Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

Improving YOLOv12 for Wind Turbine Blade Defect Detection through Edge Enhancement and Multiscale Feature Fusion

Biao Zhang
Harbin University of Science and Technology
Guoxin Li
2420510146@stu.hrbust.edu.cn

Harbin University of Science and Technology

Research Article

Keywords: wind turbine blades, defect detection, YOLOv12, multiscale feature fusion, attention mechanisms

Posted Date: November 25th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-11703794/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.

Improving YOLOv12 for Wind Turbine Blade Defect Detection through Edge Enhancement and Multiscale Feature Fusion

Biao Zhang* and Guoxin Li

School of Automation, Harbin University of Science and Technology, Harbin 150080, Heilongjiang, China

* Corresponding author: zhangbiao@hrbust.edu.cn

Abstract. Blurred boundaries, variations in defect size, and complex backgrounds can cause detectors to miss wind turbine blade defects or mistake background features for damage. We modify YOLOv12 at three points to address these problems. In the shallow backbone, edge enhancement preserves fine boundary detail in low-quality images. Deeper in the backbone, convolutional emulation of self-attention captures global context to distinguish closely spaced defects. In the neck, sequence shuffle attention fuses features across scales to help detect defects of different sizes and those that are partially occluded. The modified model achieves a precision of 88.3%, an mAP@0.5 of 88.9%, and an mAP@0.5:0.95 of 60.3%. These values exceed those of the YOLOv12 baseline by 6.6, 3.1, and 2.6 percentage points, respectively. The model improves detection accuracy and robustness on the evaluated blade images while retaining real-time inference speed, supporting its use in unmanned aerial vehicle inspections and wind turbine maintenance.

Keywords. wind turbine blades, defect detection, YOLOv12, multiscale feature fusion, attention mechanisms

Chinese Library Classification: TP394.1, TH691.9 Document code: A

1. Introduction

Wind power has expanded rapidly in recent years. Turbine blades capture wind energy, and their condition directly affects power generation and operational safety. Exposure to wind, rain, and lightning can cause surface damage, including cracks and erosion. As installed wind capacity grows, manual inspection alone struggles to deliver the speed and accuracy required across large fleets. Reliable methods for detecting blade defects are therefore important for wind farm operation and maintenance.

Blade inspection relies primarily on visual checks by personnel and sensor-based nondestructive testing. Manual inspection is slow, exposes personnel to safety risks, and depends on the inspector's judgment [1]. Infrared thermography [2], acoustic emission testing [3], and radiography [4] offer other ways to detect blade damage. However, high inspection costs, turbine shutdown requirements, and limited capacity for online monitoring can restrict their use across large wind farms.

Machine vision has also been applied to blade inspection. Canny edge detection [5], adaptive thresholding [6], and classifiers such as support vector machines [7] and random forests [8] have improved the efficiency of defect screening. These methods depend heavily on hand-crafted features and empirically chosen parameters. When backgrounds are cluttered or defects vary widely in shape and size, they struggle to classify and locate multiple defects accurately. Their limited robustness and generalization make them difficult to use under variable field conditions.

Two-stage deep learning detectors, including Faster R-CNN [9], Mask R-CNN [10], and Cascade R-CNN [11], have improved defect recognition and localization by learning features from images. Generating region proposals, however, adds computational cost and complicates real-time operation. This trade-off has prompted interest in single-stage detectors that can maintain accuracy at higher inference speeds.

Single-stage detectors such as SSD and YOLO are now widely used for blade inspection. Modifications to SSD's multiscale fusion strategy [12] and feature extraction network [13] have improved detection efficiency and accuracy. Small defects, large variations in scale, and background clutter nevertheless remain sources of missed and false detections. Transformer-based detectors such as RT-DETR offer stronger feature representations [14], although their computational requirements can constrain real-time

deployment. The YOLO family is widely studied for industrial inspection because it combines detection accuracy, fast inference, and ease of deployment.

Early applications of YOLO to blade inspection focused on YOLOv5 and YOLOv7. Zheng et al. [15] improved the accuracy of YOLOv5 by modifying its loss function and activation mechanism. Wang and Gao [16] improved detection accuracy and stability with a modified YOLOv5s, while Tang et al. [17] strengthened feature extraction in YOLOv7.

Subsequent work has focused on lighter networks, multiscale fusion, and the detection of small defects. Tong et al. [18] developed WTBD-YOLOv8, reducing the parameter count and improving detection efficiency through architectural changes. Zhang et al. [19] enhanced sensitivity to small defects, and Zhu et al. [20] modified multiscale fusion to better represent defects of different sizes. Tong et al. [21] optimized small-defect detection to reduce missed detections. Li et al. [22] reduced model complexity while maintaining detection performance.

Work on newer YOLO architectures follows similar lines. Chen et al. [23] improved detection accuracy and inference efficiency by revising the architecture and feature extraction in YOLOv10n. Wang et al. [24] modified feature fusion in YOLOv11n to improve performance and stability. Li and Tang [25] redesigned the feature extractor and detection head to improve recognition of blade surface defects.

These studies have advanced detection accuracy, lightweight design, and small-object recognition through changes to network architecture, attention, and feature fusion. When blurred edges, background clutter, and defects of different sizes occur together, however, weak defect features can still be lost. Missed detections and false alarms remain difficult to reduce while meeting real-time requirements.

This study investigates a task-oriented configuration of established EGA, ESC, and SSA mechanisms within YOLOv12 [26–28]. The design is motivated by the different roles of boundary detail, contextual information, and feature fusion in blade defect detection. We evaluate the three mechanisms individually and in combination using an eight-configuration ablation study, and compare detection metrics and throughput with representative detectors under the reported experimental settings. The analysis characterizes both the gains and the trade-offs of the combined configuration, including the loss in recall relative to ESC+EGA. The contribution is the configuration and empirical evaluation of these mechanisms for the collected blade defect dataset.

2. YOLOv12 Architecture

YOLOv12 is a single-stage detector with a backbone, a neck, and a detection head (Figure 1). Its feature extraction and fusion stages improve the representation of objects in complex scenes while supporting real-time detection.The backbone extracts image features using Conv, C3K2, and A2C2f blocks. C3K2 uses lightweight cross-layer connections to capture local features at relatively low computational cost. A2C2f combines area attention with residual connections, extending the receptive field to capture global information and distinguish objects from complex backgrounds.

The neck combines features from different backbone levels. Its top-down and bottom-up paths bring shallow spatial detail together with deeper semantic features. Multiscale fusion improves detection across object sizes, particularly for small objects.The head predicts object classes and bounding boxes. Separate branches learn classification and localization, and detection branches at multiple scales accommodate objects of different sizes.

Lightweight feature extraction, area attention, and multiscale fusion give YOLOv12 a favorable balance between detection accuracy and inference speed. We therefore use it as the baseline for blade defect detection.

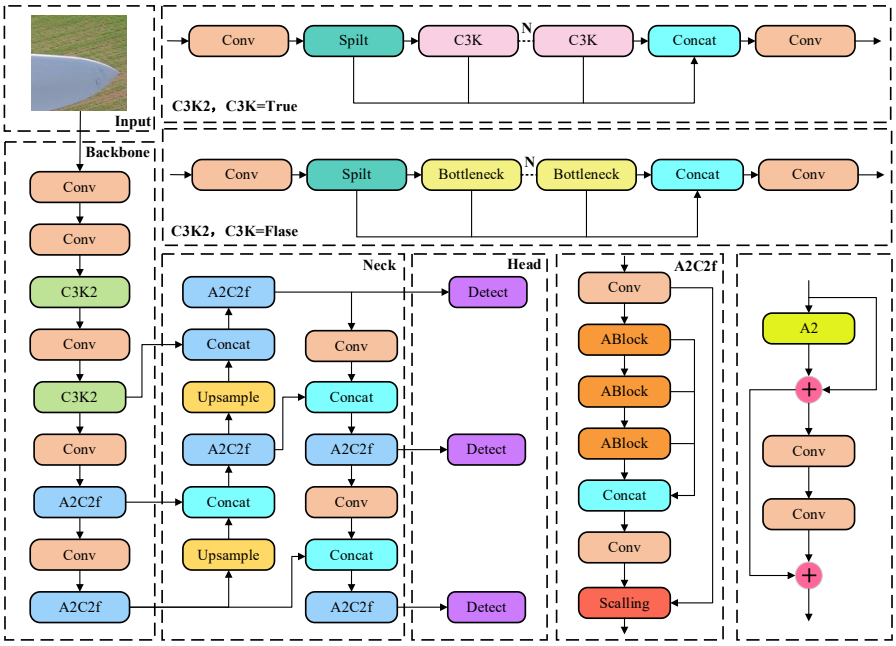


Figure 1. Architecture of the baseline YOLOv12 detector.

3. Modifications to YOLOv12

Defect information is represented differently at different network depths. The modified architecture in Figure 2 assigns the three enhancement mechanisms according to the spatial resolution of shallow features, the semantic content of deeper features, and the role of the neck in combining features from multiple levels.

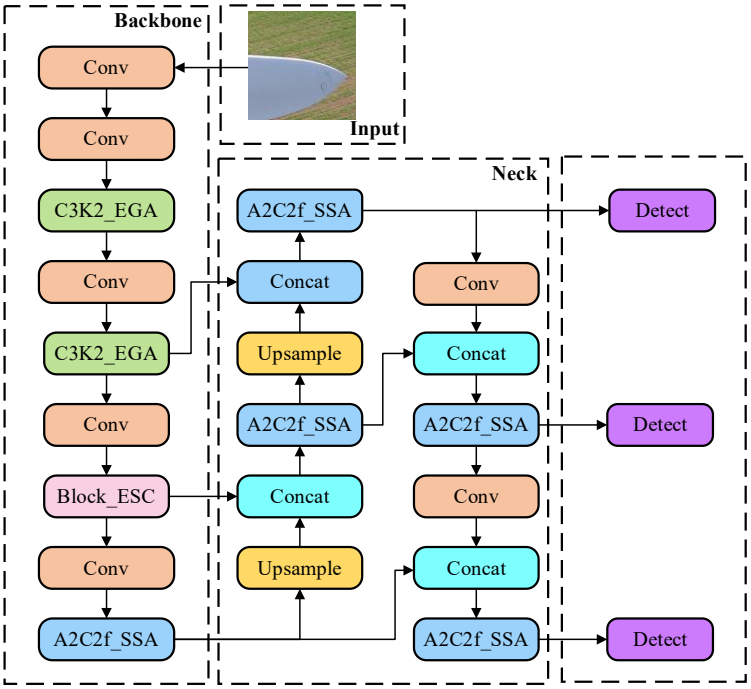


Figure 2. Architecture of the modified YOLOv12 detector.

In the backbone, C3K2-EGA strengthens the extraction of defect edges and fine detail in shallow feature maps. ESC operates on deeper features, where global context helps distinguish defects in complex scenes.

In the neck, A2C2f-SSA strengthens the interaction between features at different scales, improving the representation of defects whose apparent size varies.

These changes bring shallow edge information, deeper semantic features, and multiscale features together to represent both local details and their context. Sections 3.1–3.3 describe how each mechanism is adapted to blade inspection.

3.1. Edge–Gaussian Aggregation and Local Feature Enhancement

Fine cracks and eroded boundaries may occupy only a small region of a blade image. Preserving their local contrast and contours motivates the use of edge enhancement before repeated downsampling attenuates spatial detail. In this design, EGA and LEG are used within selected shallow C3K2 blocks (Figure 3). EGA provides edge-related feature enhancement, while LEG recalibrates channel responses. These roles motivate their placement; the ablation study evaluates their combined contribution through the C3K2-EGA configuration.

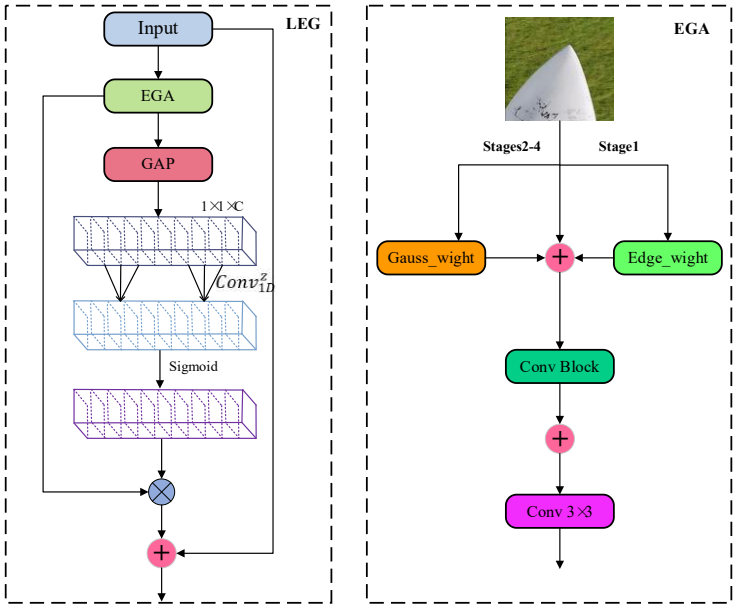


Figure 3. Structure of the EGA and LEG modules.

EGA changes how it enhances features with network depth. Stage 1 retains rich edge and texture information, so an edge branch applies the Scharr operator to the input features. For input $F_{in} \in \mathbb{R}^{H\times W\times C}$, convolution with the horizontal and vertical Scharr kernels G_x and G_y gives the directional responses in Equation (1).

$$G_x = Conv_{2D}^{S_x}(F_{in}), \quad G_y = Conv_{2D}^{S_y}(F_{in}) \tag{1}$$

The two Scharr kernels are defined in Equation (2).

$$S_x = \begin{bmatrix} -3 & 0 & 3 \\ -10 & 0 & 10 \\ -3 & 0 & 3 \end{bmatrix}, S_y = \begin{bmatrix} -3 & -10 & -3 \\ 0 & 0 & 0 \\ 3 & 10 & 3 \end{bmatrix} \tag{2}$$

The Euclidean norm of the two responses gives the edge magnitude used to form the edge attention map.

$$A_{edge} = \sqrt{G_x^2 + G_y^2} \tag{3}$$

Compared with the Sobel operator, the Scharr operator provides more stable gradient estimates and less sensitivity to edge orientation. It thus better preserves contours and helps the detector identify weak boundaries in cracks and worn areas.

At greater depths (stages 2–4), feature maps lose spatial resolution. Explicit edges become weaker, while semantic responses become more concentrated in object regions. EGA therefore uses a Gaussian kernel with fixed parameters to smooth and aggregate the input features.

$$A_{gauss} = G_{1.0}^{5\times 5}\left(F_{in}\right) \tag{4}$$

We use a Gaussian kernel with $k=5$ and $\sigma=1.0$. Its weight at the discrete position (x,y) is given by Equation (5).

$$G_{\sigma}^{\mathrm{k\times k}}(x)=\frac{1}{2\pi\sigma^{2}}e^{-\frac{x^{2}+y^{2}}{2\sigma^{2}}} \tag{5}$$

Gaussian aggregation strengthens responses within object regions and suppresses random noise, making deep semantic features more stable and discriminative. After edge or Gaussian modeling, EGA fuses and reconstructs the resulting attention features. It adds the attention map to the input elementwise and uses a convolutional block to combine local context, yielding the intermediate feature F_a.

$$F_{temp}=\mathrm{Conv}_{2D}^{3\times3}\left(\mathrm{AN}\left(\mathrm{Conv}_{2D}^{1\times1}\left(F_{in}+A_{ega}\left(F_a\right)\right)\right)\right) \tag{6}$$

$$F_a=\mathrm{Norm}\left(\mathrm{Conv}_{2D}^{1\times1}\left(\mathrm{AN}\left(F_{temp}\right)\right)\right) \tag{7}$$

Here, AN denotes the combination of activation and normalization layers. Alternating 1×1 and 3×3 convolutions reduces the channel dimension while strengthening spatial relationships within defect regions.EGA then uses a residual connection to retain the original information and produce the enhanced feature F_{ega}.

$$F_{ega}=\mathrm{Conv}_{2D}^{3\times3}\left(\left(F_{in}\otimes F_a\right)+F_{in}\right) \tag{8}$$

In Equation (8), $\otimes$ denotes elementwise multiplication. The residual connection preserves the original semantic information as responses in important regions are amplified, limiting the distortion that excessive enhancement could introduce.

LEG then recalibrates the EGA output across channels. Global average pooling summarizes spatial information, and an adaptive one-dimensional convolution models local channel dependencies to generate channel weights.

$$F_{temp}=\mathrm{Sigmoid}\left(\mathrm{Conv}_{1D}^{z}\left(\mathrm{GAP}\left(F_{ega}\right)\right)\right) \tag{9}$$

Here, Conv_{1D}^{z} denotes adaptive one-dimensional convolution with kernel size z. The kernel size varies with the number of channels to exchange information efficiently. Channel reweighting and a residual connection then produce the LEG output.

$$F_o=\mathrm{Norm}\left(\left(F_{temp}\otimes F_{ega}\right)+F_{in}\right) \tag{10}$$

This operation strengthens channels associated with defects and suppresses redundant background information.The original design used EGA as a basic backbone unit. In this model, EGA is confined to selected shallow C3K2 blocks. This places edge enhancement where the spatial resolution still supports narrow cracks and low-contrast erosion, while avoiding the cost of repeating edge operations at every depth.

3.2. Emulating Self-Attention with Convolution

In UAV inspection images, the blade may appear against the sky, clouds, the tower, or the ground. Local convolution can confuse reflections, shadows, and background textures with defects. We adapt ESC from image reconstruction to blade defect detection and place it in the deeper semantic layers of the backbone (Figure 4). At this depth, feature maps are smaller and semantic information is more concentrated. A shared large kernel extends the effective receptive field at relatively low cost, while dynamic convolution adjusts the weights to the current image. Together, they link local defect features to the structure of the blade.

ESC uses a shared large kernel (LK) and a dynamic kernel (DK) to approximate the long-range interactions and input-dependent weighting of self-attention. This design strengthens global features while limiting computational complexity. The block comprises a convolutional feed-forward network

for local feature enhancement (ConvFFN) and a convolutional attention module (ConvAttn). Residual connections stabilize training.

Given the input $F_{i-1} \in \mathbb{R}^{H\times W\times C}$, layer normalization and the ConvFFN module first enhance local features, producing the preprocessed feature F_i^{in}.

$$F_i^{in} = \mathrm{ConvFFN}\left(\mathrm{LN}\left(F_{i-1}\right)\right) \tag{11}$$

Self-attention then constructs an initial global representation and forms the first residual connection.

$$F_{i.0} = F_i^{in} + \mathrm{SelfAttn}\left(\mathrm{LN}\left(F_i^{in}\right)\right) \tag{12}$$

At iteration j of convolutional attention, the input feature $F_{i,j-1}$ is split along the channel dimension.

$$F_{i,j}^{att}, F_{i,j}^{idt} = \mathrm{Split}_{16,C-16}\left(F_{i,j}^{CF}\right) \tag{13}$$

The attention branch, $F_{i,j}^{att} \in \mathbb{R}^{H\times W\times 16}$, contains the first 16 channels and uses the shared large kernel and the dynamic kernel in parallel. The remaining channels, $F_{i,j}^{idt} \in \mathbb{R}^{H\times W\times (C-16)}$, form an identity branch. Restricting long-range modeling to a subset of the channels reduces computational cost.To model input-dependent attention, the attention branch generates the depthwise kernel weights $DK_{i,j}$ dynamically.

$$DK_{i,j} = \mathrm{Conv}_{I\times I}^{up}\left(\phi\left(\mathrm{Conv}_{I\times I}^{down}\left(\mathrm{GAP}\left(F_{i,j}^{att}\right)\right)\right)\right) \tag{14}$$

In Equation (14), ϕ denotes the GELU activation function.The dynamic kernel $DK_{i,j}$ and the shared large kernel LK (set to 13×13 in this study) are each applied to $F_{i,j}^{att}$ by depthwise convolution, denoted by $\#$. Adding the two resulting feature maps gives the enhanced feature $F_{i,j}^{\mathrm{res}}$.

$$F_{i,j}^{\mathrm{res}} = \left(F_{i,j}^{\mathrm{att}} \# \ DK_{i,j}\right) + \left(F_{i,j}^{\mathrm{att}} \# \ LK\right) \tag{15}$$

The shared kernel captures global context through its wide receptive field. The dynamic kernel adjusts the response to the input features, helping the detector accommodate defects of different sizes.

The enhanced feature $F_{i,j}^{\mathrm{res}}$ is concatenated with the identity branch $F_{i,j}^{idt}$. A 1×1 convolution then mixes information across channels.

$$F_{i,j}^{fuse} = \mathrm{Conv}_{1\times 1}^{fuse}\left(\mathrm{Concat}\left(F_{i,j}^{res}, F_{i,j}^{idt}\right)\right) \tag{16}$$

Here, Concat denotes concatenation along the channel dimension.

After M iterations of enhancement, the final output feature F_i is given by Equation (17).

$$F_i = F_{i-1} + \mathrm{Conv}_{3\times 3}\left(\mathrm{LN}\left(F_{i,M}\right)\right) \tag{17}$$

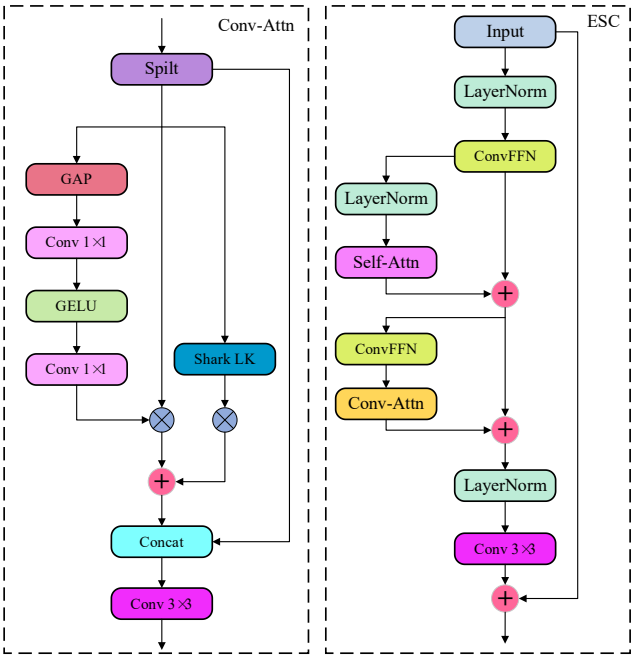


Figure 4. Structure of the ESC module.

We adapt the ConvAttn feature-modeling component of ESC for deep semantic enhancement in the detection backbone. The image super-resolution reconstruction network is omitted. The adapted unit supplies the neck with features that encode global context and help discriminate defects from the background.

3.3. Sequence Shuffle Attention for Feature Fusion

Blade defects vary widely in orientation, shape, and scale. Simple concatenation or elementwise addition in the neck cannot readily select features according to defect orientation or semantic importance. We incorporate the sequence reordering and dynamic weighting of SSA into A2C2f (Figure 5). Spatial pooling, sequence shuffling, and channel weighting across directions or branches adapt the response to crack orientation, eroded areas, and irregular damage. They also suppress redundant information from reflections, shadows, and background textures.

SSA groups channels with corresponding semantic roles across scan directions. A lightweight grouped convolution then models relationships within each group. This adaptive fusion of directional information helps represent fine defects against complex backgrounds.

Let the input comprise a set of sequences from K scan directions.

$$\{X^i\}_{i=1}^{K} \in \mathbb{R}^{D\times H\times W} \tag{18}$$

We use $K=4$, corresponding to four scan directions. The channel dimension is D, and the spatial dimensions are H and W .SSA first compresses the spatial information, then shuffles the sequences, estimates cross-direction weights, and performs weighted fusion.

Spatial average pooling and sequence concatenation

SSA applies global average pooling (GAP) to each directional sequence produced by the selective scan operation (SSO). Pooling reduces the computational cost of multidirectional modeling while retaining global semantic information from each scan. It compresses each two-dimensional spatial feature map into a channel descriptor. Concatenating the descriptors along the channel dimension forms a sequence containing information from all directions.

$$\widetilde{X}=\Phi_{cat}\left(\Phi_{AP}(\{X^i\}_{i=1}^{K}\right) \tag{19}$$

Here, $\Phi_{AP}(\cdot)$ denotes spatial average pooling, and $\Phi_{cat}(\cdot)$ denotes channel concatenation. Each pooled directional descriptor has length D. Their concatenation gives the sequence in Equation (20).

$$\widetilde{X}=[x_1^1,\ldots,x_D^1,x_1^2,\ldots,x_D^2,\ldots,x_1^K,\ldots,x_D^K] \tag{20}$$

In this expression, $L = K \times D$, and x_D^K denotes the global response for scan direction K and channel D.

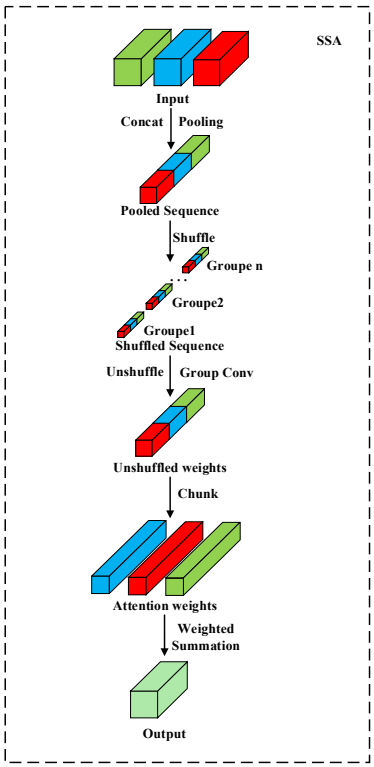


Figure 5. Sequence shuffle attention for fusing features from multiple scan directions.

Sequence shuffling

The concatenated sequence is ordered by scan direction. Channels with corresponding semantic roles are therefore separated from one another. SSA applies a shuffle operation to reorder the sequence.

$$\widehat{X} = \Phi_{ss}(X) \tag{21}$$

The reordered sequence has the following form.

$$\widehat{X} = [x_1^1, \ldots, x_1^K, x_2^1, \ldots, x_2^K, \ldots, x_D^1, \ldots, x_D^K] \tag{22}$$

The shuffle operator in Equation (21) places corresponding channels from different directions next to one another. A subsequent convolution can then learn relationships across scan directions directly.

Grouped convolution and weight unshuffling

A one-dimensional grouped convolution with group size K learns channel relationships across scan directions and generates weights for the shuffled sequence. Unshuffling restores the original directional order and yields a channel weight for each scan direction.

$$\widetilde{W} = \Phi_{su}\left(\Phi_g(\widehat{X})\right) \tag{23}$$

Here, $\Phi_g(\cdot)$ denotes one-dimensional grouped convolution, and $\Phi_{su}(\cdot)$ denotes unshuffling. The resulting directional weight sequence is given by Equation (24).

$$\widetilde{W} = [w_1^1, \ldots, w_D^1, w_1^2, \ldots, w_D^2, \ldots, w_1^K, \ldots, w_D^K] \tag{24}$$

The quantity w_D^K is the attention weight for scan direction K and channel D.

Weight splitting and weighted summation

To fuse the directional features adaptively, the chunk operation first splits the weight sequence into a separate weight vector for each direction.

$$\{W^i\}_{i=1}^K = \Phi_{chunk}(\widetilde{W}) \tag{25}$$

The resulting weights are applied channelwise to the original input sequences. The weighted sequences are then aggregated.

$$Y = \sum_{i=1}^{K} \left(W^i \square X^i \right) \tag{26}$$

Here, $\square$ denotes elementwise weighting with broadcasting along the channel dimension, and $Y \in \square^{D \times H \times W}$ is the fused output feature.

SSA originally combined multidirectional sequences for image restoration. Here, it operates within the multibranch A2C2f structure as a feature selection and fusion unit for the YOLOv12 neck. The design aims to make fusion more efficient, reduce the parameter count, and increase inference speed while retaining detection performance for real-time UAV inspection.

4. Experiments and Results

4.1. Dataset

We assembled a dataset of wind turbine blade surface defects from several public sources. Most images came from Kaggle, supplemented with blade defect images from Roboflow Universe. The sources include close-range UAV inspections, close-up views of blade surfaces, and images of artificially simulated defects. They differ in camera equipment, imaging distance, viewing angle, illumination, background, and defect scale.

We manually screened the images, removed duplicates and low-quality samples, harmonized the class definitions, and reannotated the images. The resulting dataset contains 6,450 images, resized to 640 × 640 pixels, covering four surface defect classes (crack, damage, dirt, and erosion). Combining sources increased the diversity of defect shapes, object sizes, viewing angles, and backgrounds. Combining image sources increased the variety of appearances represented in the dataset. However, the evaluation uses a random split of the pooled images rather than a source-held-out split. The resulting performance measures detection within this assembled dataset and does not establish generalization to unseen wind farms, cameras, or acquisition conditions.

YOLO-format annotations specify the class and bounding box of each defect. We randomly split the dataset into training, validation, and test sets in a 7:2:1 ratio, keeping the class distributions as similar as possible. The training set was used to learn model parameters, the validation set for tuning and model selection, and the test set for final evaluation.

4.2. Experimental Setup

Experiments ran on Ubuntu 20.04 with an Intel Xeon Gold 6330 CPU and an NVIDIA GeForce RTX 3090 GPU with 24 GB of memory. The software environment comprised Python 3.8, PyTorch 2.0.0, and CUDA 11.8. All models were trained for 300 epochs at an input resolution of 640 × 640 pixels and a batch size of 16. We used stochastic gradient descent (SGD) with an initial learning rate of 0.01, momentum of 0.937, and weight decay of 0.0005. All other training parameters followed the YOLOv12 defaults. Comparative and ablation experiments used the same hardware and training settings.

4.3. Evaluation Metrics

We evaluated the detector using precision (P), recall (R), mean average precision (mAP@0.5 and mAP@0.5:0.95), trainable parameter count, and throughput in frames per second (FPS). Precision measures the fraction of detections that are correct, and recall measures the fraction of labeled objects detected. The mAP values summarize detection performance across classes, while parameter count and FPS characterize model size and real-time performance. Equations (27)–(30) define precision, recall, average precision (AP), and mAP.

$$Precision = \frac{TP}{TP + FP} \tag{27}$$

$$Recall = \frac{TP}{TP + FN} \tag{28}$$

$$AP = \int_0^1 p(r)\,dr \tag{29}$$

$$mAP = \frac{1}{N}\sum_{i=1}^{N} AP_i \tag{30}$$

TP is the number of correctly detected objects, FP the number of false detections, and FN the number of missed objects. N denotes the number of defect classes. For a single class, AP is the area under the precision–recall (PR) curve. Averaging AP across classes gives mAP.

The metric mAP@0.5 uses an intersection-over-union (IoU) threshold of 0.5. The metric mAP@0.5:0.95 averages mAP over IoU thresholds from 0.50 to 0.95 in steps of 0.05 and is therefore more sensitive to localization accuracy. A smaller trainable parameter count indicates lower model complexity. FPS measures the number of images processed per second, with higher values indicating faster inference.

4.4. Ablation Study

Table 1 shows that the three modules affect detection accuracy and computational cost differently. EGA alone increases mAP@0.5 from 0.858 to 0.870 and mAP@0.5:0.95 from 0.577 to 0.587, with 0.10 million additional parameters. ESC alone raises the two metrics to 0.866 and 0.590, respectively, but increases the parameter count to 3.27 million and reduces throughput to 263 FPS. SSA alone reduces the parameter count from 2.55 to 2.22 million and increases throughput from 350 to 443 FPS, while recall falls from 0.834 to 0.815 and mAP@0.5:0.95 falls from 0.577 to 0.568. Its isolated contribution therefore favors efficiency over recall and performance under stricter localization criteria.

Table 1. Ablation results for ESC, EGA, and SSA on the YOLOv12n baseline.

Run	ESC	EGA	SSA	P	R	mAP@0.5	mAP@ 0.5:0.95	Params (M)	FPS
1	–	–	–	0.817	0.834	0.858	0.577	2.55	350
2	✓	–	–	0.842	0.824	0.866	0.590	3.27	263
3	–	✓	–	0.851	0.829	0.87	0.587	2.65	361
4	–	–	✓	0.831	0.815	0.862	0.568	2.22	443
5	✓	✓	–	0.852	0.872	0.883	0.598	3.37	265
6	✓	–	✓	0.856	0.844	0.88	0.598	3.00	270
7	–	✓	✓	0.84	0.842	0.865	0.573	2.38	376
8	✓	✓	✓	**0.883**	**0.846**	**0.889**	**0.603**	**3.42**	**278**

The effect of SSA also depends on the accompanying module. Adding SSA to EGA lowers mAP@0.5:0.95 from 0.587 to 0.573, which is 0.004 below the baseline. In contrast, adding SSA to ESC increases this metric from 0.590 to 0.598. These results indicate that the benefits of SSA are configuration dependent. A possible explanation is that feature reweighting can suppress weak defect responses as well as background responses, with its effect depending on the features supplied to the fusion stage. The aggregate results do not establish this mechanism, and the observed differences may also reflect training variability.

Compared with ESC+EGA, the full configuration increases precision from 0.852 to 0.883, mAP@0.5 from 0.883 to 0.889, and mAP@0.5:0.95 from 0.598 to 0.603, but reduces recall from 0.872 to 0.846. Thus, the full configuration achieves the highest precision and both mAP values among the tested configurations, whereas ESC+EGA achieves the highest recall. Precision and recall describe performance at the reported operating point, while AP summarizes the precision–recall relationship across confidence thresholds. A lower recall at that operating point can therefore coexist with higher mAP. For applications that prioritize avoiding missed defects, the recall advantage of ESC+EGA remains relevant.

Relative to the baseline, the full model adds 0.87 million parameters and reduces throughput from 350 to 278 FPS. Its gains in mAP therefore incur a computational cost. The ablation results support selecting

the full configuration when precision and mAP are prioritized under the tested hardware conditions; they do not show uniform improvement in every metric. Because Table 1 does not report variation across repeated runs, small differences between configurations should be interpreted descriptively.

Figures 6 and 7 compare test-set precision–recall and precision–confidence curves for the baseline and modified models. In Figure 6, the modified model's PR curve generally lies above the baseline curve and closer to the upper-right corner. The class-specific gains are uneven. AP increases by 8.0 percentage points for damage, 3.3 points for erosion, 0.9 points for cracks, and 0.3 points for dirt. Thus, most of the observed improvement is concentrated in damage and erosion. These class-level results do not isolate the effects of defect size, blur, or background complexity.

The largest class-specific gain was for damage, where AP rose from 0.695 to 0.775, a relative improvement of 11.5%. This category includes defects with complex shapes and substantial variation in appearance. Erosion AP rose from 0.850 to 0.883 (3.9% relative), indicating improved detection where boundaries are indistinct. Crack AP increased from 0.901 to 0.910, and dirt AP from 0.984 to 0.987. All four classes improved, with the largest gains for the more difficult damage and erosion categories.

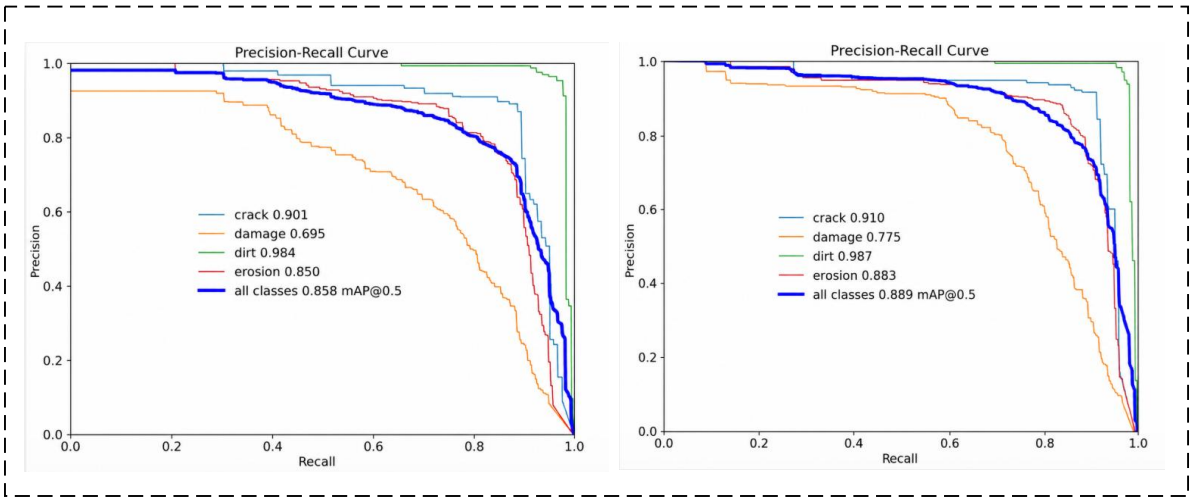


Figure 6. Test-set precision–recall curves for the baseline and modified detectors.

Figure 7 reports an all-class precision of 1.00 at a confidence threshold of 0.904 for the modified model and 0.95 at a threshold of 0.886 for the baseline. These values are obtained at different confidence thresholds and do not by themselves establish superior overall detection performance. Increasing the threshold can improve precision while excluding valid detections. The precision–confidence curves should therefore be interpreted together with the PR curves and the recall and mAP values reported in Table 1.

Taken together, the PR and precision–confidence curves support the use of ESC, EGA, and SSA to strengthen defect representations, particularly in cluttered scenes and where boundaries are blurred.

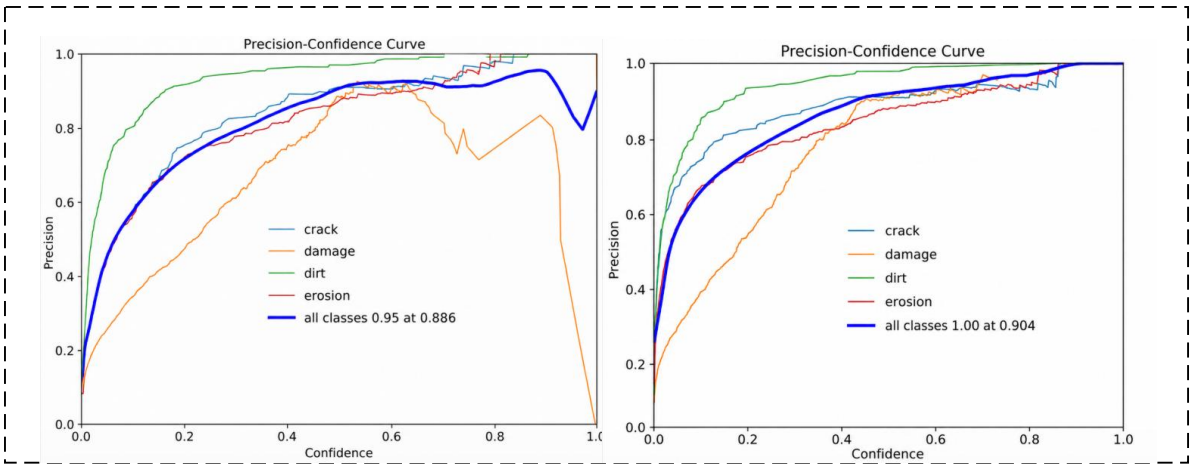


Figure 7. Test-set precision–confidence curves for the baseline and modified detectors.

4.5. Detector Comparisons and Qualitative Results

We compared the proposed detector with Faster R-CNN, RT-DETR, and lightweight models from the YOLO family. All models were trained on the same hardware with the same training parameters. Table 2 reports the results.

On the reported test platform, Faster R-CNN and RT-DETR achieved throughputs of 17 and 106 FPS, respectively, compared with 278 FPS for the proposed model. These measurements compare inference throughput under the reported benchmark settings. Whether a model meets inspection requirements depends on the target hardware and the latency of the complete acquisition and processing pipeline.

The modified YOLOv12 achieved the highest values among the detectors compared on all four accuracy metrics, with a precision of 0.883, a recall of 0.846, an mAP@0.5 of 0.889, and an mAP@0.5:0.95 of 0.603. It exceeded RT-DETR by 6.5 percentage points in mAP@0.5 and 8.8 points in mAP@0.5:0.95, with 3.42 million parameters.

Against the YOLOv12n baseline, the relative gains in precision, recall, mAP@0.5, and mAP@0.5:0.95 were 8.1%, 1.4%, 3.6%, and 4.5%, respectively. Across the YOLOv8n through YOLOv13n models in Table 2, the absolute improvements ranged from 3.1 to 5.0 percentage points for mAP@0.5 and from 2.6 to 6.5 points for mAP@0.5:0.95. The advantage on the stricter metric indicates more accurate localization.

The modified detector used 0.87 million more parameters than the baseline and its throughput fell to 278 FPS. It nevertheless retained a compact architecture and real-time inference on the test platform. The comparison shows the accuracy and speed trade-off associated with combining edge, context, and multiscale features.

Table 2. Detection performance, parameter count, and throughput of the detectors compared.

Model	P	R	mAP@0.5	mAP@0.5:0.95	Params (M)	FPS
Faster R-CNN	0.468	0.823	0.779	0.364	137	17
RT-DETR	0.873	0.796	0.824	0.515	31	106
YOLOv8n	0.84	0.832	0.855	0.563	3.01	720
YOLOv9n	0.834	0.814	0.839	0.551	2.66	440
YOLOv10n	0.815	0.803	0.839	0.538	2.7	628
YOLOv11n	0.83	0.827	0.846	0.561	2.58	631
YOLOv12n	0.817	0.834	0.858	0.577	2.55	350
YOLOv13n	0.818	0.836	0.846	0.554	2.48	338
Proposed	**0.883**	**0.846**	**0.889**	**0.603**	**3.42**	**278**

We also compared detections in individual images. Figure 8 shows predictions from Faster R-CNN, RT-DETR, the YOLO models, and the proposed detector for the four defect classes. Figure 9 compares the original images with baseline and modified YOLOv12 predictions in two difficult cases. The proposed detector identified defects in each of the four settings shown in Figure 8.

In the damage example, the defect lies close to the blade edge against a cluttered background. The proposed detector covers the damaged region more completely, and its bounding box follows the visible defect boundary more closely. In the dirt example, the discoloration resembles the surrounding blade surface and has little texture. Some detectors draw overly large or misplaced boxes, whereas the proposed model places a tighter box around the affected area.

The first example in Figure 9 contains small damaged areas whose gray levels differ little from the surrounding blade, leaving the defect boundaries indistinct. The baseline detects the main area of damage but assigns confidence scores of only 0.37 and 0.30 to two less conspicuous defects. The modified detector raises the corresponding scores to 0.40 and 0.41 while retaining the detection of the conspicuous damage. In this example, its response is stronger for small, low-contrast defects.

The second example contains elongated defects that resemble the blade texture and are difficult to distinguish from the background. The baseline detects two defects with confidence scores of 0.37 and

0.61. The modified detector raises these scores to 0.59 and 0.75, and the predicted regions follow the visible defects more closely. The revised feature extraction and fusion stages improve discrimination of narrow, small-scale defects in this example.

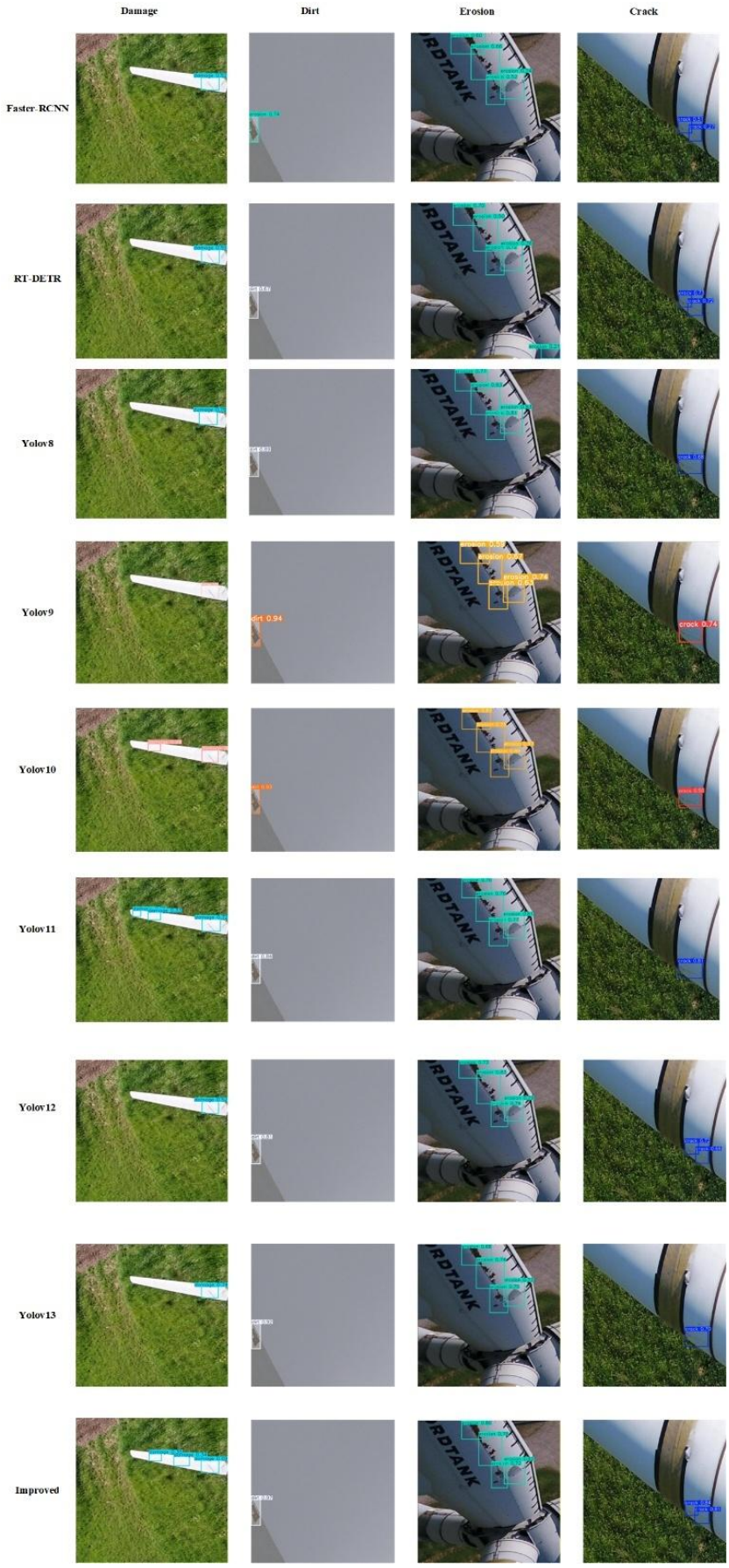


Figure 8. Qualitative comparison of defect detections across models. The columns show damage, dirt, erosion, and cracks from left to right.

In the selected examples, the modified model assigns higher confidence scores to several low-contrast defects that are also detected by the baseline. These examples illustrate changes in prediction responses, but confidence scores alone do not establish improved localization or calibrated detection reliability. The visual comparison provides illustrative evidence; the aggregate evaluation of detection performance is given by the precision, recall, and mAP results in the tables.

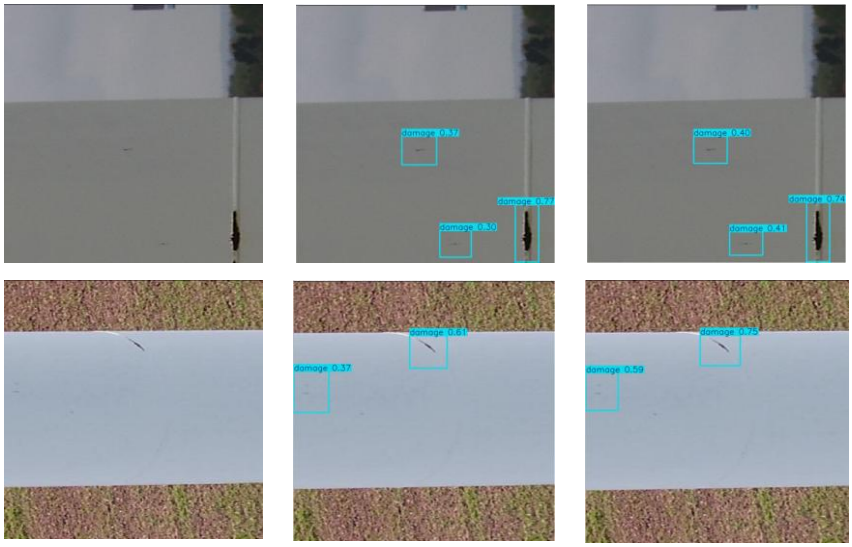


Figure 9. Original inspection images and corresponding detections from the baseline and modified YOLOv12 models.

5. Conclusions

This study investigated the integration of EGA, ESC, and SSA into YOLOv12 for wind turbine blade surface defect detection. The design combines established mechanisms for edge enhancement, contextual feature modeling, and feature fusion within a single detection framework. Its contribution lies in their task-oriented configuration and the evaluation of their individual and combined effects on the collected blade defect dataset.

The full model achieved 88.3% precision, 84.6% recall, 88.9% mAP@0.5, and 60.3% mAP@0.5:0.95. The two mAP values exceeded the baseline by 3.1 and 2.6 percentage points, respectively. The model contained 3.42 million parameters and achieved a reported throughput of 278 FPS on an RTX 3090 GPU. The ablation results showed that the full configuration provided the highest precision and both mAP values, whereas ESC+EGA provided the highest recall. The combined design therefore involves a trade-off between detection metrics and computational cost.

The evaluation used a random split of the assembled dataset. It does not establish generalization to unseen wind farms or image-acquisition domains. The reported GPU throughput also does not establish end-to-end inspection latency or performance on resource-constrained edge hardware. The results support further investigation of the method for blade inspection. Future work should assess source-separated generalization and deployment latency, memory use, and power consumption on the intended hardware.

Author Contributions

Biao Zhang conceived the study and designed the algorithm.

Guoxin Li conducted the experiments, curated the data, and performed the analysis.

Declarations

Funding

No funding was received for conducting this study.

References

[1] CARNERO A, MARTÍN C, DÍAZ M. Structural health and intelligent monitoring of wind turbine blades with a motorized telescope. In: Proceedings of the 2022 21st IEEE International Conference on Machine Learning and Applications. Piscataway: IEEE, 2022: 105–112.

[2] WANG C, GU Y. Research on infrared nondestructive detection of small wind turbine blades. Results in Engineering, 2022, 15: 100570.

[3] DING S, YANG C, ZHANG S. Acoustic-signal-based damage detection of wind turbine blades: A review. Sensors, 2023, 23(11): 4987.

[4] FANTIDIS J G, POTOLIAS C, BANDEKAS D V. Wind turbine blade nondestructive testing with a transportable radiography system. Science and Technology of Nuclear Installations, 2011, 2011(1): 347320.

[5] XU H, XU X, ZUO Y. Applying morphology to improve Canny operator's image segmentation method. The Journal of Engineering, 2019, 2019(23): 8816–8819.

[6] QIU Z, WANG S, LI M. Defect detection of wind turbine blade based on unmanned aerial vehicle-taken images. Power Generation Technology, 2018, 39(3): 277–285.

[7] DENG L, GUO Y, CHAI B. Defect detection on a wind turbine blade based on digital image processing. Processes, 2021, 9(8): 1452.

[8] YANG X Y, ZHANG Y F, LYU W, et al. Image recognition of wind turbine blade damage based on a deep learning model with transfer learning and an ensemble learning classifier. Renewable Energy, 2021, 163: 386–397.

[9] SHIHAVUDDIN A S M, CHEN X, FEDOROV V, et al. Wind turbine surface damage detection by deep learning aided drone inspection analysis. Energies, 2019, 12(4): 676.

[10] JIMÉNEZ A A, ZHANG L, MUÑOZ G Q C, et al. Maintenance management based on machine learning and nonlinear features in wind turbines. Renewable Energy, 2020, 146: 316–328.

[11] MAO Y, WANG S, YU D, et al. Automatic image detection of multi-type surface defects on wind turbine blades based on cascade deep learning network. Intelligent Data Analysis, 2021, 25(2): 463–482.

[12] QI G, YU Z, SONG J. Multi-scale feature fusion and context-enhanced spatial sparse convolution single-shot detector for unmanned aerial vehicle image object detection. Applied Sciences, 2025, 15(2): 924.

[13] YING J, LIU X, ZENG X R, et al. Research and application of wind turbine blade defect detection based on optimized SSD algorithm. Zhejiang Electric Power, 2021, 40(8): 47–52. (in Chinese)

[14] WU D, WU R, WANG H, et al. Real-time detection of blade surface defects based on the improved RT-DETR. Journal of Intelligent Manufacturing, 2026, 37(1): 313–325.

[15] ZHENG Y, LIU Y, WEI T, et al. Wind turbine blades surface crack-detection algorithm based on improved YOLO-v5 model. Journal of Electronic Imaging, 2023, 32(3): 033012.

[16] WANG J, GAO G B. Surface defect detection method for wind turbine blades based on improved YOLOv5s. China Mechanical Engineering, 2025, 36(9). (in Chinese)

[17] TANG Z J, ZHANG C J, WANG J, et al. Wind turbine blade defect detection based on improved YOLOv7. Journal of Beijing University of Aeronautics and Astronautics, 2025: 1–15. (in Chinese)

[18] TONG L, FAN C, PENG Z, et al. WTBD-YOLOv8: an improved method for wind turbine generator defect detection. Sustainability, 2024, 16(11): 4467.

[19] ZHANG Z C, YAO S C, LIANG X J, et al. Intelligent defect detection of wind turbine blades based on small-object perception enhancement. Chinese Journal of Scientific Instrument, 2025, 46(9): 159–172. (in Chinese)

[20] ZHU G, GU C, XU L Y, et al. Multiscale defect detection of wind turbine blades based on improved YOLOv8. Optics and Precision Engineering, 2025, 33(9): 1496–1514. (in Chinese)

[21] TONG W G, CAI T J, WANG Y P, et al. Small-scale defect detection algorithm for wind turbine blades based on improved YOLOv8. Journal of Electrical Engineering, 2025: 1–10. (in Chinese)

[22] LI D H, WU C Q, GAO Q, et al. Lightweight surface defect detection network for wind turbine blades based on improved YOLOv8n. Journal of Electronic Measurement and Instrumentation, 2025, 39(8): 145–155. (in Chinese)

[23] CHEN Z R, JIAO L B, MENG L, et al. Wind turbine blade defect detection based on improved YOLOv10n. Computer Measurement & Control, 2026, 34(1): 59–66, 93. (in Chinese)

[24] WANG H Q, GUAN M, YU H F, et al. Wind turbine blade defect detection algorithm based on improved YOLOv11n. Electronic Measurement Technology, 2026: 1–12. (in Chinese)

[25] LI S L, TANG L. Surface defect detection of wind turbine blades based on improved YOLO11. Electronic Measurement Technology, 2026: 1–17. (in Chinese)

[26] LU W, CHEN S B, LI H D, et al. LEGNet: A Lightweight Edge-Gaussian Network for Low-Quality Remote Sensing Image Object Detection. In: Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops. 2025: 2865–2874.

[27] LEE D, YUN S, RO Y. Emulating self-attention with convolution for efficient image super-resolution. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. 2025: 24467–24477.

[28] LI B Y, ZHAO H Y, WANG W X, et al. MaIR: A locality- and continuity-preserving Mamba for image restoration. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2025: 7491–7501.

Author Biographies



Biao Zhang (corresponding author) received his bachelor's degree from Harbin University of Science and Technology in 2009 and his master's and doctoral degrees from Harbin Engineering University in 2011 and 2020, respectively. He is an associate professor in the School of Automation at Harbin University of Science and Technology. His research interests include ship motion prediction and control, digital image processing, and deep learning algorithms.

zhangbiao@hrbust.edu.cn



Guoxin Li received his bachelor's degree from Harbin University of Science and Technology in 2023. He is pursuing a master's degree in control engineering at the same university. His research interests include image processing and object detection.

2420510146@stu.hrbust.edu.cn