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.

CXS-YOLO: a lightweight bird-nest detection model for UAV inspection of overhead transmission lines

Rui Li

ruili@gzmtu.edu.cn

School of Low-Altitude Equipment and Intelligent Control, Guangzhou Maritime University

Yaozhong Mei

School of Low-Altitude Equipment and Intelligent Control, Guangzhou Maritime University

Xiaoya Shang

School of Low-Altitude Equipment and Intelligent Control, Guangzhou Maritime University

Lixiao Wang

School of Low-Altitude Equipment and Intelligent Control, Guangzhou Maritime University

Conghuan Yang

School of Low-Altitude Equipment and Intelligent Control, Guangzhou Maritime University

Fei Xiao

School of Low-Altitude Equipment and Intelligent Control, Guangzhou Maritime University

Liangfeng Wang

Yangjiang Power Supply Bureau, Guangdong Power Grid Co., Ltd

Article

Keywords: YOLO11n, UAV, overhead transmission lines, bird nests, multiscale detection, lightweight model

Posted Date: September 29th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-10978493/v1

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

Additional Declarations: No competing interests reported.

Abstract

Bird-nest detection in UAV-based inspection of overhead transmission lines is challenged by substantial target-scale variation, background clutter and limited computing resources. We developed CXS-YOLO, a lightweight multiscale detector based on YOLO11n. C3k2_MCC integrates multi-branch contrast convolution to strengthen fine-grained and local contrast features, while lightweight exposure-aware coordinate attention reduces illumination-related interference with limited overhead. Content-adaptive weighted downsampling preserves small-target information during scale reduction, and a scale-aware lightweight shared convolutional detection head reduces prediction redundancy across feature scales. On a held-out test set of 356 images from the CXS-BirdNest dataset, CXS-YOLO achieved 89.6% precision, 84.9% mAP@0.5, 49.8% mAP@0.5:0.95 and 83.4% recall. Compared with YOLO11n, these metrics increased by 2.2, 1.6, 1.0 and 0.9 percentage points, respectively. The model used 1.107 M parameters and 4.3 GFLOPs, representing reductions of 57.1% and 31.7%. CPU-only inference tests yielded 6.17 FPS on a Raspberry Pi 5 and 35.98 FPS on an Intel Core i9-14900K platform. These results show that CXS-YOLO improves bird-nest detection while reducing model complexity, providing a compact and efficient approach for UAV-based transmission-line inspection.

Introduction

As transmission networks continue to expand, overhead transmission lines are increasingly exposed to complex natural environments over prolonged periods. Bird nests built on transmission towers can reduce electrical clearance, while bridging by nesting materials may cause partial discharge, flashover, or line-tripping faults[1]. Collisions and electrocution associated with transmission lines can also threaten individual birds and local populations[2, 3]. Timely detection and removal of bird nests are therefore important not only for maintaining a secure power supply but also for bird conservation[4, 5]. However, transmission lines are widely distributed and inspection routes are often extensive, making it difficult for conventional manual inspection to meet the combined requirements of efficiency, safety, and timely intervention. Automated bird-nest detection has consequently become an operational necessity for transmission-line maintenance.

The mobility, low operational risk and wide coverage of unmanned aerial vehicles (UAVs) have supported the transition towards automated bird-nest detection from aerial images[6, 7]. Multiscale feature fusion, attention mechanisms and lightweight architectures have improved detection performance, but scale variation and background interference remain challenging in practical inspections. Terrain, tower structure and safe flight distances produce substantial variation in the apparent scale of bird nests[8]. Distant nests occupy few pixels and contain limited contour and texture information, making fine edges and local structures susceptible to loss during successive downsampling. Close-range nests occupy larger image regions and require feature extraction over broader spatial extents[9, 10]. Illumination changes, partial occlusion and similar textures in vegetation and tower components may also reduce local contrast between nests and their surroundings. These effects can weaken target-edge responses or make them difficult to distinguish from background responses, increasing false detections and missed targets[11]. Limited public datasets and scene coverage further constrain model training and evaluation across these variations[12, 13]. Meanwhile, the computing and storage limits of onboard devices restrict the use of larger networks or additional feature-enhancement modules to compensate for these losses. Preserving discriminative bird-nest features across scales while suppressing background interference at low computational cost therefore remains an unresolved challenge for existing detection methods.

Existing studies have improved detection performance through several approaches. SCI-YOLO11 enhances transmission-line insulator defect detection by combining convolutions that preserve low-resolution information, channel attention, and an improved regression loss[14]. However, global channel reweighting may weaken the responses of small targets, and the method does not directly model the spatial locations or scale variations of bird nests. A domain-knowledge and occlusion-reasoning approach reduces false detections by restricting candidate regions and simulating occlusion[15]. Its performance, however, depends on scene-specific priors, such as tower locations and occlusion patterns, limiting its adaptability to changes in viewpoint and background. LRDS-YOLO improves small-object detection in UAV imagery through adaptive downsampling, multilevel feature fusion, attention mechanisms, and a dynamic detection head[16]. Nevertheless, coordinating multiple

components increases architectural complexity, and the method was not specifically designed to address the strong visual similarity between bird nests and their surrounding backgrounds. Overall, existing methods have not yet fully reconciled multiscale information retention, background-interference suppression, and lightweight model design.

YOLO11n provides a strong foundation in terms of detection accuracy and computational efficiency, but its native architecture is not fully suited to multiscale bird-nest detection. C3k2 relies mainly on successive local convolutions, through which the limited edge and texture cues of bird nests can be obscured by background responses during repeated feature extraction. Attention modelling in C2PSA also incurs a certain computational cost, and its attention weights may be affected by strong illumination and local overexposure. In the neck, fixed-stride convolutions progressively compress the feature maps but cannot adapt the information-retention strategy to the target content, which can result in the loss of medium- and small-object features. Moreover, different detection layers use relatively independent prediction branches, providing limited scale-specific adaptation across feature levels while introducing repeated parameters. Further optimisation of YOLO11n is therefore required to achieve a better balance between multiscale feature extraction and lightweight model design.

To address these limitations, we propose CXS-YOLO, a lightweight bird-nest detector for UAV inspection images containing substantial target-size variation. C3k2_MCC targets weak local contrast and fine-grained visual features, whereas LECA reduces attention overhead and limits illumination-related interference. CAWD is introduced to reduce information loss during downsampling, and Detect_SALSCD reduces parameter redundancy in the prediction head. The model was designed to improve overall bird-nest detection performance while maintaining low computational complexity.

The main contributions of this study are summarised as follows:

To improve the limited extraction of fine textures and local-contrast information from distant bird nests by the native C3k2 blocks in the backbone, we embed MCC_Bottleneck within the internal feature-transformation path of C3k2 to construct C3k2_MCC. This module strengthens the representation of bird-nest edges and textures and their contrast with the surrounding background.

C2PSA at the end of the backbone incurs a relatively high computational cost, and its attention responses are susceptible to strong illumination and local overexposure. We therefore propose a lightweight exposure-aware coordinate attention module, LECA, to replace C2PSA, thereby reducing the cost of attention modelling and improving the model's adaptability to illumination changes.

To reduce the loss of fine bird-nest details during scale compression, CAWD replaces the fixed-stride downsampling convolutions in the original neck. Its content-adaptive aggregation is intended to retain informative local features during downsampling, particularly for small bird-nest regions.

Detect_SALSCD replaces the native detection head to reduce redundant parameters across the prediction branches. It retains cross-level shared convolutions and introduces scale-specific modulation for each detection layer.

Related work

Existing methods for bird-nest detection on transmission lines can be broadly classified into traditional image-processing and deep-learning approaches. Traditional methods rely mainly on hand-crafted features such as colour, texture, saliency, and morphology, and show limited stability under changing illumination, complex backgrounds, or target occlusion[17, 18]. With advances in deep learning and computer vision, object detectors such as Faster R-CNN[19], EfficientDet[20], SSD[21], and the YOLO family[22] have increasingly been applied to UAV-based transmission-line inspection. Two-stage detectors generally provide strong feature-representation capabilities but incur relatively high computational costs, whereas one-stage detectors offer a better balance between detection accuracy and inference efficiency. Accordingly, recent studies have focused on occlusion-relation modelling, multiscale feature fusion, and lightweight deployment[23, 24].

Previous studies on transmission-line object detection have addressed small defects, multiple target classes and occluded objects through targeted architectural improvements. Priya and Kamlu[25] combined bidirectional feature fusion, receptive-field attention and regional context modelling within Faster R-CNN to improve small and partially occluded insulator-defect detection in complex backgrounds. Their design integrates local details with surrounding information but retains region proposal generation and region-level prediction, requiring a balance between detection gains and computational cost for onboard use. Song et al.[26] incorporated a higher-resolution feature level and coordinate attention into EfficientDet to improve the detection of power-grid components and defects, including bird nests. Although this approach strengthened spatial information for small targets, its reported inference speed remained below those of the YOLO and SSD comparators. This result illustrates the trade-off between detail preservation and inference efficiency. Zhao et al.[27] embedded an occlusion-relation module into SSD, using similarity within overlapping target regions to model occlusion relationships and improve the localisation of overlapping fittings. Their approach focuses on relationships between objects; its applicability to texture confusion between bird nests, vegetation and tower components requires further evaluation. Collectively, these studies show that multiscale fusion and feature enhancement can improve small and occluded object detection, although their benefits must be weighed against model complexity. For bird nests with substantial scale variation and strong background similarity, preserving informative details within a lightweight architecture remains a challenge.

For bird-nest detection, Tu et al.[28] proposed YOLO-nc-kd based on YOLOv5s, combining multiscale convolutional feature fusion, attention enhancement and a revised localisation loss. These modifications improved target representation and localisation, while knowledge distillation further improved detection performance. Xu et al.[29] reconstructed the backbone using FasterNet and combined ConvMixer with ODConv to improve feature fusion, reducing computational cost while maintaining detection accuracy. These advances support the use of feature enhancement and lightweight architectures for bird-nest detection. However, preserving informative features across target scales while limiting confusion with similar backgrounds remains challenging under a constrained computational budget.

YOLO11n object detection model

Based on the above analysis, we selected YOLO11n, a compact one-stage detector with relatively low computational cost, as the baseline model. The YOLO family performs detection through a one-stage prediction framework and has been widely applied to tasks requiring comparatively efficient inference[30]. YOLO11 was further optimised by Ultralytics from established YOLO architectures, and its lightweight YOLO11n variant mainly comprises an input stage, a backbone, a neck, and a detection head. Although YOLO11n provides a strong basis for lightweight deployment, its original architecture cannot fully satisfy the combined requirements of bird-nest feature extraction and onboard implementation. In the backbone, C3k2 performs feature transformation mainly through local convolutions, limiting its ability to represent the fine-grained textures of bird nests and distinguish them from complex backgrounds. C2PSA at the end of the backbone enhances features in salient regions, but its parameter count and computational cost could be further reduced, and its adaptability to complex illumination remains limited. The neck uses fixed-stride downsampling, which may weaken the edge and texture information of distant bird nests during scale compression. Moreover, the native detection head applies separate prediction transformations to features at different scales, potentially introducing parameter redundancy. We therefore retain the overall YOLO11n framework while introducing targeted modifications to the backbone, neck, and detection head to improve complex-background feature extraction, multiscale information retention, and model lightweighting.

Methods

CXS-YOLO is built on YOLO11n and targets bird-nest inspection images with substantial variation in apparent target size. Its overall architecture is shown in Fig. 1. In the backbone, C3k2_MCC replaces the original C3k2 module to encode local texture and contrast information. LECA replaces C2PSA at the end of the backbone to reduce attention overhead and limit interference from strong illumination. Two CAWD modules are introduced into the neck to retain informative features during downsampling.

Detect_SALSCD replaces the native detection head and combines shared convolutions with scale-specific modulation. These components form a compact detection architecture for UAV images containing bird nests of different apparent sizes.

Multi-branch Contrast Convolution (MCC) module

Native C3k2 improves information reuse through cross-stage connections and feature concatenation, but its internal operations rely mainly on successive local convolutions. For bird nests that occupy relatively few pixels and exhibit irregular textures, simply increasing convolutional depth or adding parallel branches does not ensure that informative features are retained. Their limited edge and texture cues may still be obscured by complex background responses during successive feature transformations. C3k2_MCC is therefore not a simple stack of convolutional and attention structures. Instead, it forms a structured processing pathway designed to address the weakening of bird-nest details caused by background interference and insufficient information transfer across branches. We construct C3k2_MCC by replacing the internal feature-transformation unit of C3k2 with MCC_Bottleneck, whose architecture is shown in Fig. 2.

To emphasise local differences between bird nests and their neighbouring backgrounds, the differential depthwise-convolution kernel is initialised with positive values in the central region and negative values in the surrounding region:

$$K[p,q] = \begin{cases} \frac{1}{s^2}, & (p,q) \in \Omega_s \\ \frac{-1}{k^2 - s^2}, & (p,q) \notin \Omega_s \end{cases} \sum_{p,q} K_{k,s}[p,q] = 0$$

1

Here, Ω_s denotes the central $s \times s$ region of the convolution kernel. By computing the response difference between the central region and its surrounding neighbourhood, this structure highlights bird-nest edges and local texture variations while further suppressing background responses.

As shown in Fig. 2, the input feature first passes through a 1×1 convolution for channel mapping and is then divided equally into four branches along the channel dimension. The first branch applies an identity mapping to preserve basic information for subsequent feature enhancement. The second branch extracts local-contrast features using differential depthwise convolution. In the third branch, the output of the second branch is added to the current feature and then subjected to a further differential transformation, allowing the extracted local differences to contribute to subsequent modelling. The fourth branch receives the output of the third branch and uses local-mean depthwise convolution and depthwise convolution with a dilation rate of 2 to aggregate neighbouring structures and broader-range background information, respectively. The outputs of the four branches are finally concatenated along the channel dimension, forming a feature representation that combines original details, local differences, progressively enhanced features, and extended-neighbourhood information. Rather than arranging multiple convolutional branches in simple parallel, this structure transfers information across branches, allowing local differences to be progressively incorporated into broader-range feature modelling.

After concatenating the four branch features, the module sequentially applies channel and spatial attention to refine the resulting representation, as shown in Fig. 2. Given the concatenated feature G, the channel- and spatial-attention operations are formulated as follows:

$$G_c = \sigma\left[g\left(GAP\left(G\right)\right) + g\left(GMP\left(G\right)\right)\right] \odot G$$

2

$$G_s = \sigma\left\{C_{7\times 7}\left[Avg_c\left(G_c\right); Max_c\left(G_c\right)\right]\right\} \odot G_c$$

3

Here, GAP and GMP denote global average pooling and global max pooling, respectively; $\mathrm{g}(\cdot)$ denotes the channel-mapping function implemented using two shared convolutions; Avg_c and Max_c denote average and max pooling along the channel dimension, respectively; σ denotes the sigmoid function; and $\odot$ denotes element-wise multiplication.

Finally, the refined features are fused through an output convolution and combined with the input via a residual connection, thereby preserving the original information and maintaining stable feature propagation.

Lightweight Exposure-aware Coordinate Attention (LECA) module

At the end of the backbone, attention weights are typically generated directly from the input features. Under strong illumination or local overexposure in UAV images, abnormal brightness alters the mean and variance of these features, making the attention responses more likely to be dominated by highlighted regions. In this case, simply increasing the attention capacity does not directly address the resulting shift in feature distribution and may instead introduce additional parameters. Therefore, rather than merely enhancing attention capacity, LECA first stabilises the feature distribution used for weight estimation and then performs directional localisation and spatial-difference modelling, as illustrated in Fig. 3.

Given an input feature tensor $X \in R^{B\times C\times H\times W}$, LECA first performs parameter-free exposure normalisation independently within each sample and channel:

$$\mu_{b,c} = \frac{1}{H\,W}\sum_{h=1}^{H}\sum_{w=1}^{W} X_{b,c,h,w}$$

4

$$X_{norm_{b,c,h,w}} = \frac{X_{b,c,h,w} - \mu_{b,c}}{\sqrt{\frac{1}{H\,W}\sum_{h=1}^{H}\sum_{w=1}^{W}\left(X_{b,c,h,w} - \mu_{b,c}\right)^2 + \epsilon}}$$

5

This operation is used solely to generate the attention weights and does not directly modify the original features in the residual pathway, thereby reducing the influence of variations in brightness range on attention estimation while preserving the input information.

The normalised features are then aggregated separately along the width and height dimensions:

$$F_{h(b,c,h,1)} = \frac{1}{W}\sum_{w=1}^{W} X_{norm_{b,c,h,w}}, \quad F_{w(b,c,w,1)} = \frac{1}{H}\sum_{h=1}^{H} X_{norm_{b,c,h,w}}$$

6

As shown in Fig. 3, the two directional descriptors are concatenated and then compressed and encoded through a shared $1\times\ 1$ convolution, GroupNorm[31] and the h-swish activation function[32]. Two separate $1\times\ 1$ convolutional branches subsequently generate the height-wise attention A_h and width-wise attention A_w. This process models only the directional features of length $\mathrm{H}+\mathrm{W}$, thereby avoiding complex attention computation over the complete two-dimensional feature map. To constrain the parameter count, the number of intermediate channels is set to $m = \max(8, C/32)$:

$$A_h = \sigma\ \left(Conv_h\left(Z_h\right)\right), \quad A_w = \sigma\ \left(Conv_w\left(\prod\ \left(Z_w\right)\right)\right)$$

7

Here, $\prod(\bullet)$ denotes the dimension-rearrangement operation. By decomposing two-dimensional spatial modelling into two one-dimensional directional encodings, this operation preserves positional information while controlling the computational cost of attention.

Directional encoding alone may be insufficient to distinguish bird-nest textures from highlighted background regions. To address this limitation, LECA constructs a spatial-contrast prior from the absolute deviation between the original features and their spatial mean:

$$P = \sigma\left(\frac{D_{b,h,w}}{\frac{1}{H\,W}\sum_{h=1}^{H}\sum_{w=1}^{W} D_{b,h,w} + \epsilon}\right)$$

8

Finally, the module fuses the two directional attention weights with the spatial-contrast prior and modulates the original input through residual gating:

$$\mathrm{Y} = \mathrm{X} \odot (1 + A_h \odot A_w \odot \mathrm{P})$$

9

Content-Adaptive Weighted Downsampling (CAWD) module

The YOLO11n neck compresses feature maps using strided convolutions, whose parameters remain fixed after training, such that the same mapping is applied across different images and spatial locations[33]. Large objects generally occupy more pixels and contain some redundancy in their local information, making them relatively less sensitive to fixed downsampling. By contrast, small bird nests contain only limited edge and texture responses. Once these cues are weakened during scale compression, they are difficult to recover fully through subsequent feature fusion[34, 35]. The key limitation therefore lies in the downsampling process itself: a fixed mapping cannot adjust the proportion of different features retained according to local content.

Based on this consideration, CAWD is used to replace the original strided convolutions in the bottom-up fusion pathway of the neck. As shown in Fig. 1, CAWD comprises a feature branch and a weight branch. The feature branch generates multiple candidate features for each downsampled location, whereas the weight branch estimates the contribution of each candidate component according to the input content. The output feature is then obtained through weighted summation.

The feature branch uses a grouped convolution with a stride of 2 to generate candidate features, which are then divided into four candidate components through channel rearrangement:

$$V = \mathscr{R}_c\left(\mathrm{C}_{g,s=2}(X)\right) = \{V_1, V_2, V_3, V_4\},$$

10

Here, $\mathrm{C}_{g,s=2}$ denotes a grouped convolution with a stride of 2, and $\mathscr{R}_c$ denotes the operation that rearranges the convolutional output into the candidate-component dimension.

In parallel with the feature branch, the weight branch first aggregates local information through average pooling, then uses a convolution to generate weight features and rearranges each downsampling region into four weight components. Softmax normalisation is subsequently applied along the candidate-component dimension:

$$A = \mathrm{Softmax}\left[\mathscr{R}_s\left(\mathrm{C}_w\left(\mathrm{AvgPool}(X)\right)\right)\right], \sum_{i=1}^{4} A_i = 1,$$

11

Here, $\mathscr{R}_s$ denotes the operation that rearranges each local spatial region into the candidate-component dimension, and C_w denotes the convolutional unit used to generate the weights.

Finally, the candidate features are multiplied element-wise by their corresponding weights and summed along the candidate-component dimension:

$$Y = \sum_{i=1}^{4} A_i \odot V_i.$$

12

In this way, CAWD integrates feature selection directly into the scale-compression process, allowing different local regions to use distinct combinations of candidate components according to their input features. For distant bird nests with limited texture information, this input-dependent aggregation provides a more flexible pathway for retaining edge and local structural cues, rather than subjecting them to irreversible compression by a single fixed convolution.

Scale-Aware Lightweight Shared Convolutional (SALSCD) Detection Head

The original YOLO11n detection head uses an independent prediction branch for each feature scale. Although this design processes the information from each detection layer separately, it introduces many structurally similar and redundant parameters. The LSCD detection head[36] reduces prediction redundancy through cross-scale shared convolutions; however, this shared mapping also applies the same feature transformation to different detection layers. Moreover, its scale layer uses only a single learnable scalar to adjust the regression output, making it difficult to represent differences in channel responses across feature levels. Reintroducing fully independent prediction branches could improve scale adaptability, but would compromise the lightweight advantage of the shared detection head. Therefore, we introduce a scale-aware dynamic generator (SADG) into LSCD to construct the Detect_SALSCD detection head, which supplements the shared convolutions with scale-specific modelling through a lightweight auxiliary structure.

SADG takes the channel-aligned feature F_i as input and compresses its global information through global average pooling (GAP). Lightweight 1×1 convolutions are then used to reduce and restore the channel dimensionality, after which the sigmoid function generates the scale-adaptive weights α:

$$\alpha = \sigma\left\{C_{3\times 3}\delta\left[C_{1\times 1}\mathrm{GAP}\left(\mathrm{F_i}\right)\right]\right\}$$

13

The scale-adaptive weights α are then multiplied element-wise by the output of the shared convolution, and the resulting features are fed into the shared classification and regression branches. The number of discrete bins used for bounding-box distribution regression is set to 8 to further control the number of regression output channels. In this way, Detect_SALSCD retains the cross-scale parameter-sharing scheme of LSCD, while SADG accounts for channel-response differences among detection layers. This design allows the detection head to process bird-nest features at different scales more selectively while maintaining a compact parameter count.

Experiments and results

Dataset

We constructed the CXS-BirdNest dataset[37] for bird-nest detection, comprising 3,554 original UAV images without data augmentation. The images were acquired using onboard visible-light cameras mounted on multiple UAV platforms, including the DJI Mavic 2 Enterprise Advanced and include both distant panoramic views and close-range local views, thereby covering variations in object scale caused by changes in inspection distance. Representative samples are shown in Fig. 4. During dataset preparation, image provenance was verified against the original videos and acquisition records. Visually similar images

from the same acquisition sequence were also manually reviewed, and only samples that met the requirements for target discernibility and annotation quality were retained.

The original images varied in spatial resolution, whereas the model input size was standardised to 640×640 pixels. All bird-nest instances were annotated with bounding boxes using LabelImg v1.8.6. To classify object scales, we followed the scale definitions of the Microsoft COCO dataset[38] and converted the normalised bounding boxes to their equivalent pixel dimensions at an input resolution of 640 × 640 pixels. Let w_n and h_n denote the normalised width and height of a bounding box, respectively. Its projected area is then calculated as $A_b = 640^2 w_n h_n$. An instance is classified as small scale when $A_b < 32^2$ pixels², medium scale when $32^2 \leq A_b < 96^2$ pixels², and large scale when $A_b \geq 96^2$ pixels².

The 3,554 original images were stratified according to object scale and divided into training, validation and test sets at a ratio of 8:1:1, comprising 2,842, 356 and 356 images, respectively. To increase the diversity of the training data under complex inspection conditions, the training set was augmented through blurring, illumination adjustment, horizontal flipping, partial occlusion and edge enhancement[39], increasing the number of training images to 4,692. The resulting dataset contained 5,404 images and 6,148 annotated bird-nest instances, as summarised in Table 1. According to the scale criteria described above, 1,806 instances were assigned to the small category, 2,910 to the medium category and 1,432 to the large category, accounting for 29.4%, 47.3% and 23.3% of all instances, respectively. The small and medium categories together contained 4,716 instances, representing 76.7% of the dataset. Thus, the dataset predominantly comprised bird nests in these two scale categories while retaining a substantial number of large instances. Figure 5 further presents the continuous distribution of bounding-box areas and the number of instances in each category. Large objects occurred more frequently in close-range UAV images, whereas small objects were generally associated with longer observation distances or a lower proportion of the image area.

Table 1
Distribution of bird-nest instances in the dataset.

Split	Images	Instances	Small	Medium	Large
Train	4692	5348	1547	2548	1253
Validation	356	404	137	184	83
Test	356	396	122	178	96
Total	5404	6148	1806	2910	1432

Experimental environment

All experiments were conducted on a 64-bit Windows 11 system equipped with an NVIDIA GeForce RTX 4060 Laptop GPU with 8 GB of video memory. The models were implemented using Python 3.10.18, PyTorch 2.9.0 and Ultralytics 8.3.9, with GPU acceleration provided by CUDA 12.8 and cuDNN 9.10.2. To ensure consistent experimental conditions, all compared models used the same dataset splits, input resolution and training configuration. The detailed settings are provided in Table 2.

Table 2
Training and evaluation settings used in this study.

Parameter	Value
Input size	640×640
Batch size	8
Epochs	300
Optimizer	SGD
Momentum	0.937
Weight decay	0.0005
Workers	4
Initial learning rate	0.01

Evaluation metrics

Detection performance was evaluated using precision, recall and mean average precision (mAP), whereas model complexity and inference efficiency were assessed using the number of parameters (Parameters), giga floating-point operations (GFLOPs) and throughput in frames per second (FPS). At specified confidence and intersection-over-union (IoU) thresholds, the predictions were classified as true positives (TP), false positives (FP) or false negatives (FN).

Precision measures the reliability of positive predictions and is defined as the proportion of predicted positive samples that are true positives:

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

14

Recall measures the model's ability to identify positive samples and is defined as the proportion of actual positive samples that are correctly detected:

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

15

Average precision (AP) is defined as the area under the precision–recall curve, and mAP is the mean AP across all object classes:

$$AP = \int_0^1 P\left(R\right) dR$$

16

$$mAP = \frac{1}{C} \sum {}_{C=1}^{C} AP_C$$

17

The number of trainable parameters (Parameters) was used to quantify model size and complexity and was calculated as the total number of parameters across all model layers.

Computational complexity during inference was quantified in giga floating-point operations (GFLOPs), where one GFLOP corresponds to 10^9 floating-point operations.

Frames per second (FPS) denotes the number of image frames processed by the model per second and was used to assess its real-time inference capability:

$$\mathrm{FPS} = \frac{\mathrm{N}}{\mathrm{T}}$$

18

Results

Ablation experiments

To investigate the effects of the proposed modules on detection performance and model complexity, ablation experiments were conducted using YOLO11n as the baseline. All configurations used the same dataset split and training settings and were evaluated on the validation set. The module configurations and experimental results are presented in Tables 3 and 4, respectively.

The single-module configurations affected different aspects of performance. C3k2_MCC increased Recall from 81.8% to 82.4% and reduced the parameter count from 2.582 M to 2.463 M, although Precision decreased by 0.2 percentage points and GFLOPs increased from 6.3 to 6.4. CAWD produced the largest single-module increase in Recall, reaching 84.0%, and increased mAP@0.5 to 83.8%, but raised GFLOPs to 6.6. Detect_SALSCD retained the baseline mAP@0.5 of 83.5%, increased Precision and Recall to 87.2% and 82.7%, and reduced the parameter count and GFLOPs to 2.292 M and 5.2, respectively. The three modules therefore produced different changes in detection metrics and computational complexity.

Performance varied across the two-module configurations. CAWD with Detect_SALSCD achieved the highest Precision and mAP@0.5, at 88.2% and 84.5%, respectively. C3k2_MCC with CAWD yielded 88.1% Precision, 83.8% mAP@0.5 and 82.9% Recall. C3k2_MCC with Detect_SALSCD reduced model complexity to 1.498 M parameters and 4.5 GFLOPs, but its mAP@0.5 remained at the baseline level. The performance gains varied across module combinations, indicating that their contributions were not simply additive.

Adding C3k2_MCC to the combination of CAWD and Detect_SALSCD increased Precision, mAP@0.5 and Recall by 0.3, 0.2 and 1.2 percentage points, respectively, while reducing the parameter count and GFLOPs to 1.167 M and 4.3. This result indicates that enhancing bird-nest details and neighbourhood information before scale transformation and prediction can further improve the combined configuration. After LECA was incorporated, the complete model achieved a Precision of 90.2% and an mAP@0.5 of 85.1%, representing increases of 1.7 and 0.4 percentage points over Experiment 8, respectively. Recall remained at 84.1%, while the parameter count was further reduced to 1.107 M without increasing GFLOPs. These results indicate that LECA complemented feature selection and prediction in the complete model without adding theoretical computational cost.

Compared with YOLO11n, the complete CXS-YOLO improved Precision, mAP@0.5 and Recall by 3.1, 1.6 and 2.3 percentage points, respectively, while reducing the parameter count and GFLOPs by 57.1% and 31.7%. Overall, the individual modules exhibited different performance characteristics, and their combined effects were not simply additive. Through the progressive coordination of feature extraction, scale transformation, attention-based feature selection and shared prediction, the complete model achieved a favourable balance between detection performance and model complexity.

Table 3
Module configurations used in the ablation experiments.

Experiment	YOLO11n	C3k2_MCC	CAWD	Detect_SALSCD	LECA
1	√	×	×	×	×
2	√	√	×	×	×
3	√	×	√	×	×
4	√	×	×	√	×
5	√	√	√	×	×
6	√	√	×	√	×
7	√	×	√	√	×
8	√	√	√	√	×
9	√	√	√	√	√

Table 4
Ablation results for CXS-YOLO.

Experiment	Precision(%)	mAP@0.5(%)	Recall(%)	Parameters(M)	GFLOPs
1	87.1	83.5	81.8	2.582	6.3
2	86.9	83.6	82.4	2.463	6.4
3	86.9	83.8	84.0	2.249	6.6
4	87.2	83.5	82.7	2.292	5.2
5	88.1	83.8	82.9	1.418	5.6
6	87.6	83.5	82.7	1.498	4.5
7	88.2	84.5	82.9	1.944	5.0
8	88.5	84.7	84.1	1.167	**4.3**
9	**90.2**	**85.1**	**84.1**	**1.107**	**4.3**

Comparative experiments

To evaluate the detection performance of CXS-YOLO, we compared it with representative lightweight object detectors spanning different generations. These included established baselines, such as YOLOv5n, YOLOv8n and YOLO11n, as well as YOLO26n, LSOD-YOLO and several improved variants of YOLO11n. All models used the same dataset splits, input resolution and training settings and were evaluated consistently on the test set.

As shown in Table 5, CXS-YOLO achieved an mAP@0.5 of 84.9%, the highest value among the compared models. This result exceeded YOLO11n by 1.6 percentage points and both YOLO11n-PFG and YOLO26n by 0.2 percentage points. CXS-YOLO also achieved the highest Precision of 89.6%, exceeding YOLO11n-CAT by 0.7 percentage points. Its Recall reached 83.4%, representing a 0.9-percentage-point improvement over YOLO11n. Under the common test protocol, CXS-YOLO therefore ranked first in Precision and mAP@0.5, whereas its Recall advantage was smaller.

Notably, CXS-YOLO required 1.107 M parameters, the lowest parameter count in Table 5. Relative to YOLO11n, the parameter count and GFLOPs decreased by 57.1% and 31.7%, respectively. Compared with LSOD-YOLO, CXS-YOLO required an additional 0.5 GFLOPs but increased Precision and mAP@0.5 by 2.2 and 0.5 percentage points. Its parameter count was also 33.3%

lower than that of LSOD-YOLO. Figure 6 shows the mAP@0.5 and Precision curves of YOLO11n, LSOD-YOLO and CXS-YOLO during training.

Under the more stringent localisation metric, CXS-YOLO achieved an mAP@0.5:0.95 of 49.8%, exceeding YOLO11n by 1.0 percentage points but remaining below the 51.2% achieved by YOLO26n. Its Recall was also slightly lower than those of LSOD-YOLO and YOLO11n-SDTA. The principal advantages of CXS-YOLO therefore lie in mAP@0.5, Precision and parameter efficiency, whereas its localisation accuracy at higher IoU thresholds remains open to further improvement. Together with the quantitative results in Table 5, these findings indicate that CXS-YOLO achieves a favourable balance between detection performance and model complexity on the current test set. We also compared CXS-YOLO with YOLO11n across three random seeds to assess variation between training runs (Supplementary Tables S1 and S2).

Table 5
Comparison of different object detection models on the test set.

Model	Year	Precision(%)	mAP@0.5(%)	mAP@0.5:0.95(%)	Recall(%)	Parameters(M)	GFLOPs
YOLOv5n	2020	85.7	82.4	48.1	80.3	2.503	7.1
YOLOv6n	2022	87.2	82.9	48.6	80.5	4.234	11.7
MAFusion[40]	2022	86.8	83.7	49.6	81.6	3.034	7.2
YOLO11n-AFPN[41]	2023	85.3	83.9	48.5	83.1	2.102	6.3
YOLOv8n	2023	86.9	82.5	48.5	80.7	3.006	8.1
YOLOv10n	2024	86.8	82.9	48.9	79.5	2.265	6.5
YOLO11n	2024	87.4	83.3	48.8	82.5	2.582	6.3
BiFPN-YOLO[42]	2025	88.1	84.6	50.2	83.1	1.923	6.3
YOLOv12n[43]	2025	86.2	83.5	50.7	82.0	2.605	6.3
YOLO11n-C2WCA[44]	2025	87.6	82.9	49.1	80.9	2.752	6.3
YOLO11n-DMSSP[45]	2026	84.4	83.6	49.5	81.8	2.893	8.8
YOLO11n-LaSEA[46]	2026	87.5	84.1	49.0	82.9	4.258	11.5
YOLO11n-FrequencyCM[47]	2026	87.3	83.6	50.2	80.2	3.277	7.9
LSOD-YOLO[36]	2026	87.4	84.4	49.3	**83.6**	1.659	**3.8**
YOLO11n-PFG[48]	2026	88.4	84.7	50.2	82.6	2.225	5.6
YOLO11n-SCGN[49]	2026	86.7	84.2	50.3	83.3	3.604	8.7
YOLO11n-ES-MoE[50]	2026	86.9	83.5	48.8	82.9	2.020	6.6
YOLO26n[51]	2026	85.8	84.7	**51.2**	83.1	2.383	5.2
YOLO11n-CAT[52]	2026	88.9	84.5	49.8	81.6	2.614	6.3
YOLO11n-WDAM[53]	2026	86.9	84.4	50.2	82.1	3.371	6.4
YOLO11n-SDTA[54]	2026	86.9	84.2	50.1	83.5	2.57	6.3
CXS-YOLO (Ours)	2026	**89.6**	**84.9**	49.8	83.4	**1.107**	4.3

Visualisation of detection results

Figure 7 presents representative detections under heavy fog, transmission-tower occlusion, strong illumination and occlusion by foreign objects. Across these examples, CXS-YOLO localized the visible bird-nest regions and produced comparatively complete bounding boxes despite indistinct contours, partial visibility or low target-to-background contrast. These qualitative results complement the test-set comparison in Table 5, in which CXS-YOLO achieved higher Precision and mAP@0.5 than YOLO11n and LSOD-YOLO.

Heatmap visualisation

Gradient-weighted class activation mapping (Grad-CAM)[55] was used to compare the activation responses of YOLO11n, LSOD-YOLO and CXS-YOLO under complex inspection conditions, as shown in Fig. 8. In the selected heavy-fog example, the high-response regions of YOLO11n and LSOD-YOLO were relatively dispersed, whereas those of CXS-YOLO were more concentrated around the bird nest. In the examples involving transmission-tower occlusion, strong illumination and occlusion by foreign objects, the principal responses of CXS-YOLO were concentrated on the visible portions of the nests, with comparatively weaker responses to the surrounding tower components and background regions. In these selected examples, CXS-YOLO produced more spatially concentrated responses around the visible bird-nest regions than the two comparison models. This observation is consistent with reduced background activation.

Prototype system and inference-speed evaluation

To examine the integration of CXS-YOLO into the bird-nest detection workflow, we developed a prototype interface. As shown in Fig. 9, the interface supports model loading and configuration, image input, detection visualisation and result export.

We subsequently evaluated CPU-only inference speed across three computing environments: a Raspberry Pi 5 equipped with a Broadcom BCM2712 processor, a mobile computing platform with an Intel Core i7-13650HX processor and a high-performance computing platform with an Intel Core i9-14900K processor. All models were tested with a batch size of 1. The reported FPS included image preprocessing, model inference and detection post-processing. After 100 warm-up iterations, each model underwent 1,000 timed iterations, and FPS was calculated from the mean processing time. The results are presented in Table 6.

CXS-YOLO achieved 18.32 FPS on the Intel Core i7-13650HX platform, exceeding LSOD-YOLO and YOLO11n by 21.3% and 28.3%, respectively. On the Intel Core i9-14900K platform, its processing speed increased to 35.98 FPS, representing corresponding improvements of 17.5% and 22.8%. This throughput exceeded 30 FPS, supporting real-time video processing under the current test conditions.

On the Raspberry Pi 5, CXS-YOLO achieved 6.17 FPS, compared with 4.52 FPS for LSOD-YOLO and 4.06 FPS for YOLO11n. These values correspond to improvements of 36.5% and 52.0%, respectively. Although the absolute throughput was lower on this resource-constrained platform, CXS-YOLO retained the highest processing speed among the three models. The consistent results across the evaluated platforms show that its inference-efficiency advantage extends across different levels of CPU computing capacity.

Table 6
Comparison of inference speeds (FPS) across different CPU platforms.

Platform	CXS-YOLO	LSOD-YOLO	YOLO11n
Broadcom BCM2712	6.17	4.52	4.06
Intel(R) Core(TM) i7-13650HX	18.32	15.1	14.28
Intel Core(TM) i9-14900K	35.98	30.62	29.3

Discussion

The test-set results indicate that the principal advantages of CXS-YOLO lie in Precision, mAP@0.5 and parameter efficiency. Relative to YOLO11n, CXS-YOLO increased mAP@0.5 by 1.6 percentage points while reducing the parameter count and GFLOPs by 57.1% and 31.7%, respectively. This performance profile suggests that the observed improvement arises from coordinated adjustments to feature extraction, scale transformation and shared prediction rather than from increased network capacity.

The improvement was less pronounced under stricter localisation criteria, with mAP@0.5:0.95 increasing from 48.8% to 49.8%. This difference may reflect the sensitivity of high-IoU evaluation to small coordinate deviations, particularly for distant or partially occluded bird nests with indistinct boundaries. The current architecture primarily strengthens feature retention and target discrimination, while it does not explicitly introduce a localisation-refinement stage. Moreover, the present findings mainly reflect static visible-light images covered by the CXS-BirdNest dataset. Future work will include samples from additional regions, seasons, flight distances and environmental conditions, together with other transmission-line defects such as insulator damage, conductor foreign objects and hardware-fitting defects.

From an application perspective, the prototype interface verified the integration of CXS-YOLO with model configuration, image input, detection visualisation and result export. CPU-based tests further showed that CXS-YOLO consistently achieved higher inference speeds than the comparison models across the Raspberry Pi 5, mobile computing and high-performance computing platforms. CXS-YOLO therefore combines a compact model structure with practical inference efficiency, supporting bird-nest detection under different computing-resource constraints.

Conclusion

CXS-YOLO is a lightweight detector designed for bird nests with varying scales and complex backgrounds in UAV-based transmission-line inspection. On the held-out test set, it achieved 89.6% Precision, 84.9% mAP@0.5, 49.8% mAP@0.5:0.95 and 83.4% Recall, with 1.107 M parameters and 4.3 GFLOPs. Compared with YOLO11n, CXS-YOLO improved Precision and mAP@0.5 while substantially reducing model complexity. The model achieved a 1.0 percentage point improvement in mAP@0.5:0.95, indicating improved detection performance across multiple localisation thresholds. A prototype user interface further verified the workflow from image input and parameter configuration to detection visualisation, target tracking and result export. CPU-only tests showed that CXS-YOLO maintained higher inference speeds than YOLO11n and LSOD-YOLO across the three evaluated platforms, ranging from 6.17 FPS on the Raspberry Pi 5 to 35.98 FPS on the Intel Core i9-14900K platform. These findings show that CXS-YOLO combines improved bird-nest detection with low model complexity and efficient CPU inference, providing a lightweight solution for automated transmission-line inspection.

Declarations

Data availability

The CXS-BirdNest dataset generated and analysed in this study, including the image annotations and dataset-split files, is publicly available through Zenodo at https://doi.org/10.5281/zenodo.22202440 or https://github.com/MEI11137/CXS-BirdNest.

Code availability

The custom source code used to produce the results reported in this study is not publicly available. To enable the editors and reviewers to reproduce and verify the reported results, the authors would be pleased to provide the complete source code, including the implementations of the proposed modules, upon request. The source code may also be made available to qualified academic researchers by the corresponding author upon reasonable request for non-commercial research purposes. Requests should be directed to Rui Li (ruili@gzmtu.edu.cn).

Author contributions

R.L. and Y.M. jointly conceived and designed the study, developed the methodology, analysed the results, and drafted the manuscript. Y.M. implemented CXS-YOLO, conducted model training and experiments, and prepared the figures. R.L. supervised and coordinated the research and revised the manuscript. X.S. contributed to dataset annotation, data curation, and validation. L.X.W. contributed to experimental validation, result analysis, and manuscript revision. C.Y. assisted with software implementation, comparative experiments, and visualization. F.X. contributed to data curation, result interpretation, and manuscript revision. L.F.W. coordinated field data acquisition with the local power supply bureau, participated in data collection, and provided access to field sites and facilities for data acquisition and experimental work. All authors reviewed and approved the final manuscript. R.L. and Y.M. contributed equally to this work.

Funding

This work was supported in part by the National Natural Science Foundation of China (Grant No. 62302034), the 2026 Guangdong Provincial Science and Technology Innovation Strategy Special Fund (Grant No. pdjh2026bkl57), and the 2026 Guangdong Provincial College Student Innovation and Entrepreneurship Training Program (Project No. S11106202614).

Additional Information

Competing interests

The authors declare no competing interests.

References

1. Mao, W. et al. Analysis and research on the relationship between bird related faults in transmission lines and bird activity patterns. E3S Web Conf. 617, 01006; (2025). https://doi.org/10.1051/e3sconf/202561701006
2. Bevanger, K. Biological and conservation aspects of bird mortality caused by electricity power lines: a review. *Biol. Conserv.* **86**, 67–76. https://doi.org/10.1016/S0006-3207(97)00176-6 (1998).
3. Jenkins, A. R., Smallie, J. J. & Diamond, M. Avian collisions with power lines: a global review of causes and mitigation with a South African perspective. *Bird. Conserv. Int.* **20**, 263–278. https://doi.org/10.1017/S0959270910000122 (2010).
4. Dwyer, J. F., Harness, R. E. & Donohue, K. Predictive model of avian electrocution risk on overhead power lines. *Conserv. Biol.* **28**, 159–168. https://doi.org/10.1111/cobi.12145 (2014).
5. Biasotto, L. D. et al. Risk of bird electrocution in power lines: a framework for prioritizing species and areas for conservation and impact mitigation. *Anim. Conserv.* **25**, 285–296. https://doi.org/10.1111/acv.12736 (2022).
6. Li, J., Yan, D., Luan, K., Li, Z. & Liang, H. Deep learning-based bird's nest detection on transmission lines using UAV imagery. *Appl. Sci.* **10**, 6147. https://doi.org/10.3390/app10186147 (2020).
7. Li, H., Dong, Y., Liu, Y. & Ai, J. Design and implementation of UAVs for bird's nest inspection on transmission lines based on deep learning. *Drones* **6**, 252. https://doi.org/10.3390/drones6090252 (2022).
8. Zhu, H. & Xie, X. Enhanced YOLOv11n for small object detection in UAV imagery: higher accuracy with fewer parameters. *Sci. Rep.* **16**, 5536. https://doi.org/10.1038/s41598-026-35301-2 (2026).
9. Du, D. et al. The unmanned aerial vehicle benchmark: object detection and tracking. In Proc. Eur. Conf. Comput. Vis. 375–391 (2018).
10. Cao, Y. et al. VisDrone-DET2021: the vision meets drone object detection challenge results. In Proc. IEEE/CVF Int. Conf. Comput. Vis. Workshops 2847–2854 (2021).
11. Xiang, Y. et al. BN-YOLO: a lightweight method for bird's nest detection on transmission lines. *J. Real-Time Image Process.* **21**, 194. https://doi.org/10.1007/s11554-024-01577-8 (2024).
12. Wang, Y. et al. The transmission line foreign body detection algorithm based on weighted spatial attention. *Front. Neurorobot*. **18**, 1424158. https://doi.org/10.3389/fnbot.2024.1424158 (2024).

13. Faisal, M. A. A. et al. Deep learning in automated power line inspection: a review. *Appl. Energy*. **385**, 125507. https://doi.org/10.1016/j.apenergy.2025.125507 (2025).
14. Wang, J., Wang, Y., Li, X., Yuan, B. & Wang, M. SCI-YOLO11: an improved defect detection algorithm for transmission line insulators based on YOLO11. *PLOS ONE*. **20**, e0322561. https://doi.org/10.1371/journal.pone.0322561 (2025).
15. Dong, N., Zhang, W., Chen, Z., Feng, H. & Jia, J. Bird's nest defect detection of transmission lines based on domain knowledge and occlusion reasoning. *IET Gener Transm Distrib.* **17**, 4946–4957. https://doi.org/10.1049/gtd2.13007 (2023).
16. Han, Y. et al. LRDS-YOLO enhances small object detection in UAV aerial images with a lightweight and efficient design. *Sci. Rep.* **15**, 22627. https://doi.org/10.1038/s41598-025-07021-6 (2025).
17. Yu, Z., Chu, L., Ma, Y., Zhou, Y. & Li, Q. Bird's nest detection of transmission lines tower based on visual saliency. *Comput. Appl. Softw.* **42**, 137–142 (2025).
18. Dai, Y., Lyu, D. & Guo, S. Transmission line rusted area detection scheme based on color and texture features. *Ind. Control Comput.* **31**, 39–40 (2018).
19. Ren, S., He, K., Girshick, R., Sun, J. & Faster R-CNN: towards real-time object detection with region proposal networks. *Adv. Neural Inf. Process. Syst.* **28**, 91–99 (2015).
20. Tan, M., Pang, R. & Le, Q. V. EfficientDet: scalable and efficient object detection. In Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recogn. 10781–10790 (2020).
21. Liu, W. et al. SSD: single shot MultiBox detector. In Computer Vision – ECCV. 21–37 (Springer, 2016). (2016).
22. Redmon, J., Divvala, S., Girshick, R. & Farhadi, A. You only look once: unified, real-time object detection. In Proc. IEEE Conf. Comput. Vis. Pattern Recogn. 779–788 (2016).
23. Qiu, Y. & Lin, Z. Lightweight model LMW-YOLO for small object detection in remote sensing images. *Sci. Rep.* **16**, 11644. https://doi.org/10.1038/s41598-026-45055-6 (2026).
24. Ge, J. et al. MFR-YOLO: advancing UAV object detection with multi-scale feature refinement via deformable convolution and global attention. *Sci. Rep.* **16**, 15587. https://doi.org/10.1038/s41598-026-45641-8 (2026).
25. Priya, P. & Kamlu, S. S. Inspection of power line insulator using an enhanced faster region-convolutional neural network. *Sci. Rep.* **16**, 23127. https://doi.org/10.1038/s41598-026-51304-5 (2026).
26. Song, L., Liu, S., Wang, K. & Yang, J. Identification method of power grid components and defects based on improved EfficientDet. *Trans. China Electrotech Soc.* **37**, 2241–2251. https://doi.org/10.19595/j.cnki.1000-6753.tces.210956 (2022).
27. Zhao, Z., Jiang, A., Qi, Y., Zhang, W. & Zhao, W. Fittings detection in transmission line images with SSD model embedded occlusion relation module. *CAAI Trans. Intell. Syst.* **15**, 656–662. https://doi.org/10.11992/tis.202001008 (2020).
28. Tu, N., Jiao, M. & Yan, X. Bird's nest target image detection model for transmission lines in complex environments. *Comput. Eng.* **50**, 216–226. https://doi.org/10.19678/j.issn.1000-3428.0068852 (2024).
29. Xu, Y. et al. Lightweight overhead transmission line bird's nest detection network based on YOLOv5s. *Electron. Meas. Technol.* **47**, 138–148 (2024).
30. Khanam, R. & Hussain, M. YOLOv11: an overview of the key architectural enhancements. Preprint at (2024). https://arxiv.org/abs/2410.17725
31. Wu, Y. & He, K. Group normalization. In Proc. Eur. Conf. Comput. Vis. 3–19 (2018).
32. Howard, A. et al. Searching for MobileNetV3. In Proc. IEEE/CVF Int. Conf. Comput. Vis. 1314–1324 (2019).
33. Deng, H., Chen, Y. & Xu, Y. ALD-YOLO: a lightweight attention detection model for apple leaf diseases. *Front. Plant. Sci.* **16**, 1616224. https://doi.org/10.3389/fpls.2025.1616224 (2025).
34. Zhu, Y. et al. Small object detection in complex open-pit mine backgrounds based on improved YOLOv11. *J. Mine Autom.* **51**, 93–99. https://doi.org/10.13272/j.issn.1671-251x.2025020018 (2025).
35. Sunkara, R. & Luo, T. No more strided convolutions or pooling: a new CNN building block for low-resolution images and small objects. In Machine Learning and Knowledge Discovery in Databases 443–459 (Springer, 2023). https://doi.org/10.1007/978-3-031-26409-2_27

36. Cai, S. et al. LSOD-YOLO: a visual object detection method for AGV perception systems based on a lightweight backbone and detection head. *Technologies* **14**, 173. https://doi.org/10.3390/technologies14030173 (2026).
37. Mei, Y. & Li, R. CXS-BirdNest Dataset: a multi-scale UAV image dataset for bird nest detection on overhead transmission lines. Zenodo https://doi.org/10.5281/zenodo.22202440 (2026).
38. Lin, T. Y. et al. Springer,. Microsoft COCO: common objects in context. In Computer Vision – ECCV 2014 740–755 (2014).
39. Jung, A. B. et al. imgaug. GitHub (2020). https://github.com/aleju/imgaug
40. Li, X., Chen, H., Li, Y. & Peng, Y. MAFusion: multiscale attention network for infrared and visible image fusion. *IEEE Trans. Instrum. Meas.* **71**, 5014116. https://doi.org/10.1109/TIM.2022.3181898 (2022).
41. Yang, G. et al. AFPN: asymptotic feature pyramid network for object detection. In Proc. IEEE Int. Conf. Syst. Man Cybern. 2184–2189 (2023).
42. Doherty, J., Gardiner, B., Kerr, E. & Siddique, N. BiFPN-YOLO: one-stage object detection integrating bi-directional feature pyramid networks. *Pattern Recogn.* **160**, 111209. https://doi.org/10.1016/j.patcog.2024.111209 (2025).
43. Tian, Y., Ye, Q. & Doermann, D. YOLOv12: attention-centric real-time object detectors. *Adv. Neural Inf. Process. Syst.* **38**, 87151–87175. https://doi.org/10.52202/085713-2627 (2025).
44. Yan, F. et al. Wavelet and prototype augmented query-based transformer for pixel-level surface defect detection. In Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recogn. 23860–23869 (2025).
45. Kong, L. et al. Context-aware and semantic-guided adaptive filtering network for infrared small target detection. *IEEE Trans. Geosci. Remote Sens.* **64**, 5000512. https://doi.org/10.1109/TGRS.2025.3646255 (2026).
46. Li, S., Ma, Q., Zhang, S. & Yang, C. DCCS-Det: directional context and cross-scale-aware detector for infrared small target. *IEEE Trans. Geosci. Remote Sens.* **64**, 4400215. https://doi.org/10.1109/TGRS.2025.3646345 (2026).
47. Kim, D., Kim, Y., Oh, Y. J. & Kim, T. H. UCMNet: uncertainty-aware context memory network for under-display camera image restoration. In Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recogn. 29939–29948 (2026).
48. Cai, X., Sun, C., Wang, Y., Yang, H. & Wu, Y. PFGNet: a fully convolutional frequency-guided peripheral gating network for efficient spatiotemporal predictive learning. In Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recogn. 38848–38858 (2026).
49. Li, H., Wu, Z., Shao, R. & Fu, Y. Statistical characteristic-guided denoising for rapid high-resolution transmission electron microscopy imaging. In Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recogn. 34050–34060 (2026).
50. Lin, X., Peng, J., Gan, Z., Zhu, J. & Liu, J. YOLO-Master: MOE-accelerated with specialized transformers for enhanced real-time detection. In Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recogn. 18440–18449 (2026).
51. Jocher, G. et al. Ultralytics YOLO26: unified real-time end-to-end vision models. Preprint at (2026). https://arxiv.org/abs/2606.03748
52. Han, D., Li, T., Wang, Z. & Huang, G. Vision transformers are circulant attention learners. Proc. AAAI Conf. Artif. Intell. 40, 21549–21557 (2026). https://doi.org/10.1609/aaai.v40i26.39303
53. Qu, L. et al. It takes two: a duet of periodicity and directionality for burst flicker removal. In Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recogn. 15516–15527 (2026).
54. Setyawan, N., Sun, C. C., Hsu, M. H., Kuo, W. K. & Hsieh, J. W. MicroViTv2: beyond the FLOPs for edge energy-friendly vision transformers. Preprint at (2026). https://arxiv.org/abs/2605.10148
55. Selvaraju, R. R. et al. Grad-CAM: visual explanations from deep networks via gradient-based localization. In Proc. IEEE Int. Conf. Comput. Vis. 618–626 (2017).

Figures

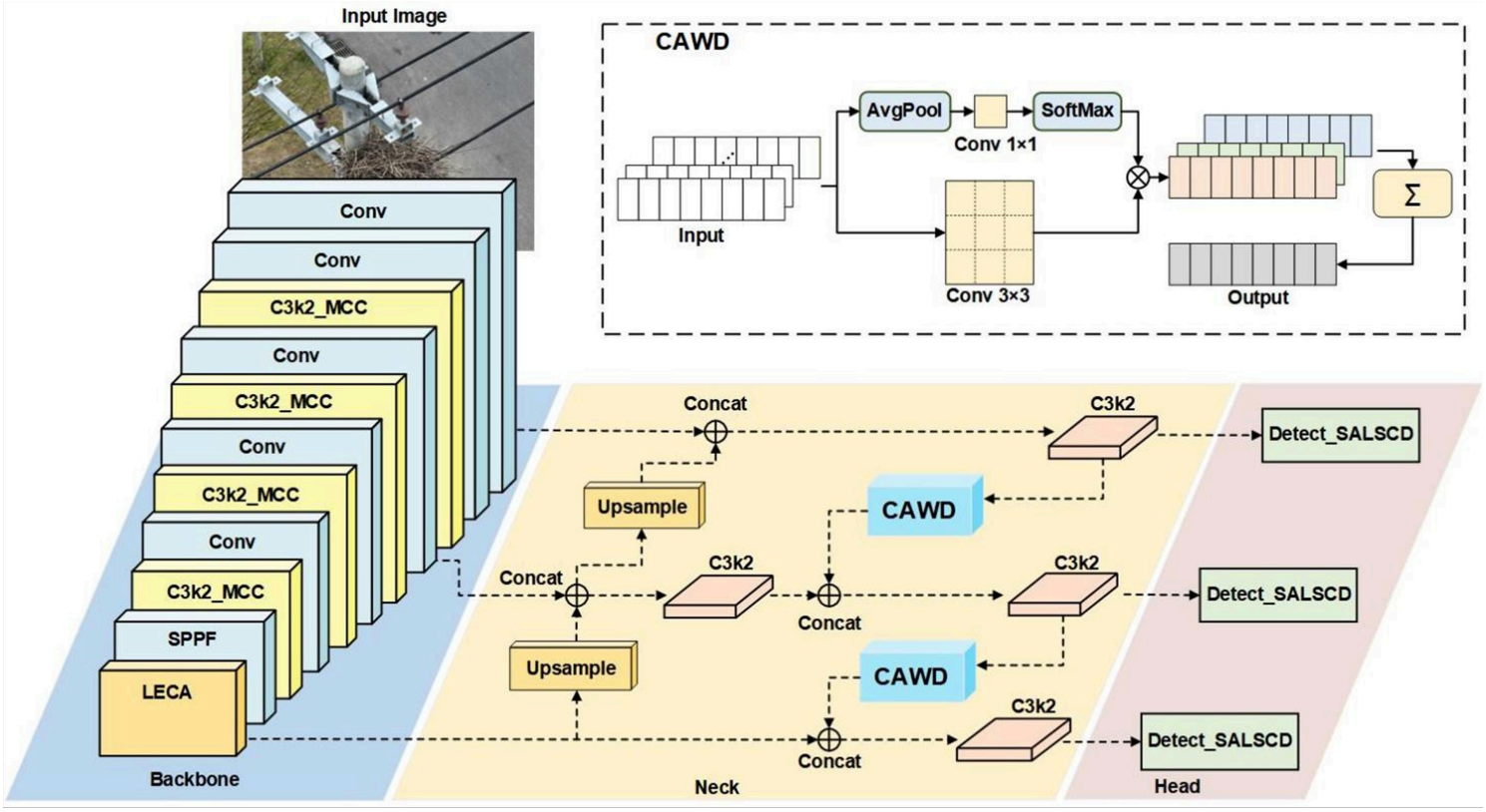


Figure 1

Overall architecture of CXS-YOLO.

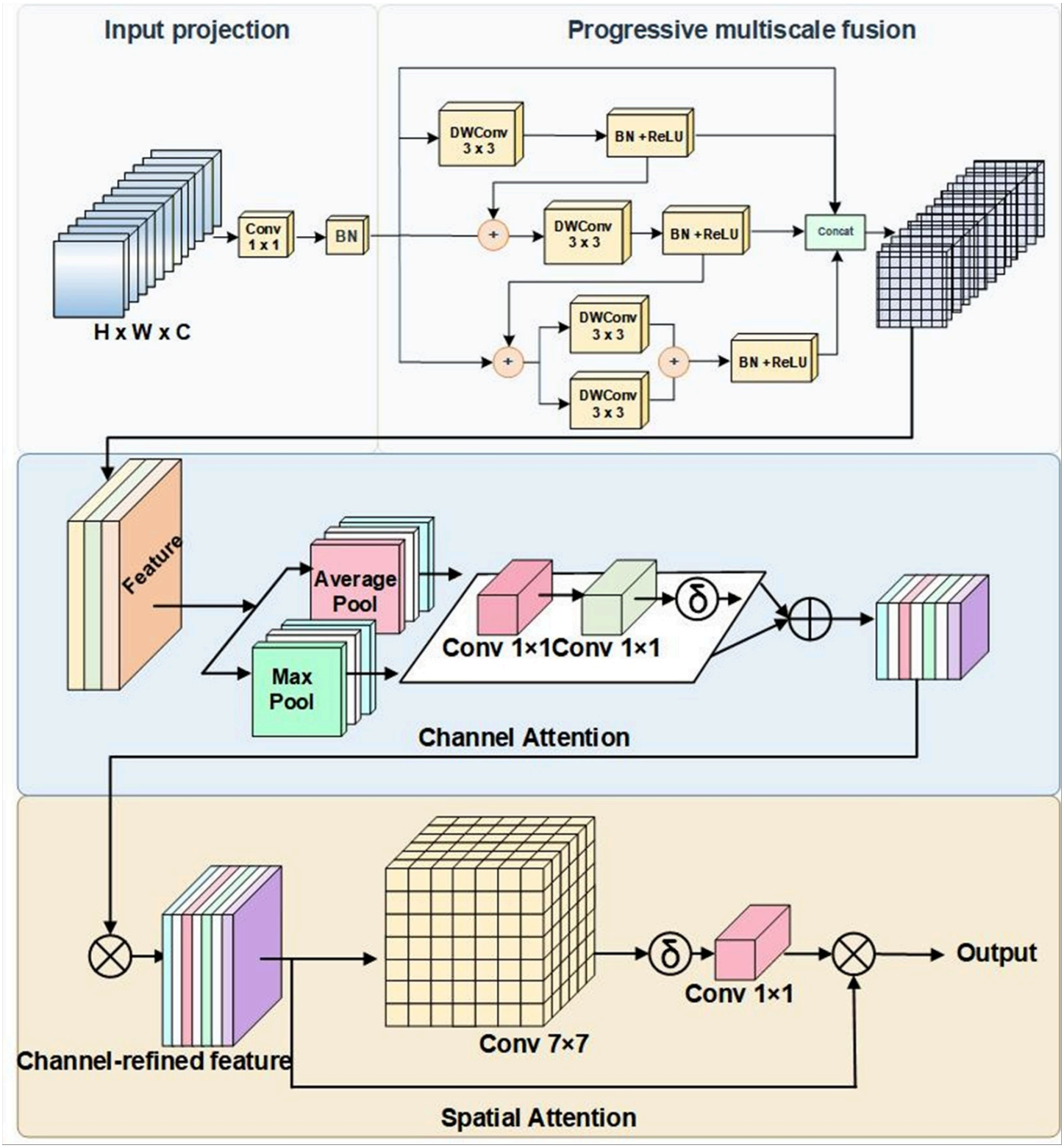


Figure 2

Architecture of MCC_Bottleneck.

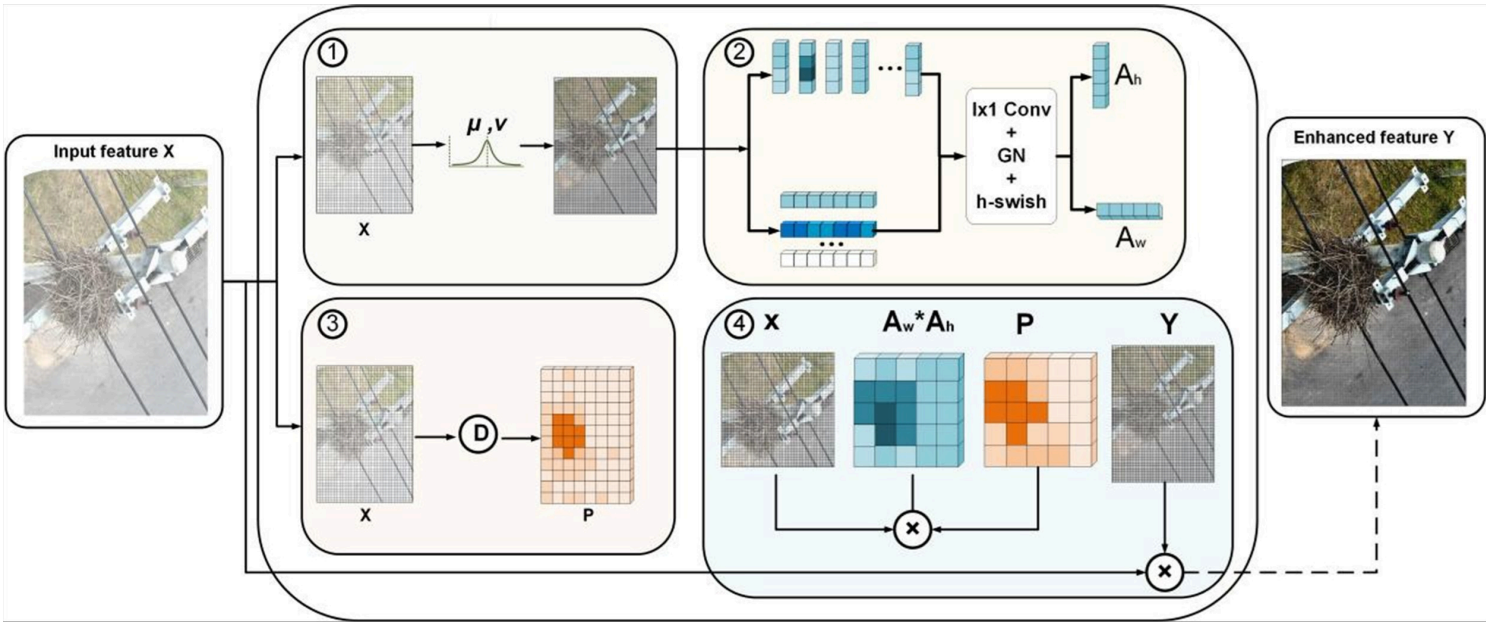


Figure 3

Schematic of the LECA attention mechanism.



Figure 4

Representative bird-nest samples at different scales.

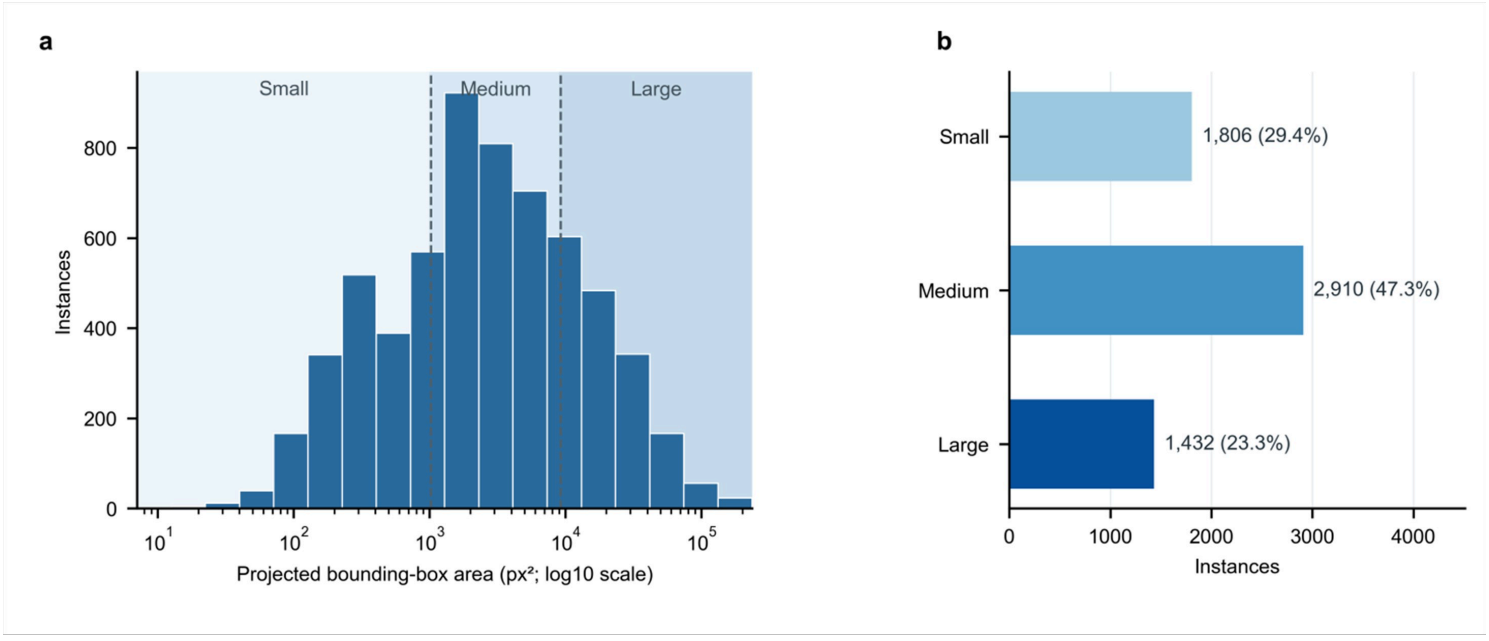


Figure 5

Distribution of bird-nest instances across different scales in the dataset

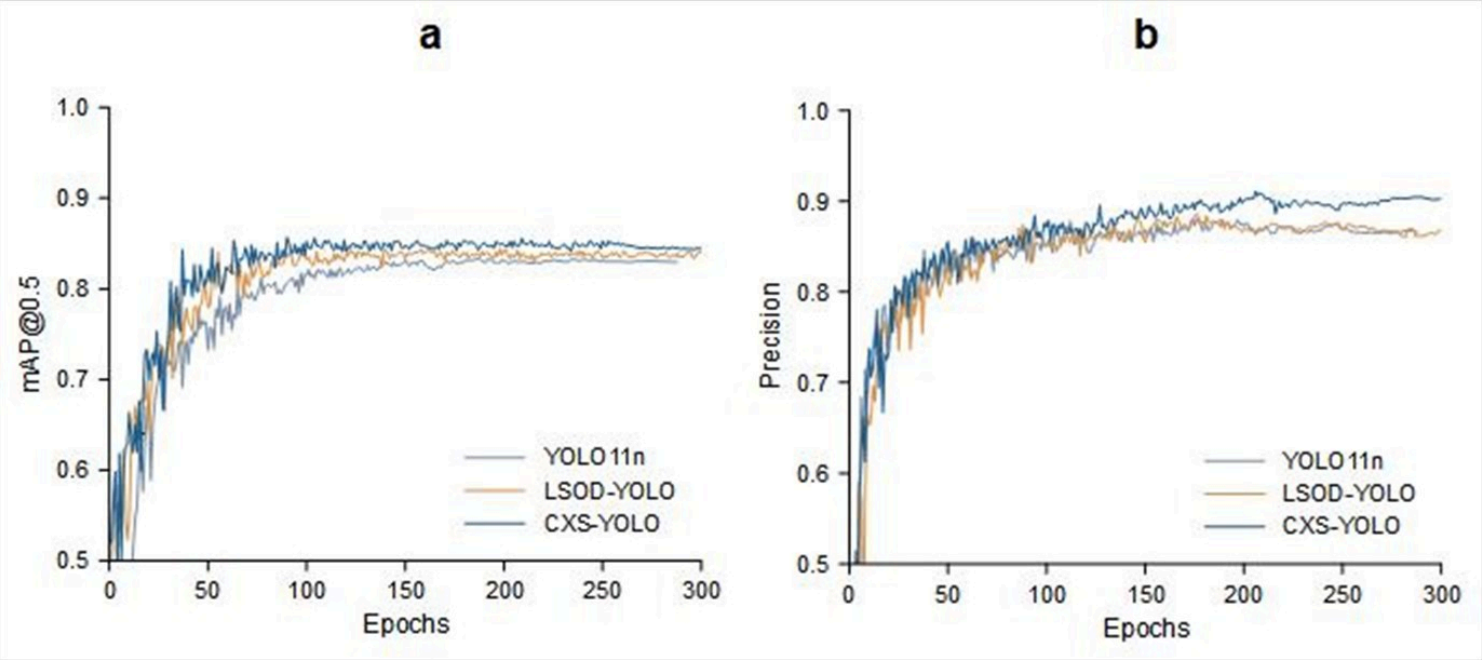


Figure 6

mAP@0.5 and Precision curves of YOLO11n, LSOD-YOLO and CXS-YOLO during training. (a) mAP@0.5. (b) Precision.

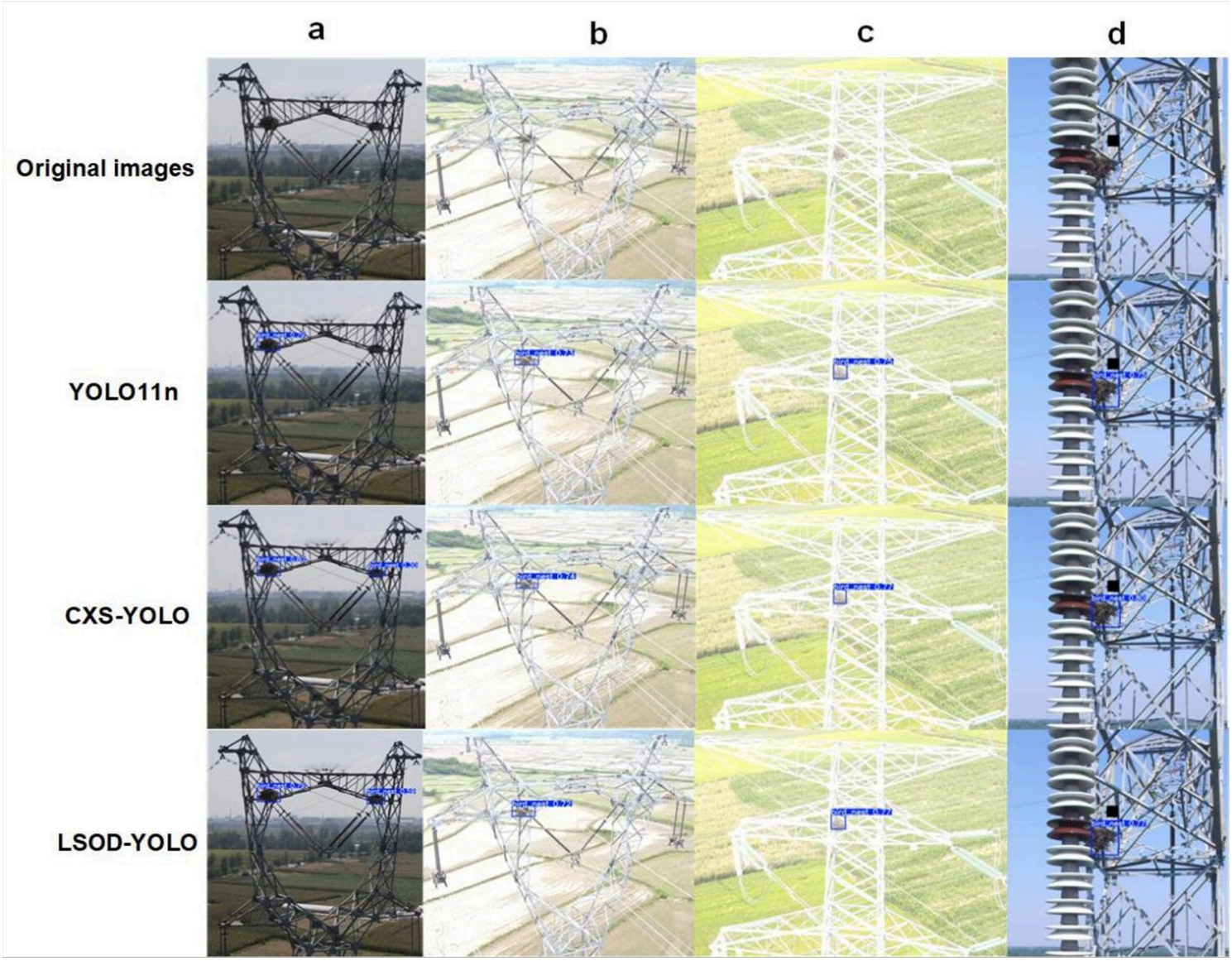


Figure 7

Detection results of different models under complex inspection conditions. The columns correspond to (a) heavy fog, (b) transmission-tower occlusion, (c) strong illumination and (d) occlusion by foreign objects.

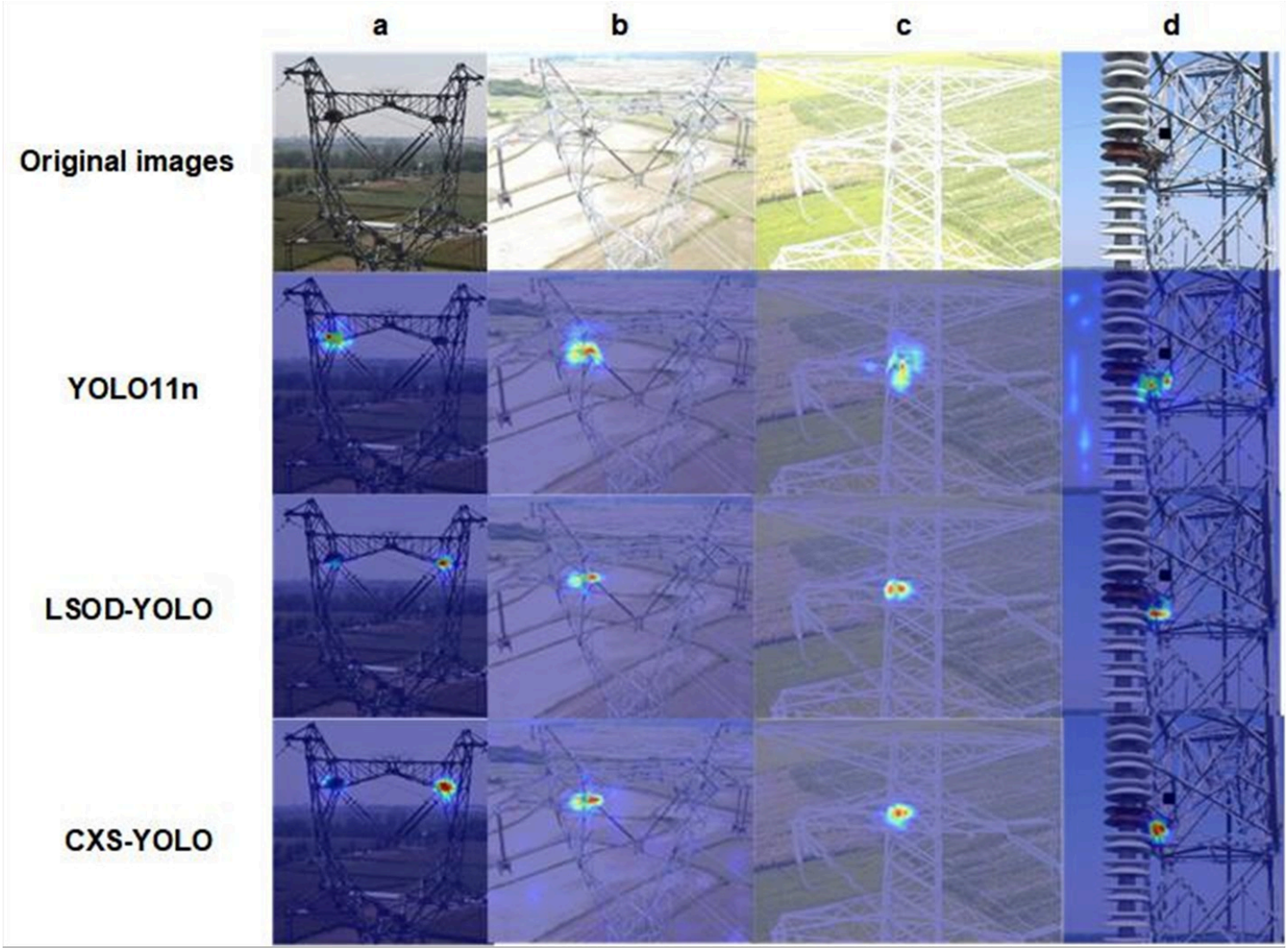


Figure 8

Grad-CAM visualisations of different models under complex power-transmission-line inspection conditions. The columns correspond to (a) heavy fog, (b) transmission-tower occlusion, (c) strong illumination and (d) occlusion by foreign objects

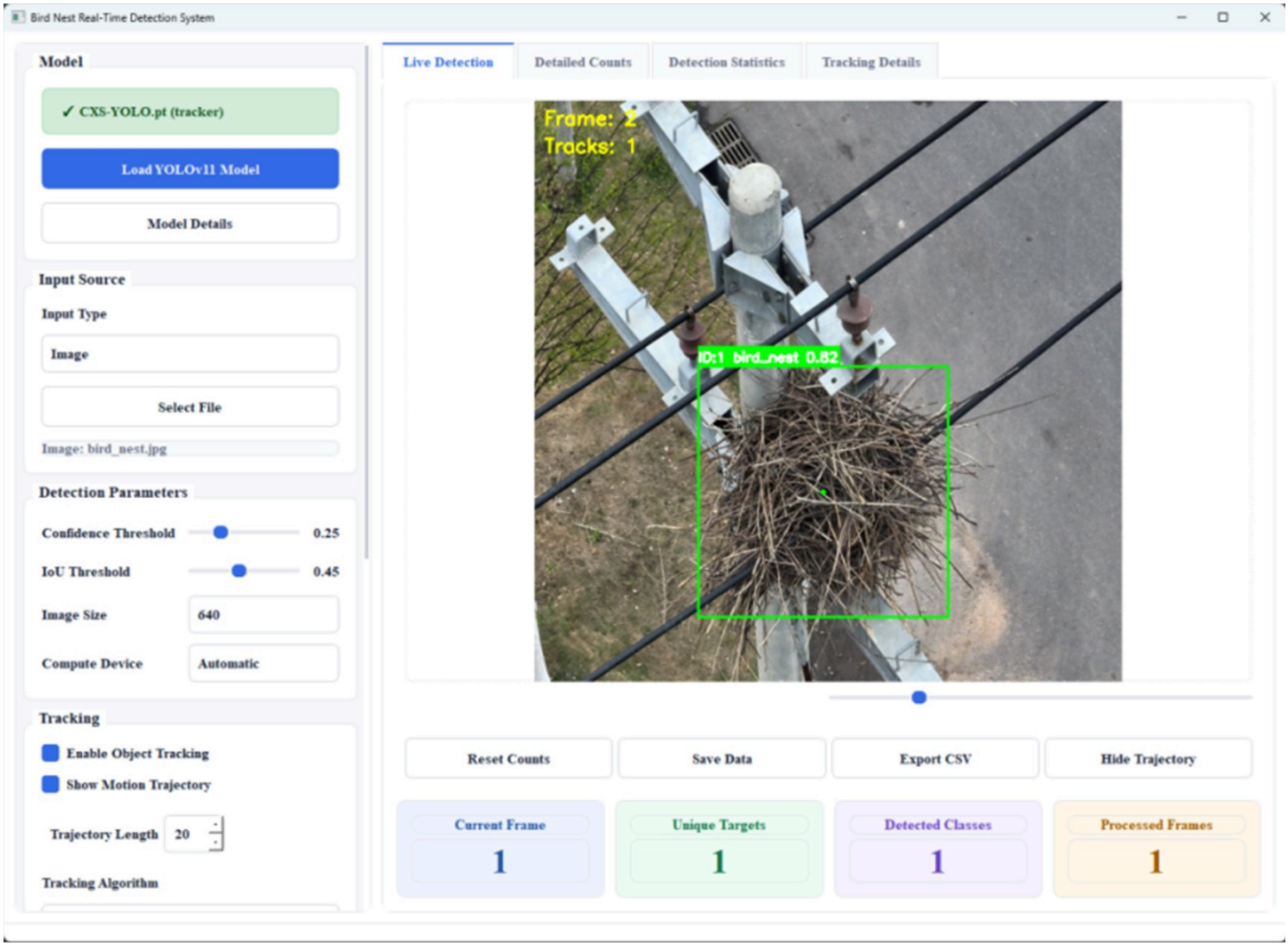


Figure 9

Prototype interface of the CXS-YOLO bird-nest detection system

Supplementary Files

This is a list of supplementary files associated with this preprint. Click to download.

- Randomseedtestresults.docx