Insulator detection based on FAYOLO network with improved feature extraction ability

Received: 21 December 2023 | Revised: 28 June 2024 | Accepted: 26 July 2024

IET Image Processing

DOI: 10.1049/ipr2.13197

The Institution of Engineering and Technology WILEY

ORIGINAL RESEARCH

Insulator detection based on FA-YOLO network with improved feature extraction ability

Yixiao Jing[1] | Tao Huang[2] | Linfeng Gao[1] | Jiangli Deng[3]

[1]Computer Engineering, University of British Columbia, Vancouver, British Columbia, Canada

[2]Department of energy, Politecnico di Torino, Torino, Italy

[3]Operations and Maintenance Department, Yibin Power Supply Company, State Grid, Yibin, Sichuan, China

Correspondence
Tao Huang, Department of energy, Politecnico di Torino, Torino, Italy.
Email: tao.huang@polito.it

Abstract

Unmanned aerial vehicle insulator detection that aims to recognize defective insulators from transmission lines has made significant progress in recent years. However, it still faces challenges, such as the complex background of aerial images and the small memory of unmanned aerial vehicles. This paper proposes a refined insulator detection algorithm that integrates the attention mechanism in YOLOv8 to improve the feature extraction ability. Specifically, this paper introduces a fast vision transformers structure in the you only look once (YOLO) v8 backbone section to enhance feature extraction by capturing local and global features. Additionally, the global attention mechanism is incorporated in the neck for additional feature extraction by merging comprehensive spatial and channel information into the output. Furthermore, we amalgamate depth-wise convolution, graph convolution, and residual operation in the global attention mechanism module. This design can mitigate the issues of gradient vanishing or exploding and meanwhile enhance the distinction between spatial attention and channel attention. The proposed model is then applied to a public dataset and a set of real images from a specific power station, and the detection results show that it outperforms many competitors in terms of accuracy, efficiency, and memory size.

1 | INTRODUCTION

Insulators serve as integral components within intricate electrical power systems, tasked with the critical role of ensuring operational security by isolating conductors from towers and mitigating current leakage. Despite their importance, insulators are inherently vulnerable to degradation due to prolonged exposure to adverse environmental conditions [1, 2], which introduces a considerable risk factor concerning the overall reliability and efficiency of power systems. The original method to inspect the insulators involved artificial detection [3], which is inefficient and prone to errors. It requires the inspectors to visualize the insulators remotely from one tower to the next, making it easy to miss hidden failures such as cracks and dirt accumulation. In contrast, unmanned aerial vehicles (UAVs), known for their low cost, robust flexibility, and compactness, have emerged as a popular alternative for such inspections [4]. These UAVs are equipped with high-resolution imaging systems that capture detailed visual data of insulators. Advanced image processing techniques are then applied to this data, enabling the automatic detection of insulators and ensuring a more accurate and efficient inspection process in the further analysis.

Insulator detection methodologies employed in this process can be broadly categorized into traditional techniques and deep-learning-based approaches [5]. Traditional methods primarily focus on feature extraction techniques. For instance, Gao et al. applied SURF, a traditional feature extraction algorithm, to improve the Capsule network [6]. Moreover, Zhai et al. [7] differentiated the insulators with and without faults from spatial morphological features analysis. However, these methods often struggle in complex scenarios, such as those involving intricate backgrounds or insulators with varying shapes and colours [8]. As a result, deep-learning-based models, which are capable of automatically learning features from data, have become the centre of attention in recent research on insulator detection [8].

Deep-learning architectures for insulator detection can be broadly categorized into two-stage detectors like faster region-based convolutional neural network (faster R-CNN) and one-stage detectors like single shot multibox detector (SSD) and you only look once (YOLO).

Starting with two-stage detectors, faster R-CNN employs a two-stage process involving region proposals followed by object

Tao Huang and Linfeng Gao contributed equally to this study.

classification, which is generally more accurate but computationally intensive [9]. Particularly, Zhao et al. incorporated feature pyramid networks into faster R-CNN to address challenges arising from intricate backgrounds [10]. Shenglong Liao and Jubai introduced a multi-scale multi-feature descriptor (MSMF) that can handle rotation and scale changes, identified spatial order features (SOF) to enhance the robustness of the algorithm [11]. Additionally, Zhao et al. proposed an enhancement to the faster R-CNN network by improving the anchor generation method, which effectively reduces false detections [12]. However, this anchor setting is suboptimal for detecting small targets, such as insulators. Moving to one-stage detectors, which normally have reduced complexity and faster prediction [13]. Specifically, SSD focuses on a single-shot detection mechanism that is efficient but may compromise accuracy. To improve its performance, Miao et al. enhanced the SSD architecture with a bifurcated fine-tuning process [14]. On the other hand, the YOLO series has been a subject of considerable research interest. Liu et al. tailored the YOLOv3 architecture for aerial insulator detection to balance the performance and the efficiency.[15]. Hao et al. introduced a variant of YOLOv5, utilizing deep weakly supervised and transfer learning techniques for insulator detection, specifically tailored for icy conditions. Nevertheless, the diverse dimensions of 2 insulators accompanied by intricate backgrounds persist as challenges to the relatively efficient one-stage model. To further enhance the model's performance on complex scenarios, attention mechanisms have been integrated into the architecture. Song et al. added self-attention and global attention modules to YOLOv5s, thereby increasing its ability to discriminate in complex backgrounds [13]. Moreover, YOLOX++ developed by Zhongqi and his team improves the detection accuracy and robustness of small targets through multi-scale cross-stage partial network (MS-CSPNet), deep convolution and dilated convolution in the object decoupling head [16]. Introduced in 2023, YOLOv8 [17] strikes a commendable balance between accuracy, computational complexity, and memory constraints, making it a notable model in the realm of insulator detection. Based on it, He et al. [18] propose an improved YOLOv8 algorithm, MFI-YOLO, incorporating MSA-GhostBlock for enhanced feature extraction and ResPANet for improved multi-scale feature fusion, significantly boosting detection accuracy. However, the custom dataset used for training is overly focused and does not account for insulator detection in complex backgrounds.

The insulator detection system not only aids in localizing insulators but also enhances defect detection. In dual-class detection, insulators and defects share features, which enhances defect detection because the model leverages the context provided by accurately detecting insulators to better identify defects situated on them. Our approach is particularly significant considering that it employs real-world data from a power station. This dataset, while rich in its representation of authentic environmental and operational conditions, naturally presented limitations in terms of the variety and quantity of defect data available for training. Consequently, we have decided to focus our efforts on accurate insulator detection to reduce the risks associated with missed detections and mislocalizations.

To address the multifaceted challenges of balancing detection accuracy in intricate scenarios, computational complexity, convergence speed, and computational resources constraints, our research refines the YOLOv8 architecture. This enhancement renders the model particularly well-suited for small-scale development environments with limited computational resources yet high precision requirements. Our main contributions are as follows:

1. A refined insulator detection algorithm, which integrates the attention mechanism in YOLOv8 to improve the feature extraction ability (FA-YOLO), is proposed. The proposed model substantially enhances the feature extraction ability and ensures a seamless integration of accuracy, model size, and GPU memory usage.
2. We replace the C2f blocks in the YOLOv8 backbone with fast vision transformers (FasterViT) layers to capture both global and local features while managing model complexity. Additionally, we introduce an enhanced global attention mechanism (GAM) incorporating depthwise convolution, graph convolution, and a residual module (GAM-GDR) to fuse spatial and channel attention, thereby enhancing feature completeness.

The remainder of this paper is organized as follows. Section 2.3 delineates the framework of our proposed model. Subsequently, Section 3.3 provides a detailed exposition of our strategic modifications. Afterward, Section 4 presents our experimental setup and results. The results presented include a comparison of our model with other comparable models and an analysis of different FA-YOLO variants, aiming to elucidate the rationale behind our modifications. Finally, Section 5 summarizes this paper.

2 | MODEL FRAMEWORK

The architecture of YOLOv8 is divided into three main sections: the backbone, the neck, and the head. The backbone is responsible for feature extraction and processes input images into various sizes for multi-scale detection. Subsequently, the neck integrates these multi-scale features and further refines them, ensuring a more comprehensive representation. Lastly, the head handles the final object detection and classification tasks, pinpointing object locations and their respective categories within the image.

To enhance the performance and efficiency of the insulator detector, YOLOv8 is improved with the integration of FasterViT layers, GAM-GDR blocks, and SCYLLA-IoU (SIoU). Figure 1 renders the full structure of our model. Particularly, C2f blocks are replaced with FasterViT layers in the backbone and GAM-GDR blocks are appended before detect blocks. The substitution of the C2f module with the FasterViT module enhances feature representation and improves the model's

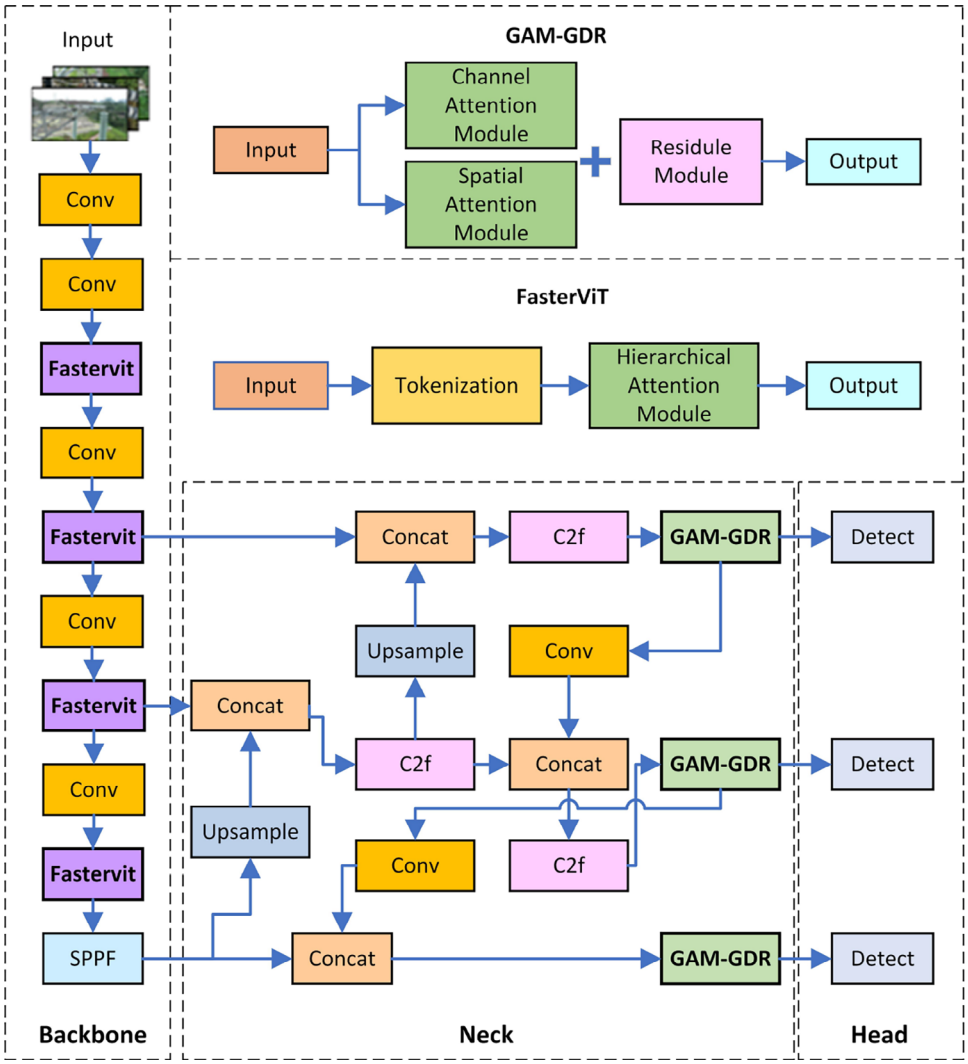


FIGURE 1 The structure of FA-YOLO.

ability to balance the features from maps with varying scaling, thereby enabling long-range dependencies capture. Incorporating GAM-GDR blocks before detect blocks helps to constrict the irrelevant features and integrate the full context into the input maps, improving the extracted features that are utilized in making the final decisions. Finally, the bounding box regression loss function is refined by substituting the conventional CIoU with SIoU to bolster the model's precision in delineating the true contours of the bounding boxes.

2.1 | Backbone

The multi-scale pyramid, achieved by the utilization of downsized convolution layers, is a basic characteristic of the YOLO architecture. According to Figure 1, the backbone of FA-YOLO contains five convolution layers, four FasterViT layers, and one SPPF layer. Particularly, the initial step of the FA-YOLO model involves reshaping the input images to a consistent resolution. This resolution is denoted as $n \times n$ pixels, which serves as the base resolution of the model. The features of the images are then extracted through two convolutional layers, which aim at discerning rudimentary visual patterns. The FasterViT layers are incorporated to replace the ordinary C2f blocks inside the backbone of YOLOv8, introducing a distinct hierarchical approach with a tokenizing mechanism. This approach is more effective in images with intricate backgrounds and varying levels of object occlusion, as detailed in Section 3.2. Post this transformative FasterViT processing, the data is channelled through a series of convolutional and FasterViT layers for further refinement. These layers continuously extract the features and downsize the feature map into five scales, including $\frac{n}{2} \times \frac{n}{2}$, $\frac{n}{4} \times \frac{n}{4}$, $\frac{n}{8} \times \frac{n}{8}$, $\frac{n}{16} \times \frac{n}{16}$, and $\frac{n}{32} \times \frac{n}{32}$. This refined data is then introduced to the spatial pyramid pooling fusion (SPPF) module, which strategically segments the image into multi-scale regions, ensuring the ability of the model to detect objects of all sizes.

2.2 | Neck

The neck module is designed to merge the multi-scale feature maps from the backbone and to conduct additional feature extraction, thereby enhancing the model's capability to discern and interpret diverse features within the input. To achieve this, three concatenation blocks, coupled with two upsample blocks, are employed for feature maps integration.

Specifically, the neck module is structured with three input pathways and three output pathways. These pathways strategically originate from the three lowest layers of the pyramid structure and are connected to three detection blocks respectively. This intentional design is aimed at extracting the diverse features extracted from different levels of the pyramid, enriching the feature representation and bolstering the detection accuracy.

The first pathway is integral for leading to the uppermost detection. It originates from the second FasterViT layer, producing feature maps of size $\frac{n}{8} \times \frac{n}{8}$, and from the third FasterViT layer, yielding feature maps of size $\frac{n}{16} \times \frac{n}{16}$. In this pathway, feature maps with a resolution of $\frac{n}{16} \times \frac{n}{16}$ are retained through the C2f block without any alteration in their size, preserving the integrity of the extracted features. These maps are then upsampled, resulting in a size of $\frac{n}{8} \times \frac{n}{8}$, enabling their concatenation with other feature maps of the same size. Thereafter, the feature maps from the concatenation refine the spatial and channel-wise details further via the C2f and GAM-GDR blocks, ensuring a robust and comprehensive feature representation. According to Figure 1, a GAM-GDR layer consists of a channel attention module, a spatial attention module, and a residual module. These components work together to offer a richer context prior to detection, as elaborated in Section 3.1.

The second pathway emerges from the combination of upsampled $\frac{n}{32} \times \frac{n}{32}$ feature maps from the SPPF and the $\frac{n}{16} \times \frac{n}{16}$ feature maps from the third FasterViT layer. This pathway is designed to enhance the merged maps via the C2f block before their integration with $\frac{n}{8} \times \frac{n}{8}$ feature maps, which have been reduced to an $\frac{n}{16} \times \frac{n}{16}$ resolution. This design is crucial for achieving a harmonious blend of features, allowing for a more detailed feature representation. Again, the details are refined further by the C2f and GAM-GDR modules before being processed by the detect block.

Finally, the third pathway has the largest receptive field. It combines the $\frac{n}{32} \times \frac{n}{32}$ feature maps from the SPPF with the $\frac{n}{16} \times \frac{n}{16}$ feature maps from the last step of the second pathway. This combination is vital for capturing the global context of the maps, allowing the model to understand the overall structure and semantics of the image. GAM-GDR is applied as well to further the comprehensive understanding of the global features.

In conclusion, the design of the neck module is a strategy to amalgamate diverse features from different scales and complexities, ensuring a richer and more robust feature representation. This enhanced representation addresses the inherent challenges posed by the variability in object scales, shapes, and contexts within the input images.

2.3 | Head

In YOLOv8, a distinctive decoupled head design is employed, enabling each block to focus on its specific task within a particular size range. As illustrated in Figure 1, FA-YOLO utilizes three detect layers as its detection head, operating at scales of $\frac{n}{8} \times \frac{n}{8}$, $\frac{n}{16} \times \frac{n}{16}$, and $\frac{n}{32} \times \frac{n}{32}$, respectively. This multi-scale approach allows for effective and accurate detection across different object sizes [19].

Building on this architecture, each detect block is responsible for generating the bounding box predictions and class predictions. Bounding box predictions determine the object localization, where the model predicts the coordinates and dimensions of a box that is supposed to encapsulate the object in the image. These predictions allow the model to accurately isolate objects of varying sizes and shapes. Furthermore, YOLOv8 employs an anchor-free strategy. Consequently, the predictions can be achieved through a regression task to directly predict the coordinates and dimensions of the bounding boxes without relying on predefined anchor boxes, allowing for more flexibility and efficiency in object detection [20].

After localized single class objects in the image, class predictions are primarily used to determine the presence or absence of the target object within the predicted bounding box. Class predictions assign a confidence score indicating the likelihood that the bounding box contains the object [21]. If the score surpasses a certain threshold, the bounding box is considered to contain the target object; otherwise, it's treated as background.

Ultimately, the integration of bounding box and class predictions are made on the scale of $\frac{n}{8} \times \frac{n}{8}$, $\frac{n}{16} \times \frac{n}{16}$, and $\frac{n}{32} \times \frac{n}{32}$ and merged to address the challenges posed by the variability in object appearances, scales, and contexts.

3 | IMPROVED MODULES

3.1 | GAM-GDR

Introduced by Liu et al., GAM has demonstrated impressive performance on both the ImageNet and CIFAR-100 datasets [22]. GAM integrates a spatial attention module and a channel attention module, enabling the amalgamation of global information derived from cross-dimensional interactions into the final output feature maps. The spatial attention module employs two convolution layers to achieve more comprehensive feature extractions. The incorporation of two multi-layer perceptrons (MLP) within the channel attention module amplifies the cross-dimensional dependencies between channels and spatial locations, thereby enhancing the robustness and reliability of the system. Incorporating GAM before the detect layers aims to refine the feature maps with richer semantic contexts,

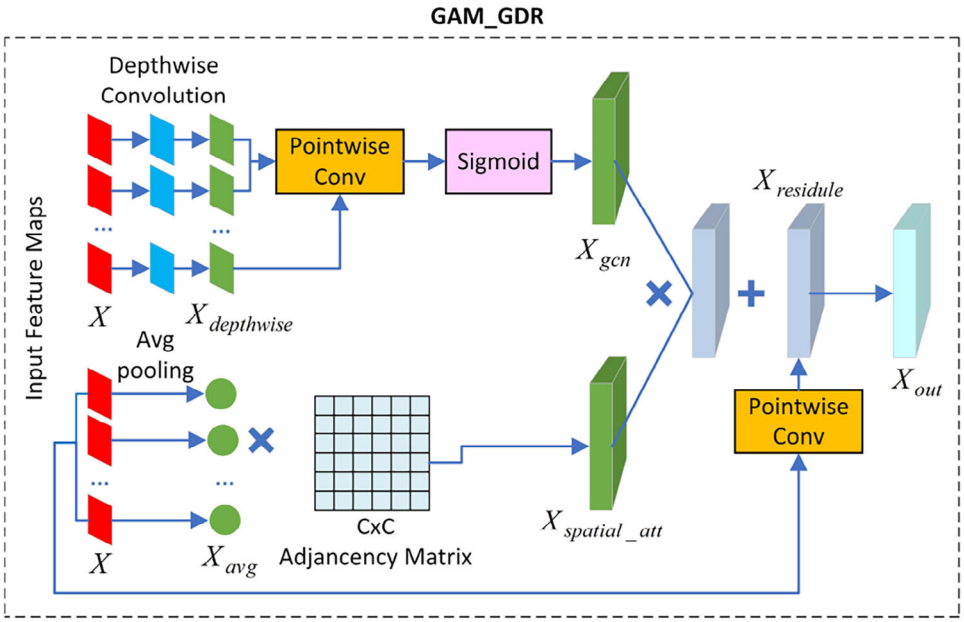


FIGURE 2 The structure of GAM-GDR.

enhancing the precision of object localization and classification in the final stages of the model.

To further the performance of the model and alleviate gradient vanishing or exploding, GAM-GDR is introduced, which incorporates depthwise convolution, graph convolution, and residual module, as shown in Figure 2.

In the standard GAM, the traditional hierarchical convolution structure forms the basis of the spatial attention mechanism. This structure integrates receptive fields of various scales and amalgamates the results derived from different input feature maps. To enhance the model's efficiency and specificity in feature processing, the first convolution layer is replaced with a depthwise convolution layer. This modification compels the model to focus more precisely on individual features within each channel, as the depthwise convolution operation applies filters separately to each channel of the input feature maps. After the depthwise convolution, an activation layer is applied. This is then followed by a pointwise convolution layer, which uses 1×1 filters to combine the outputs of the depthwise convolution. After the pointwise convolution, a sigmoid function is applied, constraining values into the range from 0 to 1. The sequence of operations can be represented as follows:

$$\begin{aligned} X_{\text{depthwise}} &= f(D_{\text{conv}}(X)) \\ X_{\text{pointwise}} &= P_{\text{conv}}(X_{\text{depthwise}}) \\ X_{\text{spatial\_att}} &= \sigma(X_{\text{pointwise}}) \end{aligned} \tag{1}$$

where f is the activation function, $X_{\text{depthwise}}$ is the feature maps from depthwise convolution, $X_{\text{pointwise}}$ refers to the feature maps from pointwise convolution, $X_{\text{spatial\_att}}$ refers to the spatial attention maps, D_{conv} denotes the depthwise convolution, P_{conv} represents the pointwise convolution, and σ is the sigmoid function.

In terms of channel attention, the original MLP reshapes each feature map, assigning a weight to each channel based on the entirety of its spatial information. To mitigate the influence of noise and potential over-reliance on spatial details, an average pooling step is incorporated. This not only helps diminish overfitting but also mitigates the emphasis on spatial information. Furthermore, the conventional MLP layers are substituted with an adjacency matrix. As indicated in Figure 2, the adjacency matrix contains $C \times C$ elements, where C represents the number of channels in X. Each element in the matrix reflects the direct relationship between the two channels. This alteration encourages the model to prioritize certain pivotal channels, further curtailing the risk of overfitting.

The graph convolution operation involves multiplying the adjacency matrix Adj with the product of the transposed average-pooled feature maps X_{avg}^{T}, containing the average value for each feature map and the weight matrix W. The adjacency matrix depicts the interrelationships between different vertices, where each channel is represented as a vertex in a graph. By doing so, the feature maps of graph convolution can be constructed, denoted as X_{gcn}, which also represents the channel attention maps. X_{gcn} can be calculated as follows:

$$X_{\text{gcn}} = \left(\text{Adj} \times \left(X_{\text{avg}}^{T} \times W\right)\right)^{T} \tag{2}$$

The main idea behind the modifications in GAM is to enhance the distinction between spatial and channel attention, allowing each to specialize in its respective domain. This approach ensures spatial attention captures intricate spatial patterns and channel attention discerns inter-channel dependencies

(a) (b)

FIGURE 3 An example that highlights the strength of GAM-GDR.

without distracting each other. Furthermore, this separation helps to filter out irrelevant or redundant information, ensuring that the model focuses on the most salient features. As a result, the model becomes more adaptable and less susceptible to noises in input data, bolstering the model's robustness.

Given the depth and the limited complexity of GAM-GD, there may be issues of gradient vanishing or exploding and potential underfitting to the training data. This limitation has led us to incorporate a residual module within the GAM structure. As Figure 2 illustrates, the residual module contains a 1×1 pointwise convolution block, allowing the output to learn from the input weights. The addition of this module not only addresses the aforementioned problems but also enhances the model's performance by controlling the decrease in the model's performance [23].

After the modification, the formula of GAM-GDR is expressed as follows:

$$X_{\text{residual}} = P_{\text{conv}}(X)$$
$$X_{\text{out}} = (X \odot X_{\text{gcn}} \odot X_{\text{spatial\_att}}) + X_{\text{residual}} \quad (3)$$

where X_{residual} represents the residual feature maps and $\odot$ denotes element-wise multiplication.

The benefits of employing GAM-GDR technology are clearly demonstrated through a comparative example. In this example, as depicted in Figure 3, both the YOLOv8n model and its advanced counterpart, augmented with GAM-GDR layers, are tasked with analysing the same image. While both models accurately detect all the insulators present in the image, YOLOv8n fails to recognize that the two separate parts depicted actually belong to a single insulator. In contrast, the YOLO model with GAM-GDR successfully identifies and interprets it.

The observed performance difference between the two models can be attributed to the capabilities of the GAM-GDR. As noted, GAM-GDR integrates both channel and spatial attention mechanisms, enabling the model to grasp more sophisticated characteristics of an insulator and maintain a more comprehensive perspective of the entire image. Therefore, YOLOv8n with GAM-GDR demonstrates improved performance in such scenarios.

3.2 | FasterVit layer

In the realm of object detection algorithms, the YOLOv8 architecture represents a significant milestone, particularly due to its incorporation of the C2f module within its backbone. The C2f module is designed to achieve feature extraction across multiple scales, thereby enhancing the model's detection capabilities [24]. In the C2f module, the input feature maps are divided into two equal parts along the channel dimension, allowing the network to diversify its feature extraction. While this enhances the model's ability to represent data, it may not be adept at capturing long-distance spatial relationships and handling intricate contexts within the feature maps.

To address these constraints, our research explores the integration of the FasterViT layer as an alternative. FasterViT is an innovative vision transformer (ViT) architecture that has demonstrated exemplary performance in various visual tasks [25]. As illustrated in Figure 4, the input undergoes a series of convolution layers followed by downsample layers. The pivotal component of this process is the deployment of the hierarchical attention mechanism (HAT).

The main idea behind FasterViT is to tokenize the input feature map into patches and employ a hierarchical attention mechanism to enhance both local and global information exchange through self-attention. In terms of tokenization, the input feature map is partitioned into multiple local windows and stretched into local tokens in downsample blocks. These local tokens primarily focus on the information within their respective windows without the capability to directly connect with tokens from other windows. To bridge this gap and facilitate cross-window information exchange, carrier token (CT) is introduced. Each local window possesses its unique set of CTs, which encapsulate the essence of the information within that window through pooling. This design allows the local tokens to remain focused on their immediate spatial context and the CTs to communicate with CTs from other windows.

As demonstrated in Figure 4, the cross-window interactions between CTs and the local interactions between local tokens and their associated CTs are both through the self-attention mechanism. This mechanism operates by assigning different importance scores to different positions in the input sequence, allowing it to focus more on relevant or important positions while processing the data [26]. With the help

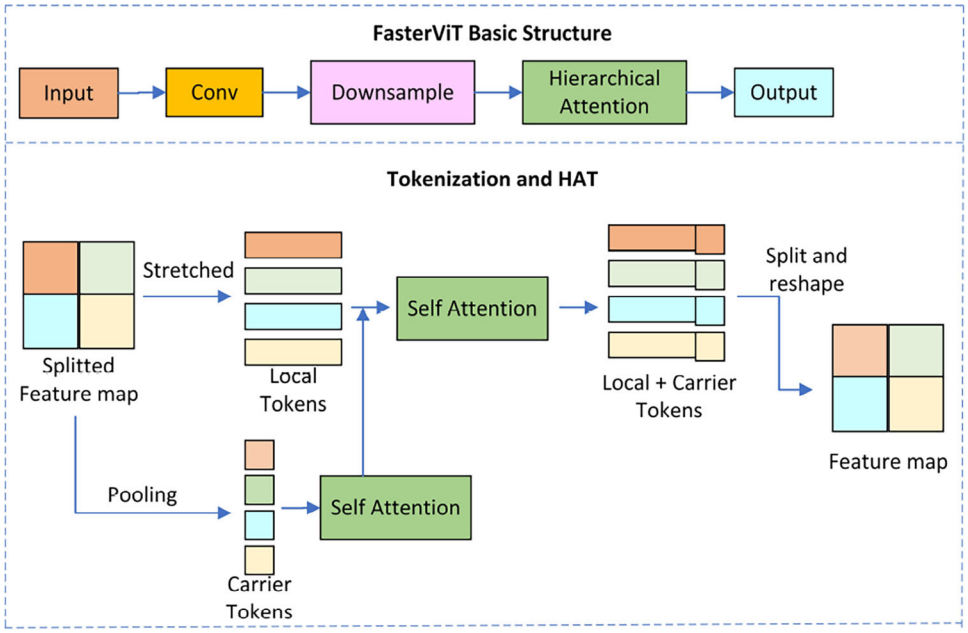


FIGURE 4 FasterViT layer structure.

(a) (b)

FIGURE 5 An example that highlights the strength of FasterViT.

of self-attention, FasterViT will not only capture long-distance relationships through cross-window interactions, but also retain the local details. Consequently, the model incorporating FasterViT modules is better equipped to process images with intricate backgrounds and navigate complex spatial relationships.

Simultaneously, compared to other ViT variants [27, 28], FasterViT captures the relationships between tokens in a more efficient manner. Traditional self-attention mechanisms construct relationships between every pair of tokens, resulting in a computational complexity of $O(n^2)$ for n tokens. In contrast, FasterViT partitions the tokens into m local windows, each containing a subset of the tokens and a carrier token. The computational complexity is then reduced to $O(\frac{n^2}{m} + m^2)$.

The advantages of FasterViT are evident in the following example. As illustrated by Figure 5, both YOLOv8n and its improved version with FasterViT layers (YOLO-F) analyse the same image. While both models successfully identify the fully visible insulators, YOLOv8n fails to detect the one that is largely obscured. Conversely, YOLO-F effectively captures it.

The difference highlighted in Figure 5 underscores the unique strength of FasterViT. As mentioned, FasterViT is adept at discerning long-distance relationships, which enables the model to focus on the intricate connections between the insulator and its surroundings, as opposed to relying solely on potentially incomplete local information. Additionally, the self-attention mechanism of FasterViT allows the model to highlight salient features, thereby gaining a more comprehensive understanding of the object's characteristics. This not only enhances object recognition but also accentuates the distinction between the object and its background during detection.

3.3 | Improved loss function

The loss function of YOLOv8 for detection tasks contains classification loss and bounding box regression loss [29]. The bounding box regression loss comprises both the distribution focal loss (DFL) and the complete intersection over union (CIoU) loss. The DFL is employed to reduce the discrepancy

between the true positive samples and the predictions made by the model [24], while the CIoU loss aims to calculate the distance between the real box and the predicted box. Specifically, the intersection of union (IoU) calculates the overlap between the real box and the predicted box, represented as follows:

$$\mathrm{IoU} = \frac{A_o}{A_u} \tag{4}$$

where A_o is the area of overlap and A_u is the area of union.

The CIoU loss, in addition to the IoU of the boxes, integrates both the distance and aspect ratio effects. However, in scenarios where the dataset contains objects with a wide range of sizes, it becomes more crucial to focus on the absolute differences in width and height rather than the aspect ratio differences. This is because the aspect ratio might remain consistent across different scales, but the absolute size differences can have a more pronounced effect on the detection accuracy. Given this observation, in our model, SIoU, which introduces a different approach to calculating the distance cost and shape cost [30], is utilized to substitute CIoU. The distance cost is calculated with the following operations denoted as follows:

$$\begin{aligned} \rho_x &= \left(\frac{S_{cw}}{cw}\right)^2, \\ \rho_y &= \left(\frac{S_{ch}}{ch}\right)^2, \\ \gamma &= \mathrm{angle\_cost} - 2, \\ d_c &= 2 - e^{\gamma\rho_x} - e^{\gamma\rho_y} \end{aligned} \tag{5}$$

where ρ_x and ρ_y represent the normalized squared distances between the centres of the two bounding boxes in the horizontal and vertical directions, respectively. γ is a factor that incorporates the angle cost. a_c represents the cost of angle differences between the predicted box and the real box. d_c represents the combined cost of the distances in both the horizontal and vertical directions. The next step is to get the shape cost as follows:

$$\begin{aligned} \omega_w &= \frac{|w_1 - w_2|}{\max(w_1 - w_2)}, \\ \omega_h &= \frac{|h_1 - h_2|}{\max(h_1 - h_2)}, \\ s_c &= (1 - e^{-\omega_w})^4 + (1 - e^{-\omega_h})^4 \end{aligned} \tag{6}$$

where ω_w and ω_h are the relative differences in the widths and heights of the two bounding boxes, respectively. s_c represents the combined cost associated with the differences in the shapes of the two bounding boxes. Finally, the distance cost and shape cost are combined to get the SIoU as follows:

$$\mathrm{SIoU} = \beta \times (\mathrm{IoU} - 0.5 \times (d_c + s_c)^{\alpha}) \tag{7}$$

TABLE 1 Hardware parameters of our experiment device.

Parameter	Value
CPU	Intel(R) Xeon(R) CPU E5-2680 v4 @ 2.40GHz
GPU	Nvidia GeForce RTX 2080Ti (11GB)
GPU acceleration library	Cuda 12.1
Training framework	PyTorch
Operating system	Windows 10

where β is a scaling factor that is used to mitigate the imbalance between positive and negative samples.

Compared to CIoU, SIoU emphasizes the absolute differences in width and height between the bounding boxes rather than just the ratio of width to height. Additionally, SIoU incorporates a modified distance cost that accounts for both the relative positions and angles of the boxes. These enhancements lead to improved performance in our model.

Given that SIoU is scaled by a factor greater than 1, it's essential to limit its value at 1. Otherwise, the SIoU loss could potentially drop below 0. In summary, the regression loss function can be represented as follows:

$$\begin{aligned} L_{\mathrm{SIoU}} &= 1 - \max(\mathrm{SIoU}, 1), \\ L_{\mathrm{box}} &= \mathrm{DFL} + L_{\mathrm{SIoU}} \end{aligned} \tag{8}$$

where L_{SIoU} represents the SIoU loss and L_{box} denotes the bounding box regression loss.

4 | EXPERIMENT RESULTS

This section starts with our experimental conditions and parameters in Sections 4.1 and 4.2. The quantitative support for the logic behind the modifications is shown from Sections 4.3 to 4.5. To ensure the robustness of the comparison, the outcomes of experiments in Sections 4.6 and 4.7 were averaged conducted with 3 different random seeds on model parameters and data split.

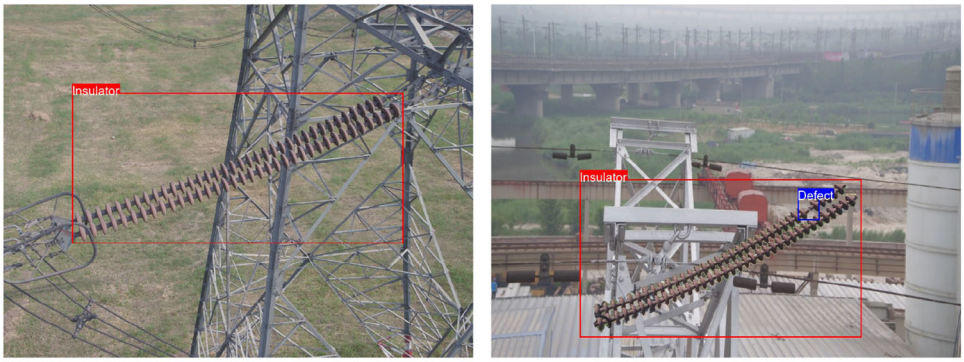
4.1 | Experimental conditions

Table 1 renders the hardware parameters of our experiment device.

In terms of the preprocessing, 1124 authentic images from drones at a specific power station are collected, including both 220 and 110 kV transmission lines. The dataset predominantly contains images of intact insulators, as the real-world maintenance scenario at the power station does not provide enough defective insulators for training. The images originally had a resolution of 5472×3078. We scaled them down and YOLO will then automatically resize the images to squares by filling them. The insulators in the dataset can be categorized into glass,

(a) (b) (c)

FIGURE 6 Photos of three types of insulators in the private dataset: (a) glass; (b) porcelain; (c) composite.



(a) (b)

FIGURE 7 Figures showing an intact insulator and an insulator with a defect from the public dataset: (a) intact insulator; (b) insulator with a defect.

porcelain, and composite, as shown in Figure 6. In this paper, this dataset will be referred to as "the private dataset".

To validate the proposed architecture and assess the model's performance on final defect detection, experiments were conducted using the public Chinese Power Line Insulator Dataset (CPLID) [31]. This dataset includes 848 images of insulators, with 248 images containing defects. Figure 7a illustrates an image with an intact insulator, while Figure 7b depicts an image featuring both an insulator and a defect. To distinguish from the private dataset, this dataset will be referred to as "the public dataset." Compared to the private dataset, the public dataset contains only porcelain insulators and features less complex backgrounds.

In the preprocessing stage, a series of augmentations are implemented: we mirror the images, introduce noise rectangles of random sizes, adjust the saturation, and vary the colour palettes. The introduction of random-sized noise rectangles is designed to mimic a range of real-world interferences, thereby fostering the model's robustness. By varying the colour palettes, the model remains indifferent to particular colour distributions, enhancing its ability to generalize across different scenarios. After these modifications, the total number of images increased to 2248. Then, the team manually annotated the insulators in these images and compiled the training dataset. Table 2 presents the parameters that are kept constant throughout our experiments.

TABLE 2 Constant parameters and their values.

Parameters	Value
Optimizer	SGD
Initial learning rate	0.01
Final learning rate	0.0001
Batch size	16
Epoch	300
Precision mode	Mixed precision

4.2 | Evaluation metrics

In our evaluation, five performance metrics and three efficiency metrics are employed. The performance metrics include precision, recall, F1 score, mAP50, and mAP50-95, while the efficiency metrics focus on training GPU memory usage, training time per epoch, model weight memory, number of parameters, and GFLOPs. Specifically, precision, recall, and

TABLE 3 Final results of the performance metrics across various models with SIoU and CIoU, with the integration of FasterViT layers and GAM combinations.

Models	Loss function	Precision	Recall	F1	mAP50	mAP50-95
YOLOv8n	CIoU	0.915	0.897	0.906	0.941	0.716
	SIoU	0.908	0.917	0.912	0.948	0.703
YOLO-F	CIoU	0.932	0.912	0.922	0.953	0.754
	SIoU	0.933	0.921	0.927	0.955	0.755
YOLO-G	CIoU	0.936	0.914	0.925	0.953	0.722
	SIoU	0.940	0.923	0.931	0.957	0.720
YOLO-FG	CIoU	0.938	0.920	0.929	0.957	0.758
	SIoU	0.948	0.925	0.936	0.961	0.753

the F1 score assess the model's overall accuracy; mAP50 and mAP50-95 measure the model's capability to minimize the discrepancy between the predicted bounding box and the actual box. Regarding efficiency metrics, training GPU memory usage and training speed indicate the model's adaptability to constrained experimental settings; the model size and the number of parameters are indicative of its storage requirements; GFLOPs indicates the computational demand and complexity of the model.

It should be noted that although many previous research papers have focused on detection speed [32–34], it is not considered in our paper. This is because most current insulator detectors loaded in UAVs are based on images rather than videos, which reduces the demands of high-speed processing ability of the model even in real-time analysis. FA-YOLO reaches FPS of 126.6 frames/s, which is far more than the requirement. Therefore, FPS is not included as an efficiency metric.

Beyond the final outcomes for each model, the performance trend lines across various metrics throughout the training process are also explored to visualize the convergence rate and performance in early epochs. For F1, mAP50, and mAP50-95 trend lines, only epochs 0 to 200 are illustrated. This decision was made as the trends stabilized after the 200th epoch, closely corresponding to the final performance shown in the tables.

4.3 | Enhancements on YOLO structure

This subsection delves into the modifications implemented on the YOLOv8 structure. Incorporating the GAM and FasterViT modules results in three models: YOLOv8 enhanced with GAM (YOLO-G), YOLOv8 combined with FasterViT (YOLO-F), and YOLOv8 with both modules integrated (YOLO-FG). Specifically, tables highlighting the optimal parameters of the models with CIoU or SIoU, based on performance and efficiency metrics, are provided. Concurrently, the trend curves from the training, capturing metrics including F1, mAP50, mAP50-95, and box loss, are showcased.

Table 3 presents the performance metrics of the four models, contrasting the effects of CIoU and SIoU. From the data, models utilizing SIoU consistently outperform those employing CIoU in the F1 and mAP50 metrics. Specifically, YOLOv8n with SIoU surpasses its CIoU counterpart by 0.06 in F1 and 0.07 in mAP50. Similarly, YOLO-F with SIoU has a lead of 0.05 in F1 and 0.02 in mAP50, YOLO-G with SIoU advances by 0.06 in F1 and 0.04 in mAP50, and YOLO-FG with SIoU excels by 0.07 in F1 and 0.04 in mAP50. However, when considering the mAP50-95 metric, the models with CIoU demonstrate a contrasting trend. In this regard, YOLOv8n using CIoU leads by 0.13, YOLO-F with CIoU trails by 0.01, YOLO-G with CIoU leads by 0.02, and YOLO-FG with CIoU advances by 0.05. The distinct advantage of SIoU is that it accounts for discrepancies between the width and height of bounding boxes rather than merely relying on a width-to-height ratio. This precision becomes especially beneficial for datasets with objects spanning diverse scales. The SIoU's incorporation of an exponential term serves to amplify the impact of minor discrepancies, resulting in a heightened distance cost for such deviations. This design choice renders SIoU less sensitive to minor discrepancies but more sensitive to pronounced ones. As a result, SIoU presents superior performance on F1 and mAP50 metrics, albeit with a slight compromise on the more rigorous mAP50-95 metric. Given these insights, SIoU is chosen as the loss function for FA-YOLO.

In assessing the performance of models that utilize SIoU, two variations of YOLOv8n are examined. The first variation integrates FasterViT layers into the backbone. According to the results, YOLO-F demonstrates enhanced outcomes in F1, mAP50, and mAP50-95 metrics. Owing to its increased complexity and superior feature extraction capabilities, YOLO-F can accurately detect complex scenarios, leading to a notable increase of 0.052 in the stringent mAP50-95 metric. The second variation focuses on the inclusion of GAM in the neck. This augmentation, which improves the global perspective of the detection blocks, yields more significant improvements in F1 and mAP50, with gains of 0.019 and 0.009, respectively. However, due to its constrained complexity, it shows a modest rise of 0.017 in mAP50-95 compared to the FasterViT layers. By combining the strengths of both modules, YOLO-FG achieves the highest scores in metrics for precision, recall, F1, and mAP50, recording values of 0.948, 0.936, 0.961, and 0.753, respectively. In terms of mAP50-95, YOLO-FG trails slightly behind YOLO-F by just 0.02. These results demonstrate the effectiveness and the enhanced performance that result from integrating FasterViT and GAM into YOLOv8.

In addition to the final performance of each model, we further analyse the performance trend during the training. In Figures 8a, 8b, and 8c, the F1, mAP50, and mAP50-95 trend curves for YOLOv8n, YOLO-F, YOLO-G, and YOLO-FG are presented. From Figures 8a and 8b, it's evident that YOLO-F and YOLO-FG significantly outperform the others up to the 80th epoch. However, as training progresses, the performance gap narrows, with the F1 and mAP50 scores of all models converging. The enhanced early learning capability provided by the FasterViT module is attributed to its attention mechanism, which rapidly identifies critical features. This accelerated identification boosts the model's understanding of both local and

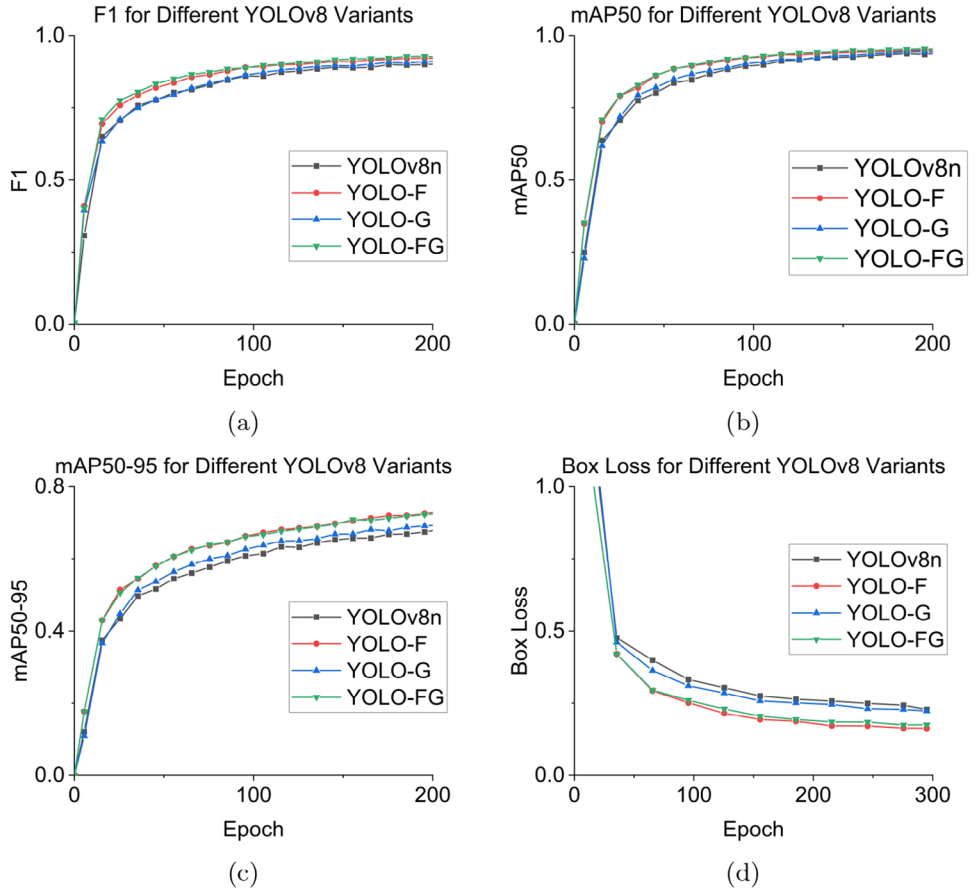


FIGURE 8 Trend curves of performance metrics and box loss across various models, with the integration of FasterViT layers and GAM combinations: (a) F1, (b) mAP50, (c) mAP50-95, (d) box loss.

global nuances, thus facilitating rapid performance improvements in the initial stages of training. In terms of mAP50-95, YOLO-F, and YOLO-FG demonstrate parallel growth rates and outperform both YOLO-G and YOLOv8n throughout the training, highlighting the advancements on mAP50-95 brought by the FasterViT module. Figure 8d illustrates the box loss trend across all models. As observed, the box loss for all models begins to converge from the 60th epoch, indicating a comparable rate of convergence. However, due to their increased complexity, YOLO-F and YOLO-FG can capture more intricate features, leading to a reduced loss throughout the learning.

Regarding the efficiency, YOLO-FG exhibits a marked increase in model size and training complexity compared to YOLOv8n. This is evident from an increase of 0.93 GB in training GPU consumption, an additional 8 s per epoch in training speed, a growth of 14,452 KB in weight memory, an increase of 7.5 million in the number of parameters, and an addition of 14.9 in GFLOPs, as indicated in Table 4. Conversely, the efficiency impact on YOLO-G is less pronounced. As shown in Table 4, YOLO-G experiences an increase of only 0.4 GB in training GPU consumption, a 2-s addition per epoch in training speed, a growth of 465 KB in weight memory, an increment of 0.3 million in the number of parameters, and a mere 0.8 rise in GFLOPs compared to YOLOv8n. Analysing the data for YOLO-F and YOLO-G, it becomes clear that the decline in model efficiency is primarily due to the FasterViT modules. While increased complexity can lead to enhanced performance, especially in stringent metrics such as mAP50-95, the results underscore the balance that must be struck with the efficiency.

4.4 | Variants based on GAM

In this section, the results of YOLO-F improved by different versions of GAM will be presented. The naming convention for the variants is based on modifications in the GAM module. The suffix "G" in GAM-G denotes the use of graph convolution for channel attention, "D" in GAM-D indicates depthwise convolution for spatial attention, and "R" in GAM-R signifies the addition of a residual module. When combined, their respective letters represent the integration of these modifications. For instance, GAM-GD refers to the GAM module with both graph and depthwise convolutions.

To evaluate the effect of the refinements, a table showcasing the performance of YOLO-F across different GAM variants will be presented. Also, for kempt, the trend curves on F1, mAP50, mAP50-95, and box loss of YOLO-F with

TABLE 4 Final results of the efficiency metrics across various models, with the integration of FasterViT layers and GAM combination.

Models	Training GPU consumption (GB)	Training speed (s/epoch)	Weight memory (KB)	Parameters (M)	GFLOPs
YOLOv8n	4.81	19	6104	3	8.2
YOLO-F	5.45	25	20301	10.3	22.3
YOLO-G	5.21	21	6569	3.3	9.0
YOLO-FG	5.74	28	20556	10.5	23.1

TABLE 5 Final results of the performance metrics with different additional GAM variants on YOLO-F on the private dataset.

Additional module on YOLO-F	Precision	Recall	F1	mAP50	mAP50-95
GAM	0.943	0.921	0.932	0.958	0.753
GAM-G	0.949	0.923	0.936	0.959	0.757
GAM-D	0.937	0.905	0.921	0.955	0.736
GAM-R	0.926	0.939	0.933	0.955	0.738
GAM-GD	0.939	0.937	0.938	0.96	0.737
GAM-DR	0.927	0.94	0.934	0.957	0.745
GAM-GR	0.944	0.908	0.926	0.955	0.746
GAM-GDR	0.950	0.926	0.938	0.961	0.757

GAM, GAM-GD, and GAM-GDR during training. Since the modifications made on a single module do not significantly impact the model's efficiency, the results on efficiency metrics are disregarded.

Table 5 shows the comparison of YOLO-FG with all GAM variants. According to the table, the model with GAM-GDR boasts the highest precision, F1, mAP50, and mAP50-95 scores, reaching 0.95, 0.938, 0.961, and 0.757, respectively. The results indicate that making isolated adjustments to either the spatial or channel attention mechanisms doesn't significantly elevate the overall efficacy of GAM because the remaining interconnections from one side still influence the other. Introducing just the residual module to GAM, which already possesses a balanced complexity, might lead to overfitting. While GAM-GD outperforms the standard GAM in terms of recall and F1 score and maintains mAP50, its reduced complexity leads to the aforementioned problem, which is mitigated by integrating a residual module, thus bolstering its mAP50-95 performance. With the refined spatial and channel attention mechanisms and the residual component, the refined GAM minimizes overfitting and counteracts the drawbacks of diminished complexity, leading to enhanced performance.

Figure 9 demonstrates the trend curves of different metrics on variants of GAM as follows:

Figures 9a, 9b, and 9c display the comparative analysis of F1, mAP50, and mAP50-95 trend rates across GAM, GAM-GD, and GAM-GDR, respectively. It is clear in Figure 9 that before the 50th epoch, both FA-YOLO and YOLO-FG with the standard GAM exhibit almost identical growth rates in F1 and mAP50, and both surpass YOLO-FG with GAM-GD. After the 50th epoch, the growth rate starts to converge, and all three models follow a similar trajectory. The reduced rate of increase observed in GAM-GD can be attributed to the complexity reduction introduced by the modifications in GAM. The incorporation of depthwise convolution adds a level of spatial granularity, allowing the model to focus on more localized features. Concurrently, the graph convolution, designed to capture intricate interchannel relationships, introduces a more focused structure that gives up some features during training. These modifications require the model to invest more time during the initial epochs to grasp complex relationships. This is addressed by the incorporation of the residual module, which maintains the learning rate at the early stage of the training and mitigates the gradient vanishing problem encountered in the deep layers of the model, ensuring a consistent improvement in performance.

There is a noticeable difference in mAP50-95 curves between YOLO-FG with GAM and YOLO-FG with GAM-GD. The targeted approach of GAM-GD with reduced complexity may lead to the omission of certain features and detailed differences. While this might not significantly impact most performance metrics, the aforementioned limitation is amplified under the stringent evaluation criteria of mAP50-95. This limitation is also mitigated by the incorporation of the residual module, which allows deeper layers to learn the features at a lower level. By integrating features in higher and earlier levels, the model captures a broader context and refines the details, thereby enhancing its performance under mAP50-95.

In conclusion, by incorporating graph convolutions, depthwise convolutions, and residual models, FA-YOLO demonstrates enhanced performance and convergence rate over YOLO-FG and all other permutations and combinations of modifications.

4.5 | Detection samples

To visualize the discrepancies, Figure 10 is included to present a comparison of the outcomes using identical input in different situations. The predicted bounding boxes for insulators are depicted as red rectangles in the images.

Based on the figures, the displayed results reflect our primary modifications: the addition of FasterViT and GAM, as well as alterations to the GAM structure. Besides, YOLOv8s is included in the comparison to demonstrate the advantages of FA-YOLO over larger models. The four columns correspond,

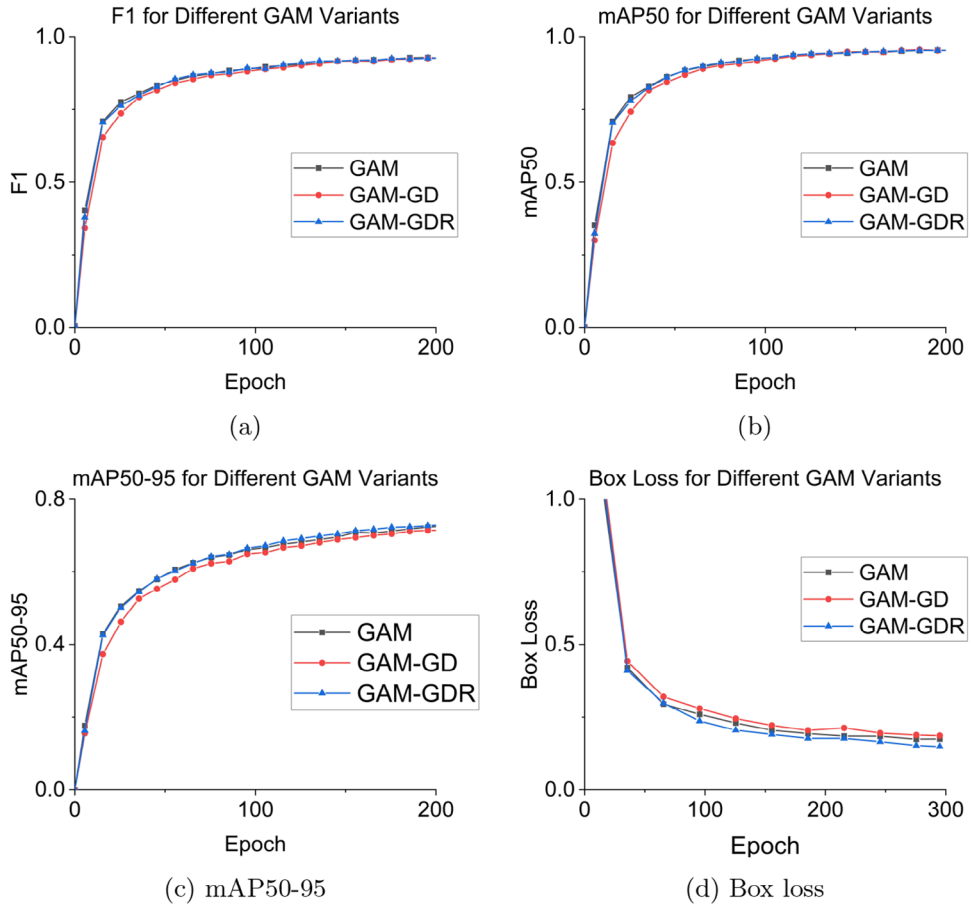


FIGURE 9 Trend curves of performance metrics and box loss across YOLO-F with different variants of GAM. (a) F1, (b) mAP50, (c) mAP50-95, (d) Box loss.

respectively, to the outputs from YOLOv8n, YOLO-FG, YOLOv8s, and FA-YOLO.

In Figure 10a–d, both the porcelain insulators and the tower pole are white, blurring the distinction between them based solely on colour variations. Faced with this challenge, YOLOv8n struggles to detect the insulators when only a small portion is visible. It misses the diminutive object at the bottom and fails to address the intricate overlap of one large box with two smaller ones. While YOLO-FG manages to discern the overlapping insulators, it neglects the minor details at the bottom. YOLOv8s, on the other hand, successfully detects the bottom insulator but makes false detection on the top overlapping. In contrast, FA-YOLO adeptly navigates these challenges, accurately identifying the correct objects.

In Figure 10e–h, the images pose a more complex overlapping challenge: beyond the mere overlap of boxes, the smaller insulator is obscured by the larger one. In this scenario, while YOLOv8n recognizes the overlap, it struggles to distinguish the insulators based on their size disparity. Conversely, YOLO-FG fails to detect the overlap. Yet, YOLOv8s and FA-YOLO adeptly navigate this intricacy and resolve the issue effectively.

Figure 10i–l presents a combination of challenges. In the top right, there's an object with a colour and disc-stacked shape similar to the insulators. The smaller insulator is obscured by the similarly coloured tower and appears split into two. Additionally, there's a complex overlap of objects in the image's centre. All models, except for FA-YOLO, misidentify the object with similar characteristics. Furthermore, YOLOv8s and YOLO-FG struggles to address the intricate overlapping in the centre and at the right, respectively. Only FA-YOLO successfully detects all the insulators without any misidentification.

Through these three concrete examples, the superior performance of FA-YOLO becomes visualized. With reduced misclassifications and misdetections, as illustrated in these examples, the enhanced accuracy in pinpointing the insulator and its location provides a more robust basis for insulator defect detection. The defect detection will be improved by a more intense comprehension of the insulator features. Therefore, FA-YOLO is especially crucial in situations marked by subtle differences in colour and shape, complex overlapping issues, and cases where objects are concealed by others.

4.6 | Comparable attention mechanisms

This section provides a comparative analysis of the introduced attention mechanisms, GAM-GDR and FasterViT, with

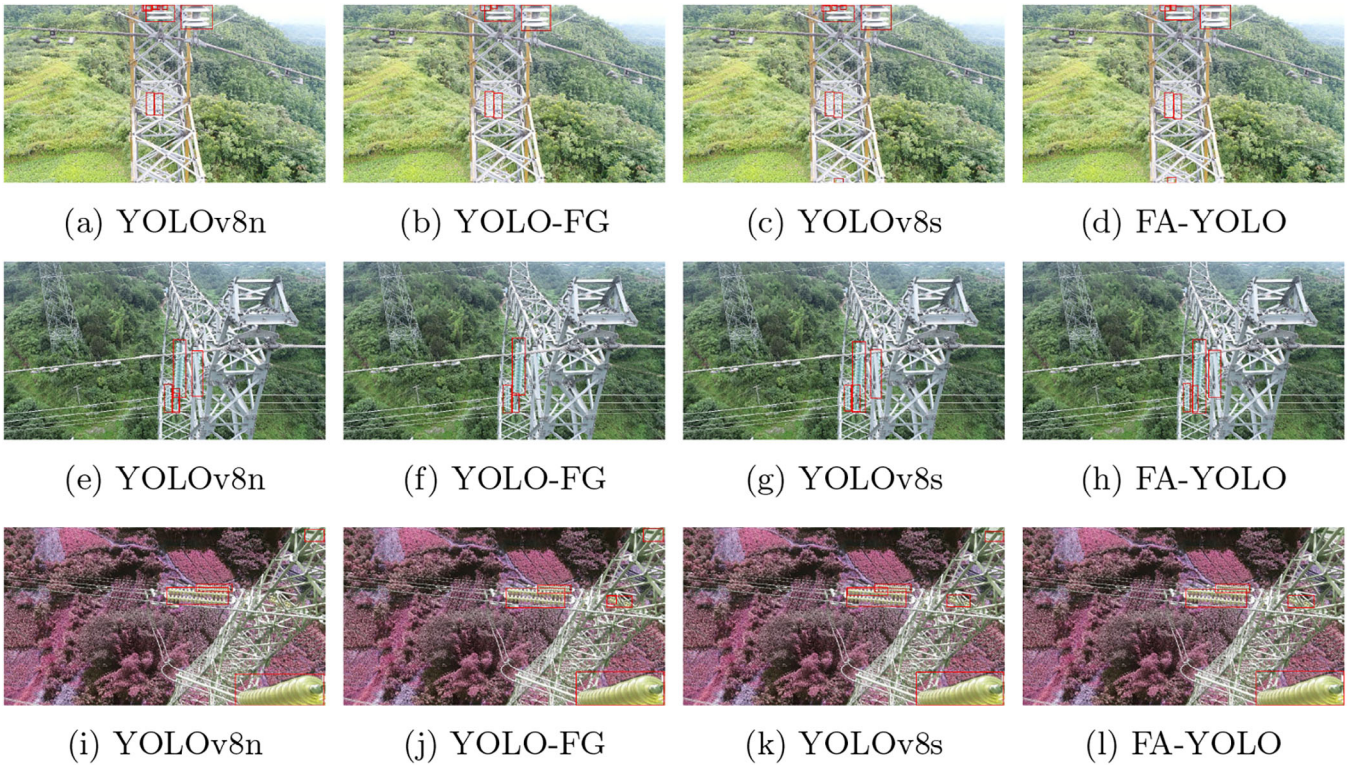


FIGURE 10 Sample outputs from different models. Each row represents different samples and each column represents different models: (a), (e), and (i) are predicted by YOLOv8n; (b), (f), and (j) are predicted by YOLO-FG; (c), (g), and (k) are predicted by YOLOv8s; (d), (h), and (l) are predicted by FA-YOLO.

other state-of-the-art attention mechanisms. As detailed in Section 3.1, GAM-GDR integrates both channel and spatial attention, which is conceptually aligned with the convolutional block attention module (CBAM) [35]. Additionally, we examine the channel attention mechanisms employed in global context network (GCNet) [36], gather-excite network (GENet) [37], and squeeze-and-excitation networks (SENet) [38], and include a comparative evaluation of FasterViT with swin transformer and revisiting mobile CNN from ViT perspective (RepViT) [39]. In the experiments, considering the similarities and complexities among these attention mechanisms, we substituted GAM-GDR with CBAM, GCNet, and SENet, and replaced FasterViT with Swin and RepViT.

The experiments were conducted on both the private dataset and the public dataset. To ensure the robustness of our results, we averaged the outcomes of experiments conducted with 3 different random seeds on model parameters and data split. For defect detection on the public dataset, mAP50 and mAP50-95 were not utilized as primary metrics because the most important metric is to accurately identify the presence of defects instead of evaluating the localization performance, which is better reflected by F1 score.

According to Table 6, incorporating spatial attention on top of channel attention improves the performance of models with CBAM and GAM-GDR in insulator detection. Although defects do not have a specific shape, a better understanding of the insulator string helps the model to localize missing shells or other defects. Consequently, the model can be more adept to identify the existence of the defects on the insulators. The addition of ResNet in GAM-GDR improves the performance of FA-YOLO with CBAM by reducing the effects of gradient vanishing in deep layers, allowing FA-YOLO to learn more complex features. As a result, the YOLOv8n incorporating GAM-GDR and FasterViT achieves an F1 score of 0.961 for defect detection and 0.997 for insulator string detection, both of which are the highest among the comparable attention mechanisms. This improvement is more evident in the private dataset, as described in the next paragraph. Compared to Swin, FasterViT is smaller in size, allowing it to be applied to different feature map scales. Additionally, with global attention, FasterViT can achieve a better comprehension of the background contextual information. This results in better performance on datasets with varying object sizes. In comparison to RepViT, which employs the reparameterization technique for detection, FasterViT demonstrates superior performance in feature capturing during the training phase.

Table 7 demonstrates the performance of these attention mechanisms on a dataset with a greater variety of insulator categories and more complex backgrounds. FA-YOLO, enhanced with both spatial and channel attention, improved feature map scale fusion, and a better understanding of complex features, shows significant improvements in performance, leading to superior results across all five metrics.

TABLE 6 Final results of the performance metrics with different attention mechanisms on the public dataset.

Attention mechanism	Class	Precision	Recall	F1 score
CBAM + FasterViT	All	0.971	0.973	0.972
	Defect	0.965	0.953	0.959
	String	0.978	0.992	0.985
GCNet + FasterViT	All	0.964	0.973	0.968
	Defect	0.956	0.946	0.951
	String	0.972	1.000	0.986
GENet + FasterViT	All	0.965	0.977	0.971
	Defect	0.947	0.954	0.951
	String	0.983	0.999	0.991
SENet + FasterViT	All	0.968	0.970	0.969
	Defect	0.963	0.940	0.951
	String	0.975	1.000	0.987
GAM-GDR + Swin	All	0.965	0.978	0.971
	Defect	0.959	0.957	0.958
	String	0.971	0.998	0.984
GAM-GDR + RepViT	All	0.965	0.975	0.970
	Defect	0.948	0.950	0.949
	String	0.982	1.000	0.991
FA-YOLO	All	0.977	0.981	0.979
	Defect	0.960	0.961	0.961
	String	0.994	1.000	0.997

TABLE 7 Final results of the performance metrics with different attention mechanisms on the private dataset.

Attention mechanism	Precision	Recall	F1 score	mAP50	mAP50-95
CBAM + FasterViT	0.928	0.908	0.918	0.948	0.735
GCNet + FasterViT	0.936	0.888	0.911	0.945	0.724
GENet + FasterViT	0.929	0.888	0.908	0.942	0.723
SENet + FasterViT	0.918	0.915	0.916	0.948	0.717
GAM-GDR + Swin	0.930	0.920	0.925	0.957	0.719
GAM-GDR + RepViT	0.930	0.914	0.922	0.950	0.730
FA-YOLO	0.948	0.922	0.935	0.958	0.754

TABLE 8 Final results of the performance metrics across different comparable models on the public dataset.

Model	Class	Precision	Recall	F1 score
Faster R-CNN	All	0.657	0.998	0.784
	Defect	0.529	1.000	0.692
	String	0.785	0.995	0.876
SSD	All	0.993	0.502	0.656
	Defect	1.000	0.375	0.545
	String	0.985	0.629	0.767
YOLOv5m	All	0.971	0.973	0.972
	Defect	0.965	0.953	0.959
	String	0.978	0.992	0.985
YOLOv7n	All	0.964	0.973	0.968
	Defect	0.956	0.946	0.951
	String	0.972	1.000	0.986
YOLOv8n	All	0.965	0.977	0.971
	Defect	0.947	0.954	0.951
	String	0.983	0.999	0.991
YOLOv8s	All	0.968	0.970	0.969
	Defect	0.963	0.940	0.951
	String	0.975	1.000	0.987
FA-YOLO	All	0.977	0.981	0.979
	Defect	0.960	0.961	0.961
	String	0.994	1.000	0.997

TABLE 9 Final results of the performance metrics across different comparable models on the private dataset.

Model	Precision	Recall	F1 score	mAP50	mAP50-95
Faster R-CNN	0.611	0.896	0.727	0.864	0.547
SSD	0.954	0.705	0.811	0.871	0.589
YOLOv5m	0.932	0.928	0.930	0.954	0.737
YOLOv7n	0.895	0.878	0.886	0.900	0.620
YOLOv8n	0.921	0.908	0.914	0.943	0.699
YOLOv8s	0.922	0.910	0.916	0.950	0.733
FA-YOLO	0.948	0.922	0.935	0.958	0.754

4.7 | Final performance against other comparable models

In this section, a comparative analysis of FA-YOLO's performance and efficiency against other notable models, including Faster R-CNN, SSD, YOLOv5, YOLOv7, and YOLOv8, is explored.

According to Tables 8 and 9, YOLO series demonstrates superior performance in terms of F1, mAP50, and mAP50-95 metrics. While faster R-CNN exhibits significantly lower precision, SSD shows substantially lower recall on both private and public datasets. This disparity is attributed to the fact that faster R-CNN and SSD require much more epochs to converge. The lower mAP50-95 scores further suggest that these models struggle to learn the complex features of insulators and backgrounds, resulting in poorer performance on more stringent evaluation metrics.

From Table 10, we can observe the advantages of the YOLO series. In addition to their superior performance, the YOLO series models exhibit lower weight memory, parameter count, and GFLOPs compared to Faster R-CNN and SSD. Furthermore, regarding training demands, while Faster R-CNN and SSD have similar GPU usage, they exhibit significantly lower training speeds compared to the YOLOv8 variants. Among

TABLE 10 Final results of the efficiency metrics across different comparable models.

Model	Training GPU consumption (GB)	Training speed (s/epoch)	Weight memory (KB)	Parameters (M)	GFLOPs
Faster R-CNN	6.29	95	110770	28.3	227.8
SSD	5.24	49	92782	23.9	30.6
YOLOv5m	6.29	95	41141	20.9	48.2
YOLOv7n	7.00	166	18974	9.3	26.7
YOLOv8n	4.81	19	6104	3.0	8.2
YOLOv8s	8.29	28	21992	11.2	28.8
FA-YOLO	5.83	26	20723	10.5	23.1

YOLO variants, there's a general trend observed: as efficiency metrics improve, performance metrics tend to decline. However, the extent of this trade-off varies based on the specific implementation. In terms of training difficulty, FA-YOLO consumes 5.83 GB of GPU memory and takes 26 s for each training epoch. As outlined in Table 10, FA-YOLO ranks second in training speed and GPU consumption. In terms of model complexity, FA-YOLO utilizes 20723 KB of weight memory, consists of 10.5 million parameters, and operates at 23.1 GFLOPs. As observed, FA-YOLO is comparable to YOLOv7n, slightly less complex than YOLOv8s, and considerably simpler than YOLOv5m. Yet, when compared to YOLOv8n, FA-YOLO exhibits a marked increase in both training requirements and model size. Notably, FA-YOLO achieves the highest scores in F1, mAP50, and mAP50-95 in both datasets. When compared with YOLOv8s, FA-YOLO exhibits an enhancement of 0.019 in F1, 0.008 in mAP50, and 0.021 in mAP50-95 in the private dataset, and an improvement of 0.01 in both defect and string classes in the public dataset. Concurrently, it demonstrates a reduction of 2.46 GB in training GPU usage, 2 s per epoch in training speed, 1269 KB in model dimensions, 0.7 million in parameter count, and 5.7 GFLOPs in computational complexity. This indicates that integrating attention mechanisms into YOLO can significantly boost performance, offering advantages beyond simply expanding the model's depth and width. Simultaneously, the model controls the increase in complexity, achieving a harmonious balance between efficiency and performance. While maintaining a relatively manageable model size, it delivers the highest performance among the models compared.

5 | CONCLUSION

Given the increasing ubiquity of UAV-based inspections, there is an escalating demand for high-precision object detection algorithms. To tackle the challenges of intricate images and memory constraints in UAVs, we propose FA-YOLO, a refined insulator detection model based on YOLOv8. This model is designed to guarantee a harmonious balance of accuracy, model complexity, and GPU memory utilization. By replacing C2f blocks with the FasterViT layers in the backbone, the model achieves a stronger feature extraction ability, particularly designed for intricate scenarios. Additionally, GAM-GDR is proposed and integrated in the neck of FA-YOLO to mitigate the issue of gradient diminishing while achieving a more comprehensive insight into the feature maps. Furthermore, the introduction of SIoU fine-tunes both the distance cost and the shape loss. Experimental results reveal that FA-YOLO outperforms existing models across multiple performance metrics, reaching the highest scores in F1, mAP50, and mAP50–95. Simultaneously, it substantially mitigates the GPU memory burden that models with similar performance require during training. Moreover, it controls the number of parameters and computation demand, making it suitable for practical applications. In future research, we plan to investigate the performance of our enhancement techniques on various types of insulators and explore strategies to elevate detection efficacy for each insulator category.

AUTHOR CONTRIBUTIONS

Yixiao Jing: Conceptualization; data curation; formal analysis; methodology; software; validation; visualization; writing—original draft; writing—review and editing. **Tao Huang**: Supervision; writing—review and editing. **Lingfeng Gao**: Data curation; formal analysis; methodology; software; writing—original draft. **Jiangli Deng**: Resources.

CONFLICT OF INTEREST STATEMENT

The authors declare no conflicts of interest.

DATA AVAILABILITY STATEMENT

The data that support the findings of this study were provided by the Yibin Power Supply Company, State Grid. The data are not publicly available due to confidentiality agreements and are restricted for use by external parties. Data inquiries can be directed to Jiangli Deng, who is a co-author of this study, for further information, subject to the approval and terms set by Yibin Power Supply Company, State Grid.

REFERENCES

1. Brancato, E.A.: Insulation aging a historical and critical review. IEEE Trans. Electr. Insul. El-13(4), 308–317 (1978)
2. Stefenon, S. F., Americo, J. P., Meyer, L. H., Grebogi, R. B., Nied, A.: Analysis of the electric field in porcelain pin-type insulators via finite elements software. IEEE Lat. Am. Trans. 16(10), 2505–2512 (2018). https://doi/org/10.1109/TLA.2018.8795129

3. Wang, L., Wang, H.: A survey on insulator inspection robots for power transmission lines. In: 2016 4th International Conference on Applied Robotics for the Power Industry (CARPI), pp. 1–6. IEEE, Piscataway, NJ (2016)
4. Wang, J., Li, Y., Chen, W.: Detection of glass insulators using deep neural networks based on optical imaging. Remote Sens. 14(20), 5153 (2022). https://doi.org/10.3390/rs14205153
5. Hao, K., Chen, G., Zhao, L., Li, Z., Liu, Y., Wang, C.: An insulator defect detection model in aerial images based on multiscale feature pyramid network. IEEE Trans. Instrum. Meas. 71, 1–12 (2022). https://doi/org/10.1109/TIM.2022.3200861
6. Hao, J., Sen, Y., Huang, X.: Based on surf feature extraction and insulator damage identification for capsule networks. In: 2019 International Conference on Computer Network, Electronic and Automation (ICCNEA), pp. 301–306. IEEE, Piscataway, NJ (2019). https://doi/org/10.1109/ICCNEA.2019.00064
7. Zhai, Y., Chen, R., Yang, Q., Li, X., Zhao, Z.: Insulator fault detection based on spatial morphological features of aerial images. IEEE Access 6, 35316–35326 (2018)
8. Gao, Z., Yang, G., Li, E., Liang, Z.: Novel feature fusion module-based detector for small insulator defect detection. IEEE Sensors J. 21(15), 16807–16814 (2021). https://doi/org/10.1109/JSEN.2021.3073422
9. Ren, S., He, K., Girshick, R., Sun, J.: Faster R-CNN: Towards real-time object detection with region proposal networks. IEEE Trans. Pattern Anal. Mach. Intell. 39(6), 1137–1149 (2016)
10. Zhao, W., Xu, M., Cheng, X., Zhao, Z.: An insulator in transmission lines recognition and fault detection model based on improved faster rcnn. IEEE Trans. Instrum. Meas. 70, 1–8 (2021). https://doi/org/10.1109/TIM.2021.3112227
11. Liao, S., An, J.: A robust insulator detection algorithm based on local features and spatial orders for aerial images. IEEE Geosci. Remote Sens. Lett. 12(5), 963–967 (2015). https://doi/org/10.1109/LGRS.2014.2369525
12. Zhao, Z., Zhen, Z., Zhang, L., Qi, Y., Kong, Y., Zhang, K.: Insulator detection method in inspection image based on improved faster R-CNN. Energies 12(7), 1204 (2019). https://doi.org/10.3390/en12071204
13. Song, Z., Huang, X., Ji, C., Zhang, Y.: Intelligent identification method of hydrophobic grade of composite insulator based on efficient GA-YOLO former network. IEEJ Trans. Elec. Electron. Eng. 18(7), 1160–1175 (2023). https://doi/org/10.1002/tee.23822
14. Miao, X., Liu, X., Chen, J., Zhuang, S., Fan, J., Jiang, H.: Insulator detection in aerial images for transmission line inspection using single shot multibox detector. IEEE Access 7, 9945–9956 (2019). https://doi/org/10.1109/ACCESS.2019.2891123
15. Liu, C., Wu, Y., Liu, J., Han, J.: MTI-YOLO: a light-weight and real-time deep neural network for insulator detection in complex aerial images. Energies 14(5), 1426 (2021). https://doi/org/10.3390/en14051426.
16. Bi, Z., Jing, L., Sun, C., Shan, M.: YOLOx++ for transmission line abnormal target detection. IEEE Access 11, 38157–38167 (2023). https://doi/org/10.1109/ACCESS.2023.3268106
17. Ultralytics. YOLOv8. https://docs.ultralytics.com/ (2023). Accessed 21 June 2023.
18. He, M., Qin, L., Deng, X., Liu, K.: MFIi-YOLO: multi-fault insulator detection based on an improved YOLOv8. IEEE Trans. Power Delivery 39(1), 168–179 (2023)
19. Oh, G., Lim, S.: One-stage brake light status detection based on YOLOv8. Sensors 23(17), 7436 (2023). https://www.mdpi.com/1424-8220/23/17/7436
20. Xu, Z., Zhang, W., Ye, M., Zhu, C., Dai, B., Li, H., Liu, Y., Wang, X.: CenterNet3D: an anchor-free object detection method for point cloud data. arXiv:2007.07214 (2020)
21. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: unified, real-time object detection. arXiv:1506.02640 (2016)
22. Liu, Y., Shao, Z., Hoffmann, N.: Global attention mechanism: Retain information to enhance channel-spatial interactions. arXiv:2112.05561 (2021)
23. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 770–778. IEEE, Piscataway, NJ (2016)
24. Zhao, X., Ding, W., An, Y., Du, Y., Yu, T., Li, M., Tang, M., Wang, J.: Fast segment anything. arXiv:2306.12156 (2023)
25. Hatamizadeh, A., Heinrich, G., Yin, H., Tao, A., Alvarez, J. M., Kautz, J., Molchanov, P.: FasterViT: fast vision transformers with hierarchical attention. arXiv:2306.06189 (2023)
26. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., Polosukhin, I.: Attention is all you need. arXiv:1706.03762 (2017)
27. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., Houlsby, N.: An image is worth 16x16 words: transformers for image recognition at scale. arXiv:2010.11929 (2021)
28. Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., Guo, B.: Swin transformer: hierarchical vision transformer using shifted windows. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), pp. 10012–10022. IEEE, Piscataway, NJ (2021)
29. Terven, J., Cordova-Esparza, D.M.: A comprehensive review of YOLO: From YOLOv1 and beyond. arXiv:2304.00501 (2023)
30. Gevorgyan, Z.: Siou loss: More powerful learning for bounding box regression. arXiv:2205.12740 (2022)
31. Tao, X., Zhang, D., Wang, Z., Liu, X., Zhang, H., Xu, D.: Detection of power line insulator defects using aerial images analyzed with convolutional neural networks. IEEE Trans. Syst., Man, Cybern.: Syst. 50(4), 1486–1498 (2018)
32. Shi, W., Lyu, X., Han, L.: An insulator detection model using bidirectional feature fusion structure based on YOLO X. In: 2022 IEEE 17th Conference on Industrial Electronics and Applications (ICIEA), pp. 881–886. IEEE, Piscataway, NJ (2022)
33. Han, G., Zhao, L., Li, Q., Li, S., Wang, R., Yuan, Q., He, M., Yang, S., Qin, L.: A lightweight algorithm for insulator target detection and defect identification. Sensors 23(3), 1216 (2023)
34. Shuang, F., Han, S., Li, Y., Lu, T.: RSIn-Dataset: an UAV-based insulator detection aerial images dataset and benchmark. Drones 7(2), 125 (2023)
35. Woo, S., Park, J., Lee, J.-Y., Kweon, I.S.: CBAM: convolutional block attention module. In: Proceedings of the European Conference on Computer Vision (ECCV), pp. 3–19. Springer, Cham (2018)
36. Cao, Y., Xu, J., Lin, S., Wei, F., Hu, H.: GCNet: non-local networks meet squeeze-excitation networks and beyond. In: Proceedings of the IEEE International Conference on Computer Vision Workshops (ICCVW), pp. 1971–1980. IEEE, Piscataway, NJ (2019)
37. Hu, J., Sun, G., Li, X., Xue, X., Qin, S.: Gather-excite: Exploiting feature context in convolutional neural networks. In: NIPS'18: Proceedings of the 32nd International Conference on Neural Information Processing Systems, pp. 9401–9411. ACM, New York (2018)
38. Hu, J., Shen, L., Sun, G.: Squeeze-and-excitation networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 7132–7141. IEEE, Piscataway, NJ (2018)
39. Ding, X., He, J., Liu, Z., Li, Z., Sun, H., Wang, Q., Han, S.: RepViT: revisiting mobile CNN from ViT perspective. arXiv:2307.09283 (2023)

How to cite this article: Jing, Y., Huang, T., Gao, L., Deng, J.: Insulator detection based on FA-YOLO network with improved feature extraction ability. IET Image Process. 18, 3600–3616 (2024). https://doi.org/10.1049/ipr2.13197