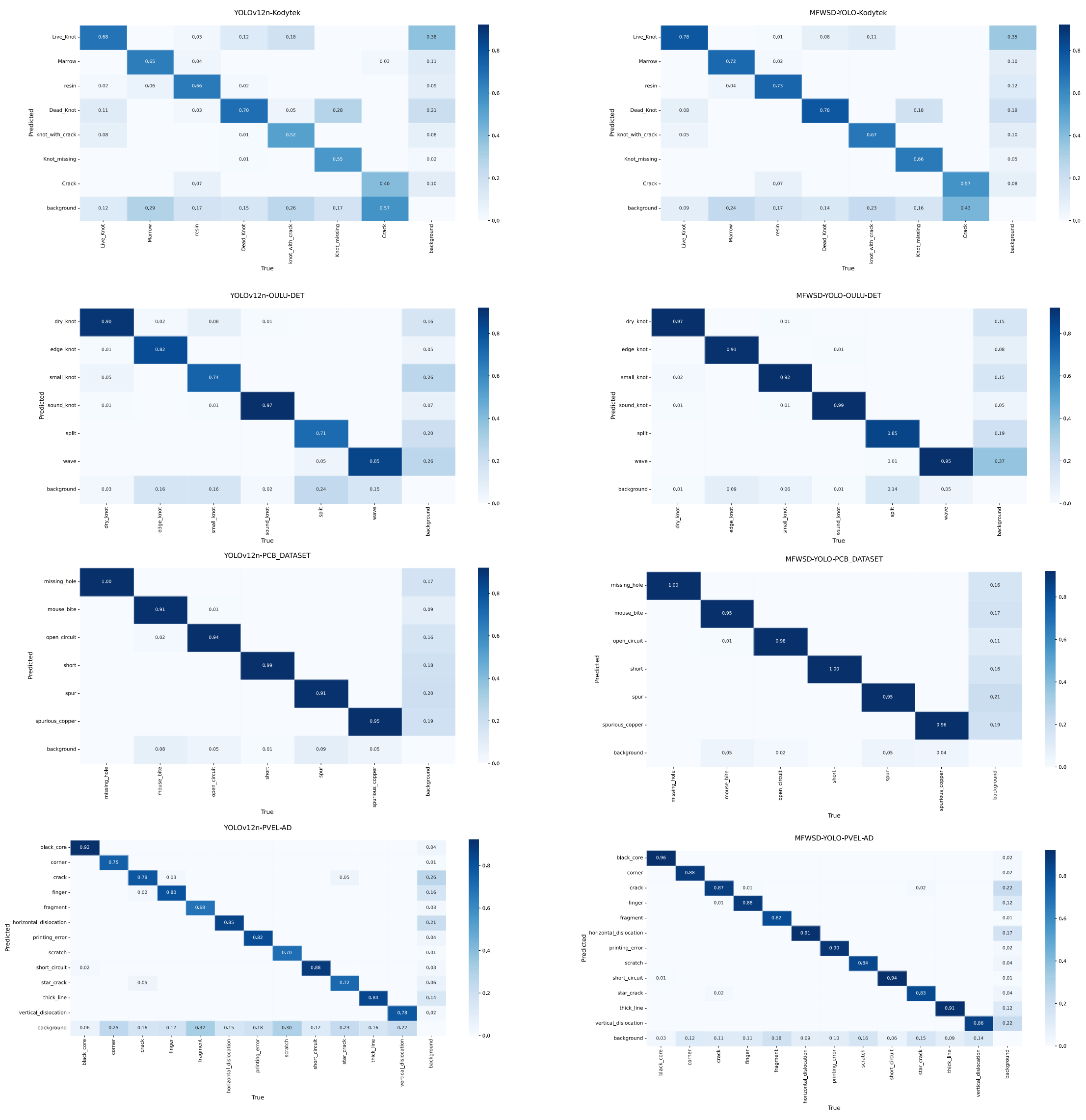

Fig. 12. Confusion matrix comparison chart

On the PVEL-AD dataset with twelve defect categories, scratch recall improved from 70% to 84%, and the confusion between crack and star_crack was notably alleviated through enhanced feature discrimination. The star_crack category showed particularly strong improvement from 72% to 83% recall, benefiting from the progressive receptive field expansion of PLRFIB that enhanced discrimination between linear and radial fracture patterns. Fragment recall also increased from 68% to 82%, demonstrating the model's enhanced capability in detecting small-scale defects. The overall matrix distributions confirm that MFWSD-YOLO systematically reduced both false negative and false positive rates through the targeted contributions of each proposed module.

CONCLUSIONS

1. The improved algorithm achieved significant performance improvements on a public wood defect dataset, with mAP@0.5 reaching 72.93% and mAP@0.5:0.95 reaching 38.70%, representing improvements of 8.90% and 5.17% respectively compared to the baseline model YOLOv12n. Meanwhile, model parameters were reduced to 1.21 M, computational complexity decreased to 4.2 G, and model size was compressed to 2.6 MB, representing reductions of 52.73%, 33.33%, and 50.94% respectively compared to the baseline model, with inference speed reaching 195.6 frames per second, achieving an effective balance between detection accuracy and model lightweight design.
2. Ablation experiments demonstrated the effectiveness of each improved module. The adaptive downsampling module (ADown) improved mAP@0.5 by 2.81%; the lightweight shared convolution and group normalization detection head (LSGNDH) improved mAP@0.5 by 5.46%; the progressive lightweight reparameterized feature integration block (PLRFIB) improved mAP@0.5 by 7.40%; and the efficient local attention mechanism (ELA) improved mAP@0.5 by 5.56%. The synergistic effect of the four modules achieved maximum performance improvement.
3. Comparative experiments demonstrated distinct advantages over mainstream algorithms. Against Transformer-based and ResNet-based architectures, the proposed method achieved parameter reductions exceeding 90% while maintaining competitive accuracy. Compared to YOLO variants, mAP@0.5 improvements ranged from 7.35% to 14.74%, validating the effectiveness of multi-scale feature fusion and attention mechanisms in detecting small defects within complex wood textures.
4. Cross-dataset generalization experiments verified the algorithm’s robustness. On the OULU-DET wood defect dataset, mAP@0.5 improved from 86.99% to 95.84%; on the cross-domain PCB_DATASET, mAP@0.5:0.95 improved from 65.05% to 74.74%; and on the PVEL-AD photovoltaic defect dataset, mAP@0.5 improved from 73.66% to 82.45%, demonstrating the algorithm's adaptability across different application scenarios.

ACKNOWLEDGMENTS

This research was financially supported by the National Natural Science Foundation of China (Grant Nos. 52074305, 62572173), and the Science and Technology Research Project of Henan Province (Grant No. 212102210005). Additionally, the work was supported in part by the Open Research Project of Henan Engineering Research Center of Optoelectronic Detection and Sensing Integration (2025) (KF202502), as well as the Basic Research Project (Category B) under the Basic Scientific Research Business Expenses of Henan Polytechnic University (NSFRF240621).

Use of Generative AI

The authors confirm that ChatGPT was used solely for the translation and polishing of certain parts of the text. No generative AI tools were employed in the creation or modification of any figures, graphs, diagrams, or data representations. All AI-assisted content has been carefully reviewed by the authors to ensure accuracy and compliance with publication ethics.

REFERENCES CITED

Cai, X., Lai, Q., Wang, Y., Wang, W., Sun, Z., and Yao, Y. (2024). "Poly kernel inception network for remote sensing detection," in: *2024 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Seattle, USA, pp. 27706-27716. https://doi.org.10.1109/CVPR52733.2024.02617

Chattopadhay, A., Sarkar, A., Howlader, P., and Balasubramanian, V. (2018). "Grad-cam++: Generalized gradient-based visual explanations for deep convolutional networks," In: *2018 IEEE winter conference on applications of computer vision (WACV)* , pp. 839-847. https://doi.org.10.1109/WACV.2018.00097

Chen, Y., Sun, C., Ren, Z., and Na, B. (2023). "Review of the current state of application of wood defect recognition technology," *BioResources* 18(1), 2288-2302. https://doi.org.10.15376/biores.18.1.Chen

Chen, Z., Feng, J., Zhu, X., and Wang, B. (2025). "YOLOv8-OCHD: A lightweight wood surface defect detection method based on improved YOLOv8," *IEEE Access* 13, 84435-84450. https://doi.org.10.1109/ACCESS.2025.3569175

Deng, Z., Liu, H., Tu, J., Ye, S., Liao, N., and Lai, G. (2025). "SGS-YOLO: Method for detecting dress code violations by airport security personnel," *Laser & Optoelectronics Progress* 62(6), 420-428. https://doi.org.10.3788/LOP241729

Ding, R., Dai, L., Li, G., and Liu, H. (2019). "TDD-Net: A tiny defect detection network for printed circuit boards," *CAAI Transactions on Intelligence Technology* 4(2), 110-116. https://doi.org.10.1049/trit.2019.0019

Feng, C., Zhong, Y., Gao, Y., Scott, M., and Huang, W. (2021). "TOOD: Task-aligned one-stage object detection," in: *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, Montreal, Canada, pp. 3490-3499. https://doi.org.10.1109/ICCV48922.2021.00349

Girshick, R. (2015). "Fast r-cnn," in: *Proceedings of the IEEE International Conference on Computer Vision*, Santiago, Chile, pp. 1440-1448. https://doi.org.10.1109/ICCV.2015.169

He, K., Gkioxari, G., Dollár, P., and Girshick, R. (2017). "Mask R-CNN," in: *2017 IEEE International Conference on Computer Vision*, Venice, Italy, pp. 2961-2969. https://doi.org.10.1109/ICCV.2017.322

Hu, H., Huang, L., Yang, H., and Chen, Y. (2024). "Improved YOLOv8n lightweight honeycomb ceramic defect-detection algorithm," *Laser & Optoelectronics Progress* 61(22), 160-169. https://doi.org.10.3788/LOP240670

Hu, K., Wang, B., Shen, Y., Guan, J., and Cai, Y. (2020). "Defect identification method for poplar veneer based on progressive growing generated adversarial network and MASK R-CNN model," *BioResources* 15(2), 3041-3052. https://doi.org.10.15376/biores.15.2.3041-3052

Huang, H., Chen, Z., Zou, Y., Lu, M., and Chen, C. (2024). "Channel prior convolutional attention for medical image segmentation," *Computers in Biology and Medicine* 178, article 108784. https://doi.org.10.1016/j.compbiomed.2024.108784

Hubbe, M. (2017). "To repair or not to repair cracked wood," *BioResources* 12(4), 6904-6906. https://doi.org.10.15376/biores.12.4. 6904-6906

Jiang, C., He, X., and Xiang, J. (2024). "LOL-YOLO: Low-Light object detection incorporating multiple attention mechanisms," *Computer Engineering and Applications* 60(24), 177-187. https://doi.org.10.3778/j.issn.1002-8331.2406-0424

Kodytek, P., Bodzas, A., and Bilik, P. (2022). "A large-scale image dataset of wood surface defects for automated vision-based quality control processes," *F1000Research* 10, article 581. https://doi.org.10.12688/f1000research.52903.2

Kurdthongmee, W., and Suwannarat, K. (2019). "Locating wood pith in a wood stem cross sectional image using YOLO object detection," in: *2019 IEEE 15th International Joint Conference on Computer Science and Software Engineering (JCSSE)*, Nakhon Si Thammarat, Thailand, pp. 1-6. https://doi.org.10.1109/TAAI48200.2019.8959823

Lau, K., Po, L., and Rehman, Y. (2024). "Large separable kernel attention: Rethinking the large kernel attention design in CNN," *Expert Systems with Applications* 236, article 121352. https://doi.org.10.1016/j.eswa.2024.121352

Li, Y., and Peng, Y. (2024). "Research on improved YOLOv8n wood surface defect detection algorithm," *Journal of Heilongjiang University of Technology (Comprehensive Edition)* 24(11), 86-93. https://doi.org.10.16792/j.cnki.1672-6758.2024.11.015

Ling, J., and Xie, Y. (2022). "Research on wood defects classification based on deep learning," *Wood Research* 67(1), 147-156. https://doi.org.10.37763/WR.1336-4561/67.1.147156

Long, Y., Xin, Z., Chu, Y., and Lin, W. (2025). "Research progress in surface defect detection for wood and its processed products based on deep learning," *World Forestry Research* 38(3), 84-91. https://doi.org.10.13348/j.cnki.sjlyyj.2025.0052.y

Luo, Q., Xu, W., Su, J., Yang, C., Gui, W., and Silvén, O. (2025). "Efficient adaptation of visual foundation models for wood defect segmentation via instance linking and feature disentanglement," *IEEE Transactions on Instrumentation and Measurement* 74, article 5031913. https://doi.org.10.1109/TIM.2025.3565348

Redmon, J., Divvala, S., Girshick, R., and Farhadi, A. (2016). "You Only Look Once: Unified, real-time object detection," in: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, USA, pp. 779-788. https://doi.org.10.1109/CVPR.2016.91

Ren, S., He, K., Girshick, R., and Sun, J. (2016). "Faster R-CNN: Towards real-time object detection with region proposal networks," *Proceedings of the IEEE Transactions on Pattern Analysis and Machine Intelligence* 39(6), 1137-1149. https://doi.org.10.1109/TPAMI.2016.2577031

Su, B., Zhou, Z., and Chen, H. (2022). "PVEL-AD: A large-scale open-world dataset for photovoltaic cell anomaly detection," *IEEE Transactions on Industrial Informatics* 19(1), 404-413. https://doi.org.10.1109/TII.2022.3162846

Wang, B., Yang, C., Ding, Y., and Qin, G. (2021). "Detection of wood surface defects based on improved YOLOv3 algorithm," *BioResources* 16(4), 6766-6780. https://doi.org.10.15376/biores.16.4.6766-6780

Wang, C., Yeh, I., and Liao, H. (2024). "YOLOv9: Learning what you want to learn using programmable gradient information," in: *2024 European Conference on Computer Vision (ECCV)*, Milan, Italy, pp. 1-21. https://doi.org.10.1007/978-3-031-72751-1_1

Xie, Y., and Ling, J. (2023). "Wood defect classification based on lightweight convolutional neural networks," *BioResources* 18(4), 7663-7680.

https://doi.org.10.15376/biores.18.4.7663-7680

Xu, W., Wan, Y., and Zhao, W. (2025). "ELA: Efficient location attention for deep convolution neural networks," *Journal of Real-Time Image Processing* 22(4), 1-14. https://doi.org.10.1007/s11554-025-01719-6

Yan, S., Zhang, H., Ji, H., Ding, Y., Bai, Y., and Yang, C. (2025). "Research on wood defect detection based on improved YOLOv8 model," *Forest Engineering* 41(4), 750-760. https://doi.org. 10.7525/j.issn.1006-8023.2025.04.010

Zhang, R., Yang, H., Wang, Y., Zhao, Y., Bi, L., Ren, S., and Wang, W. (2025). "Deep learning-based lightweight system for detection of surface defects in timber," *Journal of Forestry Engineering* 10(4), 107-117. https://doi.org.10.13360/j.issn.2096-1359.202406003

Zhao, W., Zhang, W., Liu, D., Wang, T., Wang, D., Xia, D., Zhou, L., and Li, Z. (2025). "Research on intelligent counting of gas extraction drill rods based on YOLOv5," *Journal of Henan Polytechnic University (Natural Science)* 44(3), 81-88. https://doi.org.10.16186/j.cnki.1673-9787.2024060009

Zhao, Y., Lv, W., Xu, S., Wei, J., Wang, G., Dang, Q., Liu, Y., and Chen, J. (2024). "Detrs beat yolos on real-time object detection," in: *2024 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Seattle, USA, pp. 16965-16974. https://doi.org.10.1109/CVPR52733.2024.01605

Article submitted: October 30, 2025; Peer review completed: December 1, 2025; Revised version received: December 19, 2025; Accepted: December 28, 2025; Published: January 12, 2026.
DOI: 10.15376/biores.21.1.1779-1806